Modellbasierte Mehrzieloptimierung mit Neuronalen Netzen und Evolutionsstrategien

Dissertation

zur Erlangung des akademischen Grades

Doktoringenieur
(Dr.-Ing.)

vorgelegt dem Fakultätsrat
der Fakultät für Informatik und Automatisierung
der Technischen Universität Ilmenau

von Dipl.-Ing. Dany Meyer
geb. am 17.06.1968 in Frankfurt/Oder

Gutachter:

1. Univ.-Prof. Dr.-Ing. habil. Horst Puta, TU Ilmenau
2. Univ.-Prof. Dr. Dr. h.c. Wilfried Brauer, TU München
3. Univ.-Prof. Dr.-Ing. Ralf Möller, Universität Bielefeld

Tag der Einreichung: 23.10.2003
Tag der öffentlichen wissenschaftlichen Aussprache: 06.12.2004

urn:nbn:de:gbv:ilm1-2004000176

Danksagung

An dieser Stelle sei allen ganz herzlich gedankt, die mich in den vergangenen Jahren bei der Fertigstellung meiner Dissertation unterstützt haben.

Mein ganz besonderer Dank gilt meinem Doktorvater Herrn Professor Dr. Dr.h.c. Wilfried Brauer für seine unermüdliche Unterstützung sowie die persönliche und kontinuierliche Betreuung meiner Arbeit. Durch die vielen Gespräche, Anregungen und Korrekturen verhalf er mir zu einem klaren Verfolgen meiner Ziele und ließ mir bei deren Umsetzung immer freie Hand.

Prof. Dr. Ralf Möller möchte ich für seinen konstruktiven fachlichen Rat und seine große Geduld bei unseren vielen Diskussionen danken. Er war mir auch in schwierigen Phasen der Arbeit ein wichtiger Ratgeber und immer ein guter Freund, dessen Motivation und Unterstützung für mich von unschätzbarem Wert waren.

Über die Tatsache, dass ich Herrn Prof. Dr.-Ing. Horst Puta als Betreuer gewinnen konnte, habe ich mich sehr gefreut und danke ihm für seine sofortige Bereitschaft und stets freundliche Unterstützung. Er ermöglichte mir einen fachlich fundierten und systematischen Einblick in die klassische, kontinuierliche Optimierung, für den ich ihm sehr dankbar bin.

Prof. Dr. Horst Michael Groß und den Mitarbeitern des Fachbereichs Neuroinformatik möchte ich für ihre unkomplizierte und rasche Unterstützung bei meiner Aufnahme als externe Doktorandin an der Fakultät für Automatisierung und Informatik und ihre Hilfe bei fachlichen und organisatorischen Problemen danken.

Mein großer Dank gilt allen Kolleginnen und Kollegen der Kratzer Automation AG, bei der diese Dissertation entstanden ist. Besonders danken möchte ich Dr. Margit Sturm, Dr. Michael Sturm, Dr. Klaus Eder, Dr. Markus König und Winfried Scheidler für ihre regen Diskussionsbeiträge. Von Herzen bedanke ich mich bei Gerhard Kratzer für die langjährige Unterstützung und die ausgezeichneten Arbeitsbedingungen in seinem Unternehmen.

Bei den Mitarbeiterinnen des Frauenbüros der Technischen Universität München, insbesondere bei Frau Dr. Ute Lill, möchte ich mich für die Gewährung eines Stipendiums aus dem Programm: „Chancengleichheit von Frauen in Forschung und Lehre“ bedanken. Dieses Stipendium, und die damit verbundenen regelmäßigen Treffen mit anderen Stipendiatinnen, haben mich stark motiviert und mich in meiner Arbeit beflügelt.

Meinem Mann Matthias und meinen beiden Töchtern Anne und Lena danke ich von ganzem Herzen für ihre Geduld, die sie in Zeiten der Vielfachbelastung mit mir aufbringen mussten. Besonders meine Kinder haben mir viel Kraft gegeben, mich mit ihrer Fröhlichkeit immer angesteckt und mich viele Probleme wieder in der richtigen Relation sehen lassen.

Sehr geholfen haben mir auch meine Freundinnen. Vor allem Gabi Abs, Susanne Wolf, Samar Perez, Dr. Julia Gockel und Regina Dünser möchte ich für die fröhlichen Stunden danken, bei denen ich nicht über Optimierungsprobleme gegrübelt habe.

Ohne die ausdauernde und unermüdliche Unterstützung meiner Eltern Monika und Peter Hirche und meiner Schwester Kerstin Langer, wäre die Umsetzung meiner Promotion jedoch kaum möglich gewesen. Neben ihrer ganz praktischen Hilfe haben sie mich immer gestärkt und mir Selbstvertrauen vermittelt. Besonders bei ihnen möchte ich mich daher von ganzem Herzen bedanken und ihnen diese Arbeit widmen.

Für meine Eltern und meine Schwester.

Erklärung

Ich versichere, daß ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet.

Weitere Personen waren an der inhaltlich-materiellen Erstellung der vorliegenden Arbeit nicht beteiligt. Insbesondere habe ich hierfür nicht die entgeltliche Hilfe von Vermittlungs- bzw. Beratungsdiensten (Promotionsberater oder anderer Personen) in Anspruch genommen. Niemand hat von mir unmittelbar oder mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen.

Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form einer Prüfungsbehörde vorgelegt.

Ich bin darauf hingewiesen worden, daß die Unrichtigkeit der vorstehenden Erklärung als Täuschungsversuch angesehen wird und den erfolglosen Abbruch des Promotionsverfahrens zu Folge hat.

Freising, den 21.10.2003 Dany Meyer

Abstract

Today, tasks of optimization are not excluded from any part of the modern engineering. Ever more frequently engineers must cope with complex optimization problems with conflicting goals as well as a large number of various constraints. Additionally, there is the claim to incorporate human expert knowledge in an easy and transparent manner within the solutions. Often the dependencies within the parameters of the process which is to be optimized are mathematically no more or only very imperfectly formulatable, are present in the form of rule knowledge, or example situations and must therefore be estimated by suitable models. These requirements present complex challenges for the developers of modern concepts and optimization methods, which are in most cases no longer solvable with methods of classical mathematics alone. Rather, they require the additional employment of adaptive methods, which lead by using synergies between the classical and nature-analog procedures for the development of efficient hybrid systems.

In this work a multitier system for model-based, multi-objective optimization is presented, which consists of the data driven process modeling for calculation of objectives and constraints, their multi-objective optimization as well as an interactive Decision Making System. The uniqueness of the presented approach is the development of modeling and interpolation of the generated pareto optimal solutions and their corresponding objectives after the optimization by neural networks. In this way the approach allows to perform an interpolation access within the pareto set as well as the extraction of knowledge between the process variables near the pareto set and pareto front. Besides the representation of a practical-suited methodology, extensions in the theory of evolutionary algorithms in the form of learning the evolution direction during the optimization represents a further emphasis. The additional combination with a gradient-based optimization algorithm makes the approach a multi-hybrid system, which is characterized by very good convergence characteristics and a high quality of the generated solutions.

As an example of the industrial application of the presented approach, a system for model-based, multi-objective recipe optimization in the animal fodder industry is described. The aim of the work is the development of an adaptive, multi-hybrid multi-objective evolutionary algorithm which exhibits its superiority by efficiently using synergies between different natur-analog and mathematical methods as well as the presentation of a practical methodology for engineers to optimize the production processes. This includes a more efficient, powerful design of experiments, process modeling and multi-objective optimization.

Kurzfassung

Heute sind Aufgaben der Optimierung aus keinem Bereich der modernen Technik mehr wegzudenken. Dabei zeigt sich immer häufiger, daß es sich um komplexe Optimierungsprobleme handelt, die zum einen sich widersprechende Ziele und zum anderen eine große Anzahl von unterschiedlichen Randbedingungen enthalten. Zusätzlich besteht der Anspruch, menschliches Expertenwissen in die Lösung dieser Probleme unkompliziert und transparent einzubringen und für die Problemlösung verwendbar zu machen. Oft sind die Abhängigkeiten innerhalb der zu optimierenden Prozesse mathematisch nicht mehr oder nur sehr unvollkommen formulierbar, liegen in Form von Regelwissen oder Beispielsituationen vor und müssen daher mit Hilfe von geeigneten Modellen geschätzt werden. Diese Anforderungen stellen die Entwickler von modernen Modellbildungs- und Optimierungsmethoden vor große Herausforderungen, die in den meisten Fällen nicht mehr allein mit Methoden der klassischen Mathematik lösbar sind. Vielmehr erfordern sie den zusätzlichen Einsatz lernfähiger Methoden, die durch das Ausnutzen von Synergien zwischen den klassischen und natur-analogen Verfahren zur Entwicklung leistungsfähiger hybrider Systeme führen.

In dieser Arbeit wird ein mehrstufiges Verfahren zur modellbasierten Mehrzieloptimierung vorgestellt, das sich aus der datengetriebenen Prozessmodellbildung zur Berechnung der Zielfunktionen und Randbedingungen, ihrer multikriteriellen Optimierung sowie einem interaktiven Decision-Making-Modul zusammensetzt. Die Besonderheit des hier entwickelten Ansatzes besteht darin, daß die bei der Optimierung generierte, näherungsweise pareto-optimale Lösungsmenge nach Abschluß der Optimierung durch Neuronale Netze modelliert wird und so einen interpolierenden Zugriff auf ihre Elemente sowie die Extraktion von Wissen über die Zusammenhänge zwischen den Prozessgrößen im näherungsweise pareto-optimalen Bereich gestatten. Neben der Darstellung einer praxistauglichen Gesamtmethodik stellen Erweiterungen im Bereich der Theorie Evolutionärer Algorithmen in Form des Lernens der Evolutionsrichtung während der Optimierung einen weiteren Schwerpunkt dar. Die zusätzliche Kombination mit einem gradientenbasierten Optimierungsalgorithmus machen den Ansatz zu einem Multi-Hybrid-System, das sich durch sehr gute Konvergenzeigenschaften und eine hohe Qualität der generierten Lösungen auszeichnet.

Beispielhaft wird eine mit diesem Ansatz entwickelte und im industriellen Einsatz befindliche Applikation zur modellbasierten Rezepturoptimierung in der Tierfutterindustrie beschrieben.

Ziel der Arbeit ist es, einen lernfähigen, multi-hybriden multikriteriellen Evolutionären Algorithmus zu entwickeln, der seine Überlegenheit durch das effiziente Ausnutzen von Synergien zwischen den einzelnen Verfahren zeigt, sowie eine praxisbezogene Methodik zu erarbeiten, die es dem Ingenieur gestattet, rezepturgesteuerte Produktionsprozesse – angefangen von der Datenerfassung und Versuchsplanung, über die Prozessmodellierung bis hin zur multikriteriellen Optimierung – effizienter zu gestalten.

Inhaltsverzeichnis

1 Einleitung

Eine wettbewerbsentscheidende Aufgabe moderner Fertigungsunternehmen besteht heute in der optimalen Planung und Organisation der zugrundeliegenden Produktionsprozesse.

Um Produkte definierter Qualität mit möglichst geringem Kostenaufwand herzustellen, ist eine betriebswirtschaftlich optimale Planung aller Teilprozesse des Produktionsvorganges gefordert. Durch den zunehmenden Grad der Automatisierung und der damit verbundenen Flut von Daten wird deren ganzheitliche Modellierung jedoch immer komplexer. Eine „gläserne“ Fabrik, in der alle Teilprozesse und deren Interaktionen exakt analytisch beschreib- und mathematisch modellierbar sind, ist daher unter den heutigen dynamischen technischen und betriebswirtschaftlichen Verzahnungen nicht realisierbar. Aus diesem Grunde werden immer mehr die einzelnen, miteinander interagierenden Teilprozesse in sich modelliert und optimiert. So soll es durch die Zerlegung der Problematik in definierte Teilaufgaben möglich werden, problemspezifisch die jeweils beste Lösungsstrategie zu wählen.

Eine dieser Teilaufgaben besteht z.B. bei der rezepturgesteuerten Produktion in der Modellierung der Abhängigkeiten zwischen den einzelnen Rohmaterialien, Prozessgrößen und der Produktqualität sowie deren Optimierung mit der Zielsetzung maximale Qualität der Produkte bei minimalen Herstellungskosten zu erreichen. Dabei besteht eine Schwierigkeit in der Erstellung eines geeigneten Prozessmodells, d.h. der Beschreibung der i.d.R. nichtlinearen Abhängigkeiten zwischen der großen Anzahl der an der Rezeptur beteiligten Rohmaterialien, Prozessparametern und Qualitätsgrößen. Häufig sind diese Abhängigkeiten so komplex, daß ihre vollständige mathematische Formulierung, z.B. in Form eines Systems aus Differentialgleichungen oder logischen Regeln nicht möglich ist. Mit Hilfe des Prozessmodells soll eine Vorhersage der realen Reaktion des Prozesses auf Veränderungen in der Zusammensetzung der Rohmaterialien oder der Prozessparameter gemacht werden. Diese Modellvorhersagen bilden die Grundlage der modellbasierten Optimierung der Rezeptur, bei der es um das Auffinden von Kombinationen von Rohmaterialien und Prozessparametern geht, die zum aktuellen Zeitpunkt die geringsten Kosten verursachen und eine Menge von Qualitätskriterien maximieren, die sich i.d.R. widersprechen.

Die Optimierung realer Produktionsprozesse gestaltet sich daher nicht nur wegen der hohen Prozesskomplexität als schwierig, sondern auch aufgrund mehrerer, sich u.U. widersprechender Zielsetzungen sowie der großen Anzahl der dabei zu berücksichtigenden Randbedingungen und ist mit klassischen mathematischen Methoden nicht mehr zu leisten. Es wurde daher nach alternativen Lösungsstrategien gesucht, die es gestatten, auch schwer modellierbare Prozesse hoher Komplexität zu modellieren und zu optimieren.

Die Natur als Lehrmeister zeigt, daß mit Prinzipien wie Selbstorganisation und Lernfähigkeit sowie Zufall und Konkurrenz robuste Lösungen im Bereich Modellbildung und Optimierung geschaffen werden können. Diese Prinzipien zu erkennen und für praktische Aufgaben verwendbar zu machen, ist Aufgabe der Computational Intelligence (CI), die natur-analoge Methoden der Informationsverarbeitung wie z.B. Künstliche Neuronale Netze (NN) und Evolutionäre Algorithmen (EA) zusammenfaßt.

1.1 Zielsetzung

NN und EA bieten als adaptive und lernfähige Methoden die Möglichkeit, Zusammenhänge zwischen den Prozessgrößen aufgrund von Beispielsituationen zu lernen und eine Optimierung auch in hochkomplexen und nichtlinearen Suchräumen durchzuführen. In der vorliegenden Arbeit sollen ihre Prinzipien industriell angewandt und ein System zur modellbasierten Mehrzieloptimierung (rezepturgesteuerter) Produktionsprozesse unter Berücksichtigung harter

und weicher Randbedingungen zur simultanen Minimierung mehrerer Zielfunktionen vorgestellt werden. Es stützt sich insbesondere auf die Modellierung von Zielfunktionen und -Konflikten mit Hilfe Neuronaler Netze sowie einem Optimierungskern in Form einer speziell entwickelten Evolutionsstrategie zur Mehrzieloptimierung.

Folgende Anforderungen sollen durch das zu entwickelnde Verfahren zur Lösung multikriterieller Optimierungsprobleme erfüllt werden, die sich aus der praktischen Arbeit bei der Modellierung und Optimierung höherdimensionaler Prozesse ergeben haben:

- Es ist ein robustes Verfahren zur Lösung kontinuierlicher, multikriterieller Optimierungsprobleme zu erarbeiten bei dem die zu minimierenden Zielfunktionen und Randbedingungen nicht nur in Form mathematischer Zusammenhänge sondern auch als Menge von Beispielsituationen des Prozesses vorliegen können.
- Der Optimierungsalgorithmus soll dabei ohne eine a-priori definierte exakte Gewichtung der zu minimierenden Zielfunktionen eine Menge von Kompromißlösungen generieren, aus der der Anwender nach Abschluß der Optimierung interaktiv die für ihn „beste“ Lösung auswählen kann. Es soll dabei eine Verarbeitung linearer und nichtlineare Zielfunktionen sowie linearer und nichtlinearer Restriktionen möglich sein, wobei die Einhaltung der Randbedingungen eine höhere Priorität als die Minimierung der Zielfunktionen besitzt.
- Die Richtung der Optimierung soll durch Mechanismen der Selbstadaptation gezielt in Richtung pareto-optimalen Bereich beeinflußt werden. Dabei sollen Synergien genutzt werden, die sich durch die Kombination von Methoden der CI und klassischen, gradientenbasierten Optimierungsmethoden ergeben.
- Nach Abschluß der Optimierung soll es möglich sein, Wissen über die Zusammenhänge zwischen den Prozessgrößen im Bereich der Kompromißlösungen zu extrahieren.

Mit Hilfe des hier entwickelten Ansatzes soll es möglich sein, eine modellbasierte Mehrzieloptimierung rezepturgesteuerter Produktionsprozesse durchzuführen, um eine Kostenminimierung bei maximaler Produktqualität zu erreichen und die Entwicklungszeit für neue verbesserte Rezepturen mit gewünschten Produkteigenschaften zu verkürzen.

1.2 Aufbau der Arbeit

Im folgenden wird ein kurzer Überblick über die Gliederung dieser Arbeit gegeben:

Kapitel 2: Hier werden zunächst die grundlegenden Begriffe aus dem Bereich der multikriteriellen Optimierung sowie das Konzept der Pareto-Optimalität geklärt.

Kapitel 3: In diesem Kapitel werden die in der Arbeit verwendeten Netzwerkarchitekturen und Lernalgorithmen Neuronaler Netze zur Modellbildung und Datenanalyse gegeben.

Kapitel 4: Die Grundlagen Evolutionärer Algorithmen werden in diesem Kapitel beschrieben, wobei insbesondere auf die Besonderheiten von Evolutionsstrategien eingegangen wird.

Kapitel 5: An dieser Stelle wird auf Evolutionäre Algorithmen zur Mehrzieloptimierung eingegangen, wobei zunächst die wesentlichen Konzepte und im Anschluß daran die wichtigsten Vertreter ihrer Realisierungen vorgestellt werden.

Kapitel 6: Hier wird das in dieser Arbeit entwickelte Verfahren zur modellbasierten Mehrzieloptimierung vorgestellt. Es wird detailliert in den Stufen Modellbildung, Mehrzieloptimierung und Decision Making beschrieben, sowie die Abgrenzung des entwickelten Verfahrens von anderen Ansätzen dargestellt.

Kapitel 7: In diesem Kapitel sind die durchgeführten ausführlichen Simulationen zum Benchmarking des Verfahrens dargestellt. Eine kritische Diskussion der Leistungsfähigkeit schließt sich an.

Kapitel 8: Schließlich werden in diesem Kapitel zwei Anwendungen des Verfahrens beschrieben. Es handelt sich dabei um die Darstellung der modellgestützten Optimierung der Konstruktion eines T-Trägers und eines im industriellen Einsatz befindlichen Systems zur modellbasierten Mehrzieloptimierung von Rezepturen in der Tierfutterherstellung.

Kapitel 9: In den Schlußbetrachtungen werden die mit dem hier entwickelten Verfahren erzielten Ergebnisse zusammengefaßt. Ein Ausblick auf weiterführende Entwicklungen schließt die Arbeit ab.

Im Anhang werden schließlich die verwendeten Symbole und Abkürzungen aufgeführt und eine Übersicht über einige, das Thema betreffende Informationsquellen aus dem Internet gegeben.

2 Formale Problembeschreibung und Definitionen

2.1 Multikriterielle Optimierungsprobleme

Die in Kapitel 1 beschriebenen Probleme sind Optimierungsprobleme mit mehrfacher Zielsetzung, d.h. multikriterielle Optimierungsprobleme (MOP: multi-objective problem).

Im Folgenden soll von Minimierungsproblemen ausgegangen werden, da jedes Maximierungsproblem durch einfache Konvertierung [1] in ein Minimierungsproblem umgeformt werden kann.

$$\arg\min_x f(x) = \arg\max_x \left(-f(x)\right) \qquad [1]$$

In dieser Arbeit werden statische Optimierungsprobleme betrachtet, d.h. Optimierungsprobleme, bei denen die Lage des Optimums zeitinvariant ist. Methoden zur Lösung statischer Optimierungsprobleme sind jedoch auch für dynamische Optimierungsprobleme relevant, da die Lösungsstrategie für solche Probleme i.d.R. darin besteht, den betrachteten Zeitbereich in Abschnitte aufzuteilen und in jedem der so entstandenen Zeitfenster wiederum eine statische Optimierung für die veränderten Bedingungen durchzuführen. Auf diese Weise wird ein Regelkreis gebildet, wobei die veränderlichen Parameter des Optimierungsproblems die Stellgrößen und das Qualitätskriterium die Regelgröße sei. Die Genauigkeit des Reglers hängt vom Zeitraster ab. Diese Vorgehensweise ist jedoch nur dann sinnvoll, wenn die Lage des Optimums nur durch den aktuellen Systemzustand bestimmt wird.

Im Folgenden werden einige grundlegende Definitionen bezüglich der Lösung multikriterieller Optimierungsprobleme getroffen:

Definition 1: MOP

Ein multikriterielles Optimierungsproblem[1] *MOP* wird definiert durch

- den Lösungs- oder Entscheidungsraum (*decision space*) X mit n Entscheidungsvariablen und den Vektoren $\vec{x} = [x_1, x_2, ..., x_n] \in X$,
- den k-dimensionalen Zielraum Y (*objective space*) der Vektoren $\vec{y} = [y_1, y_2 .., y_k] \in Y$,
- den Vektor $\vec{f} = [f_1(\vec{x}), f_2(\vec{x}), ..., f_k(\vec{x})]$ mit k Zielfunktionen $f_i(\vec{x})$, die Funktionen der Entscheidungsraumvektoren $\vec{x} \in X$ sind und X auf Y abbilden, sowie
- den Vektoren $\vec{g} = [g_1, g_2, ..., g_m] \in G$ und $\vec{h} = (h_1, h_2, ..., h_l) \in H$ für m bzw. l Randbedingungen (*constraints*) im Raum der Vektoren der Randbedingungsfunktionen (*constraint space*) G und H, mit $g_i = g_i(\vec{x})$ und $h_i = h_i(\vec{x})$ als Funktionen von $\vec{x}$.[2]

[1] Definitionen und Notationen für multikriterielle Optimierungsprobleme finden sich in: [Hwang und Masud, 1979], [Steuer, 1986] sowie [Fonseca und Fleming, 1993] und [Veldhuizen und Lamont, 2000].

[2] Bezüglich der Randbedingungen soll zwischen weichen (soft constraints) und harten (hard constraints) Randbedingungen unterschieden werden. Während ein MOP nur genau dann eine gültige Lösung besitzt, wenn alle harten Randbedingungen eingehalten werden, können weiche Randbedingungen in verschiedenen Ausprägungen erfüllt sein. Häufig wird der Grad ihrer Einhaltung als zusätzliche Zielwertfunktion des MOP zusammengefaßt (siehe Abschnitt 4.1.5).

Ohne Beschränkung der Allgemeinheit sei das generelle Optimierungsziel eines MOP´s die *gleichzeitige* Minimierung aller k Komponenten $f_i = f_i(\vec{x})$ des Vektors der Zielfunktionen $\vec{f}(\vec{x})$ unter Berücksichtigung der Randbedingungen $\vec{g}(\vec{x}) \geq \vec{0}$ und $\vec{h}(\vec{x}) = \vec{0}$. Die Kurzschreibweise eines MOP´s sei hier:

$$\begin{aligned} \textit{minimiere:} \quad & f_i(\vec{x}) \ \textit{für} \ i = 1,..,k \\ \textit{unter Berücksichtigung von:} \quad & g_i(\vec{x}) \geq 0 \ \textit{mit} \ i = 1,..,m \\ \textit{und:} \quad & h_i(\vec{x}) = 0 \ \textit{mit} \ i = 1,..,o \end{aligned} \qquad [2]$$

Definition 2: Menge der zulässigen Lösungen (feasible set)

Die Menge der zulässigen Lösungen eines MOP´s $X_{feasible} \subseteq X$ (*feasible set*) im nicht leeren Entscheidungsraum ist definiert als die Menge der Vektoren der Entscheidungsvariablen $\vec{x}$, die den Randbedingungen $\vec{g}(\vec{x}) \geq \vec{0}$ und $\vec{h}(\vec{x}) = \vec{0}$ genügen.

$$X_{feasible} = \left\{ \vec{x} \in X \mid \vec{g}(\vec{x}) \geq \vec{0} \wedge \vec{h}(\vec{x}) = \vec{0} \right\} \qquad [3]$$

Definition 3: Menge der zulässigen Zielfunktionswerte (feasible region)

Der Vektor $\vec{f}$ der Zielfunktionen bestimmt eine Abbildung von $X_{feasible}$ in den Zielraum, das Bild von $X_{feasible}$ ist die Menge der zulässigen Zielfunktionswerte $Y_{feasible}$ (*feasible region*):

$$Y_{feasible} = \bigcup_{\vec{x} \in X_{feasible}} \left\{ \vec{f}(\vec{x}) \right\} \qquad [4]$$

In Abbildung 2.1 werden oben definierten Begriffe für ein multikriterielles Optimierungsproblem mit $n = 2$ Entscheidungsvariablen und $k = 2$ Zielfunktionen illustriert.

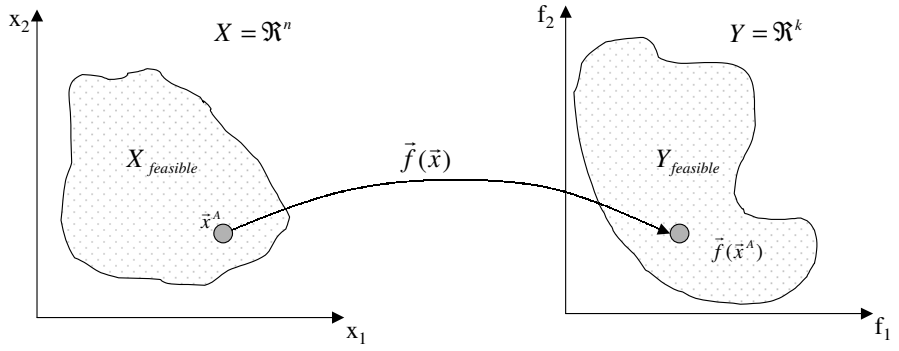


Abbildung 2.1: Definition des MOP als Abbildung des Entscheidungs- auf den Zielraum

Definition 4: Idealvektor

Es sei $f_i^* = \min\left\{ f_i(\vec{x}) \mid \vec{x} \in X_{feasible} \right\}$ der Minimalwert der Zielfunktion $f_i(\vec{x})$. Dann heißt $\vec{y}^{ideal} := \left[f_1^*, f_2^*, .., f_k^* \right]$ Idealvektor eines MOP´s.

Definition 5: Ideallösung

Im folgenden wird davon ausgegangen, daß jedes f_i^* eindeutig durch ein $\vec{x}^{*i} := \arg\min\{f_i(\vec{x}) \mid \vec{x} \in X_{feasible}\}$ bestimmt wird, wobei die $\vec{x}^{*i}$ für jedes f_i^* verschieden sein können. Sind die $\vec{x}^{*i}$ jedoch für alle f_i^* identisch, wird $\vec{x}^{*i}$ als Ideallösung $\vec{x}^{ideal}$ bezeichnet und der Idealvektor durch $\vec{f}(\vec{x}^{ideal}) = \vec{y}^{ideal}$ erreichbar.

Definition 6: best-over-all-Lösung

Die best-over-all-Lösung eines MOP´s (siehe Abbildung 2.2) auf einer Lösungsmenge $X_{feasible}$, ist diejenige Lösung $\vec{x}^B \in X_{feasible}$, für die der euklidische Abstand zwischen $\vec{f}(\vec{x}^B)$ und dem Idealvektor $\vec{y}^{ideal}$ minimal ist.

$$\vec{x}^B := \arg \min \left\| \vec{f}(\vec{x}) - \vec{y}^{ideal} \right\| \; mit \; \vec{x} \in X_{feasible} \qquad [5]$$

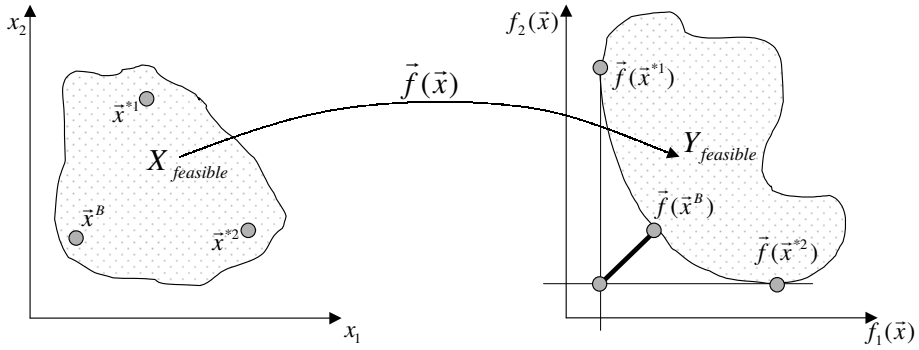


Abbildung 2.2: Definition best-over-all-Lösung eines MOP´s

Definition 7: Zielfunktionskonflikt

Ein Zielfunktionskonflikt liegt vor, wenn zwei Optimierungsziele $f_i(\vec{x})$ und $f_j(\vec{x})$ in Konflikt stehen, d.h. die Verbesserung des Zielfunktionswertes $y_i = f_i(\vec{x})$ nicht ohne die Verschlechterung des Zielfunktionswertes $y_j = f_j(\vec{x})$ möglich ist und umgekehrt. Dann ist es nicht möglich, eine Ideallösung zu finden, die die Zielfunktionen $f_i(\vec{x})$ und $f_j(\vec{x})$ gleichzeitig minimiert. Der Idealvektor ist dann nicht zu erreichen.

Das folgende Beispiel[3] (siehe Gleichung [6] und Abbildung 2.3) illustriert ein MOP mit Zielfunktionskonflikt:

[3] Entnommen aus [Fonseca und Fleming, 1995].

$$\textit{minimiere: } f_i(\vec{x}) \textit{ mit } i = 1,2$$

$$f_1(\vec{x}) = 1 - e^{-(x_1-1)^2-(x_2+1)^2}$$
$$f_2(\vec{x}) = 1 - e^{-(x_1+1)^2-(x_2-1)^2} \qquad \textit{mit} \quad x_i \in \Re;\ i = 1,2 \qquad [6]$$

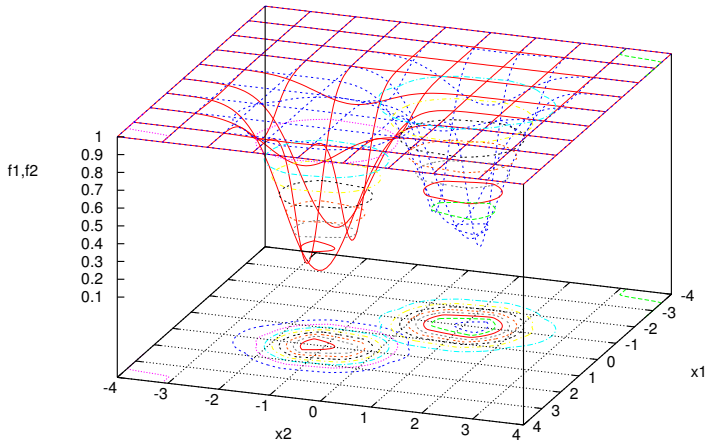


Abbildung 2.3: Visualisierung eines MOP´s mit Zielfunktionskonflikt

Für dieses Beispiel liegt ein Zielfunktionskonflikt vor. Der daher nicht erreichbare Idealvektor des Problems [6] ist: $\vec{y}^{ideal} = \left[f_1(\vec{x}^{*1}), f_2(\vec{x}^{*2})\right] = [0,0]$ mit $\vec{x}^{*1} = [1,-1]$ und $\vec{x}^{*2} = [-1,1]$.

Liegt ein MOP mit Zielfunktionskonflikt vor, kann seine Lösung nicht aus einem einzelnen Vektor der Entscheidungsvariablen bestehen, sondern muß sich aus einer *Menge* von Lösungen zusammensetzen. Entsprechend muß der Anwender daher auch unter Einbeziehung von Präferenzen aus einer Menge von sogenannten effizienten Vektoren mit unterschiedlicher Ausprägung der einzelnen Zielfunktionen, sogenannten „Trade-offs“, auswählen. Effizient heißt ein Vektor von Zielfunktionswerten dann, wenn er in keiner Dimension dem jeweiligen Gütekriterium schlechter, aber ihm in mindestens einer Dimension besser entspricht als alle übrigen Vektoren. Zur Beschreibung der Menge der Lösungen, die die Menge der „Trade-offs“ bzw. der effizienten Vektoren der Zielfunktionswerte bestimmt, wird das Konzept der Pareto-Optimalität verwendet. Es ist nach dem italienischen Gelehrten Vilfredo Pareto benannt, der schon zu Ende des 19ten Jahrhunderts Optimierungsaufgaben mit sich widersprechenden Zielen im Bereich der Wirtschaftswissenschaften untersuchte. Im Folgenden sollen einige Begriffe aus dem Bereich der Pareto-Optimalität formal eingeführt werden, um den Begriff der Lösung eines MOP´s zu definieren.

2.2 Das Konzept der Pareto-Optimalität

Die Lösung eines MOP´s liefert i.d.R. keine Einzel- oder Ideallösung, sondern eine Menge von Lösungen, die in den Raum der zulässigen Zielfunktionswerte $Y_{feasible}$ abgebildet werden. Die Elemente dieser Lösungsmenge werden als *pareto-optimal*, die entsprechenden Vektoren der Zielfunktionswerte als *effizient* bezeichnet. Der Begriff der „Pareto-Optimalität“ bezieht sich im Folgenden auf den Entscheidungsraum während der Begriff der „Effizienz“ auf den Zielraum sowie der Begriff der „Pareto-Dominanz“ sowohl auf Entscheidungs- als auch den Zielraum angewandt werden soll.

Definition 8: Pareto-Dominanz

Für zwei Vektoren von Zielfunktionswerten eines MOP´s $\vec{y}^1 = \left[y_1^1,..,y_k^1\right]$ und $\vec{y}^2 = \left[y_1^2,..,y_k^2\right]$ gelte:

Der Vektor $\vec{y}^1$ dominiert den Vektor $\vec{y}^2$ genau dann schwach (*weakly dominates*) und wird mit der Notation: $\vec{y}^1 \preceq \vec{y}^2$ bezeichnet, wenn gilt:

$$y_i^1 \le y_i^2 \mid \forall\ i \in \{1,..,k\} \qquad [7]$$

Dagegen dominiert $\vec{y}^1$ genau dann den Vektor $\vec{y}^2$ streng (*strongly dominates*) mit der Notation: $\vec{y}^1 \prec \vec{y}^2$, wenn $\vec{y}^1$ partiell echt kleiner als $\vec{y}^2$ ist, d.h.

$$y_i^1 \le y_i^2 \mid \forall\ i \in \{1,..,k\}\ \wedge\ y_j^1 < y_j^2 \mid \exists\ j \in \{1,..,k\} \qquad [8]$$

In diesem Falle wird $\vec{y}^1$ als effizient gegenüber $\vec{y}^2$ bezeichnet.

Die Vektoren $\vec{y}^1$ und $\vec{y}^2$ sind bezüglich der Pareto-Dominanz genau dann nicht vergleichbar (indifferent) und werden mit der Notation $\vec{y}^1 \sim \vec{y}^2$ bezeichnet, wenn gilt:

$$y_i^1 > y_i^2 \mid \exists\ i \in \{1,..,k\}\ \wedge\ y_j^1 < y_j^2 \mid \exists\ j \in \{1,..,k\} \qquad [9]$$

Analog sei für zwei Vektoren von Entscheidungsvariablen $\vec{x}^1 = \left[x_1^1,..,x_n^1\right]$ und $\vec{x}^2 = \left[x_1^2,..,x_n^2\right]$ definiert:

$\vec{x}^1 \prec \vec{x}^2$: $\vec{x}^1$ ist gegenüber $\vec{x}^2$ genau dann streng dominant, wenn: $\vec{f}(\vec{x}^1) \prec \vec{f}(\vec{x}^2)$

$\vec{x}^1 \preceq \vec{x}^2$: $\vec{x}^1$ ist gegenüber $\vec{x}^2$ genau dann schwach dominant, wenn: $\vec{f}(\vec{x}^1) \preceq \vec{f}(\vec{x}^2)$ [10]

$\vec{x}^1 \sim \vec{x}^2$: $\vec{x}^1$ ist gegenüber $\vec{x}^2$ genau dann indifferent, wenn: $\vec{f}(\vec{x}^1) \sim \vec{f}(\vec{x}^2)$

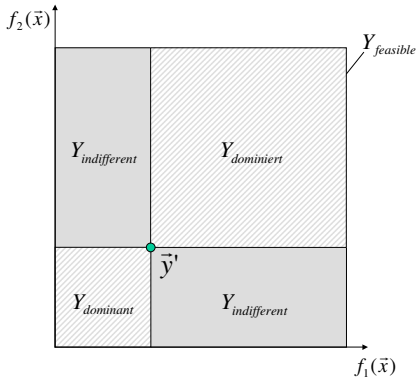


Abbildung 2.4: Pareto-Dominanz

Entsprechend der obigen Definitionen gilt in Abbildung 2.4:

- Alle Vektoren $\vec{y} \in Y_{dominiert}$ werden von $\vec{y}'$ streng dominiert, d.h. $\vec{y}'$ ist streng dominant gegenüber jedem $\vec{y}$ der Menge $Y_{dominiert}$: $\vec{y}' \prec \vec{y} \mid \forall\ \vec{y} \in Y_{dominiert}$.
- Alle Vektoren $\vec{y} \in Y_{dominant}$ dominieren $\vec{y}'$ streng, d.h. jedes $\vec{y}$ der Menge $Y_{dominant}$ ist streng dominant gegenüber $\vec{y}'$: $\vec{y} \prec \vec{y}' \mid \forall\ \vec{y} \in Y_{dominant}$.
- Alle Vektoren $\vec{y} \in Y_{indifferent}$ sind hinsichtlich der Pareto-Dominanz nicht vergleichbar mit $\vec{y}'$, d.h. $\vec{y} \sim \vec{y}' \mid \forall\ \vec{y} \in Y_{indifferent}$
- Die Lösung $\vec{x}'$, die $\vec{y}'$ bestimmt, ist hinsichtlich aller Lösungen $\vec{x}$, die alle $\vec{y} \in Y_{dominiert}$ bestimmen, streng dominant, d.h.: $\vec{x}' \prec \vec{x} \mid \forall\ \vec{x}\ mit\ \vec{y}' = \vec{f}(\vec{x}') \prec \vec{y} = \vec{f}(\vec{x}) \in Y_{dominiert}$.
- Alle Lösungen $\vec{x}$, die alle $\vec{y} \in Y_{dominant}$ bestimmen, sind bezüglich der Lösung $\vec{x}'$, die $\vec{y}'$ bestimmt, streng dominant, d.h.: $\vec{x} \prec \vec{x}' \mid \forall\ \vec{x}\ mit\ \vec{y} = \vec{f}(\vec{x}) \in Y_{dominant} \prec \vec{y}' = \vec{f}(\vec{x}')$.

Definition 9: nicht-dominierte Menge (non-dominated set)

Bezüglich einer Menge $A \subseteq X_{feasible}$ oder $A \subseteq Y_{feasible}$, ist die nicht-dominierte Menge (*non-dominated set*) $ND(A)$ diejenige Teilmenge von A deren Elemente von keinem Vektor aus A stark dominiert werden.

Definition 10: Pareto-Menge (pareto-optimal set)

Für ein definiertes MOP sei die Menge aller pareto-optimalen Lösungen P_{true} oder die Pareto-Menge (*pareto-optimal set)* definiert als:

$$P_{true} := \left\{ \vec{x} \in X_{feasible} \mid \neg\ \exists\ \vec{x}' \in X_{feasible} : \vec{f}(\vec{x}') \preceq \vec{f}(\vec{x}) \right\} \qquad [11]$$

Die Pareto-Menge P_{true} ist die nicht-dominierte Menge bezüglich $X_{feasible}$, d.h. $P_{true} = ND(X_{feasible})$, und wird auch als die globale Pareto-Menge bezeichnet.

Während sich der Begriff der Pareto-Menge auf den Raum der Entscheidungsvariablen bezieht, wird deren Abbildung in den Zielraum als Pareto-Front bezeichnet.

Definition 11: Pareto-Front

Für ein definiertes MOP mit $\vec{f}(\vec{x})$ und seine pareto-optimale Lösungsmenge P_{true} sei die globale Pareto-Front PF_{true} definiert als:

$$PF_{true} := \left\{ \vec{f}(\vec{x}) = \left[f_1(\vec{x}), ..., f_k(\vec{x}) \right] \mid \vec{x} \in P_{true} \right\} \qquad [12]$$

Nach Horn [Horn und Nafpliotis, 1993] ist die Pareto-Front eines MOP´s mit k Zielfunktionen eine beschränkte Hyperfläche von höchstens der Dimension $(k-1)$[4].

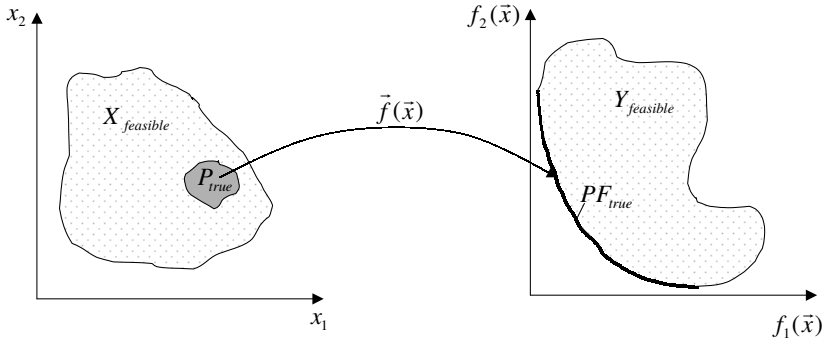


Abbildung 2.5: Pareto-Menge und Pareto-Front

Definition 12: Approximation der Pareto-Menge

Als Approximation (Näherung) P^* der globalen Pareto-Menge wird jede Menge nicht-dominierter, zulässiger Lösungen bezeichnet, wobei P^* nicht mit P_{true} übereinstimmen muß.

Definition 13: Approximation der Pareto-Front

Die durch P^* bestimmte Menge PF^* von nicht-dominierten, zulässigen Zielfunktionswerten wird als Approximation der Pareto-Front oder nicht-dominimierte Front bezeichnet, wobei auch hier PF^* nicht mit PF_{true} übereinstimmen muß.

Definition 14: Lösung eines MOP´s

Lösung $\vec{x}^{Lsg.}$ eines MOP´s ist jeder Vektor der Pareto-Menge, d.h $\vec{x}^{Lsg.} \in P_{true}$. Jedes $\vec{x}^{Lsg.}$ bestimmt einen Vektor der globalen Pareto-Front des MOP´s, d.h. $\vec{f}(\vec{x}^{Lsg.}) \in PF_{true}$.

Das Optimierungsziel eines MOP´s, d.h. die simultane Minimierung der k Zielfunktionen $f_i(\vec{x})$, kann also nur realisiert werden, indem eine Menge von pareto-optimalen Lösungen (Pareto-Menge) generiert wird, die eine Menge von effizienten Zielfunktionswertevektoren (Pareto-Front) bestimmen. Dabei soll die Abbildung der gefundenen Lösungen in den Zielraum die globale Pareto-Front PF_{true} des MOP´s möglichst gut approximieren. Zur Lösung eines MOP´s ist also ein Algorithmus zu konstruieren, der in der Lage ist, eine Approximation P^* der Pareto-Menge P_{true} zu generieren, die eine nicht-dominierte Front PF^* bestimmt, die sich so eng wie möglich

[4] Veldhiuzen und Lamont beweisen in [Veldhuizen und Lamont, 1999], daß die Pareto-Front eines MOP´s für $k = 2$ Zielfunktionen höchstens eine beschränkte Kurve und für $k \geq 3$ Zielfunktionen höchstens eine beschränkte Hyperfläche ist.

an PF_{true} anschmiegt. Liegen keine Präferenzen zur Eingrenzung des Zielraums vor, soll PF^* über Elemente verfügen, die eine möglichst hohe Variablität besitzen.

Besitzen alle $f_i(\vec{x})$ eines MOP´s jeweils genau ein Optimum, spricht man von einem *unimodalen* MOP. Liegen dagegen bei mindestens einem $f_i(\vec{x})$ mehrere lokale Optima vor und ist das globale Optimum zu ermitteln, handelt es sich um ein *multimodales* MOP.

Das folgende aus der Literatur bekannte Beispiel (Schaffers F2 aus [Fonseca und Fleming, 1995]) soll die bisherigen Begriffe der Pareto-Dominanz illustrieren: Es sei das folgendes MOP zu lösen, d.h. $f_1(x)$ und $f_2(x)$ simultan zu minimieren:

$$X = \Re, \quad \vec{f} = [f_1, f_2], \quad G = H = \varnothing$$
$$\textit{für: } f_1 = |x|^2, f_2 = |x-c|^2 \ \textit{ mit } \ 0 \neq c \in \Re \qquad [13]$$

Für die Zielfunktionen f_1 und f_2 liegt ein Zielfunktionskonflikt vor, der daher nicht erreichbare Idealvektor des Problems ist: $\vec{y}^{ideal} = [f_1(\vec{x}^{*1}), f_2(\vec{x}^{*2})] = [0,0]$ mit $\vec{x}^{*1} = 0$ und $\vec{x}^{*2} = c$. Für das MOP [13] ist die Pareto-Menge P_{true} durch den Ausdruck [14] und die globale Pareto-Front PF_{true} durch den Ausdruck [15] beschrieben.

$$P_{true} = \{x \in \Re \mid 0 \leq x \leq c\} \qquad [14]$$
$$PF_{true} = \{[f_1(x), f_2(x)] \mid x \in P_{true}\} \qquad [15]$$

Da f_1 und f_2 nur Funktionen von x sind, kann die funktionale Abhängigkeit zwischen $f_1(x)$ und $f_2(x)$ durch den Ausdruck [16] beschrieben werden.

$$f_2(x) = |x-c|^2 = x^2 - 2cx + c^2 = f_1(x) - 2c\sqrt{f_1(x)} + c^2$$
$$\textit{speziell für } c = 2 \textit{ gilt: } f_2(x) = f_1(x) - 4\sqrt{f_1(x)} + 4 \qquad [16]$$

Abbildung 2.6 (links) veranschaulicht für $c = 2$ die funktionale Abhängigkeit von $f_1(x)$ und $f_2(x)$. In Abbildung 2.6 (rechts) ist $Y_{feasible}$ als eine Menge von Paaren $[f_1(x), f_2(x)]$ und PF_{true} für $c = 2$ dargestellt.

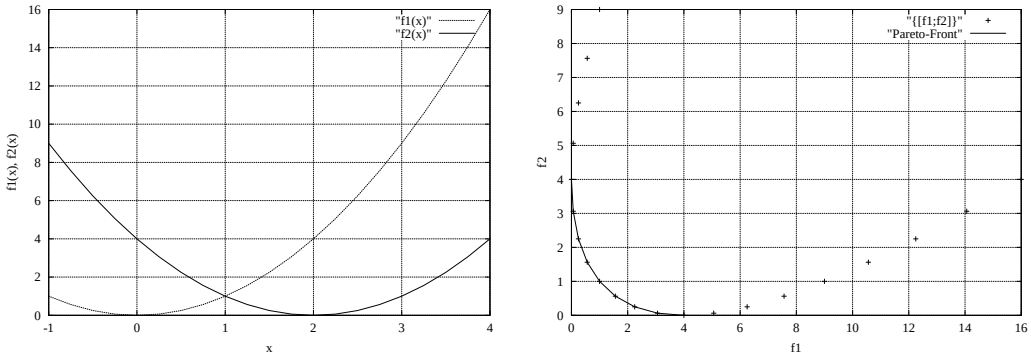


Abbildung 2.6: $f_1(x)$, $f_2(x)$ **(links) und Pareto-Front (rechts) für [13] bei** $c = 2$

3 Datengetriebene Modellierung mittels Neuronaler Netze

3.1 Grundlagen

Viele Leistungen die von biologischen Gehirnen zur Anpassung der Individuen an ihre natürliche Umwelt mühelos erbracht werden, sind auch für technische Systeme gefordert, jedoch mit klassischen mathematischen Verfahren in der notwendigen Komplexität nicht zu erreichen. Gerade in den Bereichen Assoziation, Klassifikation und Modellierung ist die Fähigkeit von biologischen Gehirnen beeindruckend, anhand von Beispielen selbständig zu lernen, d.h. ihre Struktur so zu adaptieren, daß die Individuen bei ähnlichen Situationen angemessen agieren können. Diese große Leistungsfähigkeit[5] wird dabei durch die massiv parallelisierte Signalverarbeitung durch einfache, miteinander kommunizierende Basiselemente erreicht. Seit Mitte des 20-ten Jahrhunderts[6] wird versucht, die durch die moderne Hirnforschung in biologischen Gehirnen erkannten Organisationsprinzipien auszunutzen, um künstliche Neuronale Netze (NN) zu konstruieren, die eine Informationsverarbeitung nach dem Vorbild der Natur anstreben. Ziel ist es dabei, die wichtigsten Eigenschaften biologischer Gehirne zu simulieren, um adaptive, lernende Algorithmen zu entwickeln, die für alternativ nicht lösbare technische Aufgabenstellungen z.B. im Bereich der Mustererkennung, Datenanalyse oder Modellierung einsetzbar sind.

Der prinzipielle Aufbau eines künstlichen Neuronalen Netzes ist biologisch motiviert: es besteht aus einer Vielzahl unabhängiger, einfacher Prozessoren (Neurone), die i.d.R. in Schichten angeordnet, über gewichtete Verbindungen (Synapsen) in einer gewissen Nachbarschaft jeweils miteinander verschaltet sind und Information (Erregung) austauschen. Dabei wird durch die Stärke der Kopplung die Reaktion des Neurons auf eingehende Signale moduliert. Durch Lernregeln können diese synaptischen Gewichte verändert werden und auf diese Weise eine Adaptation des Netzes auf die eintreffenden Signale erreicht werden. In Abhängigkeit von der Anordnung, Verschaltung und Aktivierung der Neurone wird zwischen verschiedenen Netzwerktypen unterschieden, die auch für verschiedene Problemklassen einsetzbar sind. Auf eine ausführliche Einführung in die Theorie neuronaler Netze soll an dieser Stelle verzichtet und statt dessen auf die Literatur (z.B. [Rojas, 1993] und [Zell, 1994]) verwiesen werden.

Im folgenden werden diejenigen Netzwerkarchitekturen und Lernregeln kurz vorgestellt, die in der vorliegenden Arbeit eingesetzt werden. Es handelt sich dabei NN die durch unüberwachtes

[5] Die minimale menschliche Reaktionszeit z.B. zur Betätigung einer Taste nach dem Ertönen eines Klingeltones oder die Reaktion auf ein unerwartetes Hindernis auf der Autobahn beträgt jeweils nur einige hundert Millisekunden – obwohl beide Aufgabenstellungen in ihrer Komplexität und der zu verarbeitenden Signalfülle sehr unterschiedlich sind – und sind nur durch die massiv parallele Signalverarbeitung durch die Milliarden von miteinander kommunizierender Nervenzellen möglich [Goos, 1998].

[6] Die Anfänge wurden dabei von dem Neurobiologen W.S. McCulloch und dem Statistiker W. Pitts mit der Entwicklung des ersten formalen Modells eines Neurons [McCulloch und Pitts, 1943] gelegt. Nach der Veröffentlichung der Erkenntnisse von Minsky und Papert [Minsky und Papert, 1969], daß mit einschichtigen NN keine nicht separierbaren Probleme (wie z.B. das XOR-Problem) gelöst werden können, entwickelte sich die Theorie der NN nur sehr langsam weiter und erlebte erst mit der Entwicklung von Lernverfahren für mehrschichtige NN, die das Problem der Nichtseparierbarkeit lösten, durch Parker [Parker, 1985] und Rummelhart et al [Rumelhart und McClelland, 1986] einen neuen Aufschwung. Heute hat sich die Methode NN aufgrund der Entwicklung einer Vielzahl von Netzwerkstrukturen und effizienten Lernalgorithmen zu einer robusten, praxistauglichen Methode etabliert und wird in vielen industriellen Anwendungen genutzt.

Lernen eine Datenanalyse des Signalraumes (Abschnitt 3.2.1) und durch überwachtes Lernen die Interpolation seiner Vektoren und so seine Modellierung (Abschnitt 3.2.2) gestatten.

3.2 Verwendete Netzwerkstrukturen und Lernalgorithmen

3.2.1 Datenanalyse mit „Self-Organizing-Maps“ (SOM) und „Learning-Vector-Quantization“ (LVQ)

T. Kohonen entwickelte 1982 [Kohonen, 1982] sogenannte topologie-erhaltende oder selbstorganisierende Merkmalskarten (SOM), um zu simulieren, wie in biologischen Gehirnen die Zellen einer Neuronenschicht sich in ihrer Erregbarkeit so aufeinander abstimmen, daß sie auf Eingabesignale in einer gesetzmäßigen Weise reagieren, die von ihrem Ort abhängt [Goos, 1998]. SOM sollten erklären helfen, wie biologische Gehirne sich organisieren um Signalähnlichkeiten in Lagenachbarschaften erregter Neurone umzusetzen und auf diese Weise topologie-erhaltende Karten der Eingabesignale erzeugen. Die Herausbildung solcher Karten ist ein wichtiges biologisches Prinzip bei der Selbstorganisation von Nervenzellen: beispielsweise erregen auf der Hautoberfläche benachbarte Tastrezeptoren des Menschen auch benachbarte Neurone im somatosensorischen Rindenfeld und stellen so eine Projektion der Körperoberfläche dar [Kaas et al, 1983]. Ein bekanntes Beispiel für die Herausbildung topologie-erhaltender Karten wird in [Ritter et al, 1990] dargestellt. Dort wird die Entstehung von neuronalen „Frequenz-Karten“ im auditiven Kortex der Fledermaus, aufgrund der von ihr ausgesandten und empfangenen Ultraschall-Echolotsignale beschrieben, die der Fledermaus eine extrem genaue Orientierung im Raum ermöglichen.

Das von Kohonen entwickelte und weit verbreitete SOM-Modell arbeitet auf Basis des Wettbewerbslernens und besteht aus nur zwei Neuronen-Schichten: einer Eingabe- und einer i.d.R. zweidimensionalen Ausgabe- oder Wettbewerbsschicht, deren n_{SOM} Neurone in einer zweidimensionalen Topologie[7] miteinander vernetzt sind und die topologie-erhaltende Karte repräsentieren (Abbildung 3.1). Jedes Neuron i der Wettbewerbsschicht ist durch einen n-dimensionalen Gewichtsvektor $\vec{w}_i$ mit jedem der n Neurone der Eingabeschicht verbunden. Beim unüberwachten Lernen einer Menge von Trainingsvektoren $\{\vec{v}\}$, wird für jeden Eingabevektor $\vec{v}$ dasjenige Neuron i' (Winner) auf der Wettbewerbsschicht bestimmt, das aufgrund seines Gewichtsvektors die maximale Aktivität besitzt, also [17] erfüllt.

$$\left\|\vec{v}-\vec{w}_{i'}\right\| = \min_{i}^{n_{SOM}}\left\{\left\|\vec{v}-\vec{w}_i\right\|\right\} \qquad [17]$$

Anschließend werden durch einen Adaptationsschritt die Gewichtsvektoren des Winner-Neurons i' und aller der Neurone i, die sich in der durch $h_{ii'}$ definierten Umgebung zu ihm befinden in Richtung $\vec{v}$ verschoben [18].

$$\vec{w}_i^{neu} = \vec{w}_i^{alt} + \eta \cdot h_{ii'}\left(\vec{v}-\vec{w}_i^{alt}\right) \qquad [18]$$

Die vom Zentrum i' abnehmende Umfelderregung wird durch die Nachbarschaftsfunktion $h_{ii'}$ beschrieben, die i.d.R. die Form einer Gausschen Glockenkurve besitzt [23].

[7] Die Neuronen der Wettbewerbsschicht sind i.d.R. in Form eines zweidimensionalen, rechteckigen Gitters angeordnet. Diese Topologie ist jedoch nicht zwingend; auch Gitter in Form eines Sechsecks, Dreiecks oder Kreises können bei entsprechender Anpassung der Nachbarschaftsfunktion verwendet werden.

$$h_{ii'} = \exp\left(-\frac{(i-i')^2}{2\cdot\sigma^2}\right) \qquad [19]$$

Der Parameter σ bestimmt die Umgebung der Umfelderregung und wird ebenso wie die Lernrate η mit zunehmender Anzahl von Iterationsschritten verringert, was eine nur noch geringe Veränderungen der Gewichtsvektoren zum Ende der Iteration und damit eine sukzessive Verfeinerung der Konturen der Karte bewirkt [20].

$$\eta^t = \eta^{start}\cdot\left(\frac{\eta^{end}}{\eta^{start}}\right)^{\frac{t}{t_{\max}}} \qquad \sigma^t = \sigma^{start}\cdot\left(\frac{\sigma^{end}}{\sigma^{start}}\right)^{\frac{t}{t_{\max}}} \qquad [20]$$

Mit dem Modell von Kohonen werden Signalähnlichkeiten durch eine zweidimensionale Projektion in Lagenachbarschaften auf der topologie-erhaltenden Karte umgesetzt. Dabei ist jedes Neuron der Wettbewerbsschicht für die Erkennung einer Menge von Datensätzen gleicher Eigenschaften (Cluster) zuständig, wobei benachbarte Neurone ähnliche Cluster im Signalraum repräsentieren. SOM unterdrücken redundante Signale und setzen nur wichtige Information in ihre Struktur um. Sie sind daher zur Analyse hochdimensionaler Datensätze (Data-Mining) geeignet, indem sie die Identifikation der Cluster im Datenraum und durch Interpretation der Kopplungsmatrix die Extraktion der Zusammenhänge zwischen den Komponenten der Signalvektoren, die zur Herausbildung der Cluster führen, ermöglichen. Dazu wurden verschiedene Visualisierungstechniken entwickelt, die i.d.R. auf der schichtweisen Darstellung jeweils nur einer Dimension der Gewichtsvektoren basieren [Kohonen, 1995]. Häufig wird zur zweidimensionalen Visualisierung der n-dimensionalen Kopplungsmatrix auch die Sammon-Projektion [Sammon, 1969] verwendet, die eine verzerrte Darstellung des Neuronen-Gitters in Abhängigkeit der Werte der Komponenten der Gewichtsvektoren der einzelnen Neuronen realisiert. In dieser Arbeit wurde sowohl die Sammon-Projektion als auch eine in Abschnitt 6.5.1 näher dargestellte Hyper-Cube-Visualisierungstechnik eingesetzt, die die simultane Darstellung mehrerer Dimensionen der Kopplungsmatrix gestattet.

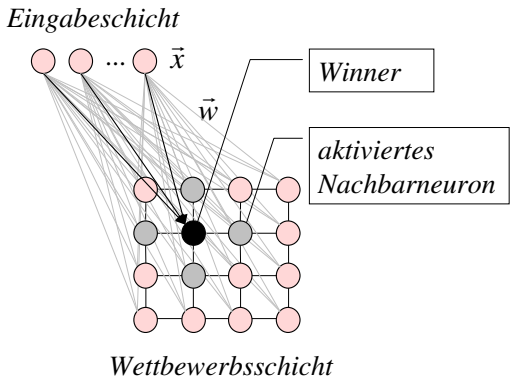


Abbildung 3.1: Struktur einer SOM

Eine den SOM eng verwandte Methode der Datenanalyse ist die ebenfalls von T. Kohonen vorgestellte Methode des „Learning-Vector-Quantization“ (LVQ) [Kohonen, 1995]. Sie dient der Clusteranalyse[8] eines n-dimensionalen Signalraumes und hat zum Ziel, ähnliche Signalvektoren zu identifizieren und einer geeigneten Klasse (Cluster) zuzuordnen sowie typische Repräsentanten dieser Klasse zu finden. Der LVQ-Algorithmus arbeitet auf einer

[8] Methoden der klassischen Clusteranalyse werden ausführlich in [Duran und Odell, 1974] und [Steinhausen und Langer, 1977] dargestellt.

Menge von n-dimensionalen kompetetiven Einheiten, die als Codebook-Vektoren bezeichnet und jeweils einem Cluster-Zentrum zugeordnet werden. Während des Trainings werden die Codebook-Vektoren so verschoben, daß sie möglichst gut den Schwerpunkt eines Clusters im Signalraum repräsentieren. Dies geschieht nach dem gleichen, wie oben für SOM dargestellten Prinzip des Wettbewerbslernen, d.h. der jeweilige Winner wird in Richtung Signalvektor in Abhängigkeit von seiner euklidischen Distanz zum Signalvektor und einer Lernrate verschoben. Jedoch erfolgt hier im Unterschied zu SOM keine Umfelderregung benachbarter Einheiten, d.h. keine Berücksichtigung einer Nachbarschaftsfunktion. Nach Abschluß des Trainings können die Signalvektoren jeweils einem der Codebook-Vektoren zugeordnet werden, wobei die Summe der Abstände der Signalvektoren zu ihren zugeordneten Codebook-Vektor durch die Adaptation minimiert wurde. Mit Hilfe des LVQ-Algorithmus ist es also möglich, eine bestimmte Anzahl von Vektoren derart zu positionieren, daß sie optimal die Schwerpunkte der Cluster im Signalraum repräsentieren, ohne etwaige Nachbarschaftsbeziehungen der Cluster zu berücksichtigen.

3.2.2 Modellbildung mittels „Radial-Basis-Function"-Netzwerken (RBF-Netze)

Ein funktionaler Zusammenhang zwischen Prozessparametern, für den eine definierte Anzahl von Stützstellen bekannt ist, kann durch eine Funktion genähert werden, indem diese Stützstellen durch eine geeignete Abbildungsvorschrift (Modell) interpoliert werden. Um lineare und nichtlineare Funktionen zu approximieren, die nur von einem Parameter abhängen, gibt es eine Reihe einfacher mathematischer Methoden. Hierzu zählen insbesondere die Approximationen mit den Interpolationsspolynomen von Lagrange oder Newton – ihre exakte Darstellung findet sich u.a. in [Bronstein und Semendjajew, 1978]. Ist die Funktion jedoch von mehreren Eingabeparametern abhängig, sind diese Ansätze i.d.R. nicht mehr einsetzbar. Es wurde daher auf dem Gebiet der klassischen Parameterschätzung und Modellbildung nach effizienten Algorithmen gesucht, die mathematische Modelle realisieren, die auf der experimentellen Beobachtung des Prozesses beruhen[9] und seine Modellierung aufgrund der beobachteten Beispielsituationen gestatten.

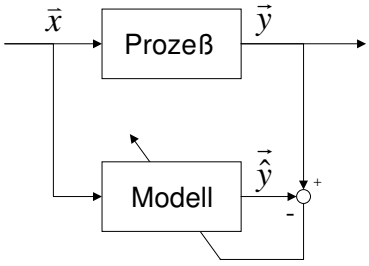


Abbildung 3.2: Belehren eines Prozessmodells mittels Vorhersagefehler

Mehrschichtige, vorwärtsgerichtete Neuronale Netze wie z.B. Multi-Layer-Perceptron´s (MLP[10]) und Radial-Basis-Functions-Netze (RBF-Netze[11]) eignen sich als allgemeine

[9] An dieser Stelle sei die Lektüre von Isermann [Isermann, 1974] und Wernstedt [Wernstedt, 1989] zur experimentellen Prozessanalyse empfohlen.

[10] Das MLP-Modell wurde erstmals ausführlich in [Rumelhart und McClelland, 1986] dargestellt, der Beweis seiner Eignung zur allgemeinen Funktionsapproximation findet sich in [Hornik et al, 1989].

[11] Das RBF-Netz wurde erstmals von Powell [Powell, 1962] und später nochmals von Poggio und Girosi in [Poggio und Girosi, 1989] vorgestellt, verschiedene Varianten dieses Netzwerktyps sind in [Butz, 1997] ausführlich dargestellt.

Funktionsaproximatoren eines mehrdimensionalen Eingangsvektors (Input: $\vec{x}$) auf einen mehrdimensionalen Ausgangsvektor (Output: $\vec{y}$) und können eine datengetriebene Modellierung nichtlinearer, statischer funktionaler Abhängigkeiten zwischen den Eingangs- und Ausgangsgrößen realisieren. Dabei wird durch Lernregeln die innere Struktur des Modells (Kopplungsmatrix) in Abhängigkeit von der Abweichung zwischen Modell-Output und Trainings-Output angepaßt (Abbildung 3.2). Nach Belehrung entsteht so ein empirisches Prozessmodell, das zwischen bekannten Stützstellen des Datenraumes interpolieren kann. An dieser Stelle soll das in dieser Arbeit mehrfach verwendete RBF-Netz näher vorgestellt werden.

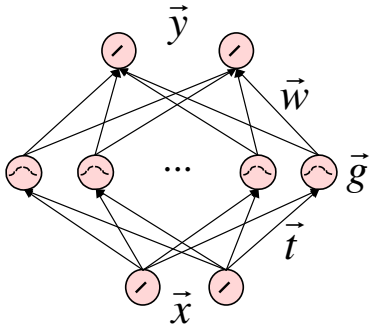


Abbildung 3.3: Struktur eines RBF-Netzes

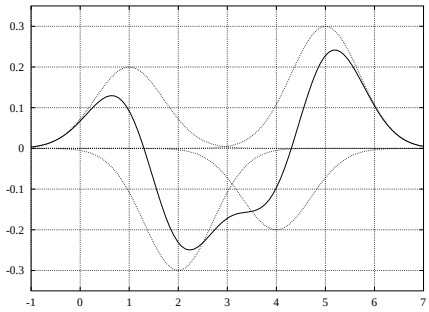


Abbildung 3.4: Überlagerung von vier radialen Basisfunktionen

Ein RBF-Netz ist ein dreilagiges, voll vernetztes, vorwärtsgerichtetes NN, das aus einer Schicht (Input-Schicht) von Eingabeneuronen (Input-Neurone), einer Schicht (Hidden-Schicht) von Zwischenneuronen (Hidden-Neurone) und einer Schicht (Output-Schicht) von Ausgabeneuronen (Output-Neurone) besteht (Abbildung 3.3). Die Neurone der Hidden-Schicht verfügen über eine gaussförmige (radialsymmetrische) Aktivierungsfunktion, wogegen die Neurone der Input- und Output-Schicht lineare Aktivierungsfunktionen besitzen. RBF-Netze realisieren daher eine, ähnlich wie in Abbildung 3.4 dargestellte, gewichtete lineare Kombination von nichtlinearen radialsysmmetrischen Basisfunktionen (z.B. Exponentialfunktionen). Die Neurone der Hidden-Schicht dienen der Kodierung der Eingabevektoren $\vec{x}$, so daß die Zuordnung zu den entsprechenden Ausgabevektoren $\vec{y}$ durch ein Anpassen der Kopplungsmatrix $\vec{w}$ zwischen der Hidden- und der Output-Schicht gelernt werden kann. Die Aktivität $\vec{g}$ der Hidden-Neurone berechnet sich durch Anwenden z.B. einer normierten Gaussfunktion auf den Abstand des aktuellen Eingabevektors $\vec{x}$ zur Position $\vec{t}$ der Hidden-Neurone im Eingangsraum (Gleichung [21]). Die Aktivität eines Output-Neurons $\hat{y}_i$ eines RBF-Netzes mit n Hidden-Neuronen wird daher nach Gleichung [22] bestimmt.

$$g_j = \exp\left(-\frac{\left\|\vec{x}-\vec{t}_j\right\|}{2\sigma_j^2}\right) \qquad [21]$$

$$\hat{y}_i = \sum_{j=1}^{n} w_{i,j} \cdot g_j \qquad [22]$$

Die Parameter des Netzwerkes, d.h. die Positionsvektoren $\vec{t}$, die Größe der radialen Basisfunktionen $\vec{\sigma}$ sowie die Gewichte $\vec{w}$ zwischen Hidden- und Output-Schicht können in Abhängigkeit vom Vorhersagefehler des Netzwerkes mittels einer Fehlerfunktion durch ein Gradientenabstiegsverfahren angepaßt werden. Dies ist jedoch sehr zeitaufwendig und setzt voraus, daß eine ausreichende Menge von Trainingsdaten zur Verfügung steht. Alternativ

werden daher häufig Clustertechniken (siehe Abschnitt 3.2.1) zur Bestimmung der Parametervektoren $\vec{t}$ und $\vec{\sigma}$ verwendet [Moody und Darken, 1989]. Das Belehren des RBF-Netzes erfordert dann im Anschluß daran nur noch eine Anpassung der Gewichtsvektoren $\vec{w}$ auf Basis des Vorhersagefehlers des Netzes und kann mit Gradientenabstiegsverfahren realisiert werden. Auch der in dieser Arbeit verwendete Lernalgorithmus für RBF-Netze folgt dieser Empfehlung: die Hidden-Neurone werden auf die durch LVQ ermittelten Clusterschwerpunkte des Eingangsraumes positioniert und die Größe der Basisfunktionen als Funktion des Abstands der Datensätze zu den im LVQ gefundenen Codebook-Vektoren geschätzt. Liegt nur eine geringe Anzahl von Trainingsvektoren vor, wird vereinfachend die Anzahl der Hidden-Neurone, der Anzahl der Trainingsvektoren gleichgesetzt und die Hidden-Neurone auf die Input-Vektoren der Trainingsdaten positioniert. Die Größe der Basisfunktionen wird für alle Hidden-Neurone einheitlich nach Gleichung [23] bestimmt, wobei d die maximale euklidische Distanz zwischen zwei Input-Vektoren der Trainingsdaten und n_{hidden} die Anzahl der Hidden-Neurone sei.

$$\sigma = \frac{d}{\sqrt{2 \cdot n_{hidden}}} \qquad [23]$$

Im Anschluß daran erfolgt eine Anpassung der Gewichtsvektoren $\vec{w}$ durch Singulär-Werte-Zerlegung (SVD) [Numerical Recipies, 1994], die gegenüber einem herkömmlichen Gradientenabstiegsverfahren um den Faktor 100 schneller arbeitet [Meyer und Eder, 1999].

3.2.3 Modellinvertierung

Häufig ist es nicht nur erforderlich, durch eine vorwärtsgerichtete Modellbildung die funktionalen Abhängigkeiten zwischen Prozesseingangs- und Ausgangsgrößen zu ermitteln, sondern auch durch eine invertierte Modellbildung diejenigen Prozesseingangsgrößen zu ermitteln, die eine festgelegte Kombination von Prozessausgangsgrößen bestimmen. Grundsätzlich existieren zwei verschiedene Ansätze, um eine Modellinvertierung zu realisieren [Jordan und Rumelhart, 1992]: die direkte und indirekte Modellinvertierung.

Bei der direkten invertierten Modellbildung wird der Prozessausgang $\vec{y}$ als Modelleingang und der Prozesseingang $\vec{x}$ als der zu lernende Modellausgang eines vorwärtsgerichteten Modells (Abbildung 3.5) verwendet. Auch hier ist der Vorhersagefehler das Kriterium, das zur Anpassung der Modellstruktur durch Standardlernverfahren z.B. auf Basis von Gradientenabstiegsverfahren verwendet wird. Gerade bei einer many-to-one-Beziehung[12] zwischen Prozesseingangs- und Ausgangsgrößen ist ein inverses Modell mit diesem Ansatz jedoch nicht ermittelbar [Jordan und Rumelhart, 1992].

Einen Ausweg bietet die indirekte invertierte Modellbildung, die sich auf das Prozessmodell zur Abbildung der Prozesseingangsgrößen auf die Prozessausgangsgrößen (siehe oben) stützt und bei der durch ein Optimierungsverfahren die Prozesseingangsgrößen solange variiert werden, bis sie bei Propagierung durch das Prozessmodell hinreichend genau die geforderten Prozessausgangsgrößen bestimmen.

[12] Unter einer *many-to-one* Beziehung sei eine Zuordnung zwischen den Prozesseingangsgrößen $\vec{x}$ und den Prozessausgangsgrößen $\vec{y}$ verstanden, bei der mindestens zwei verschiedenen Vektoren $\vec{x}_i$ ein identischer Vektor $\vec{y}$ zugeordnet werden kann.

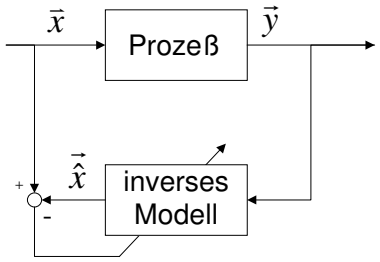


Abbildung 3.5: Belehren eines direkten, invertierten Prozessmodells mittels Vorhersagefehler

3.3 Vor- und Nachteile Neuronaler Netze

Systeme die nach den Prinzipien künstlicher Neuronaler Netze konzipiert sind, unterscheiden sich grundsätzlich von denen, die nach der von Neumann-Architektur arbeiten. Während die „klassischen“ von Neumann-Rechner sequentiell Information verarbeiten und speichern, arbeiten Neuronale Netze hochgradig parallel und speichern Daten assoziativ, d.h. inhaltsbezogen.

Neuronale Netze besitzen die Fähigkeit aus Beispielen lernen zu können, ohne das sie im herkömmlichen Sinne programmiert werden müssen. Sie können sich durch spezielle Lernverfahren selbst adaptieren und verfügen über eine gewisse Generalisierungsfähigkeit, indem sie wichtige Zusammenhänge in den Trainingsdaten in ihrer Struktur abbilden. Neuronale Netze sind aufgrund ihrer verteilten Wissensrepräsentation fehlertolerant und können auch unvollständige Daten sinnvoll verarbeiten.

Neben diesen Vorteilen besitzen NN auch eine Reihe von Nachteilen. Der Wissenserwerb erfolgt bei ihnen ausschließlich durch Lernen. Logisches oder regelbasiertes Wissen, das sehr leicht in regelbasierte Expertensystemen integriert werden kann, ist in NN schwerer modellierbar. NN sind „black-box“-Modelle, die nur sehr kompliziert oder gar nicht eine Analyse ihrer Modellentscheidung ermöglichen. Das Training kann u.U. sehr zeitaufwendig sein. NN arbeiten datengetrieben, d.h. es kann nur der Ausschnitt der Realität durch sie beschrieben werden, der auch durch die Trainingsdaten ausreichend repräsentiert wird. Gerade bei der Modellbildung durch NN in der experimentellen Prozessanalyse kommt einer geeigneten Versuchsplanung[13] daher eine große Bedeutung zu.

Trotz dieser Nachteile werden die verschiedenen Typen von NN heute vielfältig in der industriellen Praxis insbesondere zur Mustererkennung und –Vervollständigung, Datenanalyse und für Klassifikationsaufgaben überall dort erfolgreich eingesetzt, wo klassische Methoden der Informatik nicht mehr greifen und sich die Aufgaben einer analytischen Modellbildung entziehen[14].

[13] Zur Einführung in die verschiedenen Methoden der Versuchplanung wird als Lektüre [Kleppmann, 1998] empfohlen.

[14] Das Stuttgarter Institut für Mikrotechnik gab für das Jahr 1995 eine Anzahl von ca. 200 industriellen Anwendungen Neuronaler Netze an [Goos, 1998].

4 Evolutionäre Algorithmen als Optimierungsmethode

4.1 Evolutionäre Algorithmen – Grundprinzipien

Unter Evolutionären Algorithmen (EA) wird eine Klasse von stochastischen Optimierungsstrategien verstanden, die den Prozess der biologischen Evolution nachempfinden[15].

EA werden dadurch charakterisiert, daß sie über mehrere Generationszyklen t auf einer Menge, der *Population* $P(t)$, von *Individuen* agieren. Dabei entspricht jedes Individuum $\vec{a}$ einer Lösung des Optimierungsproblems und repräsentiert in geeigneter Weise sowohl die Entscheidungsvariablen $\vec{x}$ als auch mögliche Strategieparameter.

In Anlehnung an die Biologie - bei der die Erbinformation als Genotyp und das äußere Erscheinungsbild eines Individuums als Phänotyp bezeichnet wird – wird die Ausprägung der Entscheidungsvariablen, Zielfunktionswerte und Randbedingungen einer Lösung als *Phänotyp*[16] bezeichnet. Die Information, die zur Ausprägung des Phänotyps führt (hier der Wert der Entscheidungsvariablen bzw. ihre Kodierung mittels einer geeigneten Funktion Ω) sowie diverse Strategieparameter $\vec{s}$ werden unter dem Begriff *Genotyp* zusammengefaßt.

$$\vec{a} = \left(\Omega(\vec{x}), \vec{s}\right) \qquad [24]$$

Mit Hilfe einer Dekodierungsfunktion Γ wird der Genotyp $\vec{a}$ in den Phänotyp $\vec{x}$ transformiert:

$$\vec{x} = \Gamma(\vec{a}) \qquad [25]$$

Die Kodierung der Entscheidungsvariablen ist jedoch nicht zwingend erforderlich. Sind Ω und Γ Identitätsfunktionen, so gibt es für die Entscheidungsvariablen keinen Unterschied zwischen Genotyp und Phänotyp.

Den Individuen wird in Abhängigkeit ihrer Qualität ein Gütemaß, die sogenannte *Fitness* $\Phi(\vec{a})$ zugeordnet, das ihre Reproduktionstauglichkeit widerspiegelt. Unter Qualität soll hier ein Bewertungsmaß in Abhängigkeit der Erreichung eines oder mehrerer Optimierungsziele sowie der Einhaltung von Randbedingungen verstanden werden. Die Fitness eines Individuums beeinflußt unmittelbar die Wahrscheinlichkeit seiner Auswahl (*Selektion*) für den Fortpflanzungsprozess (*Reproduktion*), bei dem es seine die Qualität bestimmenden Eigenschaften – also die Ausprägung der Entscheidungsvariablen - an die Nachkommen weitergeben kann.

Während der Reproduktion wird das natürliche Prinzip von Variation in Form von Vermischen der Elterninformationen (*Rekombination*) und Verändern der Kindinformationen (*Mutation*) umgesetzt und auf diese Weise zum einen das Durchsetzen „guter“ Lösungen und zum anderen die Einhaltung der Variabilität (*Diversität*) der Lösungen erreicht.

---

[15] Eine gut verständliche Einführung in EA findet sich in den Büchern von H.P. Schwefel [Schwefel, 1995], I. Rechenberg [Rechenberg, 1994] sowie T. Bäck [Bäck et al, 1997] und V. Nissen [Nissen, 1997].

[16] Häufig wird unter dem Phänotyp eines Individuums seine technische Problemrepräsentation in der realen Welt verstanden - beispielsweise die reale Flugzeugtragfläche, die einen minimalen Luftwiderstand besitzen soll. Hier werden dagegen alle die Parameter, die zur Beschreibung des (technischen) Sachverhaltes notwendig sind, als Phänotyp zusammengefaßt – in diesem Beispiel also die geometrischen Abmessungen der Tragfläche und ihre Widerstandswerte.

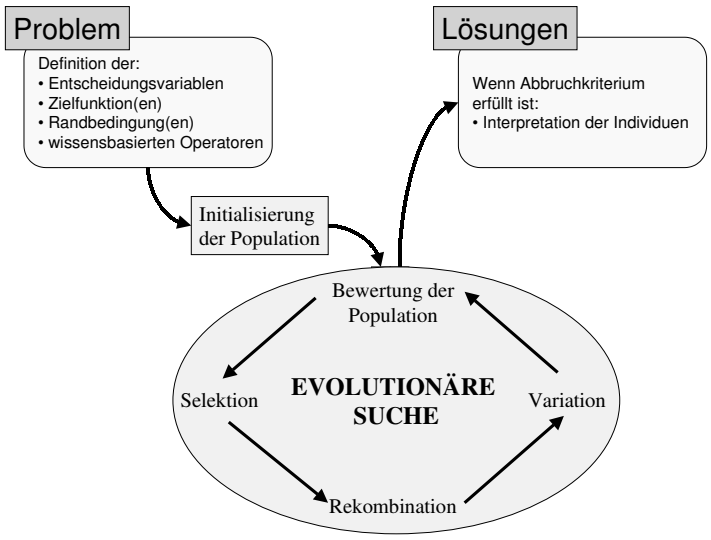


Abbildung 4.1: Grundprinzip Evolutionärer Algorithmen

Das Zusammenwirken von ungerichteter[17] Variation der Lösung durch Mutation und Rekombination sowie Selektion der besten Lösungen führt im Verlaufe vieler Generationszyklen sukzessiv zur Herausbildung qualitativ hochwertigerer Lösungen (siehe Abbildung 4.1 und Algorithmus 1 zum allgemeinen Ablaufschema von EA).

1	Problemrepräsentation	//	Kodierung, Anpassung der Evolutionsoperatoren
2	Wahl der Strategieparameter		
3	Initialisierung der Ausgangspopulation $P(0); t = 0$		
4	Bewertung der Individuen aus $P(0)$	//	Berechnung der Gütekriterien (*Fitness*)
5	$t = t + 1$		
6	Selektion	//	Auswahl der Eltern (*Selektion*)
7	Rekombination	//	Nachkommen generieren (*Rekombination)*
8	Variation	//	Nachkommen verändern (*Mutation*)
9	Bewertung der Nachkommen	//	Berechnung der Gütekriterien (*Fitness*)
10	Bilden der neuen Population $P(t)$	// //	Überlebende Kinder und Eltern werden in die nächste Generation aufgenommen (*Selektion*)
11	Wenn Abbruchkriterium erreicht ist, gehe zu Schritt 12 sonst gehe zu Schritt 5		
12	Ausgabe der Ergebnisse		
13	Stop		

Algorithmus 1: Pseudocode für den Ablauf eines allgemeinen EA

Abbildung 4.2 veranschaulicht die Anpassung der Entscheidungsvariablen der Individuen in Abhängigkeit von der Generationsanzahl für ein Optimierungsproblem mit nur einer Zielfunktion und zwei Entscheidungsvariablen (Gleichung [26]).

[17] An dieser Stelle sei angemerkt, daß sich durch den Einsatz von Mechanismen der Selbstadaption - wie etwa der Anpassung der Mutationsschrittweiten bei Evolutionsstrategien - die ungerichtete in eine gerichtete Suche umwandeln läßt (siehe dazu auch Abschnitt 4.1.1).

$$min: \ f(\vec{x}) = x_1^2 + x_2^2 \qquad [26]$$

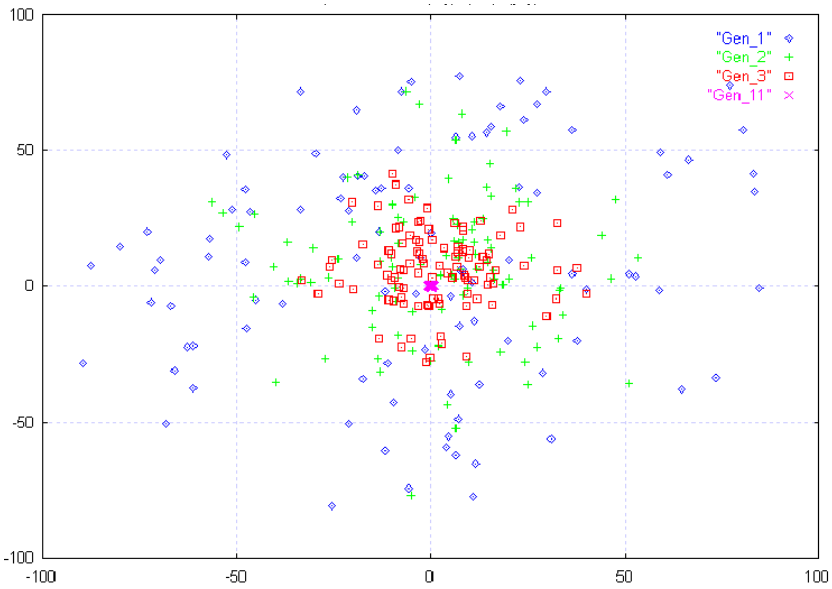


Abbildung 4.2: Lösungsverteilung der Population in Abhängigkeit der Generationszahl für das Optimierungsproblem [26] **(Abszisse: x_1, Ordinate: x_2)**

Im Folgenden sollen die oben genannten wichtigsten Ablaufelemente Evolutionärer Algorithmen kurz beschrieben werden, über deren unterschiedliche Ausprägungen sich hauptsächlich die verschiedenen Strömungen von EA definieren:

Problemrepräsentation: Bei der Wahl einer geeigneten Problemrepräsentation muß zunächst das Optimierungsproblem analysiert werden. Es sind die freien Parameter der Optimierung, d.h. die Entscheidungsvariablen zu identifizieren und eventuell zu kodieren. Wird eine Kodierungsfunktion Ω verwendet, sind die Ober- und Untergrenzen sowie die Genauigkeit der Kodierung (Anzahl der Bits pro Entscheidungsvariable) festzulegen. In Abhängigkeit von der Art des Optimierungsproblems sind verschiedene Lösungsrepräsentationen (z.B. reellwertige oder binäre Entscheidungsvariablen, Permutationen von Reihenfolgen, Baumstrukturen usw.) denkbar.

Initialisierung: In der Initialisierungsphase werden die Werte der Entscheidungsvariablen der Individuen der Anfangspopulation $P(0)$ stochastisch gesetzt. Häufig wird eine Gleichverteilung der Individuen über den Lösungsraum empfohlen, jedoch kann auch an dieser Stelle bereits problemspezifisches Wissen integriert und die Initialisierung auf bestimmte Bereiche dieses Raums beschränkt werden.

Bewertung der Individuen - Berechnung der Fitness: EA sind ableitungsfreie Verfahren, die die Güte (d.h. Fitness) einer Lösung als Optimierungskriterium verwenden. Die Fitness Φ eines Individuums $\vec{a}$ berechnet sich in den meisten Fällen direkt aus den Werten der Zielfunktionen oder seines Pareto-Rangs unter Berücksichtigung einer eventuellen Verletzung der Randbedingungen. Die einzelnen Zielfunktionen müssen dabei nicht in Form von algebraischen Gleichungen vorliegen, sondern können eine beliebige Abbildungssvorschrift enthalten, z.B. in

Form von experimentellen Simulationen oder als Berechnung der Zielfunktionswerte mit Hilfe eines Modells. Zur Berechnung der Fitness zur Lösung von MOP, siehe Abschnitt 5.2.

Selektion: Hauptsuchoperatoren Evolutionärer Algorithmen sind die Selektionsoperatoren. Es werden verschiedene Selektionsmechanismen unterschieden z.B.:

- Wettkampfselektion (*tournament selection*), bei der zwei oder mehrere Individuen stochastisch aus der Population ausgewählt und bezüglich ihrer Fitness miteinander verglichen werden. Der Sieger des Vergleichs wird als Elter der Folgegeneration ausgewählt.
- Fitnessproportionale Selektion (*roulette wheel selection*[18]), bei der die Selektionswahrscheinlichkeit der Individuen direkt proportional zu ihrer Fitness ist.
- Rangselektion (*ranking*), bei der die Selektionswahrscheinlichkeit der Individuen direkt proportional ihrer Position in einer aufgrund der Fitness berechneten Reihenfolge innerhalb der Population ist.

Evolution ist grundsätzlich nur dann möglich, wenn Mechanismen der Konkurrenz wirken. Die Stärke der Konkurrenz (z.B. die Anzahl der Wettkämpfer bei der Wettkampfselektion) bestimmt den Selektionsdruck. Ist dieser zu hoch gewählt, dominieren nur die fittesten Individuen, was zu einer vorzeitigen Konvergenz der Individuen auf nur eine Lösung und zu einer Verringerung der Variabilität der Gesamtpopulation führen kann.

Rekombination: Bei der Rekombination erfolgt die Generierung neuer Individuen durch das Mischen der Elterninformationen. Die Hauptformen der Rekombination sind:

- diskrete bzw. dominante Rekombination (crossover): Sie realisiert die zufällige Übernahme der Werte einzelner Entscheidungsvariablen – bzw. einzelner Kodierungselemente - aus den Entscheidungsvariablen der Eltern – bzw. deren Kodierungselementen.
- intermediäre Rekombination: Sie erzeugt die Werte neuer Entscheidungsvariablen durch Schwerpunktbildung der entsprechenden Entscheidungsvariablen der Eltern, z.B. durch Mittelwertbildung.

Durch Rekombination können „erfolgreiche“ Individuen, ihre Geninformationen an die Nachkommen weitergeben.

Mutation: Die Mutation ist neben der Rekombination der Evolutionsoperator, der durch die Veränderung der Entscheidungsvariablen die Variation einer Lösung bewirkt. Sowohl die Mutationswahrscheinlichkeit als auch die Mutationsrate können durch exogene Strategieparameter gesteuert werden, sind u.U. selbstadaptiv und unterliegen i.d.R. Elementen des Zufalls.

Erst durch die Anwendung von Mechanismen der Mutation können die Entscheidungsvariablen der Nachkommen den durch die Werte der Entscheidungsvariablen der Eltern definierten Wertebereich verlassen. Evolution ohne Mutation besitzt also zwangsläufig eine geringere Wahrscheinlichkeit, das Optimum zu erreichen als Evolution mit Mutation, da sie auch eine Veränderung der Entscheidungsvariablen jenseits der Werte der Eltern ermöglicht.

[18] Die Bezeichnung „roulette wheel selection“ leitet sich aus der Veranschaulichung der Fitnessproportionalen Selektion mit Hilfe eines Glücksrades ab: die Breite der Glücksrad-Segmente entspricht den Fitnesswerten der Individuen. Wird das Glücksrad n-mal gedreht, bleibt das Glücksrad mit einer gewissen Wahrscheinlichkeit in den einzelnen, den Individuen zugeordneten, Abschnitten hängen. Dabei wächst die Wahrscheinlichkeit der Auswahl proportional mit der Breite der Glücksradsegmente. Auf diese Weise besitzt ein Individuum mit einer hohen Fitness eine größere Auswahlwahscheinlichkeit, als ein Individuum geringerer Fitness – jedoch hat selbst das Individuum mit noch so kleinem Fitnesswert eine (entsprechend geringe) Auswahlwahrscheinlichkeit.

Die Anfänge Evolutionärer Algorithmen reichen bereits in die 1950er Jahre zurück[19]. Die EVOP-Methode von Box[20] zählt zu einem der bekanntesten frühen Ansätze von EA. In den 1960er Jahren haben sich zunächst unabhängig voneinander 3 Hauptströmungen Evolutionärer Algorithmen entwickelt:

- Evolutionsstrategien (ES) – *Rechenberg* [Rechenberg, 1965], [Rechenberg, 1973], [Rechenberg, 1994], *Schwefel* [Schwefel, 1968], [Schwefel, 1977], [Schwefel, 1995],
- Evolutionäres Programmieren (EP) - *Fogel, Owens, Walsh* [Fogel et al, 1966] – und [Fogel, 1992]
- Genetische Algorithmen (GA) – *Holland* [Holland, 1975], *De Jong* [De Jong, 1975], [De Jong und Spears, 1993] und *Goldberg* [Goldberg, 1989].

Einen weiteren, an dieser Stelle nicht diskutierten Ansatz stellt das Genetische Programmieren (GP) als Spezialfall der GA zur Optimierung von Programmcode dar (siehe z.B. [Blickle, 1996] und [Koza, 1992]).
In den darauffolgenden Jahren entstanden immer neue Varianten und Teilströmungen, so daß die anfänglichen Unterschiede immer mehr verschwammen. Heute betrachtet man die unterschiedlichen Varianten Evolutionärer Algorithmen als Instanzen einer gemeinsamen Methodenklasse, die in ihrer Grundstrategie übereinstimmen und sich nur in ihrer Problemrepräsentation und ihren Operatoren unterscheiden ([Nissen, 1994] S.11). So können beispielsweise die sogenannten reellwertigen Genetischen Algorithmen ([Mühlenbein et al, 1993] und [FlorMond_1996]) als Zwischenglied zwischen GA und ES betrachtet werden.

Obwohl die grundlegenden Prinzipien der EA bereits in den 1970er Jahren geschaffen wurden, hat sich erst seit Beginn der 1990er Jahre diese Klasse der Lösungsstrategien als robuste, für die verschiedensten Problemklassen einsetzbare Methode etabliert. Das hat verschiedene Ursachen: zum einen waren erst mit der Entwicklung der modernen Rechentechnik die Voraussetzungen für die Abarbeitung der sehr rechenintensiven Algorithmen gegeben. Zum anderen stellen sich die Probleme moderner Fertigungsunternehmen heute derart komplex dar, daß ihre Lösung mit traditionellen Methoden des OR nicht mehr zu leisten ist und daher die Entwicklung neuer Lösungsmethoden fordern.

Zur theoretischen Erklärung der Wirkungsweise Evolutionärer Algorithmen existieren heute zwei Schulen, die sich auf den Gebieten der Genetischen Algorithmen und Evolutionsstrategien entwickelt haben: Während sich die von I. Rechenberg begründete Schule [Rechenberg, 1973] um eine mathematisch exakten Analyse der Fortschrittsgeschwindigkeit für ES bemüht, was jedoch bisher nur für die zweigliedrige ES und das Korridor- und Kugelmodel (als Modelle des Fitnessgebirges) gelungen ist, versuchen Anhänger der Schema-Theoreme für GA [Goldberg, 1989] die Wirkungsweise von GA statistisch über Erfolgswahrscheinlichkeiten der Mutationen von Bit-Folgen zu deuten. Eine ausführliche Darstellung der Schema-Theoreme findet sich in: [Holland, 1975] und [Goldberg, 1989]. Für beide Ansätze gibt es eine Reihe von Kritikpunkten.

---

[19] Einen Überblick über die frühen Entwicklungen im Bereich EA geben Goldberg [Goldberg, 1989] und Fogel [Fogel, 1992].

[20] EVOP: „Evolutionary Operation“. Diese Methode basierte darauf, jeweils nur 2 oder 3 der möglichst einflußreichsten Parameter zu verändern. Um den aktuellen Zustand wird dann entsprechend dieser Parameter ein Quadrat bzw. ein Würfel konstruiert und bestimmte Zustände getestet. Der Zustand höchster Qualität wird als Ausgangspunkt der Konstruktion des folgenden Quadrats bzw. Würfels gewählt usw. ([Box, 1957], [Schwefel, 1977] S. 11). Das Verfahren wurde hauptsächlich zur dynamischen Optimierung chemischer Prozesse gewählt und ist auch heute noch als Methode der Versuchsplanung im Einsatz ([Kleppmann, 1998] S. 244 ff.). Der Gedanke, nur die einflußreichsten Parameter zu variieren, hat auch bei modernen EA nach wie vor Gültigkeit. Auch sollte an dieser Stelle die Ähnlichkeit der EVOP-Heuristik mit klassischen Optimierungsverfahren wie der Simplex-Methode nicht unerwähnt bleiben.

So setzt der Ansatz nach Rechenberg eine mathematische Beschreibbarkeit der Zielfunktionen voraus, was in der Praxis nicht immer gewährleistet ist. Kritiker der Schema-Theoreme heben dagegen hervor, daß diese Theoreme auf der Gleichverteilung der Schemata (Instanzen von Bitfolgen {1,0,*} mit einer Anzahl freier Bits „*“) innerhalb der Population beruht, was nach der ersten Anwendung der Selektionsmechanismen nicht mehr gewährleistet ist ([Salomon, 1997] S.8).

Eine vollständige exakte, mathematische Erklärung der Wirkungsweise von EA existiert jedoch bisher nicht. Im Folgenden sollen die Grundprinzipien, Unterschiede und Besonderheiten der 3 Hauptströmungen von EA bezüglich der Individuenrepräsentation und Evolutionsoperatoren zusammengefaßt werden.

4.1.1 Evolutionsstrategien (ES)

ES wurden für Optimierungsprobleme mit reellwertigen Entscheidungsvariablen entworfen. Die ersten bekannten Problemlösungen mit ES waren hauptsächlich Anwendungen aus dem ingenieurtechnischen Bereich, die ihren Ursprung in einer praxisorientierten evolutionären Experimentiermethodik hatten[21].

Der grundsätzliche Ablauf einer Standard-ES entspricht dem in Algorithmus 1 dargestellten generischen EA. Die Evolutionsoperatoren Selektion, Rekombination und Mutation manipulieren jedoch direkt den Phänotyp d.h. verwenden die natürliche Problemrepräsentation ohne genetische Kodierung. Sie beinhalten das problemspezifische Wissen.

Eine Besonderheit von Evolutionsstrategien besteht darin, daß sie über Mechanismen der Selbstadaptation verfügen, die die effiziente Einstellung der Mutationsschrittweiten betreffen. Wie I. Rechenberg nachwies [Rechenberg, 1973], kann pro Generation nur dann ein relevanter Fortschritt erzielt werden, wenn die Schrittweite der Mutation in einem geeigneten Bereich – dem sogenannten Evolutionsfenster – liegt. Bei zu kleiner Schrittweite ist die Fortschrittsgeschwindigkeit[22] zu gering, und es besteht die Gefahr, daß Individuen in der Nähe lokaler Minima die Population dominieren. Ist die Schrittweite dagegen zu groß gewählt, kann es sogar zu einer Verschlechterung der Güte der Individuen und einem „Überspringen“ des Optimums kommen.

Um einen Regel-Mechanismus zu realisieren, der die Muationsschrittweiten genau in diesem Evolutionsfenster hält, wurden verschiedene Ansätze entwickelt: die einfachste Realisierung einer solchen Adaptation sieht dabei die Anwendung von sogenannten Meta-Regeln zur Beeinflussung der Muationsschrittweite(n) vor. Eine solche Regel ist z.B. die von Rechenberg definierte „1/5-Erfolgsregel“[23]. Eine weitere Möglichkeit zur Realisierung der Selbstadaptation

---

[21] Zu den ersten Anwendungen gehörten z.B. die Optimierung eines Tragflügelprofils [Lichtfuss, 1965], die Optimierung einer Hochdruckdüse [Schwefel, 1968] und die Bestimmung der optimalen Biegekurve für ein Rohr, das um 90° abgewinkelt werden sollte [Rechenb erg, 1965].

[22] Als Fortschrittsgeschwindigkeit bezeichnet man den Erwartungswert der Abstandsänderung zum Optimum pro Generation [Rechenberg, 1973]. Evolutionäre Algorithmen mit einer hohen Fortschrittsgeschwindigkeit sind jedoch nicht zwangsläufig effizienter als diejenigen mit niedriger Fortschrittsgeschwindigkeit, da die Fortschrittsgeschwindigkeit nicht die Anzahl der Parameteranpassungen pro Generation berücksichtigt. Häufig wird die Anzahl der Fitnessberechnungen als Maßstab für den Vergleich der Effizienz zweier EA verwendet [Bäck und Schwefel, 1993], doch auch hier können bei gleicher Anzahl der Fitnessberechnungen die CPU-Rechenzeiten stark variieren, da keine Aussagen über die Anzahl der tatsächlichen Anpassungen eventueller interner Strategieparameter gemacht werden.

[23] In Abhängigkeit von der Erfolgswahrscheinlichkeit der Mutation P_S - d.h. der Wahrscheinlichkeit, daß ein Nachkomme eine höhere Qualität als sein Elter hat - wird die Mutationsstärke bei P_S>1/5 vergrößert oder

besteht im Einsatz hierarchisch geschachtelter Meta-ES [Herdy, 1992]. Hierbei handelt es sich um eine geschachtelte ES mit zwei Populationsebenen, wobei die erste der Optimierung der Mutationsschrittweiten und die zweite Ebene der Anpassung der Entscheidungsvariablen unter Verwendung dieser optimierten Mutationsschrittweiten dient. Am häufigsten wird jedoch zur Selbstadaptation eine Vorgehensweise gewählt, die darauf basiert, die Mutationsschrittweiten der einzelnen Entscheidungsvariablen in Form von internen Strategieparametern selbst mit in den Evolutionsprozess der ES mit einzubeziehen. So können sich diejenigen Individuen im Verlaufe der Generationen durchsetzen, die über eine ideal eingestellte Mutationsschrittweite ihrer Entscheidungsvariablen verfügen. In Abhängigkeit davon, wie diese Strategieparameter gestaltet sind, unterscheidet man zwischen isotroper und anisotroper Selbstadaptation. Da ES, deren Selbstadaptation sowohl auf Meta-Regeln als auch auf geschachtelte ES basieren, eher selten angewandt werden, soll auf ihre ausführliche Darstellung an dieser Stelle verzichtet werden. Vielmehr werden die grundlegenden Prinzipien einer ES mit anpassungsfähigen internen Strategieparametern übersichtsartig beschrieben[24].

Im Gegensatz zur Standardform Evolutionärer Algorithmen enthalten die Individuen $\vec{a}$ hier nicht nur einen Vektor aus n i.d.R. reellwertigen Entscheidungsvariablen $\vec{x} \in \Re^n$, sondern können darüber hinaus einen Vektor von reellwertigen Strategieparametern $\vec{\sigma}$ zur Anpassung der Mutationsschrittweiten besitzen.

Im Falle der isotropen Selbstanpassung enthält ein Individuum lediglich eine Mutationsschrittweite σ für alle Entscheidungsvariablen, d.h. $\vec{a} = (\vec{x}, \sigma)$. Wird dagegen ein Vektor $\vec{\sigma}$ von n_σ Mutationsschrittweiten σ_i mit $(n_\sigma \leq n)$[25] verwendet, d.h. gilt: $\vec{a} = (\vec{x}, \vec{\sigma})$, spricht man von anisotroper Selbstanpassung der ES. Die Evolutionsoperatoren werden nicht nur auf die Entscheidungsvariablen selbst, sondern auch auf ihre Mutationsschrittweiten angewandt. In vereinfachten ES-Varianten enthält ein Individuum keine adaptierbaren Mutationsschrittweiten, sondern ausschließlich den Vektor der Entscheidungsvariablen, d.h. $\vec{a} = (\vec{x})$.

Der Ablauf einer ES folgt prinzipiell der Darstellung in Algorithmus 1, besitzt jedoch einige Besonderheiten: Im Gegensatz zum Standard-EA, der eine Gleichverteilung der Individuen der Startpopulation über den Entscheidungsraum empfiehlt, erfolgt in den meisten praktischen ES-Anwendungen eine inhomogene Verteilung der μ Startindividuen durch die Berücksichtigung von Vorwissen. Ist ein solches Vorwissen nicht vorhanden, jedoch die Ober- und Untergrenzen der Entscheidungsvariablen bekannt, empfiehlt Schwefel in [Schwefel, 1995], diese mit einer bezüglich ihres Wertebereichs gleichverteilten Zufallszahl zu initialisieren. Für die Initialisierung der Mutationsschrittweiten empfiehlt Bäck einen Wert von $\sigma_i = 3.0$ [Bäck, 1996]. Es erscheint

---

bei P_S<1/5 verkleinert ([Rechenberg, 1973]). P_S berechnet sich aus dem Verhältnis der Zahl der Erfolge zur Zahl der Versuche (Mutationen) [Schwefel, 1977].

[24] Die Arbeiten von M.Milano et al stellen in [Milano et al, 2001] einen weiteren, neuen Aspekt der Adaptation der Schrittweiten bei ES dar. Hier werden mittels einer durch die aktuelle Population belehrten SOM die Reproduktionsoperatoren angepaßt und auf diese Weise ein Lernen des Evolutionspfades realisiert. D.Büche et al haben in [Büche et al, 2002] diesen Ansatz aufgegriffen und in ihrem SOM-MOEA eingearbeitet. Zur Diskussion dieser Herangehensweise siehe auch Abschnitt 5.4.3.2.

[25] Weniger häufig kommen ES-Varianten zum Einsatz, bei denen die Anzahl der Mutationsschrittweiten n_σ geringer als die Anzahl der Entscheidungsvariablen n ist, d.h. eine Mutationsschrittweite mehreren Entscheidungsvariablen zugeordnet wird.

jedoch sinnvoll, den u.U. bekannten Wertebereich der Entscheidungsvariablen bei der Initialisierung der σ_i zu berücksichtigen.

Nach der Initialisierung und Bewertung der Startpopulation werden λ Nachkommen erzeugt, indem jeweils mit gleicher Wahrscheinlichkeit von $1/\mu$ stochastisch ρ Eltern-Individuen selektiert (Ziehen mit Zurücklegen), rekombiniert und die so entstandenen Nachkommen mutiert werden.

Die Rekombination wird sowohl auf $\vec{x}$ als auch auf $\vec{\sigma}$ angewandt. Für die Entscheidungsvariablen wird i.d.R. die diskrete bzw. dominante Rekombination verwendet. Dabei wird für jede Komponente des Vektors der Entscheidungsvariablen des Kindindividuums x_i^K stochastisch der Wert der Entscheidungsvariable der ρ Eltern in der Dimension i übernommen. Welche Komponenten i der Entscheidungsvariablen rekombiniert werden, wird zufällig bestimmt[26]. Auf die Mutationsschrittweiten wird dagegen typischerweise die intermediäre Rekombination angewandt, d.h. eine oder mehrerer Mutationsschrittweiten σ_i^K des Kindes werden dadurch erzeugt, daß der Schwerpunkt der entsprechenden Mutationsschrittweiten σ_i^{Ej} mit $j = 1,..,\rho$ der ρ Eltern gebildet wird (siehe Gleichung [27]).

$$\sigma_i^K = \frac{1}{\rho}\sum_{j=1}^{\rho}\sigma_i^{Ej} \qquad [27]$$

Der Rekombination schließt sich zunächst die Mutation aller σ_i aus $\vec{\sigma}$ und dann die Mutation jeder Komponente x_i von $\vec{x}$ des neu gebildeten Nachkommens an (siehe [28] und [29]), was einer Mutationswahrscheinlichkeit von $p_m = 1$ entspricht. Dabei wird die Mutation der Entscheidungsvariablen x_i wird durch Addition einer Zufallszahl realisiert, die durch die Mutationsschrittweite σ_i der Entscheidungsvariablen sowie eine $N(0,1)$-normalverteilte Zufallszahl bestimmt ist.

Bei der Standard-ES werden $N(0,1)$-normalverteilte Mutationen verwendet, d.h. Zufallszahlen mit einem Erwartungswert von 0 und einer Standardabweichung von 1. Da bei ES die Entscheidungsvariablen nicht explizit in ihrem Wertebereich beschränkt sind, sind auf diese Weise sehr große Variationen der einzelnen Entscheidungsvariablen, wenngleich auch nur mit einer geringen Wahrscheinlichkeit, möglich. Das hat den Vorteil, daß sich die Optimierung nicht nur auf die Bereiche des Zielraums beschränkt, die durch die vom Anwender (u.U. ungünstige) Definition von Ober- und Untergrenzen der Entscheidungsvariablen bestimmt werden[27].

Für die Mutation der Mutationsschrittweiten σ_i werden dagegen logarithmisch normalverteilte Zufallszahlen verwendet. Die mutierte Mutationsschrittweite σ_i' berechnet sich nach Gleichung [28] ([Bäck und Schwefel, 1993] S.4) und berücksichtigt neben einer Zufallszahl nach einer einmalig für alle Entscheidungsvariablen berechneten Normalverteilung $N(0,1)$ eine weitere Zufallszahl, die nach einer für jedes $i = 1,..,n$ neu zu bestimmenden Normalverteilung $N_i(0,1)$

[26] Eine Erweiterung dieses Schemas, das die Wahrscheinlichkeit der Auswahl von Entscheidungsvariablen entsprechend ihrer Relevanz für den Optimierungserfolg steuert, wird im Abschnitt 6.4 vorgestellt.

[27] Sind jedoch solche Grenzen in Form von Randbedingungen bei der Optimierung zu berücksichtigen, werden diese i.d.R. direkt in die Evolutionsoperatoren integriert.

berechnet wird. Der Einfluß beider Zufallszahlen auf $\sigma_i^{'}$ wird durch die exogenen Konstanten τ und $\tau^{'}$ bestimmt. Der globale Faktor $\tau^{'} \cdot N(0,1)$ beeinflußt dabei die Veränderungen aller Schrittweiten einheitlich, wogegen der Faktor $\tau^{'} \cdot N_i(0,1)$ die individuelle Anpassung einzelner Schrittweiten pro Entscheidungsvariable realisiert.

$$\sigma_i^{'} = \sigma_i \cdot \exp(\tau^{'} \cdot N(0,1) + \tau \cdot N_i(0,1))$$
$$\tau^{'} \approx \frac{1}{\sqrt{2 \cdot n}} \qquad \tau \approx \frac{1}{\sqrt{2 \cdot \sqrt{n}}} \quad \textit{empfohlene Werte} \qquad [28]$$

Sind die Mutationsschrittweiten neu bestimmt worden, werden die Entscheidungsvariablen selbst nach folgendem Schema mutiert:

$$x_i^{'} = x_i + \sigma_i^{'} \cdot N(0,1) \qquad [29]$$

Im Anschluß daran wird die Fitness $\Phi(\vec{a})$ der so veränderten Nachkommen berechnet. Alle generierten Nachkommen gelangen danach in eine Zwischenpopulation, die in Abhängigkeit von der Wahl des Selektionsschemas nicht nur die λ Nachkommen („Komma"-Strategie), sondern auch die μ Eltern („Plus"-Strategie) enthält.

Der Vorgang von Selektion, Rekombination und Mutation wiederholt sich für alle λ Nachkommen. Sind diese erzeugt, werden die bezüglich der Fitness μ besten Individuen der Zwischenpopulation in die neue Generation übernommen und der Generationszyklus wiederholt sich, bis ein Abbruchkriterium erfüllt ist. Häufig ist das Abbruchkriterium so definiert, das der Generationszähler t einen bestimmten Wert $t_{\max}$ nicht überschreiten darf oder ist direkt an die Qualität der Individuen der aktuellen Population gebunden. So schlägt beispielsweise Schwefel in [Schwefel, 1995] vor, eine ES dann zu beenden, wenn die Differenz zwischen dem besten und schlechtesten Fitnesswert innerhalb einer Population bezogen auf die Populationsgröße einen bestimmten Wert unterschreitet. Oft gilt eine ES auch dann als terminiert, wenn die durchschnittliche Fitness der Populationsmitglieder über eine bestimmte Anzahl von Generationen unterhalb einer parametrierbaren Schwankungsbreite bleibt.

Folgende Nomenklatur wurde von H.P. Schwefel zur Spezifizierung einer ES eingeführt: (μ/ρ,λ)-ES[28] für eine ES mit „Komma"-Strategie und (μ/ρ+λ)-ES für eine ES mit „Plus"-Strategie. Dabei ist μ die Anzahl der Eltern, ρ die Anzahl der an einer Rekombination beteiligten Eltern und λ die Anzahl der Nachkommen. Empfohlen wird ein Verhältnis von $\mu/\lambda \approx 1/7$ und $\mu >> 1$ (z.B. $\mu=15$, $\rho = 2$ und $\lambda=100$ [Schwefel, 1995]). Wird ρ nicht angegeben, handelt es sich um eine Evolutionsstrategie ohne Rekombination wie z.B. die zweigliedrige (1+1)-ES.

Eine interessante, jedoch aufwendige und daher selten verwendete Variante Evolutionärer Strategien ist die ES mit korrelierten Mutationen ([Schwefel, 1981] und [Hoffmeister und Bäck, 1992]). Hier wird als zusätzlicher interner Strategieparameter ein Vektor von Rotationswinkeln $\vec{\alpha}$ verwendet - damit ist $\vec{a} = (\vec{x}, \vec{\sigma}, \vec{\alpha})$. Die Mutation der Entscheidungsvariablen erfolgt durch $\vec{x}^{'} = \vec{x} + \vec{N}(\vec{0}, \vec{\sigma}^{'}, \vec{\alpha}^{'})$ wobei $\vec{N}(\vec{0}, \vec{\sigma}^{'}, \vec{\alpha}^{'})$ eine verallgemeinerte n-dimensionale Normalverteilung mit dem Erwartungsvektor $\vec{0}$ und einer durch $\vec{\sigma}^{'}$ und $\vec{\alpha}^{'}$ beschriebenen Kovarianzmatrix ist ([Nissen_1997] S.167). Durch die Verwendung von $\vec{\alpha}$ bei der Berechnung der mutierten Entscheidungsvariablen wird eine Koordinatenrotation erreicht, die die Mutationen

[28] Auf geschachtelte ES der Isolationszahl γ sollen hier nicht näher eingegangen werden, siehe dazu [Herdy, 1992] und [Schwefel, 1995].

der Entscheidungsvariablen in Richtung höherer Erfolgswahrscheinlichkeit zerrt, da der Gradient als beste Suchrichtung mit den Koordinatenachsen in Übereinstimmung gebracht wird. Obwohl der Einsatz korrelierter Mutationen im Gegensatz zu unabhängigen Mutationen nach Gleichung [22] eine höhere Fortschrittsgeschwindigkeit bewirkt, führt der zusätzliche Aufwand, der durch der Einbeziehung der $\vec{\alpha}$ in den Evolutionsprozess nötig ist, nicht zwangsläufig zu einer Effizienzsteigerung des Optimierungsalgorithmus. Daher wird von den meisten Autoren empfohlen, zunächst eine ES mit Selbstanpassung ohne korellierte Mutationen zu verwenden.

Die Selbstadaptation der Strategieparameter ist eine hervorstechende Eigenschaft von ES, die sie von anderen EA-Formen hervorhebt. Durch diesen Mechanismus kann die ungerichtete stochastische Suche in eine gerichtete Suche überführt werden. Neuere Untersuchungen haben gerade aufgrund dieser Eigenschaft die Überlegenheit der ES gegenüber anderen Formen Evolutionärer Algorithmen hervorgehoben [Salomon, 1996].

4.1.2 Evolutionäre Programmierung (EP)

Eine der ES ähnliche Variante Evolutionärer Algorithmen ist die Evolutionäre Programmierung (EP). Sie wurde in den 1960er Jahren von Fogel, Owens und Walsh konzipiert [Fogel et al, 1966]. Das Ziel der ersten Arbeiten auf dem Gebiet der EP bestand darin, endliche Automaten zu kreieren, die in der Lage waren, Signalsequenzen vorherzusagen. Die Individuen waren dabei Zustandsgraphen, deren Ausgangssymbole und Zustandsübergänge mutiert werden konnten. Die Fitness der entsprechenden Individuen berechnete sich aus den „Treffern“ des Automaten, indem die tatsächlichen mit den prognostizierten Ausgabesignalen verglichen wurden ([Nissen, 1994] S.177ff). In den 1990er Jahren wurden die EP von D.B. Fogel auf die Anwendung von Optimierungsproblemen mit reellwertigen Entscheidungsvariablen und normalverteilten Mutationen erweitert [Fogel, 1992].

Das Schema einer EP ist einer reinen $(\mu+\lambda)$-ES mit $\mu=\lambda$ sehr ähnlich. Auch bei der EP werden die Entscheidungsvariablen nicht kodiert, sondern eine möglichst problemnahe phänotypische Individuenrepräsentation verwendet. Als Operatoren werden neben der Selektion lediglich die Mutation eingesetzt. Der Ablauf einer EP entspricht grundsätzlich dem Schema in Algorithmus 1 ohne Rekombination.

Nach der Initialisierung, bei der zufällig μ Startindividuen erzeugt und bewertet werden, wird durch Kopieren aus jedem Elter ein Nachkomme erzeugt und anschließend mutiert. Die Mutation [30] der reellwertigen Entscheidungsvariablen x_i berechnet sich durch Addition einer normalverteilten $N(0,\sigma)$-Zufallszahl. Dabei wird σ in Abhängigkeit von der durch k_i gewichteten Fitness des Elternindividums $\Phi(\vec{a})$ und der Grundvarianz z_i bestimmt. Sowohl der Skalierungsfaktor k_i als auch die Grundvarianz z_i können für jedes x_i verschieden sein.

$$x_i^{'} = x_i + N(0,\sqrt{k_i \cdot \Phi(\vec{a}) + z_i}) \qquad [30]$$

Die EP verfügt ebenso wie die ES über einen Anpassungsmechanismus der Mutationsschrittweite – jedoch wird diese nicht mit in den Evolutionsprozess einbezogen, sondern durch Meta-Regeln berechnet. Diese Regeln passen die Mutationsschrittweite σ in Abhängigkeit der Güte der Individuen in der Form an, daß bei Annäherung an das Optimum eine zunehmende Verkleinerung der Mutationsschrittweite erfolgt. Voraussetzung dafür ist jedoch, daß die einzelnen Zielfunktionswerte derart in die Berechnung der Fitness eingehen, daß das zu erreichende Optimum einem Wert der Fitnessfunktion von 0 entspricht, d.h. in diesem Fall die Fitness minimiert wird.

Sind alle μ Nachkommen erzeugt und mutiert worden, erfolgt die Auswahl der Individuen, die in die neue Generation aufgenommen werden. Bei der Standard-EP wird dazu als Selektionsmechanismus die Wettkampfselektion in folgender Form benutzt: es werden für jedes der μ Eltern *und* λ Nachkommen stochastisch q Kandidaten aus der Vereinigungsmenge aus Eltern und Nachkommen ausgewählt, gegen die diese konkurrieren müssen. Für jedes Individuum wird dabei die Anzahl der Siege gegen seine Konkurrenten gezählt, die gesamte Population anschließend entsprechend dieser Maßzahl sortiert und schließlich nach dieser Rangfolge die μ besten Individuen in die neue Generation übernommen. So ist es möglich, daß auch qualitativ schlechtere Individuen eine positive Selektionswahrscheinlichkeit erhalten. Durch Variation der Zahl der Wettkämpfer q läßt sich Selektionssensitivität steuern. Einen sehr hohen Selektionsdruck, erreicht man durch die Wahl großer Werte von q, was im Extremfall bei $q = \mu$ einer Eliteselektion entspricht.

Ausgehend von der dargestellten Standardvariante EP haben sich eine ganze Reihe von Erweiterungen und Modifizierungen vor allem bezüglich der Selektion und Steuerung der Mutationsschrittweite[29] herausgebildet.

EP finden ihre Anwendung z.B. in der Strukturoptimierung, sind aber nicht so stark verbreitet wie ES oder GA.

4.1.3 Genetische Algorithmen (GA)

Genetischen Algorithmen (GA) wurden unabhängig zur Entwicklung der ES und EP von J. Holland [Holland, 1975] und K. A. De Jong [De Jong, 1975] in den 1970er Jahren vor allem in den USA als eine weitere Richtung Evolutionärer Algorithmen entwickelt. Sie wurden ursprünglich zur Modellierung adaptiver Prozesse entworfen und stellen heute die am weitesten verbreitete Variante von EA dar.

Die hervorstechenste Eigenschaft, die GA von anderen Formen Evolutionärer Algorithmen wie ES und EP unterscheidet, besteht in ihrer speziellen Problem- bzw. Individuenrepräsentation. Die Individuen entsprechen auch hier diskreten Problemlösungen, deren Phänotyp jedoch im Genotyp kodiert wird. GA arbeiten also nicht direkt auf den Entscheidungsvariablen, sondern auf deren Kodierung[30]. Der Genotyp einer Lösung $\vec{x}$ wird durch eine endliche Binärzeichenfolge dargestellt. Dabei wird jeder Entscheidungsvariable eine Teilzeichenfolge[31] zugeordnet - der Anwender muß also entscheiden, wie und mit wie vielen Bits die einzelnen Entscheidungsvariablen kodiert werden. Zugleich ist es (im Gegensatz zu ES und EP) erforderlich, jeder zu kodierenden Objektvariable eine obere und untere Grenze zuzuordnen. Zur Berechnung der Fitness muß der Genotyp des Individums durch eine anwendungsspezielle Dekodierungsfunktion transformiert werden d.h. die Fitnessfunktion setzt sich aus der Dekodierungsfunktion und der Zielfunktion zusammen. Die Kodierung des Phänotyps wird oft

[29] In [Fogel, 1992] werden u.a. verschiedene Meta-Regeln zur Steuerung der Mutationen vorgeschlagen z.B. in Abhängigkeit von der Anzahl der Generationen, unter Berücksichtigung verschiedener Zufalls-Verteilungen oder auch unter Einbeziehung von Gradienteninforamtionen und problemspezifischen Heuristiken.

[30] Typischerweise verwendet man zur Kodierung den Standard-Binärcode, der jedoch in Hinblick auf die Einhaltung des Kausalitätsprinzips Nachteile aufweist: natürlicherweise sollten kleine bzw. große Veränderungen an der Struktur von Systemen auch zu kleinen bzw. großen Veränderungen an den Eigenschaften des Systems führen, was mit den Standardbinärcode nicht gewährleistet werden kann. Eine Möglichkeit, dieses Problem zu umgehen, liegt in der Anwendung der Gray-Kodierung.

[31] Eine Außnahme stellen hier die reellwertigen GA ([FlorMond_1996], [Mühlenbein et al, 1993]) dar.

damit begründet, daß auch in der natürlichen Evolution eine Kodierung der phänotypischen Eigenschaften von Organismen durch Nukleotidbasen erfolgt.

GA folgen dem in Algorithmus 1 dargestellten prinzipiellen Ablaufschema von EA: im Anschluß an die Problemrepräsentation, d.h. nach der Definition der Anzahl der Bits, die zur Kodierung einer Problemlösung notwendig sind, erfolgt die stochastische Initialisierung der Startpopulation mit μ Individuen[32]. Dabei werden die einzelnen Bits des Genotyps aller Individuen stochastisch mit 0 oder 1 belegt. Im Anschluß an die Initialisierungsphase wird die Fitness der Individuen berechnet, indem aus dem Genotyp zunächst der Phänotyp berechnet wird, auf den dann die Bewertungsfunktion angewandt wird.

Nach der Bewertung der Ausgangspopulation erfolgt die stochastische Selektion (mit Zurücklegen) und Replikation der ebenfalls μ Eltern-Individuen. Auf diese Weise wird eine Zwischenpopulation - der sogenannte *mating pool* - gebildet. Während Holland den klassischen Fitness-proportionalen Selektionsoperator vorschlug, arbeiten neuere Varianten häufig mit rangbasierter oder Wettkampfselektion[33].

Ist der mating pool mit μ Eltern-Individuen aufgefüllt, werden aus ihm mit gleicher Wahrscheinlichkeit von $1/\mu$ die Eltern ohne Zurücklegen gezogen, aus denen dann durch Anwenden der Operatoren Rekombination (*crossover*) und Mutation die Individuen der nächsten Generation entstehen.

Das Auftreten von Rekombination und Mutation wird durch die exogenen Wahrscheinlichkeitskonstanten p_c und p_m gesteuert. Dabei bestimmt p_c die Wahrscheinlichkeit der Anwendung der Operatoren für die Rekombination und p_m die der Mutation. Für p_c empfehlen Schwefel und Bäck in [Bäck und Schwefel, 1993] einen Wert von $p_c \geq 0.6$.

Für die Wahl effizienter Mutationswahrscheinlichkeiten gibt es eine Reihe von Empfehlungen: De Jong z.B. schlägt in [De Jong, 1975] für $p_m = 0.001$, Grefenstette dagegen in [Grefenstette, 1986] einen Wert von $p_m = 0.01$ vor. Mühlenbein berechnet die empfohlene Mutationswahrscheinlichkeit bei einer Kodierung mit l Bits und einer Populationsgröße von μ mit: $p_m \leq 1.7/\left(\mu \cdot \sqrt{l}\right)$ [Mühlenbein, 1992]. Bei reellwertigen GA, die n Entscheidungsvariablen als Fließkommazahlen darstellen, wird $p_m \leq 1/l$ empfohlen ([Salomon, 1997] S.5).

Im Unterschied zu ES und EP verwenden GA also sehr kleine Mutationswahrscheinlichkeiten und verwenden die Mutation lediglich als Hintergrundoperator und im Gegensatz dazu die Rekombination als Hauptsuchoperator. Motiviert wird dieses Vorgehen mit der Tatsache, daß auch in der natürlichen Evolution das Mischen der Erbinformationen wahrscheinlicher ist als das Auftreten phänotypisch wirksamer Mutationen.

GA verwenden zum Mischen der Erbinformation von q Individuen (i.d.R. ist $q = 2$) eine diskrete Rekombination das sogenannte crossover. Dabei wird hauptsächlich das traditionelle N-Punkt-

[32] Empfohlene Werte für die Populationsgröße liegen für GA zwischen 30 und 500 ([Nissen, 1997] S.37).

[33] Ein Vergleich der Effizienz der unterschiedlichen Selektionsoperatoren findet sich in [Thiersen und Goldberg, 1994]. Thiersen und Goldberg kommen hier zu dem Schluß, daß eine (μ,λ) Selektion mit Übernahme des besten Individuums also einer sogenannten *elitist truncation selection* zu den günstigsten Konvergenzraten führt. (Siehe dazu auch [Salomon, 1997] S.5)

oder das uniform-crossover angewandt[34]. Beim N-Punkt-crossover werden die Bitstrings der beteiligten Eltern identisch in (N+1) zufällig ausgewählte Substrings unterteilt. Die Bitstrings der Nachkommen ergeben sich, indem abwechselnd die Substrings der Eltern ausgetauscht werden. Für den Grad der Durchmischung, also die Anzahl der crossover-Punkte empfehlen Bäck und Schwefel in [Schwefel und Bäck, 1992] Werte von 2 oder 8. Beim uniform-crossover hingegen wird für jede Position auf den Bitstrings der Eltern mit einer bestimmten Wahrscheinlichkeit p_{ux} ein Austausch der Bits durchgeführt. In [Mitchell, 1996] wird für diese bitbezogene Wahrscheinlichkeit ein Wert von $0.5 \leq p_{ux} \leq 0.8$ angegeben.

Nach Abschluß der Rekombination erfolgt bei den neu generierten Individuen (pro Rekombination entstehen 2 Nachkommen) die Mutation, bei der jedes Bit mit einer Wahrscheinlichkeit von p_m invertiert wird. Auch hier sind wiederum dynamische Anpassungen der Mutationswahrscheinlichkeiten möglich ([Salomon, 1997] S.5).

Nachdem alle μ Individuen erzeugt wurden, beginnt der oben beschriebene Generationszyklus von Selektion, Rekombination und Mutation von neuem.

Neben dem klassischen Anwendungen GA auf dem Gebiet der Parameteroptimierung, finden sich gerade im Bereich der kombinatorischen Optimierung (z.B. Travelling-Salesman-Probleme) eine Vielzahl von Anwendungen.

Auf eine weitere Hauptform Evolutionärer Algorithmen - die Genetische Programmierung (GP) - soll an dieser Stelle nicht näher eingegangen werden. Sie wird häufig als Unterform GA angesehen und arbeitet mit Genomen, die symbolische Informationen repräsentieren. Hauptanwendungsfelder der GP ist der Entwurf von Computercode - vor allem in der Form von LISP-Programmen ([Koza, 1992] und [Koza, 1994]).

4.1.4 Vergleich von ES, EP und GA

Die Frage, welche der dargestellten EA-Hauptformen für eine bestimmte Problematik zu wählen ist, ist nicht eindeutig zu beantworten. Historisch gesehen wurden die einzelnen EA-Formen für verschiedene Problemklassen angewandt: so werden z.B. ES traditionell zur reellwertigen Parameteroptimierung und GA für die Lösung von Permutationsproblemen eingesetzt – wobei es, wie bereits erwähnt, zunehmend zu einer Vermischung der verschiedenen Lösungsstrategien kommt.

Eine fundierte theoretische Erklärung der Funktionsmechanismen Evolutionärer Algorithmen fehlt sowohl für die ES als auch GA und EP bis heute. Die in der Literatur zu findenden vergleichenden Betrachtungen können sich daher immer nur auf experimentelle Beobachtungen stützen, die in ihrer Qualität natürlich stark von der Wahl der Testprobleme abhängen. Salomon hat diese Problematik besonders kritisch bezüglich der Effizienz von GA in [Salomon, 1997] hinterfragt.

> *„Bei den vorgeschlagenen kleinen Mutationswahrscheinlichkeiten wird im statistischen Durchschnitt nur eine Variable je Nachkomme geändert, was einer Gauss-Seidel Strategie entspricht und bei linear separierbaren Funktionen sehr effizient ist. Da für die Konvergenz die konkrete Wahrscheinlichkeitsdichte des Mutationsoperators nur eine untergeordnete Bedeutung spielt, ist die erhaltene Konvergenzgeschwindigkeit nicht ein*

[34] Weitere Formen des cross-overs sind z.B. das shuffle-cross-over [Eshelmann et al, 1989], das diagonal-cross-over [Eiben und Kemenade, 1995], das average-cross-over [Davis, 1991b] oder das blend-cross-over [Eshelmann und Schaffer, 1993].

> *Resultat effizienter genetischer Operatoren, sondern resultiert aus der Eigenschaft dieser Testfunktionen." ([Salomon,* 1997] *S.9)*

Salomon bestätigt durch ein einfaches Experiment seine These: wird eine dieser Testfunktionen (z.B. die Rastriginfunktion) durch Rotation des Koordinatensystems verändert, so wird die Konvergenzgeschwindigkeit des GA drastisch verlangsamt. Durch die Rotation wird eine Epistasis der Objektvariablen – also Nichtseparabilität der Effekte einzelner Entscheidungsvariablen – erreicht. Im Gegensatz zu GA, welche Mutationen entlang der Koordinatenachsen stark bevorzugen, sind ES gegenüber Effekten der Epistasis unempfindlich, da hier die Nachkommen aufgrund der Mutationen kugelförmig mit einem Radius von $\sigma \cdot \sqrt{n}$ verteilt sind ([Salomon, 1997] S.10).

Um aber tatsächlich die Effizienz der verschiedenen evolutionären Lösungsstrategien vergleichen zu können, wurden eine Reihe künstlicher Testfunktionen entwickelt – eine Auswahl findet sich in [Bäck und Schwefel, 1993]. Dort haben H. P. Schwefel und T. Bäck anhand verschiedener Testfunktionen mit bis zu 30 Entscheidungsvariablen verschiedene EA (isotrope und anisotrope ES, EP und GA mit Gray-Code-Kodierung) bezüglich ihrer Konvergenzgeschwindigkeit miteinander verglichen. Ihre Ergebnisse zeigen klar die Überlegenheit der ES gegenüber GA. Schwefel und Bäck weisen experimentell nach, daß eine richtig parametrierte Evolutionsstrategie aufgrund ihres Mechanismus´ der Selbstadaptation der Mutationsschrittweiten bezüglich der Zielwertberechnungen schneller (bezüglich der Anzahl der Zielfunktionsevaluierungen) das Funktionsminimum erreicht als EP und GA. Weiter wird gerade bei multimodalen Zielfunktionen der Wert des Rekombinationsoperators und der Komma-Strategie hervorgehoben. ES und EP verhalten sich zusätzlich sehr robust bei verrauschten Zielfunktionen, was gerade unter dem Aspekt der praktischen Verwertbarkeit des Verfahrens von Bedeutung ist.

Ein weiteres Argument, das ES gegenüber den GA vorteilhafter erscheinen läßt, betrifft die Kodierung der Information. Wie bereits erwähnt, ist mit der binären Kodierung das Prinzip der „starken Kausalität" nicht mehr garantiert. Die Mutation eines einzigen Bits kann die Entscheidungsvariable um Größenordnungen verändern. GA mit Gray-Kodierungen sollten daher immer den Vorrang haben. Bei ES und EP stellt sich dieses Problem jedoch nicht, weil eine Kodierung wie in GA nicht zum Einsatz kommt. I. Rechenberg wies in [Rechenberg, 1994] darauf hin, daß mit den Kodierungen der Entscheidungsvariablen eine Transformation in einen neuen Raum erfolgt und die GA-Evolutionsoperatoren eben auf diesen Raum und nicht auf den Raum der Entscheidungsvariablen zugeschnitten sind. Damit ist auch das unmittelbare Einbringen von Expertenwissen bei den Operatoren nur sehr schwer möglich. In der Praxis zeigt sich jedoch, daß bei den Standard-Versionen Evolutionärer Algorithmen oft das Einbringen problemspezifischer Heuristiken notwendig ist, um eine tatsächlich taugliche Lösung zu generieren. Evolutionsstrategien eignen sich hingegen für die Integration spezieller Heuristiken bei den Evolutionsoperatoren besser als GA.

Evolutionäre Algorithmen in ihren Standardformen gehören zu den schwachen Methoden, d.h. sie verfügen über einen sehr hohen Grad der Allgemeinheit und einen vergleichsweise geringen Grad der Leistungsfähigkeit[35]. Durch den Einsatz problemspezifischer Operatoren und Lösungsrepräsentationen lassen sie sich jedoch in starke, speziell für ausgezeichnete Probleme zugeschnittene Methoden, umwandeln. Besonders ES und EP eignen sich hierbei für die direkt bei den Evolutionsoperatoren ansetzende Integration von problemspezifischem Wissen.

[35] Zum Interessenskonflikt zwischen Leistungsfähigkeit und breiter Anwendbarkeit von EA siehe [Nissen, 1997] S.18ff. und [Angeline, 1993].

Zusammenfassend wird festgestellt, daß für viele praktische Probleme Evolutionsstrategien - aufgrund ihrer Fähigkeit zur Selbstadaptation und Integration von problemspezifischen Expertenwissen - gegenüber GA zu bevorzugen sind.

4.1.5 Berücksichtigung von Randbedingungen in EA

In der ingenieurtechnischen Praxis sind fast alle Optimierungsprobleme durch Randbedingungen beschränkt. Daher besitzt ihre explizite Berücksichtigung bei der Optimierung eine entscheidende Bedeutung. Wie im Kapitel 2 dargestellt, hat ein MOP nur dann gültige Lösungen, wenn alle harten Randbedingungen erfüllt sind und die weichen Randbedingungen möglichst gut eingehalten werden.

Randbedingungen stellen sich i.d.R. dar in Form von: linearen oder nichtlinearen Gleichungen und Ungleichungen, als Ganzzahligkeitsbedingungen und Beschränkungen der Entscheidungsvariablen und Zielfunktionswerten.

Eine sehr effiziente Methode der Berücksichtigung von Randbedingungen besteht darin, das beschränkte Optimierungsproblem bereits vor dem Einsatz des Optimierungsverfahrens in ein unbeschränktes Optimierungsproblem zu konvertieren. Leider ist diese Methode nur für eine geringe Anzahl von Randbedingungen und Optimierungsproblemen geeignet – z.B. bei der Berücksichtigung linearer Gleichungen durch Koordinatentransformation in Form von Variablensubstitution.

EA eignen sich für die Optimierung in diskontinuierlichen und zerklüfteten Suchräumen wie sie sich bei stark beschränkten Optimierungsproblemen darstellen und sind aus diesem Grunde für die Anwendung auf beschränkte Optimierungsprobleme hervorragend geeignet. Es existieren verschiedene Konzepte zur Integration von Randbedingungen und Beschränkungen in Evolutionäre Algorithmen, die insbesondere die Behandlung ungültiger Lösungen bzw. deren Vermeidung betreffen[36].

Berücksichtigung von Randbedingungen durch Straffunktionen: Eine der einfachsten Methoden der Integration von Randbedingungen in EA besteht darin, ungültige Lösungen durch eine Verringerung ihrer Fitness durch den Einsatz einer zusätzlichen Funktion, der sogenannten *penalty function*[37] $p(\vec{a}(\vec{x}))$, zu bestrafen. Auf diese Weise erhalten Individuen, deren Lösungen den Randbedingungen bei gleicher Ausprägung der Zielfunktionswerte nicht genügen, eine geringere Reproduktionswahrscheinlichkeit als Individuen, deren Lösungen die Randbedingungen nicht verletzen. Im Verlaufe der Generationen werden so Lösungen generiert, die gleichzeitig ihre Gütefunktion optimieren und die Straffunktion gegen Null minimieren. Die modifizierte Fitness $\Phi'(\vec{a}(\vec{x}))$ berechnet sich entsprechend:

$$\Phi'(\vec{a}(\vec{x})) = \begin{cases} \Phi(\vec{a}(\vec{x})); & \text{für } \vec{x} \in X_{feasible} \\ \Phi(\vec{a}(\vec{x})) - p(\vec{a}(\vec{x})); & \text{für } \vec{x} \notin X_{feasible} \end{cases} \qquad [31]$$

Die Straffunktion sollte so gestaltet werden, daß mit zunehmenden Grad der Verletzung der Randbedingungen auch der Wert der Straffunktion zunimmt (häufig werden lineare und exponentielle Funktionen verwendet). Sind m Randbedingungen zu berücksichtigen, so setzt

---

[36] Ausführliche Darstellungen der wichtigsten Konzepte zur Berücksichtigung von Randbedingung in EA finden sich in [Michalewicz, 1996], [Fonseca und Fleming, 1998] sowie [Hoffmeister und Sprave, 1996].

[37] Gilt für Minimierungsprobleme: $p(\vec{a}(\vec{x})) = +\infty$ bei $\vec{x} \notin X_{feasible}$ und $p(\vec{a}(\vec{x})) = 0$ bei $\vec{x} \in X_{feasible}$, wird $p(\vec{a}(\vec{x}))$ auch als Barriere-Funktion bezeichnet.

sich die Straffunktion aus m Komponenten $p_i(\vec{a}(\vec{x}))$ zusammen, die zu einem Term aggregiert werden können[38].

Eine effiziente Parametrierung von $p(\vec{a}(\vec{x}))$ ist problemabhängig und wirkt sich sensitiv auf den Erfolg des Optimierungsverfahrens aus. Michalewicz schlägt in [Michalewicz, 1996] vor, die Bestrafung zu Beginn der Optimierung (d.h. bei niedrigen Generationszähler t) weniger stark zu gestalten als zu ihrem Ende und verwendet für die Berechnung der Straffunktion die Gleichung [32].

$$p(\vec{a}(\vec{x}),t) = \frac{1}{2 \cdot \tau_t} \sum_{i=1}^{m} \left[p_i(\vec{a}(\vec{x})) \right]^2 \; ; \; \tau_t \in \Re; \tau_t \geq \tau_{\min} \qquad [32]$$

Empfohlene Werte sind dabei ([Nissen, 1997] S. 83):

$$\tau_o = 1 \;\; \textit{und} \;\; \tau_{t+1} = 0{,}1 \cdot \tau_t \;\; \textit{sowie} \;\; \tau_{\min} = 10^{-5} \qquad [33]$$

Häufig wird durch die Einführung eines multiplikativen Faktors für $p(\vec{a}(\vec{x}))$ in Gleichung [31] die Abhängigkeit der Fitness eines Individuums vom Wert der Zielfunktion(en) oder der Straffunktion(en) beeinflußt. So ist es möglich, gerade bei weichen Randbedingungen deren Einfluß auf die Optimierung zu verändern. Problematisch an der Gestaltung der Straffunktion nach Gleichung [31] ist, daß bei der Berechnung der Fitness Φ' zwei Funktionen unterschiedlicher Kategorien ($\Phi(\vec{a}(\vec{x}))$ und $p(\vec{a}(\vec{x}))$) aggregiert werden. So sind z.B. zwei Individuen $\vec{a}_1$ und $\vec{a}_2$ bezüglich ihrer Fitness Φ' gleichwertig, wenn sie sich um den gleichen Betrag Δ in Φ und p unterscheiden, d.h. für die gilt:

$$\begin{gathered} \Phi(\vec{a}_1) = \Phi(\vec{a}_2) + \Delta \text{ und } p(\vec{a}_1) = p(\vec{a}_2) - \Delta \\ \text{da: } \Phi'(\vec{a}_1) = \Phi(\vec{a}_1) + p(\vec{a}_1) = \Phi(\vec{a}_1) + \Delta + p(\vec{a}_2) - \Delta = \Phi'(\vec{a}_2) \end{gathered} \qquad [34]$$

Ist die Optimierung entsprechend der Gütefunktion und die Minimierung des Verletzungsgrades der Randbedingungen gleichwertig, ist die Integration verschiedenster Randbedingungen nach Gleichung [31] sinnvoll. Allerdings müssen die einzelnen $p_i(\vec{a}(\vec{x}))$ in einen einheitlichen Wertebereich normiert werden. Sind dagegen alle Randbedingungen hart, sind Lösungen nur dann zugelassen, wenn sie allen Randbedingungen entsprechen. In diesem Falle hat die Einhaltung aller Randbedingungen i.d.R. eine höhere Priorität, als die Optimierung der Gütefunktion(en). Der Ansatz nach Gleichung [31] ist dann weniger sinnvoll als eine hierarchische Fitnessfunktion, wie sie z.B. in Gleichung [44] (S.74) oder Gleichung [87] (S. 154) dargestellt ist. Hier ist gewährleistet, daß Individuen, die den Randbedingungen auch in noch so geringem Maße nicht entsprechen, eine geringere Fitness zugeordnet wird, als Individuen, die keine Randbedingung verletzen. Die Optimierung verläuft also zunächst in Richtung $X_{feasible}$ und bewegt sich erst dann auf das Optimum bzw. die pareto-optimalen Lösungen zu.

Berücksichtigung von Randbedingungen in Form zusätzlicher Zielfunktionen: Besonders für die Berücksichtigung *weicher* Randbedingungen eignet sich der Ansatz, diese durch Techniken der Mehrzieloptimierung zu integrieren. Die unterschiedlichen m Randbedingungen erweitern den Vektor der Zielfunktionen $\vec{f}(\vec{x})$ um j Komponenten mit $j \leq m$. Dabei entsprechen die zusätzlichen j Zielkriterien der Optimierung den einzelnen oder zusammengefaßten Straffunktionen der Randbedingungen. Auf diesen Zielfunktionsvektor

[38] So können beispielsweise auch Präferenzen bezüglich der einzelnen Randbedingungen bei der Berechnung von $p(\vec{a}(\vec{x}))$ durch die Einführung von Wichtungsfaktoren berücksichtigt werden.

können die in Abschnitt 5.2 beschriebenen Techniken der Mehrzieloptimierung angewandt werden. Kommen Techniken zum Einsatz, die die Generierung einer Menge nicht-dominierter Zielfunktionswertevektoren ermöglichen, so ist es nach Ablauf der Optimierung möglich, den Einfluß der verschiedenen Randbedingungen auf die Zielfunktionswerte zu analysieren. Nach der Optimierung kann der Anwender interaktiv Lösungen aus der Menge der näherungsweise pareto-optimalen Lösungen auswählen, die die Randbedingungen unterschiedlich stark verletzen sowie den Gütekriterien verschieden gut entsprechen. Auch hier gilt: a-posteriori Techniken der Mehrzieloptimierung (siehe Abschnitt 5.2.2) zur Integration von Randbedingungen sind prinzipiell rechenzeitintensiver, gestatten aber, nach Abschluß der Optimierung unter mehreren Szenarien eine geeignete Lösung auszuwählen. Dieser Ansatz ist dann besonders sinnvoll, wenn ein Konflikt zwischen der Einhaltung der Randbedingungen und der Optimierung der Zielfunktionenen besteht und der Anwender interaktiv die Auswahl der Lösungen beeinflussen möchte.

Berücksichtigung von Randbedingung durch spezielle Operatoren und Kodierungen: Häufig werden harte Randbedingungen jedoch nicht in Form von Straffunktionen oder zusätzlichen Zielfunktionen berücksichtigt. Da nur Lösungen bei der Optimierung zugelassen sind, die allen harten Randbedingungen entsprechen, wird häufig schon die Generierung ungültiger Individuen[39] durch Anpassungen bei der Kodierung (GA,GP) und den Evolutionsoperatoren (ES, reellwertige GA,EP) vermieden. So ist es beispielsweise deutlich effizienter, die Einhaltung des Wertebereichs von Entscheidungsvariablen bei ES nicht über Straffunktionen zu realisieren, sondern bei den Operatoren für die Initialisierung, Reproduktion und Mutation eine Variation der einzelnen Entscheidungsvariablen nur im geforderten Wertebereich zuzulassen und Individuen, deren Zielfunktionswerte sich außerhalb des geforderten Wertebereichs befinden, sofort zu verwerfen.

Während bei ES und reellwertigen GA eine Integration harter Randbedingungen hauptsächlich über die Evolutionsoperatoren erfolgt, wird das anwendungsspezifische Wissen bei binärkodierten GA und GP in die Anpassung der Kodierungs- und Dekodierungsfunktion eingebracht. So können die Werte der Entscheidungsvariablen und Zielfunktionen nur in dem bei der Kodierung festzulegenden Bereich variieren, eine zusätzliche Wertebereichsprüfung während der Optimierung ist also nicht erforderlich.

Die von Z. Michalewicz in [Michalewicz, 1996] vorgestellten reellwertigen Genetischen Algorithmen GENOCOP-I und GENOCOP-II zeigen die Integration unterschiedlicher Randbedingungen nach den oben beschriebenen Methoden. So berücksichtigt GENOCOP-I Randbedingungen in Form von Beschränkungen der Wertebereiche der Entscheidungsvariablen sowie linearer Gleichungen und Ungleichungen. Dabei werden die linearen Gleichungen bereits vor der Optimierung durch Variablensubstitutionen eliminiert. Spezielle Evolutionsoperatoren gewährleisten während der Optimierung die Berücksichtigung der Wertebereichsbeschränkungen sowie linearer Ungleichungen und gewährleisten die Erzeugung stets gültiger Lösungen. In GENOCOP-II werden zusätzlich nichtlineare Gleichungen und Ungleichungen berücksichtigt, die in Form von Straffunktionen in die Berechnung der Fitness integriert wurden ([Nissen, 1994] S. 84).

4.2 Evolutionäre Algorithmen vs. deterministische Optimierungsverfahren

Ein durch die Praxis gegebenes Optimierungsproblem ist in den wenigsten Fällen analytisch zu lösen. Die Ursachen dafür liegen häufig darin, daß sich entweder die Probleme nicht

[39] Siehe dazu Definition 15 S. 44

mathematisch exakt genug beschreiben lassen oder die entstandenen Systeme aus i.d.R. nichtlinearen Gleichungen und Ungleichungen so komplex werden, daß auch diese sich nicht mehr ohne weiteres zu lösen sind. Daher sind für diese Probleme Methoden einzusetzen, die sich iterativ der Optimallösung nähern. Diese Strategien werden als Hill-climbing-Verfahren bezeichnet, da sie sich - ähnlich einem blinden Bergsteiger, der sich vom Tal zum Gipfel herantastet – sukzessive dem Extremum im Suchraum nähern [Schwefel, 1977].

In Abhängigkeit davon, ob bei diesen Näherungsmethoden Elemente des Zufalls verwendet werden, wird zwischen stochastischen und deterministische Suchverfahren unterschieden.

In diesem Abschnitt sollen nur statische nichtlineare Optimierungsstrategien[40] zur Minimierung einer einzelnen Zielfunktion mit $n \geq 1$ Entscheidungsvariablen betrachtet werden. Zu ihnen zählen eine Reihe von verbreiteten Verfahren wie die direkten oder lokalen Suchstrategien sowie Gradientenmethoden, Newton-Verfahren und Quasi-Newton-Verfahren. Vertreter dieser Strategien spielen in der modernen Optimierungspraxis nach wie vor eine große Rolle und sollen daher im Folgenden kurz beschrieben werden. Da es sich bei diesen „klassischen" Optimierungsverfahren jedoch um Verfahren handelt, die i.d.R. der Minimierung bzw. Maximierung einer einzelnen Zielfunktion dienen, ist es für ihre Anwendung auf MOP notwendig, die einzelnen Zielfunktionen geeignet zusammenzufassen (siehe Abschnitt 5.2.1.2).

4.2.1 Direkte Suchstrategien

Direkte Suchstrategien arbeiten ableitungsfrei. Sie verwenden zur Optimierung kein Modell der Zielfunktion, sondern die Zielfunktionswerte selbst und benutzen zur Bestimmung der Suchrichtung und Schrittweiten Heuristiken. Ihre Wirkungsweise basiert auf dem „Versuch-und-Irrtum"-Prinzip [Schwefel, 1977].
Bekannte Beispiele dieser Optimierungsverfahren sind die Gauss-Seidel-Strategie (Koordinatenstrategie, Relaxation), das Verfahren der rotierenden Koordinaten nach Rosenbrock [Rosenbrock, 1960], die Simplex-Strategie nach Nelder und Mead [Nelder und Mead, 1965] oder die Complex-Strategie von Box[41] [Box, 1965]. An dieser Stelle soll kurz auf die Verfahren nach Rosenbrock sowie Nelder und Mead eingegangen werden:

Das Verfahren der rotierenden Koordinaten nach Rosenbrock realisiert eine achsenparallele Suche nach dem Minimum entlang eines im $n-$dimensionalen Raum rotierenden Koordinatensystems. Während der Iteration wird jeweils eine der Koordinatenachsen in die Richtung rotiert, die dem maximalen Fortschritt entspricht und die übrigen $(n-1)-$Achsen orthogonal zu ihr ausgerichtet. Die Entscheidung wann welche Koordinatenachse und in welche Richtung rotiert wird, wird heuristisch ermittelt.

Der Simplex-Strategie nach Nelder und Mead basiert auf der Spiegelung, Expansion und Kontraktion einer geometrischen Figur - dem sogenannten Simplex - die in n Dimensionen $n+1$ Eckpunkte besitzt. Dabei repräsentiert jeder der Eckpunkte j des Simplex eine Lösung $\vec{x}^{\,j}$ des Optimierungsproblems $\min f(\vec{x})$ mit $\vec{x} = [x_1,..,x_n]$, $n>1$. Durch Spiegelung am

[40] Zu den eindimensionalen Optimierungsstrategien zählen u.a. die Eliminationsverfahren wie die „Fibonacci-Teilung" und der „Goldene Schnitt" sowie Interpolationsverfahren wie die „Regula Falsi-Iteration", die „Newton-Raphson-Iteration" sowie die „Lagrange-Interpolation". Einen Überblick über die verschiedenen numerischen Optimierungsverfahren gibt Schwefel in [Schwefel, 1995].

[41] Die Complex-Strategie („constrained Simplex") von Box stellt eine Erweiterung des Simplex-Algorithmus nach Nelder und Mead dar und wurde zur Berücksichtigung von Nebenbedingungen in Form von Ungleichungen entwickelt.

Schwerpunkt des Simplex desjenigen Eckpunktes, der dem Optimalitätskriterium am schlechtesten entspricht, wird eine „Bewegung“ im Suchraum realisiert, durch Kontraktion und Expansion eine Anpassung der jeweiligen Schrittweite. Wird ein lokales Optimum erreicht, kontrahiert der Simplex um die Lösung.

4.2.2 Gradientenverfahren

Gradientenverfahren zählen zu den bekanntesten und am häufigsten eingesetzten deterministischen Optimierungsverfahren. Sie verwenden zur Ermittlung des Minimums den lokalen Gradienten $\nabla f(\vec{x})$ der zu optimierenden Funktion, d.h. die 1. Ableitung der Funktion bezüglich des Parametervektors. Die Suche nach dem Minimum erfolgt in entgegengesetzter Richtung des Gradienten. Gleichung [35] beschreibt die Berechnung von $\vec{x}$ für den Iterationsschritt $(t+1)$ in Abhängigkeit des Gradienten. Dabei definiert der Parameter η die Schrittweite der Iteration und wird problemabhängig gesetzt.

$$\vec{x}_{t+1} = \vec{x}_t - \eta \cdot \nabla f(\vec{x})\big|_{x_t} \qquad \textit{mit:} \quad \nabla f(\vec{x}) = \frac{\partial f(\vec{x})}{\partial \vec{x}} \qquad [35]$$

Gradientenverfahren sind generell nur dann einsetzbar, wenn das zu optimierende Problem stetig und differenzierbar ist. Sie garantieren nur die Konvergenz auf das lokale Minimum in der Nachbarschaft des momentanen Punktes und vermögen es nicht, ein lokales Minimum zugunsten des globalen Minimums zu verlassen. Häufig werden Gradientenverfahren mit einer zufälligen Suche kombiniert, um die Zielfunktionslandschaft mit unterschiedlichen, zufällig gewählten Anfangsbedingungen zu durchsuchen, was jedoch auch hier das Auffinden des globalen Minimums nicht garantiert. Einen Überblick über die zahlreichen Gradientenverfahren wird in [Numerical Recipies, 1994] und [Luenberger, 1984] gegeben. An dieser Stelle seien als bekannte Vertreter der Gradientenverfahren der Levenberg-Marquardt-Algorithmus[42] und das Verfahren der konjugierten Gradienten genannt.

Der Levenberg-Marquardt-Algorithmus stellt eine effiziente Erweiterung des Newton-Verfahrens zur Minimierung von Quadratsummen dar und hat sich heute als Standardverfahren für Optimierungsprobleme mit einer geringen Anzahl von Entscheidungsvariablen, insbesondere bei der nichtlinearen Modellbildung, durchgesetzt [Numerical Recipies, 1994].

Bei dem Verfahren der konjugierten Gradienten werden nicht nur die Gradienten, sondern zudem die Hessematrix verwendet, d.h. nicht nur Approximationen der 1. sondern auch der 2. Ableitung der Zielfunktionen. Es ist damit Standard-Gradienten-Verfahren bezüglich der Konvergenzgeschwindigkeit überlegen. Durch Auswertung bereits genommener Suchschritte werden bei der Methode der konjugierten Gradienten die häufigen Wechsel der Suchrichtung, die bei der Verwendung eines Standardgradientenverfahrens entstehen, vermieden. Die neuen Schrittrichtungen werden bei diesem Verfahren unter Berücksichtigung des Gradienten der vorhergehenden Schritte[43] bestimmt, was zu einer erheblichen Konvergenzsteigerung führt. Es existieren verschiedene Verfahren, die konjugierten Richtungen zu berechnen; als Beispiel soll an dieser Stelle das Verfahren von Powell[44] genannt und auf die Einführung in [Shewchuk, 1994] verwiesen werden.

---

[42] Einige Autoren ordnen den LM-Algorithmus den Quasi-Newton-Verfahren zu.

[43] Ein Spezialfall der konjugierten Richtungen ist die Methode des steilsten Anstieges, wobei der Korrekturfaktor zur Berücksichtigung bereits genommenener Suchrichtungen nicht berücksichtigt wird.

[44] Powell gibt in [Powell, 1962] eine Möglichkeit an, die konjugierten Richtungen auch ohne Berechnung der 2. Ableitungen zu ermitteln [Schwefel, 1995].

4.2.3 Quasi-Newton-Verfahren

Quasi-Newton-Verfahren sind deterministische Optimierungsverfahren, die in der klassischen numerischen Mathematik insbesondere zur Lösung von Optimierungsproblemen mit einer mittleren bis hohen Anzahl von Entscheidungsvariablen angewandt werden. Sie basieren darauf, im aktuellen Startpunkt mit Hilfe des Taylor-Reihen-Ansatzes 2. Ordnung ein quadratisches Modell der Zielfunktion zu konstruieren und durch Nullsetzen des Gradienten dieser Modellfunktion das Minimum zu berechnen. Zur Berechnung der Suchrichtungen werden sowohl die ersten partiellen Ableitungen der Zielfunktionen als auch Approximationen der inversen Hessematrix verwendet.

Bekannte Beispiele für Quasi-Newton-Strategien sind die Verfahren nach Broydon-Fletcher-Goldfarb-Shanno (BFGS) [Fletcher_1987] und nach Davidon-Fletcher-Powell (DFP) [Fletcher und Powell, 1963].

4.2.4 Vergleich deterministischer und evolutionärer Optimierungsverfahren

Werden Evolutionäre Algorithmen und deterministische Suchverfahren - insbesondere ableitungsbasierte Strategien – gegenübergestellt, sind einige gravierende Unterschiede, jedoch auch Gemeinsamkeiten feststellbar. Die große Stärke deterministischer Suchverfahren liegt in ihrer im Vergleich zu EA sehr hohen Konvergenzgeschwindigkeit auf das lokale Minimum; ihr entscheidender Nachteil jedoch in ihrer Eigenschaft, i.d.R. auf das dem Startpunkt näheste lokale Minimum zu konvergieren. Versagen müssen deterministische Suchverfahren bei unstetigen, nicht differenzierbaren Zielfunktionen.

Daher hat trotz der Existenz effizienter deterministischer Optimierungsalgorithmen die Suche nach Verfahren, die auch für die schwierigsten Problemklassen (z.B. unstetige, multimodale Zielfunktionen) anwendbar sind, nicht aufgehört. Probabilistische Suchverfahren vergrößern durch die Einbeziehung von Elementen des Zufalls die Wahrscheinlichkeit, das globale Optimum zu finden – Evolutionäre Algorithmen vermögen dazu zusätzlich, den Lösungsraum in mehreren Richtungen massiv parallel zu durchsuchen. Allerdings können sie das Auffinden des globalen Minimums nicht garantieren.

Bei Gradientenverfahren gilt bei zunehmender Annäherung an das (lokale) Minimum: $\Delta\vec{x} \to 0$. Aber auch einige Evolutionäre Algorithmen (z.B. ES) verfügen mit Hilfe der Anpassung der Mutationsschrittweiten ihrer Entscheidungsvariablen über einen ganz ähnlichen Mechanismus. Sie sind ohne die Verwendung von Gradienteninformationen in der Lage, durch eine Anpassung der Schrittweitenänderungen der Entscheidungsvariablen eine Annäherung an ein lokales Minimum zu realisieren und sein Überspringen zu verhindern. Während noch in [Rechenberg, 1973] dargestellt wird, daß ES i.d.R. ebenso wie ableitungsbasierte, deterministische Suchverfahren dem Gradienten folgen, wurde in [Salomon, 1998] gezeigt, daß dies nur für Probleme mit einer geringen Anzahl von Entscheidungsvariablen und Evolutionsstrategien mit einer sehr großen Anzahl von Nachkommen gilt.

Es existieren eine Reihe von Ansätzen, die EA mit deterministischen Optimierungsverfahren kombinieren, um die Nachteile der einen Methode durch die Vorteile der jeweils anderen auszugleichen und vice versa (siehe dazu auch Abschnitt 5.3). Anwendungen, die die Individuen einer Population jeder Generation lediglich als Startpunkte für lokale deterministische Suchstrategien verwenden, wie sie sich beispielsweise in [Bremer, 1998] findet, erzielen zwar sehr schnell konvergierende Optimierungsalgorithmen, geben dem EA jedoch nicht die Chance, seine eigentlichen Stärken der Selbstanpassung zu entwickeln und unterliegen dem Risiko vorzeitiger Konvergenz in lokalen Minima. Daher kommen solche Ansätze eher einem mit zufällig variierten Startpunkten vielfach wiederholten deterministischen Suchverfahren als einem beschleunigten EA nahe. Sinnvoll ist dagegen, nach Konvergenz eines EA das bis zu einem definierten Konvergenzkriterium berechnete beste Individuum als Startpunkt für ein

ableitungsbasiertes Suchverfahren zu verwenden. Hier kann davon ausgegangen werden, daß die Wahrscheinlichkeit sehr hoch ist, daß sich der Startpunkt in der Nähe des globalen Minimums befindet und der Nachteil von EA, zum Ende der Iteration schlechte Finetuning-Eigenschaften zu besitzen, durch die Konvergenzeigenschaften ableitungsbasierter Verfahren kompensiert werden.

Vergleicht man die erwähnten, klassischen, deterministische Suchverfahren mit EA unter dem Aspekt ihrer Anwendbarkeit für multikriterielle Optimierungsprobleme insbesondere mit Zielfunktionskonflikten, so sind EA in der Lage, auch bei einmaliger Anwendung eine *Menge* von Lösungen zu produzieren. Durch spezielle Techniken (siehe Abschnitt 5.2) kann diese Lösungsmenge der Menge der pareto-optimalen Lösungen des Optimierungsproblems genähert werden. Sie sind daher für MOP mit Zielfunktionskonflikten besonders geeignet. Deterministische Suchverfahren verfügen hingegen nicht über diese Eigenschaft da sie i.d.R. für Probleme mit nur einer Zielfunktion konzipiert sind. Hier wird bei jedem Optimierungslauf genau eine Lösung generiert; um ein MOP zu lösen, müssen die einzelnen Zielfunktionen daher geeignet zusammengefaßt und der Optimierungsalgorithmus erneut durchlaufen werden. Durch Aggregation jedoch effizient die Menge pareto-optimaler Lösungen zu generieren ist u.U. sehr schwierig und unübersichtlich (zu den Nachteilen und Schwächen von Aggregationsansätzen siehe Abschnitt 5.2.1.2).

Zusammenfassend können die folgenden Charakteristiken[45] zur Abgrenzung Evolutionärer Algorithmen von klassischen Optimierungsverfahren genannt werden:

1. Die Problemlösungen werden entweder direkt (ES,EP) oder kodiert (GA) durch die Entscheidungsvariablen, Zielfunktionswerte sowie etwaiger interner Strategieparameter in Form von Individuen repräsentiert.
2. EA verwenden Prinzipien der natürlichen Evolution in Form der Evolutionsoperatoren (Mutation, Rekombination und Selektion), die auf einer Menge von Lösungsalternativen (Population(en)) aufsetzen. Auf diese Weise kann der Zielraum parallel in mehreren Suchrichtungen durchschritten werden. Damit wird die Wahrscheinlichkeit, sich in lokalen Minima zu fangen, drastisch verringert. Die am Ende der Iteration entstehenden Lösungen können im Suchraum über mehrere lokale Optima ähnlicher Güte verstreut sein oder sich um ein globales Optimum mit deutlich höherer Güte konzentrieren.
3. Als Selektionskriterium sind lediglich Informationen über die Güte der aktuellen Lösungen (Individuen), jedoch keine abgeleiteten Größen (z.B. Gradienten etc.) notwendig. EA stellen keine Forderungen wie Stetigkeit oder Differenzierbarkeit an die Zielfunktionen.
4. Es werden bei den Evolutionsoperatoren Elemente des Zufalls berücksichtigt. Diese realisieren jedoch im Gegensatz zum „Random Search“ keine blinde stochastische Suche, sondern konzentrieren sich sukzessive auf die Bereiche des Suchraumes, die den Gütekriterien am besten entsprechen.

Evolutionäre Algorithmen werden von ihren Kritikern häufig mit einer reinen stochastischen Suche gleichgesetzt und als „ernsthafte“ Lösungsmethode angezweifelt. Die Stärken von EA im Bereich der Selbstanpassung sowie im Zusammenspiel zwischen Variation und Selektion werden dabei jedoch oft vernachlässigt. Die Ursachen dieser Zweifel liegen sicher in dem stark heuristischen Charakter von EA begründet: EA verwenden zwingend Elemente des Zufalls – es ist daher nicht exakt berechenbar, wie sich die Individuen im einzelnen im Suchraum während er Iteration bewegen. Durch die Mechanismen von Selektion, Rekombination und Variation ist es nicht, wie beispielsweise bei klassischen Gradientenverfahren, mathematisch exakt vorhersagbar, wie die Iteration im einzelnen verläuft.

[45] Vergl. Dazu auch [Nissen, 1994] S. 13 ff.

Evolutionäre Algorithmen sind daher trotz ihrer großen Popularität nicht unumstritten. Obwohl sie sich seit den 1990er-Jahren auch in der industriellen Praxis etabliert haben, kritisieren ihre Gegner auch heute noch hauptsächlich ihre fehlende Opimimalitätsgarantie. Gerade bei Problemen, bei denen klassische Methoden des Operation Research versagen oder aufwendig angepaßt werden müssen, überzeugen EA jedoch durch ihre Einfachheit und Fähigkeit, auch in hochkomplexen Suchräumen Lösungen zu finden. So können heute auch für mathematisch nicht formulierbare Zielfunktionsgebirge mit akzeptablem Aufwand gute Näherungslösungen gefunden werden.

4.3 Vor- und Nachteile Evolutionärer Algorithmen

Zusammenfassend sollen im Folgenden stichpunktartig die Vor- und Nachteile Evolutionärer Algorithmen zusammengefaßt werden:

Vorteile Evolutionärer Algorithmen:

1. Einfachheit und Universalität: EA besitzen leicht verständliche Grundprinzipien, sind in ihren Basisalgorithmen leicht zu implementieren und können sukzessive an die spezielle Problematik angepaßt werden. EA sind auf ein sehr breites Spektrum von Problemklassen anwendbar.
2. Breite Anwendbarkeit auf komplexe Suchräume: EA können als ableitungsfreies Verfahren auch auf mathematisch nicht formulierbare Zielfunktionsgebirge angewandt werden und sind für die Optimierung in komplexen Suchräumen - wie z.B. unstetige, nichtdifferenzierbare, nichtlineare und stark zerklüftete Zielfunktionen - geeignet. Die Integration verschiedenster Zielfunktionen und Randbedingungen gestaltet sich einfach. Vorwissen ist nicht notwendig, kann aber einbezogen werden. Da EA bei nichtelitärer Selektion eine Verschlechterung der Fitness der Individuen toleriert, ist es möglich, auch lokale Optima zu verlassen und so das globale Optimum aufzufinden.
3. Möglichkeit multipler Lösungen: EA sind für multimodale Zielfunktionen geeignet und können auch bei unimodalen Zielfunktionen für einen oder mehrere Zielfunktionswerte mehrere Kombinationen der Entscheidungsvariablen generieren.
4. Anytime-Eigenschaft: Zu jedem Zeitpunkt der Iteration ist eine Lösung abrufbar, die nicht schlechter als eine eventuell vorhandene Startlösung ist, d.h. es ist i.d.R. gewährleistet, daß eine Ergebnislösung generiert wird. Dies setzt jedoch voraus, daß die jeweils besten Lösungen archiviert werden.
5. Leichtes Einbringen problemspezifischen Wissens: Durch die Anpassung der Evolutionsoperatoren und eventuellen Kodierungen kann sehr einfach und übersichtlich problemspezifisches Wissen in den Suchprozess integriert werden. Es ist kein spezielles Vorwissen über das zu lösende Problem notwendig.
6. Gute Parallelisierbarkeit: EA sind massiv parallelisierbar – aufgrund dieser Eigenschaft kann mit entsprechender Hardware eine erhebliche Steigerung ihrer Performance erreicht werden.
7. Einfache Hybridisierung mit anderen Optimierungsverfahren: EA können sehr einfach mit anderen Optimimierungsverfahren kombiniert werden. So ist es beispielsweise in der Initialisierungsphase oft sinnvoll, spezielle Heuristiken einzusetzen. Oft werden EA auch mit Gradientenverfahren kombiniert, die am Ende der Evolutionären Suche ein schnelles Finden eines lokalen Optimums gewährleistet.

Nachteile Evolutionärer Algorithmen:

1. Blackbox-Verhalten: Evolutionäre Algorithmen sind ähnlich wie Neuronale Netze sogenannte Black-Box-Methoden, d.h. der Anwender ist nicht in der Lage, die Berechnung der Lösung in jedem Schritt zu reproduzieren. Gerade diese Eigenschaft wird von den Gegnern der EA häufig als Kritikpunkt benutzt.
2. Fehlende Optimalitätsgarantie: Evolutionäre Algorithmen besitzen einen stark heuristischen Charakter und können aufgrund ihres stochastischen Grundprinzips das Auffinden des theoretisch möglichen Optimums in einer begrenzten Zeit nicht garantieren.
3. Hoher Rechenaufwand: Evolutionäre Algorithmen erfordern durch ihr Populationskonzept einen relativ hohen Rechenaufwand. Diese Tatsache machte sie in den 1970er und 1980er Jahren gegenüber Alternativmethoden für die breite praktische Anwendbarkeit relativ uninteressant. Mit der sich rasch entwickelnden Rechentechnik relativieren sich jedoch diese Probleme und spielen heute bei der Wahl eines Optimierungsalgorithmus keine vorrangige Rolle mehr[46].
4. Anfälligkeit gegenüber einer ineffizienten Parametrierung des Algorithmus: Die Effizienz von EA hängt in nicht geringem Maße von der richtigen Wahl seiner Strategieparameter (Anzahl der Individuen, Kinder, Migrationskonzept, Steuerung der Mutationsschrittweite usw.) ab, für die jedoch in der Literatur Empfehlungen gegeben werden.
5. Ineffizienz bei Finetuning: Gerade zum Ende der Optimierung, wenn die Lösungen sich bereits in einer gewissen Nähe zum Optimum befinden, sind EA klassischen Verfahren unterlegen. Aus diesem Grunde werden sie sehr oft mit Gradientenabstiegsverfahren (z.B. Levenberg-Marquardt) kombiniert (siehe Abschnitt 4.2.4).

Aufgrund der oben beschriebenen Vor-und Nachteile können Evolutionäre Algorithmen in folgenden Anwendungsbereichen besonders effizient eingesetzt werden[47]: bei Problemen

- hoher Komplexität, für die keine speziellen Lösungsalgorithmen bestehen,
- die einer hohen Dynamik unterliegen, d.h. deren Problemstruktur und Parameter sich oft ändern sowie
- bei Problemen, die starken stochastischen Einflüssen z.B. bei der Lösungsbewertung ausgesetzt sind.

Dagegen sind Evolutionäre Algorithmen als Optimierungsverfahren ungeeignet wenn

- alternative, spezielle Lösungsverfahren vorhanden sind,
- das Auffinden des globalen Optimums garantiert werden muß oder
- durch eine sehr aufwendige Zielfunktionsberechnung der Rechenaufwand zur Lösung des Problems auf ein nicht mehr tolerierbares Maß steigt.

Natürlich liegt es im Interesse eines jeden Anwenders ein Optimierungswerkzeug zu besitzen, das ohne großen Aufwand an die verschiedenen Problemklassen anpaßbar ist, garantiert mit wenig Aufwand die optimalen Lösungen findet und anderen Verfahren in jeder Hinsicht überlegen ist. Einen solchen Algorithmus kann es aufgrund des Konfliktes zwischen Allgemeingültigkeit und Effizienz eines Verfahrens nicht geben – und auch Evolutionäre

[46] Natürlich gibt es im Echtzeitbereich (z.B. embedded systems bei der Kennfeldoptimierung in der Fahrzeugindustrie) Anwendungen, bei denen Laufzeiten eine sehr große Rolle spielen.

[47] In [Nissen, 1997] wird ein ausführlicher Einblick in die Anwendungsbereiche EA gegeben.

Algorithmen können trotz der erwähnten Vorteile dem Anspruch nicht genügen, ein „ideales Optimierungswerkzeug“ zu sein.

In diesem Zusammenhang soll das 1995 von Wolpert und Macready aufgestellte sogenannte „No-Free-Lunch-Theorem“ [Wolpert und Macready, 1995] erwähnt werden, aus dem man theoretisch herleiten kann, inwieweit EA tatsächlich konkurrierenden Optimierungsverfahren überlegen sind.

Das Theorem bezieht sich auf deterministische Suchalgorithmen, die kein spezielles Problemwissen berücksichtigen. Nach Wolpert und Macready können die Ergebnisse jedoch auch unmittelbar auf stochastische Suchalgorithmen wie EA angewendet werden. Das Theorem besagt, daß stochastische Suchverfahren summierend über die verschiedensten (gleichverteilten) Optimierungsprobleme die gleiche Performance besitzen d.h. immer Optimierungsprobleme existieren, für die definierte stochastische Algorithmen besonders gut und solche für die sie weniger gut geeignet sind.

Mit diesem Theorem ist bewiesen worden:

- daß es ist nicht möglich ist, einen für alle Problemklassen in der Performance gleich guten stochastischen Optimierungsalgorithmus zu entwickeln,
- daß aus den Resultaten, die mit einem Algorithmus für eine Problemklasse erzielt wurden, nicht auf die Performance des Algorithmus bezüglich einer anderen Problemklasse geschlossen werden kann,
- sich stochastische Suchverfahren (und damit auch EA) daher auf Problemklassen spezialisieren müssen[48] und
- daß es unmittelbar erforderlich ist, die charakteristischen Merkmale des zu lösenden Problems zu identifizieren, auf deren Grundlage eine Problemklassifizierung und schließlich eine Wahl des geeignetsten Lösungsalgorithmus vorzunehmen ist.

In der Literatur löste dieses Theorem eine ganze Reihe von Diskussionen und Kritiken aus. So ist beispielsweise die unterstellte Gleichverteilung der verschiedenen Optimierungsprobleme in der Natur und Technik nicht gegeben. Auch werden nur Aussagen über die Wahrscheinlichkeit des Erreichens einer bestimmten Qualität des Algorithmus‘ gemacht – Aspekte, die seine Effizienz (beispielsweise CPU-Laufzeiten) betreffen, werden jedoch nicht berücksichtigt. Gerade diese Aussagen sind aber u.U. für den Praktiker von großer Bedeutung.

Die Schlußfolgerung, daß sich Optimierungsalgorithmen auf Problemklassen spezialisieren müssen, ist durch die Entwicklung der EA bestätigt worden. Durch zahlreiche technische Anwendungen von EA hat sich bestätigt, daß das Einbringen von problemspezifischen Expertenwissen in Form von speziellen Kodierungen (GA) bzw. Operatoren für die Initialisierung, Rekombination und Mutation (ES) für ein effizientes Anwenden der EA-Theorie notwendig ist.

[48] Diese Entwicklung hat sich beispielsweise auch auf dem Gebiet der Neuronalen Netze vollzogen. Hier wurden nach anfänglicher Euphorie - z.B. ein universelles trainierbares Klassifikationssystem zu besitzen – problemklassenangepaßte, spezialisierten Netzstrukturen und Lernalgorithmen entwickelt (SOM, MLP, RBF, TDNN..)

5 Multikriterielle Evolutionäre Algorithmen (MOEA)

Wie in Kapitel 2 dargestellt ist es für die Lösung eines MOP´s nicht ausreichend, eine einzelne Lösung zu generieren. Vielmehr ist es notwendig, eine Menge pareto-optimaler Lösungen zu erzeugen, unter denen der Anwender dann aufgrund von Präferenzen *seine* Lösung auswählt. Algorithmen wie die EA, die aufgrund ihres Wirkprinzips auf einer Menge von Lösungen agieren, eignen sich für die Lösung eines solchen Problems besonders. Aber nicht nur die Anwendung des Populationsprinzips, d.h. der massiv parallelen Suche, machen EA für die Lösung von MOP´s interessant, sondern auch ihre wie in Kapitel 4 dargestellte Fähigkeit, sich auch in stark zerklüfteten Zielfunktionslandschaften nicht in lokalen Minima zu fangen. Diese Eigenschaften zusammen mit der breiten Anwendbarkeit von EA und ihres einfachen aber leistungsfähigen Grundprinzips machen sie zu hervorragenden Kandidaten zu Lösung multikriterieller Optimierungsprobleme.

A priori realisiert ein Standard-EA jedoch die Maximierung der Fitness seiner Individuen, was einer Optimierung mit nur einer Zielfunktion entspricht. Es ist daher notwendig, ihn so anzupassen, daß er für die Lösung eines MOP´s geeignet ist. Ein so angepaßter EA wird als multikriterieller Evolutionärer Algorithmus (MOEA, multiobjective EA) bezeichnet. Die speziellen Techniken werden in den folgenden Abschnitten systematisch dargestellt. Neben der Darstellung dieser grundlegenden Prinzipien von MOEA werden in diesem Kapitel ebenso die bekanntesten Ansätze zur Realisierung MOEA überblicksartig beschrieben und ihre Vor- und Nachteile dargestellt. Eine ausführliche und für den tieferen Einblick in das Thema MOEA empfehlenswerte Übersicht gibt K. Deb in [Deb, 2002].

Während der 1990er Jahre hat sich die Entwicklung der verschiedensten MOEA sprunghaft beschleunigt. Während bei den MOEA der ersten Generation (z.B. MOGA, NSGA, NPGA und NPGA2, siehe unten) der Schwerpunkt auf der Entwicklung pareto-optimaler Selektionsmechanismen lag, führte bei den Algorithmen der zweiten Generation (z.B. PAES, PESA, SPEA, SPEA2 und NSGA2, siehe unten) die Entwicklung spezieller Selektions- und Archivierungsmechanismen zu einer Steigerung der Konvergenzgeschwindigkeit und einer deutlich effizienteren Ausprägung von Lösungen, die nicht nur dominant sind, sondern auch effiziente Zielfunktionswertevektoren bestimmen, die breit und homogen über den Zielraum verteilt sind. Dies wurde vor allem durch die konsequente Anwendung des Elite-Prinzips in Form des Speicherns generierter nicht-dominierter Individuen in einer zusätzlichen Population und hierfür geeigneter Archivierungsmechanismen erreicht.

Es können an dieser Stelle nicht alle existierenden MOEA-Ansätze[49] beschrieben werden. Vielmehr soll die Darstellung auf die bekanntesten MOEA beschränkt bleiben, die dann aus Gründen der Vergleichbarkeit in die Performance-Tests dieser Arbeit aufgenommen wurden. Für weitere MOEA – die i.d.R. Erweiterungen der hier vorgestellten Ansätze darstellen - soll daher auf die entsprechenden Literaturstellen (z.B. die Kurzanalyse von Zitzler in [Zitzler und Thiele, 1998] bzw. die ausführliche Darstellung in [Deb, 2002]) verwiesen werden.

Um die in großer Anzahl entwickelten unterschiedlichen Lösungsansätze und Algorithmen vergleichen zu können, wurde es nötig, sowohl spezielle Testprobleme (siehe Abschnitt 7.4) als auch Vergleichsmetriken (siehe Abschnitt 7.1) zu entwickeln. Die in dieser Arbeit durchgeführten Simulationen orientieren sich an diesen Vorgaben und gestatten so einen objektiven Vergleich mit den bisher bekannten MOEA.

[49] Weitere Ansätze finden sich u.a. bei: Greenwood, Hu und D'Ambrosio in [Greenwood et al, 1996], Lis und Eiben in [Lis und Eiben, 1997] sowie Cunha, Oliviera und Covas in [Cunha et al, 1997].

In dieser Arbeit wurde ein hybrider MOEA entwickelt. Daher soll neben der Darstellung der wichtigsten Prinzipien von MOEA ein kurzer Überblick über die bekannten Möglichkeiten ihrer Hybridisierung gegeben werden (Abschnitte 5.3 und 5.4.3). Da es sich hierbei i.d.R. um anwendungsbezogene Arbeiten handelt, lagen zum aktuellen Zeitpunkt keine Daten bezüglich ihrer Leistungsfähigkeit in Hinblick auf die in Abschnitt 7.4 verwendeten Testprobleme vor. Aus diesem Grunde konnten diese Ansätze nicht für die in Kapitel 7 dargestellten Benchmarks verwendet werden.

5.1 Ausgewählte Definitionen für MOEA

Im Folgenden werden einige Begriffe eingeführt, die sich auf die Individuen eines MOEA zur Lösung eines MOP´s nach Definition 1 beziehen. Entsprechend der in Kapitel 2 gegebenen Definitionen seien die Begriffe gültiges, dominantes sowie nicht-dominiertes Individuum gegeben.

Definition 15: gültiges Individuum

Ein Individuum $\vec{a}$, wird genau dann als gültig (zulässig) bezeichnet, wenn die durch ihn repräsentierte Lösung $\vec{x}$ gültig ist, d.h. $\vec{x} \in X_{feasible}$.

Definition 16: dominantes Individuum

Ein Individuum $\vec{a}^1$, wird genau dann als streng dominant gegenüber dem Individuum $\vec{a}^2$ bezeichnet, wenn die durch $\vec{a}^1$ repräsentierte Lösung $\vec{x}^1$ streng dominant gegenüber der durch $\vec{a}^2$ repräsentierten Lösung $\vec{x}^2$ ist.

Analog ist ein Individuum $\vec{a}^1$ genau dann schwach dominant gegenüber dem Individuum $\vec{a}^2$, wenn die durch $\vec{a}^1$ repräsentierte Lösung $\vec{x}^1$ schwach dominant gegenüber der durch $\vec{a}^2$ repräsentierten Lösung $\vec{x}^2$ ist.

Die Individuen $\vec{a}^1$ und $\vec{a}^2$ sind indifferent bezüglich der Pareto-Dominanz, wenn auch die durch sie repräsentierten Lösungen indifferent sind.

Definition 17: nicht-dominiertes Individuum

Ein Individuum $\vec{a}$, wird genau dann als nicht-dominiert gegenüber den Individuen einer Population P bezeichnet, wenn $\vec{a}$ von keinem Individuum aus P stark dominiert wird.

Die Menge aller nicht-dominierten, zulässigen Lösungen, die ein MOEA nach t Generationen liefert, wird im Folgenden mit $P^*(t)$ bezeichnet und ist eine Approximation der globalen Pareto-Menge des zu lösenden MOP´s. Die durch $P^*(t)$ bestimmte Menge von nicht-dominierten, zulässigen Zielfunktionswerten wird mit $PF^*(t)$ bezeichnet und ist eine Approximation der Pareto-Front des MOP´s.

Nach Abschluß der Optimierung wird die durch den MOEA generierte Approximation der Pareto-Menge mit P^* und die durch sie bestimmte Approximation der Pareto-Front mit PF^* bezeichnet[50].

Das Ziel bei der Konstruktion eines MOEA besteht darin, P^* möglichst gut der Pareto-Menge und PF^* möglichst gut der Pareto-Front zu nähern, d.h. folgenden Terme zu erreichen: $P^* = PF_{true}$ und $PF^* = PF_{true}$.

5.2 Spezielle Techniken zur Realisierung von MOEA

In diesem Abschnitt wird auf die speziellen Erweiterungen im Ablaufschema Evolutionärer Algorithmen eingegangen, die Standardformen von EA in die Lage versetzen, multikriterielle Optimierungsprobleme zu lösen.

Seit den 1990er Jahren wurde mehrere Arbeiten veröffentlicht, die die bestehenden Techniken der MOEA analysieren und vergleichen[51]. Da an dieser Stelle keine vollständige Beschreibung aller existierenden MOEA-Ansätze, sondern nur eine kurze Charakterisierung seiner bekanntesten Vertreter gegeben werden kann, soll für deren ausführlichere Darstellung auf die folgenden Quellen verwiesen werden: [Fonseca und Fleming, 1995], [Tamaki et al, 1996], [Coello, 1996], [Horn, 1997], [Zitzler und Thiele, 1998], [Veldhuizen und Lamont, 2000] und [Deb, 2002][52].

EA benötigen ein Gütekriterium zur Bewertung und Selektion der einzelnen Individuen $\vec{a}$, d.h. eine Fitnessfunktion $\Phi(\vec{a})$. Besteht das Optimierungsziel darin, nur eine Zielfunktion $f(\vec{x})$ zu minimieren, wird die Fitness direkt durch den Zielfunktionswert, den das Individuum repräsentiert, berechnet.

Wie bereits in Abschnitt 1 beschrieben, sind Optimierungsprobleme aus der technischen Praxis jedoch häufig mehrdimensional und beinhalten konkurrierende Ziele, d.h. es ist das Auffinden von Kompromißlösungen zwischen den einzelnen Optimierungszielen $f_i(\vec{x})$ gefordert. Zur Lösung eines MOP´s sind die Standardformen Evolutionärer Algorithmen also so anzupassen, daß eine simultane Optimierung der einzelnen Komponenten der Fitnessfunktion der Individuen $f_i(\vec{x})$ möglich wird. Die Techniken, die einen Standard-EA zu einem MOEA machen, betreffen

- die Berechnung der Fitness: Hierarchische Optimierungsmethoden (S.48), Aggregationsansätze (S.49), Pareto-basiertes Ranking (S.54),
- die Berücksichtigung der „Ähnlichkeit“ von Lösungen: Nischentechniken (S.52),
- die Selektionsmechanismen von Individuen: Selektion mit wechselnden Zielen (S.51)
- und das Populationskonzept selbst: 5.2.2.5 (S.55)

Standardformen Evolutionärer Algorithmen führen i.d.R. zur Herausbildung einer Population von Individuen in der Nähe nur *eines* Optimums. Dieser Effekt wird als „Genetische Drift“ bezeichnet, der seine Ursache im Selektionsdruck (Lösungen mit kleiner Fitness verschwinden),

---

[50] Siehe auch Definition 12, S.10 und Definition 13, S.10.

[51] Eine ständig aktualisierte Übersicht über die bestehenden MOEA-Aufsätze findet sich im Internet unter der Adresse: http://www.lania.mx/~ccoello/EMOO/EMOObib.html. Im Frühjahr 2003 betrug die Anzahl der Puplikationen zum Thema MOEA ca. 1000.

[52] An dieser Stelle sei gleichzeitig auf die Überblicksaufsätze zur Mehrzieloptimierung von Rosenthal [Rosenthal, 1985] und Korhonen, Moskowitz und Wallenius [Korhonen et al, 1992] verwiesen.

Selektionsrauschen (äquivalente Lösungen verschwinden durch Zufallsauswahl) und der oft vorhandenen Unstetigkeit der Evolutionsoperatoren (Zerstörung guter Lösungen durch crossover) hat [Zitzler und Thiele, 1998]. EA sind aber nur dann in der Lage, den gesamten Raum der Zielfunktionsvektoren effizient parallel zu durchsuchen, wenn die Populationsmitglieder eine gewisse Diversität (*diversity)* aufweisen. Insbesondere bei multimodalen und mehrdimensionalen Zielfunktionen ist die Herausbildung der Verschiedenartigkeit der Lösungen notwendig, um tatsächlich entweder die einzelnen lokalen Optima oder aber die pareto-optimalen Lösungen zu finden.

Die Wahl der Kompromißlösungen zwischen den einzelnen Optimierungszielen durch den Anwender wird durch einen sogenannten *Decision-Maker* (DM) realisiert. In ihm werden die Präferenzen, Zielgewichtungen oder Prioritäten des Anwenders definiert, eine interaktive Beeinflussung des Optimierungsalgorithmus realisiert oder eine definierte Auswahl einer Lösung aus der berechneten Trade-off-Menge ermöglicht.

Im trivialen Fall ist der Anwender selbst der DM, der entweder vorab eine Gewichtung oder Priorisierung der einzelnen $f_i(\vec{x})$ vornimmt, während des Optimierungsalgorithmus durch eine geeignete Rückkopplung die Suchrichtung verändert und so interaktiv in den Optimierungsverlauf eingreifen kann oder aber aus einer vom Optimierungsalgorithmus generierten Lösungsmenge nach Durchlaufen des Optimierungsalgorithmus eine Lösung auswählt.

Um eine vom Anwender akzeptierte Lösung, d.h. einen effizienten Kompromiß zwischen eventuell existierenden Zielkonflikten eines MOP zu erhalten, ist also nicht allein der Optimierungsalgorithmus als solcher wichtig. Entscheidend für die Güte einer Lösung ist ebenso die Effizienz des DM. Aus Sicht des Decision-Makers (DM) unterscheiden Hwang und Masud in [Hwang und Masud, 1979], Horn in [Horn, 1997] sowie D.A. Veldhuizen und G.B. Lamont in [Veldhuizen und Lamont, 2000] drei verschiedene Lösungsstrategien für MOP:

a priori–Techniken: Hier werden bereits ***vor*** Beginn des Optimierungsalgorithmus die einzelnen Zielfunktionen mittels einer geeigneten skalaren Aggregation zu einer einzelnen Zielfunktion zusammengefaßt. Dieser Ansatz wird auch bevorzugt bei nichtevolutionären Optimierungsverfahren eingesetzt. So kann jedes beliebige multikriterielle Optimierungsproblem in ein konventionelles Optimierungsproblem mit nur einer Zielfunktion konvertiert werden. Als Ergebnis wird genau eine Lösung generiert. Traditionell wird diese Methode am häufigsten eingesetzt.

progressive Techniken: Bei dieser Strategie erfolgt eine Interaktion zwischen dem DM und dem Optimierungsalgorithmus ***während*** des Optimierungslaufes selbst. So ist es möglich, die Suchrichtung im Lösungsraum interaktiv zu beeinflussen. Dieser Ansatz erfordert jedoch sowohl ein gewisses problemspezifisches Vorwissen als auch eine Überwachung bzw. Steuerung durch den Anwender während der Optimierung.

a posteriori–Techniken: Bei dieser Lösungsstrategie wird durch den Optimierungsalgorithmus ohne die Einbeziehung von Vorwissen eine Menge Lösungen generiert, aus der dann ***nach*** dem Optimierungsdurchlauf vom Anwender mit Hilfe des DM eine Lösung ausgewählt wird.

Während diese Systematik den Zeitpunkt des Einsatzes des DM berücksichtigt, geht eine von Fonseca und Fleming in [Fonseca und Fleming, 1995] vorgeschlagene und von V.Nissen in [Nissen, 1997] übernommene Klassifizierung mehr von der verwendeten Technik innerhalb der MOEA aus. Sie unterscheiden zwischen: Aggregationsansätzen, Ansätzen mit wechselnden Zielen, Nischentechniken und pareto-basierte Ansätzen.

Sowohl der Ansatz mit wechselnden Zielen, Nischentechniken als natürlich auch pareto-basierte Ansätze führen zur Herausbildung einer Lösungsmenge und gehören zu den a posteriori– oder trade-off-Techniken. Aggregationsansätze hingegen zu den a priori-Techniken. Die folgende Graphik veranschaulicht die beschriebenen Klassifizierungen und ihre Beziehung zueinander.

Im folgenden Abschnitt soll der in Abbildung 5.1 dargestellten Systematik gefolgt und ihre einzelnen Komponenten überblicksartig beschrieben werden.

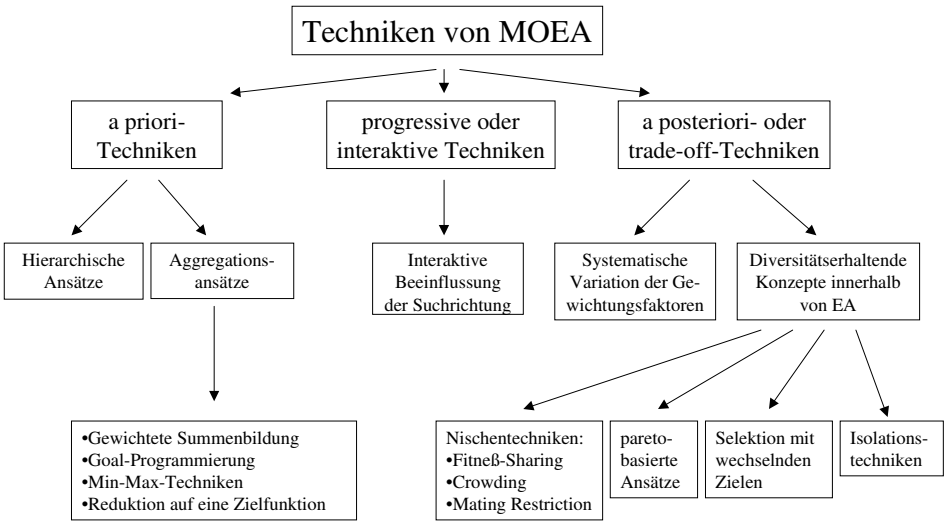


Abbildung 5.1: Konzepte für MOEA (angelehnt an [Veldhuizen und Lamont, 2000] S. 130)

5.2.1 A Priori – Techniken

Bei den a priori-Techniken wird ein MOP im Vorfeld durch eine geeignete Transformationsfunktion in ein Optimierungsproblem mit nur einer Zielfunktion (SOP oder *single objective problem*) überführt. Dies setzt jedoch voraus, daß eine Aggregationsbeziehung zwischen den einzelnen $f_i(\vec{x})$ besteht – z.B. indem sich durch Normierung alle Zielfunktionswerte in den gleichen Wertebereich transformieren und anschließend durch gewichtete Summation zusammenfassen lassen oder eine Hierarchiebeziehung zwischen den einzelnen Optimierungszielen besteht. Kann diese Aggregationsbeziehung jedoch nicht angegeben werden, ist dieser Ansatz immer kritisch zu hinterfragen, insbesondere dann, wenn die zu erwartenden Ober- und Untergrenzen der einzelnen $f_i(\vec{x})$ nicht bekannt sind und so deren Normierung schwer möglich ist. Der große Vorteil der a priori-Techniken besteht jedoch darin, daß bestehende Algorithmen zur Lösung eines SOP´s mit geringem Aufwand angepaßt und dann zur Lösung eines MOP´s verwendet werden können.

A priori-Ansätze führen zur Herausbildung nur einer einzelnen Lösung, d.h. es wird keine Menge nicht-dominierter Lösungen und damit keine durch sie bestimmte Approximation der Pareto-Front generiert. Jedoch ist es durch mehrmaliges Durchlaufen eines solchen Optimierungsalgorithmus (z.B. mit systematisch variierten Aggregationen oder Prioritäten) möglich, verschiedene näherungsweise pareto-optimale Lösungen zu generieren (siehe Abschnitt 5.2.2). Es ist jedoch mit diesem Ansatz nicht garantiert, daß die gesuchte Pareto-Front in ihrem tatsächlichen Verlauf gefunden wird. Vielmehr bleibt es dem Geschick des

Anwenders überlassen, durch eine geeignete Parametrierung des Algorithmus' (z.B. der Wahl verschiedener Kombinationen von Gewichtungsfaktoren beim „*weighted-sum*"-Ansatz) eine Approximation der Pareto-Front zu erzeugen, die der PF_{true} des Problems möglichst nahe kommt.

Bei MOEA, die zur Berechnung der Fitness a priori-Techniken verwenden, spezialisieren sich die Individuen der Population häufig auf die Optimierung nur eines Ziels, was im Extremfall einem Eckpunkt der globalen Pareto-Front entspricht. Um eine solche Konvergenz der Population auf wenige Lösungspunkte zu erschweren, sollten MOEA auf der Basis von a priori-Techniken daher mit diversitätserhaltenden Techniken (z.B. Nischentechniken - siehe Abschnitt 5.2.2.3) kombiniert werden[53].

Bei den a priori-Ansätzen wird zwischen hierarchischen Techniken und Aggregationsansätzen unterschieden.

5.2.1.1 Hierarchische Optimierungsmethoden

Die hierarchische Optimierung[54] basiert auf der Einführung einer Ordnungsrelation (bzw. einer Priorisierung) zwischen den k Komponenten des Vektors der Zielfunktionen $\vec{f}(\vec{x})$. Entsprechend einer durch den Anwender festzulegenden Reihenfolge werden die einzelnen $f_i(\vec{x})$ nacheinander separat minimiert. Dabei gilt als Nebenbedingung für die Minimierung der i-ten Komponente, daß die bereits ermittelten Minima der $(i-1)$-ten Komponenten nicht um einen für jedes $f_i(\vec{x})$ in Form von ε_i definierten Betrag überschritten werden dürfen. Ein Spezialfall der Hierarchischen Optimierung ist die von Ben-Tal vorgeschlagene lexikographische Optimierung [Ben-Tal, 1980], für die $\varepsilon_i = 0$ gilt.

Werden hierarchische Ansätze bei MOEA angewandt, so handelt es sich hierbei i.d.R. um lexikographische Ansätze. Ein Nachteil dieses Ansatz als Methode zur Lösung eines MOP´s ist, daß es nicht möglich ist, einen einmal ermittelten Minimalwert einer Zielfunktion höherer Priorität zu Gunsten eines Minimalwertes einer Zielfunktion niedrigerer Priorität zu verlassen. Unterschiedliche Kompromißlösungen, wie in den Kapiteln 1 und 2 beschrieben, sind mit diesem Ansatz also nicht generierbar und ein systematisches Auffinden pareto-optimaler Lösungen schwierig.

Trotzdem haben hierarchische Methoden zur Lösung eines MOP´s ihre Berechtigung und empfehlen sich immer dann an, wenn der Anwender bereits vor der Optimierung in der Lage ist, eine klare Prioritätenzuordnung zwischen den einzelnen Optimierungszielen festzulegen.

Ein evolutionärer Algorithmus mit lexikographischen Ansatz zur Lösung eines MOP´s wird z.B. in [Fourman, 1985] beschrieben. Hier wurde ein GA zur Optimierung der Kompaktheit beim Schaltungsentwurf eines wafers verwendet, wobei folgende Zielfunktionen entsprechend einer heuristisch festgelegten Priorität zu optimieren waren: die äußeren Abmessungen des wafers, die Anzahl der Verletzungen spezieller Designregeln und die Plazierung der Rechtecke.

---

[53] So verwenden beispielsweise Hajela und Lin in ihrem HLGA-Ansatz (siehe Abschnitt: 5.4.1.2) eine Kombination aus „*weighted-sum"*-Ansatz*, sharing* und *mating restriction.*

[54] Zu nichtevolutionären, hierarchischen Ansätzen in der ingenieurtechnischen Praxis siehe auch die Arbeiten von [Walz, 1967] und [Ocyczka, 1984].

5.2.1.2 Aggregationsansätze

Skalare Aggregationsansätze[55] zählen in der ingenieurtechnischen Praxis zu den meist verwendeten Techniken zur Lösung eines MOP´s. Die Fitness eines Individuums berechnet sich durch die skalare Aggregation der einzelnen Zielfunktionswerte des MOP´s. Im Folgenden sollen die gebräuchlichsten Techniken zur Aggregation der Zielfunktionen zur Fitnessfunktion beschrieben werden.

Gewichtete Summenbildung: Als Beispiel für eine lineare Aggregation soll hier der bekannte *weighted-sum*-Ansatz, d.h. gewichtete Summenbildung, genannt werden. Beim weighted-sum-Ansatz wird die Summe jedes einzelnen der k Zielfunktionswerte $f_i(\vec{x})$ multipliziert mit einem Gewichtungsfaktor w_i gebildet, die dann während der Optimierung minimiert wird.

$$\textit{min}: \; S(\vec{x}) = \sum_{i=1}^{k} w_i \cdot f_i(\vec{x}) \qquad \textit{mit}: \; \sum_{i=1}^{k} w_i = 1 \qquad [36]$$

Der weighted-sum-Ansatz ist eine einfache und in der Praxis häufig verwendete Methode zur Lösung eines MOP´s, deren Einsatz dann besonders effizient ist, wenn die Gewichtungen w_i in Form von Präferenzen der einzelnen Optimierungsziele $f_i(\vec{x})$ bekannt sind, sich die einzelnen Zielfunktionswerte im selben Wertebereich befinden und die Generierung einer den Zielraum möglichst homogen abdeckenden Pareto-Front nicht erforderlich ist.

Ist die globale Pareto-Front PF_{true} des zu lösenden MOP´s jedoch konkav und besteht das Ziel der Optimierung darin, eine den Zielraum breit und homogen abdeckende Approximation von PF_{true} zu generieren, sollte dieser Ansatz nicht ohne vorherige, geeignete Transformation der einzelnen Zielfunktionen gewählt werden. Der Grund dafür besteht darin, daß Algorithmen, die auf dem weighted-sum-Ansatz basieren nicht in der Lage sind, effiziente Zielfunktionswertevektoren in konkaven Regionen einer Pareto-Front aufzufinden:

> *...points in concave regions of the trade off cannot be found by optimizing a linear combination of the objectives, for any set of weights*. [Fonseca und Fleming, 1995]

Ein Ausweg aus dieser Problematik besteht darin, die einzelnen Zielfunktionen derart zu tranformieren, daß eine Veränderung der Konkavität der Pareto-Front des MOP´s erreicht wird. Fonseca und Flemming beschreiben eine solche Transformation in [Fonseca und Fleming, 1995]: indem der Exponent $\alpha = \{1,..,9\}$ im Term $\left[f_i(\vec{x})\right]^{\alpha}$ mit $f_i(\vec{x}) \in [0,1]$ variiert wird, kann die Konvexität der globalen Pareto-Front verändert werden.

Eine Anwendung auf der Basis eines MOEA mit dem weighted-sum-Ansatz findet sich in [Broekmeulen, 1995]. In dieser Applikation aus dem Management in der Lebensmittelbranche sollen sowohl die Qualitätseinbußen der Produkte bei der Lagerung als auch die dabei entstehenden Kosten minimiert werden. Zur Steigerung der Effizienz des Algorithmus wird zusätzlich eine auf das *crowding* basierende Selektion (siehe Abschnitt 5.2.2.3) verwendet.

Trotz der verlockenden Einfachheit des weighted-sum-Ansatzes für den Praktiker soll an dieser Stelle darauf hingewiesen werden, daß sich die Fitnesslandschaft bei der gewichteten Summenbildung mehrerer Zielfunktionen immer nur als Menge einzelner, weniger Punkte im Zielraum und nicht wie z.B. beim pareto-basierten ranking (siehe Abschnitt 5.2.2.4) als durch die Optima der einzelnen $f_i(\vec{x})$ begrenzte Hyperebene darstellt, die sich idealerweise der globalen

[55] Siehe dazu auch die Quellen in [Rosenthal, 1985], [Fonseca und Fleming, 1995], [Nissen, 1994] und [Coello, 1996] sowie [Veldhuizen und Lamont, 2000].

Pareto-Front nähert. Daher kann mit diesem Ansatz die Generierung einer guten Approximation von PF_{true} nicht gelingen.

„Goal"-Programmierung: Bei der „Goal"-Programmierung[56] [Steuer, 1986] wird diejenige Lösung $\vec{x}$ gesucht, die den durch $\vec{w}$ gewichteten metrischen Abstand $S(\vec{x})$ zwischen den durch $\vec{x}$ bestimmten Zielfunktionswertevektor $\vec{f}(\vec{x})$ und einem vorgegebenen (Goal-)Vektor im Zielraum $\vec{y}^{goal}$ minimiert [37].

$$min:\ S(\vec{x}) = \vec{w} \cdot \left| \vec{f}(\vec{x}) - \vec{y}^{goal} \right| \qquad [37]$$

Als Metrik $\left| \cdot \right|$ wird hierbei häufig die euklidische Distanz verwendet. Auch hier können durch die Gewichtungsfaktoren w_i Präferenzen des Anwenders über die Bedeutung der Zielerreichung der einzelnen $f_i(\vec{x})$ eingebracht werden.

MOEA auf der Basis der Goal-Programmierung sind immer dann effizient einsetzbar, wenn der Goal- oder Zielvektor $\vec{y}^{goal}$ bekannt ist. Häufig liegt dieses Wissen jedoch nicht vor – oder ist nur vermeintlich bekannt. Die Generierung einer Lösungsmenge, die die globale Pareto-Front PF_{true} bestimmt, ist mit diesem Ansatz nicht möglich.

MinMax-Techniken: Auch bei den sogenannten MinMax-Techniken ist, ähnlich wie bei der Goal-Programmierung, das Vorhandensein eines Zielvektors notwendig. Im Unterschied zur Goal-Programmierung wird hier jedoch nicht ein Abstandsmaß zum Goal-Vektor minimiert, sondern gewichtet die maximal auftretende Abweichung einer einzelnen Komponente $f_i(\vec{x})$ zu $\vec{y}^{goal}$, d.h. es wird der in Gleichung [38] ausgedrückte Term minimiert:

$$min:\ S(\vec{x}) = \max_{i=1,..,k} \left| \frac{f_i(\vec{x}) - y_i^{goal}}{w_i} \right| \qquad [38]$$

MinMax-Techniken werden ähnlich wie die Goal-Programmierung immer dann in Verbindung mit MOEA eingesetzt, wenn ein gewünschter Zielfunktionswertevektor bekannt ist.

Aggregationsansätze sind einfach zu realisierende und häufig eingesetzte Methoden zur Lösung eines MOP´s. EA können in ihren Standardformen verwendet werden, indem die einzelnen Zielfunktionen zu einem Fitnesswert aggregiert werden. Der Nachteil von Aggregationstechniken ist, daß auf diese Weise lediglich genau eine Lösung die Population dominiert, die stark von der Wahl der Präferenzen (z.B. Gewichtungsfaktoren) der einzelnen Zielkriterien anhängt. Diese Vorgehensweise setzt jedoch ein gründliches Wissen über den zu optimierenden Prozess voraus, das typischerweise nicht vorliegt. Werden genau diese Präferenzen nun vom Anwender korrigiert, um ein anderes Szenario zu simulieren, muß der Optimierungsalgorithmus abermals durchlaufen werden. Eine zusätzliche Schwierigkeit dieses Ansatzes liegt in der Aggregation von Funktionswerten mit unterschiedlichen Wertebereichen, die häufig durch die Abbildung mittels einer Skalierungs- oder Normierungsfunktion für die einzelnen Optimierungsziele gelöst wird.

Liegen die Präferenzen bzw. Gewichtungen oder Prioritäten jedoch bereits für den Anwender fest, sind Aggregationsmethoden sehr effiziente und empfehlenswerte Lösungsstrategien, da sie gestatten, den Lösungsraum in genau eine Richtung - nämlich in die des Gradienten der Aggregationsfunktion - zu durchlaufen.

[56] Häufig wird dieser Ansatz auch als „Zielprogrammierung" oder „Target-Vector"-Technik bezeichnet.

5.2.2 A Posteriori – Techniken:

Bei dieser Lösungsstrategie wird durch den Optimierungsalgorithmus ohne die Einbeziehung von Vorwissen eine Menge von Lösungen generiert, aus der dann nach dem Optimierungsdurchlauf vom Anwender mit Hilfe des Decision-Makers eine Auswahl getroffen werden kann. Da Evolutionäre Algorithmen aufgrund ihres Prinzips bereits mit einer Lösungsmenge agieren, eignen sie sich besonders gut für diese Lösungsstrategie ([Fonseca und Fleming, 1995], [Zitzler und Thiele, 1998]).

Als schwierig gilt bei diesem Ansatz jedoch zum einen die Realisierung effizienter Visualisierungs- und Zugriffstechniken für die berechnete Menge näherungsweise pareto-optimaler Lösungen und der durch sie bestimmten Zielfunktionswertevektoren sowie die Gewährleistung deren Heterogenität und homogenen Abdeckung im Zielraum.

Trotz dieser Schwierigkeiten wird dieses Konzept seit den 90er Jahren von den meisten modernen Multikriteriellen Evolutionären Algorithmen verwendet. So wurden spezielle Techniken entwickelt, um die Heterogenität (diversity) der unterschiedlichen Lösungen zu gewährleisten und eine Dominanz nur einer Lösung zu verhindern. Diese speziellen Techniken betreffen die Populationskonzepte, Selektionsmechanismen sowie die Berechnung der Fitness und sollen in den folgenden Abschnitten näher vorgestellt werden.

5.2.2.1 Mehrfache, systematische Aggregation der Zielfunktion

Bei diesem Ansatz, der in der Literatur häufig als *independent sampling* Technik bezeichnet wird, wird ein Standard-EA mehrfach mit jeweils variierter Aggregation der Zielfunktionswerte eingesetzt. Die über mehrere Optimierungsläufe berechneten Lösungen repräsentieren idealerweise jeweils einen Punkt auf der hochdimensionalen Pareto-Front.

Vorteilhaft an dieser Methode ist ihre sehr einfache Umsetzung und die Möglichkeit, vorhandene Standard-EA ohne spezielle Anpassungen verwenden zu können. Sie wird daher häufig eingesetzt, um sich einen ersten Eindruck von der Lösungsmenge eines MOP´s zu verschaffen. Schnell werden jedoch auch die Nachteile deutlich: besonders bei höherdimensionalen Zielfunktionsvektoren steigt der Aufwand, der nötig ist, um eine ausreichende Anzahl von Kombinationen der Aggregation durchzutesten, enorm an. Zusätzlich ist nicht gewährleistet, daß durch die Wahl der Aggregationskombinationen tatsächlich alle Bereiche der Pareto-Front ausreichend repräsentiert werden.

Eine Applikation, die diesen Ansatz zur Lösung eines MOP´s verwendet, findet sich z.B. in [Chang, 1995]. Hier wird ein GA mit independent sampling Technik zur Optimierung eines Eisenbahnantriebs – mit dem Ziel der gleichmäßigen Lastverteilung und minimalen Energiekosten – beschrieben.

Standardformen Evolutionärer Algorithmen führen wie oben dargestellt i.d.R. zur Herausbildung einer Population von Individuen in der Nähe nur *eines* Optimums. EA sind daher nur dann in der Lage, den gesamten Zielraum effizient parallel zu durchsuchen, wenn die Populationsmitglieder eine gewisse Heterogenität (*diversity)* aufweisen. Im folgenden Abschnitt sollen die wichtigsten heterogenitätserhaltenden Methoden für EA dargestellt werden.

5.2.2.2 Selektion mit wechselnden Zielen

Die erste Technik, die entwickelt wurde, um das Populationskonzept von EA zur Lösung eines MOP´s direkt auszunutzen, ist die *criterion selection* oder die Selektion mit wechselnden Zielen. Die Grundidee dieser Technik besteht darin, bei k Zielfunktionen die stochastische Selektion in k Teilschritte zu zerlegen und in jedem dieser Schritte jeweils eine Menge von Individuen bezüglich ihrer Güte entsprechend nur eines der Zielkriterien auszuwählen. Nach der so

gestalteten Auswahl in den mating-pool werden die Rekombinations- und Mutationsoperatoren auf den gesamten mating-pool angewandt und so wieder ein Durchmischen der Population erreicht. Schaffers VEGA (siehe Abschnitt 5.4.1.1) und die MOES von Kursawe (siehe Abschnitt 5.4.1.3) sind Beispiele für diese Technik. MOEA mit wechselnden Zielen besitzen die Tendenz zur vorzeitigen Konvergenz und Überspezialisierung der Individuen und werden daher häufig mit Nischentechniken kombiniert. Da sie indirekt auf einem Aggregationsansatz basieren, sind sie nicht in der Lage, konkave Elemente der Pareto-Front aufzufinden (siehe Abschnitt 5.2.1.2).

5.2.2.3 Nischentechniken

Das Grundprinzip von Nischentechniken basiert darauf, daß Individuen einer definierten Umgebung (Nische) nur eine bestimmte Kapazität von Ressourcen zur Verfügung steht, die sie sich teilen müssen: je mehr Individuen sich in einer definierten Nachbarschaft befinden, desto mehr verringert sich ihre Fitness. Die Nachbarschaft zweier Individuen $\vec{a}_i$ und $\vec{a}_j$ wird durch ein Distanzmaß $d(i, j)$ definiert, dessen Einfluß auf die Fitness der Individuen von der Wahl eines sogenannten Nischenradius σ_{share} bzw. σ_{mate} abhängt[57].

Nischentechniken können biologisch als Kampf um beschränkte Ressourcen interpretiert werden, wobei diejenigen Individuen eine höhere Fortpflanzungswahrscheinlichkeit erhalten, die sich mit möglichst wenig anderen Artgenossen in der selben Nachbarschaft bzw. Nische befinden und so um die dort vorhandenen Ressourcen mit möglichst wenigen Individuen konkurrieren müssen.

Fitness sharing: Am häufigsten wird als heterogenitätserhaltende Maßnahme das sogenannte Fitness-sharing eingesetzt, das zur Herausbildung stabiler Subpopulationen (Nischen) führt ([Goldberg und Richardson, 1987], [Zitzler, 1999] S.25ff). Die modifizierte Fitness $\Phi'(\vec{a}_i)$ eines Individuums i einer Population mit n Elementen berücksichtigt mittels der sogenannten sharing-Funktion S, wie viele Individuen j sich innerhalb seiner Nachbarschaft $d(i, j)$ befinden (siehe Gleichung [39]). Auf diese Weise werden Individuen, in deren Nachbarschaft sich relativ wenige andere Individuen befinden, gegenüber den Individuen mit sehr vielen „ähnlichen“ Nachbarn bevorzugt.

In Abhängigkeit davon, in welchem Repräsentationsraum die Distanzmetrik angewendet wird, unterscheidet man zwischen genotypischem und phänotypischem sharing. GA verwenden beim genotypischen sharing häufig den Hammingabstand, wogegen beim phänotypischen sharing meist die euklidische Distanz berechnet wird. Bei ES wird i.d.R. phänotypisches sharing eingesetzt.

$$\Phi'(\vec{a}_i) = \frac{\Phi(\vec{a}_i)}{\sum_{j=1}^{n} S(d(i, j))} \qquad [39]$$

Die Konstante $\beta \in \Re$ bestimmt die Form der sharing-Funktion *S*: im einfachsten Fall erhält man für $\beta = 1$ eine lineare Abhängigkeit für $S(d(i, j))$. Häufig wird jedoch eine polynomial verlaufende Form mit $\beta > 1$ benutzt.

[57] In [Goldberg und Richardson, 1987] und [Deb und Goldberg, 1989] werden Empfehlungen zur geeigneten Wahl der Nischenparameter gegeben. Fonseca und Fleming zeigen in [Fonseca und Fleming, 1998] die Abhängigkeit des Nischenradius‘ von der Anzahl und dem Wertebereich der Entscheidungsvariablen und der Zielfunktionen.

$$S(d(i,j)) = \begin{cases} 1 - \left(\dfrac{d(i,j)}{\sigma_{share}} \right)^{\beta} & \text{für} \quad d(i,j) < \sigma_{share} \\ 0 & \text{für} \quad d(i,j) \geq \sigma_{share} \end{cases} \qquad [40]$$

Das Konzept des Fitness-sharings ist eine wirksame und einfache Methode zur Herausbildung von Populationen hoher Diversität, die jedoch nur dann effizient wirkt, wenn der Nischenradius σ_{share} sinnvoll gewählt wurde. Deb und Goldberg geben in [Deb und Goldberg, 1989] für seine Wahl eine Reihe von Empfehlungen. So berechnen sie z.B. bei phänotypischem sharing der Entscheidungsvariablen einer Population P, mit q herauszubildenden Nischen und der euklidische Distanz als Distanzmetrik den Nischenradius nach folgender Formel:

$$\sigma_{share} = \frac{\sqrt{\sum_{i=1}^{n} \left(\max_{j=1}^{|P|} x_i^j - \min_{j=1}^{|P|} x_i^j \right)^2}}{2 \cdot \sqrt[n]{q}} \qquad [41]$$

Diese Empfehlung impliziert jedoch zum einen, daß eine Gleichverteilung der Individuen im Lösungsraum auch ihre Gleichverteilung im Zielraum zur Folge hat. Zum anderen setzt sie voraus, daß die Elemente der Pareto-Front ebenfalls gleichverteilt sind. Beide Voraussetzungen sind jedoch nicht allgemeingültig. Methoden, die ohne den sensitiven Parameter σ_{share} Nachbarschaften berücksichtigen, sind daher dem Standard-sharing vorzuziehen. So berücksichtigen Zitzler und Thiele in ihrem SPEA (siehe Abschnitt 5.4.2.4) nicht nur den Pareto-Rang der Individuen sondern auch die Anzahl der Populationsmitglieder, die sich in dem von ihnen dominierten Zielraum befinden. Damit ist auch ohne die Benutzung eines Nischenradius die Berücksichtigung einer Nachbarschaftsrelation möglich.

Oei, Goldberg und Chang zeigen in [Oei et al_1991], daß ein Standard-sharing in Verbindung mit Wettkampfselektion zu einem chaotischen Verhalten bei GA führt. Aus diesem Grunde wurden eine Reihe von Anpassungen für das Standard-sharing entwickelt. So realisierten Hajela und Lin für ihrem HLGA (siehe Abschnitt: 5.4.1.2) eine angepaßte Version des sharings: das sogenannte *continuously updated sharing*, bei der der Nischenzähler (d.h. der Nenner in Gleichung [39]) der aktuellen Generation von der nächstfolgenden, teilweise generierten Generation, berechnet wird. Auch Horn und Nafpliotis folgen für ihren NPGA siehe (Abschnitt 5.4.2.2) dieser Empfehlung ([Zitzler und Thiele, 1998] S.27). Fonseca und Fleming realisierten in ihrem MOGA ([Fonseca und Fleming, 1993] und Abschnitt 5.4.2.1) dagegen einen sharing-Mechanismus im Zielraum, bei dem der Nischenradius nach jeder Generation neu berechnet wird. Dabei wird davon ausgegangen, daß die Elemente einer guten Approximation einer Pareto-Front idealerweise äquidistant verteilt sein sollen und sich so in jeweils einer Nische befinden. Fonseca und Fleming bestimmen den Nischenradius nach jeder Generation daher aufgrund der Anzahl der berechneten nicht-dominierten Lösungen sowie des Wertebereichs der durch sie bestimmten Zielfunktionswertevektoren.

Crowding: Eine weitere Nischentechnik ist das *crowding* [De Jong, 1975], das ebenso wie das Fitness-sharing hauptsächlich bei GA Anwendung findet. Der Grundgedanke des crowdings besteht darin, den Selektionsdruck auf ähnliche Individuen stärker zu gestalten als auf Individuen, die sich „stärker" voneinander unterscheiden, d.h. je Generation ersetzen die Nachkommen immer diejenigen Individuen, denen sie am ähnlichsten sind. Auf diese Weise soll ein „Zusammenklumpen" von Individuen mit ähnlichen Eigenschaften verhindert werden. Als Maß für die Ähnlichkeit wird auch hier eine Abstandsmetrik verwendet, die sowohl auf den Phänotyp als auch auf den Genotyp anwendbar ist. Im Gegensatz zur Standardform

Genetischer Algorithmen wird bei einem GA mit crowding generell ein crossover durchgeführt und nach der Erzeugung der Nachkommen je ein Elter und ein Nachkomme zusammengefaßt. Dabei erfolgt die Entscheidung, welcher Elter mit welchem Nachkommen ein Paar bildet, aufgrund des oben erwähnten Ähnlichkeitsmaßes. Es werden immer diejenigen Individuen der Elterngeneration von denjenigen Individuen der Nachkommengeneration ersetzt, denen sie am „ähnlichsten" sind. Auf diese Weise wird die Heterogenität innerhalb der Population erhalten.

Mating-restriction: Eine weitere heterogenitätserhaltende Technik ist das Verfahren des *mating restriction* [Goldberg, 1989], das allerdings im Bereich der multikriteriellen Optimierung nicht sehr verbreitet ist (Awendungen finden sich in: [Hajela und Lin, 1992] und [Fonseca und Fleming, 1993]).

Beim *mating restriction* ist es zwei Individuen nur dann gestattet sich zu rekombinieren, wenn sie eine bestimmte Distanz zueinander nicht überschreiten. Analog der Vorgehensweise des Fitness-sharings wird entweder die phäno- oder genotypische Distanz $d(i, j)$ zweier Individuen berechnet und mit einem vorgegebenen mating-Radius σ_{mate} verglichen. Bei EA besitzen unterschiedliche Lösungen u.U. eine extrem unterschiedliche genetische Repräsentation. Indem Rekombination nur in einer bestimmten lokalen Umgebung zugelassen wird, verringert sich zum einen die Wahrscheinlichkeit, lethale Nachkommen zu erzeugen, und erhöht sich so die on-line Performance des Algorithmus. Zum anderen wird jedoch auch die Herausbildung stabiler Nischen ermöglicht und so die Variabilität der Population erhöht. Allerdings kann über mehrere Generationen ein langsames Verschieben und Verschmelzen der mating-areas möglich sein und sich so die Effizenz des mating restriction verringern.

Gegenüber der Methode des Fitness-sharings ist der Nutzen des mating restriction nicht eindeutig bewiesen – und hängt in seiner Effizienz von der Landschaft der Zielfunktionen ab. Zitzler und Thiele haben in [Zitzler und Thiele, 1998] eine Reihe von Simulationen dokumentiert, bei denen sie den Einfluß von σ_{mate} untersuchten. Dabei stellten sie keinen nennenswerten Einfluß auf die Performance des Algorithmus fest.

5.2.2.4 Pareto-basiertes Ranking

Ein erfolgreiches Konzept zur Lösung eines MOP´s im Sinne der Generierung pareto-optimaler Lösungen ist das pareto-basierte Ranking. Es basiert auf der Berechnung der Fitness der Individuen in Abhängigkeit ihrer Pareto-Dominanz gegenüber einer Vergleichsmenge von Individuen. Individuen, denen ein niedriger Pareto-Rang zugeordnet wird, d.h. die dominant sind, erhalten eine höhere Fitness, als Individuen höheren Pareto-Rangs. Die Fitnessberechnung auf Basis des Pareto-Rangs kann sowohl mit einer fitnessproportionalen Selektion als auch mit einer Wettkampfselektion kombiniert werden und ist nicht auf eine reine Rangselektion beschränkt. Für die Berechnung des Pareto-Rangs eines Individuums gibt es zwei verbreitete Ansätze:

Berechnung des Pareto-Rangs nach Goldberg: Goldberg [Goldberg, 1989] berechnet den Rang eines Individuums nach folgendem Schema: nicht-dominierte Individuen erhalten den Rang 1 und werden bei der Bestimmung der weiteren Rangplätze nicht mehr berücksichtigt. Aus der Restpopulation werden wiederum die nicht-dominierten Lösungen bestimmt – sie erhalten den um 1 erhöhten Rang und werden wiederum bei der Berechnung der Ränge der restlichen Individuen nicht mehr berücksichtigt usw..

Berechnung des Pareto-Rangs nach Fonseca und Fleming: Dagegen ist nach Fonseca und Fleming [Fonseca und Fleming, 1993] der Rang eines Individuums innerhalb der Population gleich der um 1 erhöhten Anzahl der Individuen, die von ihm dominiert werden. Damit erhalten alle diejenigen Individuen, die kein anderes Individuum dominieren den Rang 1.

Einige vereinfachende Ansätze ([Veldhuizen und Lamont, 2000] S.135) setzen den Rang eines Individuums immer dann gleich 0, wenn kein Individuum innerhalb der Population existiert, das es streng dominiert. Wird es dagegen von mindestens einem anderen Individuum streng dominiert, wird ihm der Rang 1 zugeordnet.

Vergleicht man die von Goldberg und Fonseca und Fleming vorgeschlagene Vorgehensweise, so erlaubt der Ansatz nach Fonseca und Fleming eine feinere Berechnung des Rangs als Goldbergs Ansatz und führt so zu einer effektiveren Gestaltung der Fitnesslandschaft [Horn, 1997] und letzlich zu einer homogeneren Approximation der Pareto-Front [Thomas, 1998].

Pareto-basiertes Ranking gestattet das korrekte Auffinden nicht-dominierter Individuen, kann jedoch allein keine Garantie über ihre homogene Verteilung im Zielraum oder Konvergenz in Richtung PF_{true} geben [Fonseca und Fleming, 1995]. In einer Fitnesslandschaft mit mehreren gleichwertigen Optima, konvergieren die Individuen einer Population aus Gründen der „genetischen Drift“ (siehe oben) i.d.R. nur in Richtung eines dieser Optima. Um jedoch bei der Lösung eines MOP´s mittels MOEA eine Approximation der Pareto-Front mit möglichst homogen und breit im Zielraum verteilten Elementen zu generieren, ist es daher notwendig, bei der Berechnung der Fitness auch die Verteilung der Individuen im Ziel- oder Lösungsraum zu berücksichtigen. Dies geschieht wie bei fast allen modernen pareto-basierten MOEA (siehe Abschnitt 5.4.2) i.A. durch die Kombination von pareto-basierter Fintessberechnung und Nischentechniken sowie speziellen Selektionstechniken.

5.2.2.5 Isolationstechniken

Eine weitere – allerdings nicht sehr weit verbreitete – Möglichkeit, MOEA zu realisieren, basiert auf dem Multipopulationskonzept ([Herdy, 1992] und [Schwefel, 1995]). Hier wird bei k Zielfunktionen die Gesamtpopulation in k Teil-Populationen (sogenannte Deme) aufgeteilt. Dabei erfolgt die Selektion in jeder der Teilpopulationen entsprechend der Minimierung je einer der k Zielfunktionen. Zwischen den einzelnen Teilpopulationen ist ein begrenzter Austausch von Individuen in Form von Migration und Emigration[58] (z.B. nach einer definierten Anzahl von Generationen oder bei relativer Stagnation der Populations-Fitness) erlaubt. Die Anwendung des Multipopulationskonzepts zur Lösung eines MOP´s besitzt Analogien zum MOEA-Ansatz mit wechselnden Zielen (z.B. VEGA-System). Auch hier besteht die Tendenz der Spezialisierung der Individuen, so daß im Extremfall lediglich die Komponenten des Idealvektors generiert werden und die Berechnung einer homogen und breit über den Zielraum verteilten Pareto-Front nicht zwangsläufig gewährleistet ist. Der hohe Parametrieraufwand dieses Ansatz (Definition der Größe und Anzahl der Deme, der Rate und des Intervalls der Migration bzw. Emigration sowie Festlegung, zwischen welchen Demen ein Individuenaustausch erlaubt ist) macht ihn zusätzlich anfällig.

5.2.3 Progressiv – Techniken

Bei dieser Strategie erfolgt eine Interaktion zwischen dem DM und dem Optimierungsalgorithmus *während* des Optimierungslaufes selbst. So ist es möglich, die Suchrichtung im Lösungsraum interaktiv zu beeinflussen. Dieser Ansatz erfordert jedoch sowohl ein gewisses problemspezifisches Vorwissen als auch eine Überwachung bzw. Steuerung durch den Anwender während der Optimierung. Typischerweise ist die Struktur des Lösungsraumes jedoch nicht bekannt und es ist daher auch selten abschätzbar, ob die eingeschlagene

[58] Der Unterschied zwischen Migration und Emigration besteht in diesem Zusammenhang darin, daß bei Emigration die Individuen ihrer Ursprungspopulation verloren gehen, während bei Migration Kopien von ihnen angelegt werden ([Nissen, 1997] S.256).

Suchrichtung bzw. der aktuelle Optimierungskorridor tatsächlich verworfen bzw. akzeptiert werden kann, da gerade in stark zerklüfteten Lösungsräumen nicht klar ist, ob sich nicht neben einem lokalen ein globales Minimum befindet. Trotz der genannten Schwierigkeiten birgt dieser Lösungsansatz sicher ein großes Entwicklungspotential ([Fonseca und Fleming, 1995] S.26ff).

5.3 Hybride MOEA

Zunehmend gelangen hybride MOEA in den Mittelpunkt des Interesses der MOEA-Entwickler. Während sich EA durch ihre breite Anwendbarkeit auf viele Probleme der Optimierung und ihre relativ geringe Spezialisierung auf Problemklassen auszeichnen, tragen effiziente, klassische Methoden der Parameteroptimierung eben genau dieser Spezialisierung Rechnung und können so bei speziellen, gut untersuchten Problemen deutlich effizienter arbeiten. Um die Vorteile der einen Methode gegen die Nachteile der anderen und vice versa zu kompensieren, werden MOEA bei konkreten praktischen Anwendungen oft mit alternativen Spezialalgorithmen kombiniert.

Dies kann im einfachsten Fall geschehen, indem schon während der Initialisierungsphase gezielt anwendungsspezifische Heuristiken eingesetzt werden und so die Individuen der Startpopulation gezielt in eine dem Optimum nahe Region des Zielraumes plaziert werden.

Eine andere und weit verbreitete Möglichkeit besteht darin, konventionelle Lösungsverfahren wie z.B. lokale Suchstrategien sequentiell zur evolutionären Suche einzusetzen. Dabei können diese mittels spezieller Evolutionsoperatoren sowohl während als auch nach Abschluß der Evolution angewandt werden (siehe Abschnitt 5.3.1). Insbesondere die ersten hybriden Ansätzen zwischen EA und konventionellen Optimierungsverfahren stellten sich hauptsächlich als sequentielle Kombination zwischen EA und klassischen Verfahren der Parameteroptimierung (siehe Abschnitt 4.2) dar. Auf diese Weise konnte vor allem in der Phase des Finetunings eine Beschleunigung der Optimierung erreicht werden [Meyer und Eder, 1999].

Schließlich seien an dieser Stelle auch diejenigen Ansätze erwähnt, die die evolutionäre Suche verwenden, um die eingesetzten konventionellen Optimierungsverfahren optimal zu parametrieren.

Heute werden zur Hybridisierung überall dort die verschiedensten Techniken verwendet, wo die entwickelten MOEA der ersten und zweiten Generation an ihre Grenzen stoßen und die Vorteile dieser Methoden den Optimierungsverlauf effizienter gestalten. Man kann hybride MOEA daher als MOEA der dritten Generation interpretieren.

Es soll im Folgenden kurz auf die Kombination zwischen MOEA und lokalen Suchstrategien und MOEA mit natur-analogen Verfahren eingegangen werden. In Abschnitt 5.4.3 werden einige - unter dem Aspekt dieser Arbeit - interessante Realisierungen dieser Herangehensweise zitiert.

5.3.1 Hybridisierung zwischen MOEA und lokalen Suchstrategien

Bei der Kombination der MOEA mit lokalen Suchalgorithmen wird zwischen zwei Strategien unterschieden.

1. Ein lokaler Suchalgorithmus kann bereits *während* der evolutionären Suche nach bestimmten Kriterien angestoßen oder
2. erst *nach* Abschluß der Optimierung auf Individuen der Endpopulation angewandt werden.

Beide Herangehensweisen haben ihre Vor- und Nachteile. Häufig wird bei Methoden der ersten Strategie der Vorteil eines besseren Konvergenzverhaltens gegen den Nachteil eines größeren Rechenaufwandes aufgehoben. Unter Umständen „fangen“ sich die lokalen Suchoperatoren in stark zerklüfteten Fitnesslandschaften schon zu früh in lokalen Minima – ein Verhalten, das ja

gerade durch den Einsatz von EA verhindert werden soll. Den Kriterien, wann und auf welche Individuen die lokalen Suchoperatoren angewandt werden, oder wann es effizienter ist, den MOEA ohne lokale Suche anzuwenden, kommt daher eine entscheidende Bedeutung zu. Da EA jedoch nicht stetig konvergieren, ist die Entscheidung, ab wann eine vorzeitige Konvergenz vorliegt, schwierig. Aus diesem Grunde erfolgt die Auswahl der Individuen, auf die lokalen Suchstrategien angewandt werden i.d.R. pro Generation gleichverteilt zufällig nach einer Wahrscheinlichkeit, die im einfachsten Fall für alle Individuen gleich oder in Abhängigkeit von der Fitness der Individuen unterschiedlich hoch sein kann.

Die zweite Strategie, lokale Operatoren auf die Elemente der durch den Algorithmus generierten Approximation PF^* der Pareto-Front nach Abschluß der Optimierung anzuwenden, umgeht zwar die obengenannten Nachteile, bringt den MOEA jedoch auch um die Möglichkeit, große Evolutionssprünge während der Optimierung durchzuführen. Die lokalen Operatoren realisieren hier das oben erwähnte Finetuning, d.h. die Elemente aus PF^*, die sich am Rande eines pareto-optimalen Tales befinden, werden in dieses hineingezogen.

Die Kombination zwischen MOEA und lokalen Suchalgorithmen hat sich als sehr sinnvoll erwiesen und wurde aus diesem Grunde auch in dieser Arbeit verwendet.

5.3.2 Hybridisierung zwischen MOEA und Methoden der Computational Intelligence

An dieser Stelle sollen hybride MOEA genannt werden, die die Kombination eines MOEA mit Methoden der Computational Intelligence realisieren.

Hierzu gehören u.a. MOEA in Kombination mit Neuronalen Netzen („Neuro-EA“). Es existieren heute eine große Anzahl verschiedener Hybridisierungsmöglichkeiten zwischen Neuronalen Netzen und EA. Die überwiegende Mehrzahl der Ansätze verwendet hierbei EA, um optimale Netzwerkstrukturen oder –parameter zu finden, während des Lernprozesses die optimalen Lernregeln auszuwählen oder die Kopplungsstärken der Gewichte selbst auf die Trainingsdaten optimal anzupassen. In vielen Arbeiten werden NN auch zur Berechnung der Gütefunktionen und zur Feststellung der Verletzung der Randbedingungen genutzt. Allen diesen Ansätzen ist gemeinsam, daß sie NN als zusätzliche, vom eigentlichen Optimierungsprozess abgekoppelte Module verwenden. Einen anderen Ansatz verfolgen diejenigen Arbeiten, die NN direkt in die evolutionäre Suche einbeziehen. Dies kann geschehen, indem die Struktur des NN selbst für die evolutionäre Suche genutzt wird, z.B. indem die Individuen der aktuellen Generation als Input-Schicht über die Hidden-Schicht, die einem Nachkommen entspricht, auf die Output-Schicht, deren Neurone den einzelnen Zielfunktionen zugeordnet werden, abgebildet werden. Eine Realisierung dieser Idee stellt das Nessy-System von M.Köppen und S.Rudlof [Koeppen und Rudlof, 1998] dar. Eine weitere Möglichkeit besteht darin, das NN dafür zu nutzen den Evolutionspfad zu lernen und die Reproduktionsoperatoren entsprechend des gelernten Wissens anzupassen. Diesen Ansatz verfolgen beispielsweise D.Büche et al mit ihrem SOM-MOEA, der in Abschnitt 5.4.3.2 näher dargestellt wird.

Es existieren noch weitere Möglichkeiten der Verknüpfung MOEA mit anderen, biologisch motivierten Optimierungsmethoden. Da im Rahmen dieser Arbeit bezüglich der Hybridisierung jedoch gradientenbasierte lokale Suchalgorithmen und Neuronale Netze im Vordergrund stehen, sollen auf weitere Möglichkeiten der Kombination MOEA mit Verfahren der CI auf die Literatur verwiesen werden: MOEA in Kombination mit

- Methoden der Fuzzy-Logik: [Gomez et al, 1998], [Gueugnuiaud et al, 1997] und [Ishbuchi und Murata, 2001],
- tabu search: [Hansen, 1997], [Gomez et al, 1998] und [Knowles, 2002] sowie

- simulated annealing: [Knowles, 2002].

Die Verbindung zwischen MOEA und lernfähigen Systemen wird deshalb als sehr sinnvoll angesehen, weil es möglich ist:

- mittels der NN die Evolutionsrichtung zu lernen und damit die evolutionäre Suche zielgerichteter zu gestalten sowie zu beschleunigen und
- ein Einsatz NN es dem Anwender gestatten, Wissen aus dem Evolutionsprozess zu extrahieren und damit den MOEA von einer „black box"- in eine „gray-box"-Methode zu überführen.

Beide Vorzüge werden in der vorliegenden Arbeit von dem realisierten modellbasierten MOEA (MOMBES) ausgenutzt.

5.4 Darstellung bekannter MOEA-Realisierungen

In diesem Abschnitt werden die bekanntesten Vertreter von MOEA beschrieben. In Kapitel 7 werden diese Ansätze in ausführlichen Benchmarks den typischen Testproblemen unterworfen. Es wird in Abhängigkeit davon, ob zur Fitnessberechnung und Selektion das Konzept der Pareto-Dominanz verwendet wird, zwischen nicht pareto-basierten und pareto-basierten Ansätzen unterschieden.

5.4.1 Nicht pareto-basierte Ansätze

5.4.1.1 „Vector Evaluated Genetic Algorithm" (VEGA) von Schaffer

Der erste für ein multimodales Optimierungsproblem angepaßte EA wurde schon 1984 von Schaffer als „Vector Evaluated Genetic Algorithm" (VEGA) in [Schaffer, 1984] und [Schaffer, 1985] vorgestellt. Sein Ansatz setzt auf einem Standard-GA des GENESIS-System von Grefenstetten [Grefenstette, 1984] auf und erweiterte dessen Selektionsmechanismen. Die stochastische Selektion wurde bei k unterschiedlichen Zielfunktionen in k Teilschritte zerlegt: bei jedem der k Teilschritte wurde ein gleich großer Anteil (k / N bei einer Populationsgröße N) von Individuen unter Berücksichtigung nur eines der k Zielkriterien fitnessproportional ausgewählt und in den mating-pool gespeichert. Diese MOEA-Technik verwendet eine sogenannte „Selektion mit wechselnden Zielen" (siehe dazu Abschnitt 5.2.2.2). Alle anderen Mechanismen, wie crossover und Mutation, werden standardmäßig ausgeführt und setzen auf diesem mating-pool auf.

Auf den ersten Blick erscheint diese Vorgehensweise erfolgversprechend, da Individuen, die mehreren Zielkriterien gut entsprechen, eine höhere Reproduktionswahrscheinlichkeit erhalten als Individuen, die nur für eine Zielfunktion minimale Werte generieren. Andererseits haben aber Individuen mittlerer Performance, d.h. die in keiner Zielfunktionsdimension minimal, aber in allen relativ nahe dem Minimum sind, weniger Chancen, sich zu reproduzieren, als überspezialisierte Individuen, die nur eine Zielfunktionsdimension minimieren. Mit diesem Ansatz können also die Eckpunkte der Pareto-Front eines MOP´s relativ sicher gefunden werden, jedoch nicht zwangsläufig die dazwischen liegenden Kompromißlösungen. In späteren Erweiterungen des VEGA-Systems wurde dieser Nachteil teilweise wieder ausgeglichen, indem spezielle Heuristiken zur „Durchmischung" des Genpools angewandt wurden. Trotzdem stellt VEGA keinen echten pareto-basierten Ansatz dar, da er im mating-pool eine Mittelwertbildung der Fitness entsprechend der einzelnen Teilziele realisiert:

> *... shuffling sub-populations together, or having different objectives affecting different tournaments, corresponds to averaging the fitness components associated with each of the objectives.* [Fonseca und Fleming, 1995] S. 8

Trotz dieser Nachteile hat das VEGA-System als erster Schritt bei der Entwicklung von MOEA eine große historische Bedeutung und erzielt bei Performance-Vergleichen erstaunlich gute Ergebnisse.

5.4.1.2 „Weighting-based Genetic Algorithm“ (HLGA) von Hajela und Lin

Ebenfalls einen nicht pareto-basierten Ansatz stellt der von P. Hajela und C.-Y. Lin in [Hajela und Lin, 1992] beschriebene „Weighting-based Genetic Algorithm“ (HLGA) dar. Dort wird ein weighted-sum-Ansatz zur Aggregation der einzelnen normierten und gewichteten Zielfunktionen verwendet. Die einzelnen Gewichtungsfaktoren werden mit in den Genotyp aufgenommen und nehmen so am Evolutionsprozess teil. Um die Diversität der Kombinationen der Gewichtungsfaktoren zu gewährleisten, wird auf ihnen ein phänotypisches sharing angewandt. Auf diese Weise wird eine parallele Suche mehrerer Lösungen - entsprechend der unterschiedlichen Kombination der Gewichtungsfaktoren - realisiert. Zusätzlich zum sharing der Gewichtungsfaktoren wird mating restriction als weitere Nischentechnik verwendet, um auch die Diversität der Lösungen selbst zu erhöhen. Als Selektionsmethode wird Wettkampfselektion eingesetzt. Um die Nachteile der Kombination aus sharing und Wettkampfselektion (siehe oben) zu umgehen, wurde das Standard-sharing in ein continuously updated sharing (siehe Abschnitt 5.2.2.3.) modifiziert. Vorteilhaft bei dieser Methode ist, daß bei der Auswahl der Lösung die entsprechend berechneten Gewichtungsfaktoren mit angegeben werden können – allerdings ist der praktische Nutzen dieser Informationen nicht sehr hoch anzusetzen. Innerhalb weniger Generationen werden mit diesem Ansatz gute Ergebnisse erzielt. Allerdings konvergiert er selbst bei großen Populationen sehr rasch auf eine einzelne Lösung und ist sehr sensitiv gegenüber Parametereinstellungen (z.B. dem Nischenradius). Da die Zielfunktionswerte normiert aggregiert werden, muß der Anwender eine ungefähre Vorstellung über deren Wertebereich besitzen.

5.4.1.3 „Multiobjective Evolutionary Strategy“ (MOES) von Kursawe

Die meisten entwickelten MOEA basieren auf dem Grundprinzip Genetischer Algorithmen - ein Fakt, der nicht in der Leistungsfähigkeit der einzelnen EA-Varianten begründet liegt (siehe Abschnitt 4.1.4), sondern hauptsächlich historisch bedingt ist. So ist die Anzahl von MOEA auf der Basis von ES weitaus geringer als die der GA-basierten MOEA, obwohl ES in ihrer Leistungsfähigkeit den GA mindestens ebenbürtig sind[59].

Als Beispiel für ein System zur Mehrzieloptimierung auf der Basis einer Standard-ES soll an dieser Stelle die von F. Kursawe in [Kursawe, 1991] beschriebene „Multiobjective Evolutionary Strategy“ (MOES) erwähnt werden. Ähnlich dem VEGA von Schaffer wurden auch hier die Standard-Selektionsmechanismen derart erweitert, daß jeder Auswahlschritt der Individuen entsprechend eines stochastisch gewählten Zielkriteriums erfolgt (siehe zur „Selektion mit wechselnden Zielen“ Abschnitt 5.2.2.2). Eventuelle Präferenzen und Zielgewichtungen lassen sich durch unterschiedliche Auswahlwahrscheinlichkeiten der Minimierung der einzelnen Zielfunktionen darstellen. Kursawe verwendet in seinem Ansatz diploide Individuen, d.h. Individuen, die entsprechend der Minimierung der einzelnen Zielfunktionen rezessiv einen

[59] Allerdings besitzen GA gegenüber anderen Formen von EA aufgrund ihrer Lösungskodierung einen gewissen Vorteil bezüglich der Gewährleistung der Diversität. Durch die Diskretisierung und binäre Kodierung von Entscheidungsvariablen und Zielfunktionswerte können sich die Lösungen nur auf einen gewissen Wert einander nähern.

Parametersatz gespeichert haben und sich so sehr schnell an sich ändernde Auswahlkriterien anpassen können, was zu einer erheblichen Performancesteigerung führt.

Da dieser Ansatz quasi auf einer gewichteten Mittelung der Komponenten der Zielfunktionen und nicht auf der Pareto-Dominanz beruht, kann die Kritik von Fonseca und Fleming an MOEA mit gewichteter Summenbildung auch auf diesen Ansatz direkt übertragen werden.

5.4.2 Pareto-basierte Ansätze

5.4.2.1 „Multiobjective Genetic Algorithm“ (MOGA) nach Fonseca und Fleming

Als erste nutzten Fonseca und Fleming [Fonseca und Fleming, 1993] einen pareto-basierten Ansatz zur Konstruktion eines MOEA in Form ihres „Multiobjective Genetic Algorithm“ (MOGA). Die Berechnung der Fitness eines Individums erfolgt hier aufgrund seines Pareto-Ranges (siehe Abschnitt 5.2.2.4), jedoch nicht wie bei Goldberg [Goldberg, 1989] direkt aus dem Pareto-Rang des Individuums berechnet, sondern hängt von der Anzahl der von ihm dominierten Individuen innerhalb der Population ab, was zu einer homogeneren Approximation der Pareto-Front führt [Thomas, 1998]. Als Selektionsmechanismus wird Wettkampfselektion, i.d.R. mit jeweils 2 Kandidaten, eingesetzt. Um zusätzlich die Diversität der Population zu erhöhen, wird im Zielraum ein sharing zwischen Individuen des selben Ranges verwendet. Fonseca und Flemming verwenden ein modifiziertes sharing, dessen Nischenradius nach jeder Generation neu bestimmt wird.

Fonseca und Fleming zeigen bei Erweiterungen ihres Ansatzes ([Fonseca und Fleming, 1998] S.15-16) die Möglichkeit, Zielpräferenzen in Form eines Goal-Vectors zu integrieren und so den Focus der Optimierung auf definierte Bereiche der Pareto-Front zu lenken.

5.4.2.2 „Niched Pareto Genetic Algorithm“ (NPGA) nach Horn und Nafpliotis

J.Horn und N.Nafpliotis kombinieren in ihrem in [Horn und Nafpliotis, 1993] vorgestellten „Niched Pareto Genetic Algorithm“ (NPGA) das Konzept der Pareto-Dominanz mit einer erweiterten Form der Wettkampfselektion. Bei der von ihnen verwendeten Selektion werden stochastisch sowohl zwei Wettkampfkandidaten, als auch eine Vergleichsmenge von Individuen aus der Population ausgewählt. Die Größe der zum Vergleich benutzten Subpopulation bestimmt den Selektionsdruck und wird durch den Parameter t_{dom} (meist in Prozent der Populationsgröße N angegeben) definiert. Wird einer der beiden Wettkampfkandidaten von keinem Individuum der Vergleichspopulation dominiert, geht er als Sieger der Selektion hervor. Werden dagegen beide Kandidaten nicht dominiert oder dominieren alle Individuen der Vergleichspopulation, wird der Kandidat ausgewählt, in dessen phänotpyischer Nische sich die wenigsten Individuen der Vergleichspopulation befinden.

Der NPGA gilt bei nicht zu groß gewähltem Nischenradius als einer der schnellsten MOEA, der in der Lage ist, eine den Zielraum breit abdeckende Approximation PF^* der gesuchten Pareto-Front zu generieren ([Coello, 1996] S.383). Nachteilig an diesem Ansatz ist, daß sich die beiden Parameter t_{dom} und σ_{share} sehr sensitiv auf die Konvergenz des Algorithmus und Gestalt von PF^* auswirken.

5.4.2.3 „Nondominated Sorting Genetic Algorithm“ (NSGA und NSGA2) nach Srinivas und Deb

N. Srinivas und K. Deb beschreiben in [Srinivas und Deb, 1994] einen weiteren pareto-basierten MOEA, den „Nondominated Sorting Genetic Algorithm“ (NSGA). In ihm wird das von Goldberg in [Goldberg, 1989] vorgeschlagene Schema zu Berechnung des Pareto-Rangs (siehe

Abschnitt 5.2.2.4) erweitert. Auch hier erfolgt die Berechnung der Fitness der Individuen in mehreren Stufen. Zunächst werden nach dem Schema von Goldberg alle nicht-dominierten Individuen identifiziert, zu einer Front zusammengefaßt und erhalten einen temporären Fitnesswert, der proportional zur Populationsgröße ist. Anschließend erfolgt zwischen den Individuen dieser ersten Front ein phänotypisches sharing im Raum der Entscheidungsvariablen unter Berücksichtigung eines definierten σ_{share}, es wird also sharing nur zwischen Individuen gleichen Rangs durchgeführt. Anschließend wird die temporäre Fitness auf einen Wert gesetzt, der geringer als der kleinste durch das sharing entstandene Fitnesswert ist und der Vorgang wiederholt sich für die folgende Front, ohne Berücksichtigung der Individuen denen bereits ein Fitnesswert zugeordnet wurde. Zusätzlich verwenden Deb und Srinivas eine stochastische fitnessproportionale Selektion in Form der roulette wheel selection [60].

Kritiker dieses Ansatzes kennzeichnen die Reduktion der einzelnen Zielfunktionswerte auf die temporäre Fitness als nachteilig ([Coello, 1996] S.115). Der NSGA ist in der Lage, eine Menge nicht-dominierter Lösungen zu generieren, diese sollte jedoch extern gespeichert werden, da sie nicht stabil über eine große Anzahl von Generationen (mehr als 100) innerhalb der Population repräsentiert wird. Durch das phänotypische sharing im Raum der Entscheidungsvariablen ist es möglich, Lösungen zu generieren, die für identische Zielfunktionen eine unterschiedliche Kombination der Entscheidungsvariablen zulassen, was für den Praktiker von großer Bedeutung sein kann. Auch bei diesem Ansatz gilt als Kritikpunkt, daß die Performance des Algorithmus' von der Wahl des Nischenradius σ_{share} abhängt, der problemabhängig justiert werden muß. Bei Berücksichtigung der Empfehlungen von Deb und Goldberg [Deb und Goldberg, 1989] (siehe Abschnitt 5.2.2.3) können jedoch sehr gute Ergebnisse erzielt werden [Deb, 2002].

Eine aktuelle Erweiterung des NSGA ist der ebenfalls von Deb et al in [Deb, 2000] vorgestellte NSGA2. In ihm wird im Gegensatz zum NSGA das Eliteprinzip verwendet und damit u.a. die Ergebnisse von Zitzler (siehe Abschnitt 5.4.2.4), daß Eliteselektion die Herausbildung von nicht-dominierten Individuen begünstigt, berücksichtigt. So wird die jeweils neue Generation nicht mehr wie bisher ausschließlich aus der Menge der Nachkommen, sondern aus der Vereinigungsmenge der Eltern- und Nachkommenindividuen generiert.

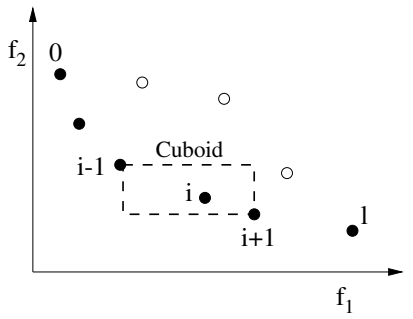


Abbildung 5.2: Crowding distance als mittelere Kantenlänge des durch die Nachbarelemente von i aufgespannten Cuboids (aus [Deb, 2000] S.5).

Das aufgrund seiner Parameterabhängigkeit kritisierte niching wird durch einen erweiterten Selektionsmechanismus ersetzt, bei dem die Nachbarschaft von Individuen mit identischem Pareto-Level (d.h. Individuen einer nicht-dominierten Front, siehe oben) durch eine sogenannte

[60] Srinivas und Deb folgen dem Ansatz von Hajela Lin zur Umgehung der bekannten Nachteile der Kombination aus sharing und Wettkampfselektion und verwenden statt des Standard-sharing´s ebenfalls das continuously updated sharing: siehe Abschnitt 5.2.2.3.

crowding distance berücksichtigt wird. Die crowding distance eines Individuums berechnet sich aus der mittleren euklidischen Distanz seines Zielfunktionswertevektors zu den Zielfunktionswertevektoren seiner unmittelbaren Nachbarindividuen (bei $k = 2$ Zielfunktionen entspricht das der mittleren Seitenlänge eines Rechtecks; siehe Abbildung 5.2) seiner Front. Um eine möglichst homogene Verteilung der Elemente von PF^* zu erreichen, werden bei der Selektion diejenigen Individuen begünstigt, deren crowding distance größer ist.

Durch den Einsatz von schnellen Sortieralgorithmen, konnte die Komplexität des Algorithmus' gesenkt und damit seine Effizienz gesteigert werden. Der NSGA2-Algorithmus zählt derzeit, neben dem SPEA2-Algorithmus zu den effektivsten MOEA.

5.4.2.4 „Strength Pareto Evolutionary Algorithm" (SPEA und SPEA2) von Zitzler

Der „Strength Pareto Evolutionary Algorithm" (SPEA) von Zitzler ([Zitzler und Thiele, 1998] und [Zitzler, 1999]) geht im Gegensatz zu den bisher vorgestellten MOEA streng von einem Maximierungsproblem aus. Auch Zitzler verwendet als Basis der Fitnessberechnung den Pareto-Rang von Lösungen. Jedoch werden nach jeder Generation alle nicht-dominierten Individuen aus der aktuellen Population P_t entfernt und mit ihnen eine externe Elitepopulation $\overline{P}_t$ aktualisiert. Damit die Größe der externen Elitepopulation eine definierte Anzahl von Elementen nicht überschreitet, verwendet Zitzler ein Clusterverfahren. Ist $\overline{P}_t$ aktualisiert, erhält jedes seiner Elemente j in Abhängigkeit davon, wieviele Individuen der Population P_t es schwach dominiert, eine Maßzahl (*strength)* s_j (siehe Gleichung [42]).

$$s_j = \frac{count}{N+1} \qquad [42]$$

Hierbei sei $count$ die Anzahl der von $\vec{a}_j$ schwach dominierten Individuen und N die Populationsgröße von P_t. Die Fitness $\Phi(\vec{a}_i)$ des Individuums $\vec{a}_i$ aus der Population P_t berechnet sich in Abhängigkeit von s_j derjenigen Elemente j aus $\overline{P}_t$, die $\vec{a}_i$ schwach dominieren (siehe Gleichung [43]).

$$\Phi(\vec{a}_i) = 1 + \sum_{j,\ a_j \preceq a_i} s_j \qquad [43]$$

Nach der Berechnung der Fitness erfolgen Selektion und Reproduktion nach dem Standard-GA, wobei die Selektion (Wettkampfselektion) auf der Vereinigungsmenge von P_t und $\overline{P}_t$ aufsetzt.

Der große Vorteil von SPEA ist, daß es gelungen ist, die Fitness der Individuen unter Berücksichtigung von Nachbarschaftsbeziehungen zu berechnen, ohne einen vom Anwender zu justierenden Parameter (wie den Nischenradius beim sharing) einzusetzen. Jedoch wird mit der Größe der Elitepopulation ein weiterer, auch sensitiver Parameter eingeführt (Zitzler gibt ihn i.d.R. mit ca. 20% der Grundpopulation an). Vorteilhaft ist, daß durch die Anwendung eines Clusterverfahrens eine Approximation der Pareto-Front mit über den Zielraum gleichmäßig verteilten Elementen generiert wird. Mit seiner Arbeit zeigt Zitzler, daß es für MOEA grundsätzlich günstig ist, das Eliteprinzip zu verwenden.

Eine aktuelle Weiterentwicklung des SPEA-Algorithmus wird mit dem SPEA2-Algorithmus in [Zitzler et al, 2001] vorgestellt. Die Erweiterungen betreffen die Berechnung der Fitness der Individuen und den Auswahlalgorithmus, der auf der Elitepopulation $\overline{P}_t$ angewandt wird. Bei SPEA2 wird bei der Berechnung der Fitness der Individuen aus P_t zusätzlich zur strength s_j

nach Gleichung [43] auch die Dichte der Population, in Form eines Dichtemaß $D(i)$, additiv berücksichtigt. Dabei wird $D(i) \in [0,1]$ nach dem „nearest neighbor"-Verfahren berechnet. So wird Individuen, die sich in einer Region hoher Dichte der Population befinden und denen nach Gleichung [43] u.U. die selbe Fitness zugeordnet wurde wie Individuen in Regionen geringerer Dichte, jetzt eine geringere Fitness zugewiesen. Auf diese Weise ist eine noch feinere Berechnung der Fitness möglich, was zu einer Erhöhung der Variabilität der Individuen innerhalb der Population führt. In SPEA2 wird der Clusteralgorithmus, der in SPEA zur Reduzierung der Anzahl der Elemente der Elitepopulation angewandt wird, durch einen einfachen Selektionsalgorithmus ersetzt: ist die Anzahl der dominanten Individuen größer als die maximal zulässige Anzahl der Individuen der Elitepopulation, werden diejenigen Individuen mit dem geringsten Dichtemaß aus $\overline{P}_t$ entfernt.

5.4.2.5 „Pareto Envelope-based Selection Algorithm" (PESA) von Corne und Knowles

Der „Pareto Envelope-based Selection Algorithm" (PESA) von Corne und Knowles stellt eine Weiterentwicklung der „Pareto Archived Evolution Strategy" (PAES) von Corne und Knowles [Knowles und Corne, 2000] dar und verwendet zur Berechnung der Fitness der Individuen eine sogenannte hyper-grid-Strategy. Dabei wird der Lösungsraum durch ein Hyper-Gitter in gleichmäßige Segmente aufgeteilt. Jedes Individuum erhält – ähnlich wie die strength bei SPEA – eine zusätzliche Maßzahl: den sogenannten *squeeze*-Faktor. Diese Maßzahl entspricht der Anzahl der Lösungen, die sich innerhalb desjenigen Hyper-Gitter-Segments befinden, daß dem Individuum zugeordnet werden kann. Ähnlich wie SPEA, arbeiten auch PAES und PESA auf zwei Populationen: einer internen Population IP und einer externen Population EP, die der Archivierung der dominanten Individuen dient. Die Größe der externen Population wird i.d.R. deutlich höher als die der internen Population gewählt z.B. im Verhältnis 10:1. Nach jedem Generationszyklus wird EP mit den dominanten Lösungen aus IP aktualisiert. Übersteigt die Anzahl der Individuen in EP einem bestimmten Wert, werden solange diejenigen Individuen mit dem größten squeeze-Faktor aus der externen Population entfernt, bis die erforderliche Größe der Population erreicht wurde. Die interne Population wird nach jeder Generation vollständig gelöscht und durch Nachkommen der folgenden Generation ersetzt. Die Selektion der Eltern im Generationszyklus erfolgt daher generell nur aus EP, d.h. aus der Menge der nicht dominierten Lösungen. Es wird Wettkampfselektion verwendet, wobei dasjenige Individuum selektiert wird, das den kleineren squeeze-Faktor besitzt. Crossover und Mutation werden nach dem Standard-GA durchgeführt.

PESA verwendet, ähnlich wie SPEA, konsequent das Eliteprinzip. Damit können zwar einmal gefundene nicht-dominierte Lösungen nicht mehr verloren gehen; jedoch besteht die Gefahr vorzeitiger Konvergenz durch die Verringerung der Variabilität der Population. Die Sensitivität des Algorithmus gegenüber der Parametrierung des Hyper-Gitters stellt einen weiteren Nachteil dieses Ansatzes dar. Trotzdem konnten bei Benchmarks [Corne et al, 2000] und [Zitzler et al, 2001] gute Ergebnisse im Vergleich mit SPEA erzielt werden.

5.4.3 Ausgewählte Hybride MOEA

5.4.3.1 Ansätze für die Kombination zwischen MOEA und lokalen Suchstrategien

D.Quagliarella und A.Vicini kombinieren in [Quagliarella und Vicini, 1997] den MOGA-Algorithmus mit einem lokalen hill-climbing-Operator[61], der während der evolutionären Suche mit einer gewissen Wahrscheinlichkeit auf Individuen der aktuellen Population aufsetzt. Da seine Auswahlwahrscheinlichkeit von der Fitness der Individuen und damit ihrer Dominanz abhängt, werden nicht-dominierte Individuen bevorzugt in die lokale Suche einbezogen. Individuen, die dominiert werden, werden jedoch von der Anwendung der lokalen Suche nicht vollständig ausgeschlossen, sondern lediglich mit geringerer Wahrscheinlichkeit ausgewählt.

K.Deb und T.Goel verwenden hingegen für ihren hybriden Ansatz in [Deb und Goel, 2001] den NSGA2-Algorithmus, der ebenso wie bei D.Quagliarella und A.Vincini mit einem lokalen hill-climbing-Operator verknüpft wird, der jedoch erst nach Durchlaufen des MOEA auf jedes Element der Approximation PF^* der Pareto-Front angewandt wird. Dabei wurde PF^* zuvor, ähnlich wie beim SPEA-Algorithmus, durch ein Clusterverfahren ausgedünnt. Bei der anschließenden Anwendung des lokalen Suchoperators auf den Elementen von PF^* sollten sich idealerweise die einzelnen Elemente von PF^* entlang eines hochdimensionalen Korridors in Richtung globaler Pareto-Front PF_{true} bewegen und dabei ihren relativen Abstand zueinander möglichst einhalten, d.h. es sollte ein „Verklumpen“ mehrerer Elemente von PF^* nach Anwendung des hill-climbing-Operators vermieden werden. Ob dies in jedem Fall gelingt, hängt zum einen von der Fitnesslandschaft selbst und zum anderen von der Leistungsfähigkeit des lokalen Suchoperators ab.

5.4.3.2 „Self-Organizing Maps for Multi-Objective Optimization“ (SOM-MOEA) von Büche et al.

Einen interessanten Ansatz der Kombination zwischen MOEA und NN haben D.Büche, M. Milano und P. Koumoutsakos mit ihrem SOM-MOEA in [Büche et al, 2002] vorgestellt. Sie gehen in ihrer Arbeit von einem NSGA2- und SPEA2-Algorithmus aus und verknüpfen diese mit einem Neuronalen Netz in Form einer selbstorganisierenden Merkmalskarte (SOM). In ihrer Arbeit erweitern sie das in [Milano et al, 2001] vorgestellte Konzept, eine SOM für Rekombinationsoperatoren eines EA zu verwenden, auf multikriterielle Optimierungsprobleme.

So belehren Büche et al eine SOM nach jeder Generation mit den Entscheidungsvariablen der Elterngeneration der aktuellen Population. Dabei wird die Dimension der Trainingsvektoren für die SOM der Anzahl der Entscheidungsvariablen des MOP´s gleichgesetzt. Die Dimension der Kopplungen innerhalb des Gitters (Kohonenkarte) entspricht der Dimension der Pareto-Front, die bei k Zielfunktionen die Dimension $(k-1)$ besitzt (siehe Abschnitt 2.1). Die Anzahl der Neurone auf dem Gitter variiert in Abhängigkeit von der Dimensionalität der Zielfunktionen (Büche et al geben für eine eindimensionale Kohonenkarte, also für 2 Zielfunktionen, 20 Neurone an). Die Belehrung der Kohonenkarte erfolgt nach den Standardverfahren für SOM (siehe Abschnitt 3.2.1 und [Kohonen, 1995]). Sowohl die verwendeten Mutations- als auch Rekombinationsoperatoren des MOEA setzen während der Generationszyklen auf der sich ständig aktualisierenden SOM auf. So wird bei der Rekombination zwischen zwei benachbarte

[61] I.d.R. werden unter diesem Begriff klassische, meist gradientenbasierte Optimierungsverfahren verstanden (siehe Abschnitt 4.2). Um diesen Ansatz lokal auf ein Individuum anwenden zu können, wird das MOP durch geeignete Aggregation der Zielfunktionswerte (siehe Abschnitt 5.2.1.2) in ein SOP überführt.

Gitterpunkte innerhalb der Kohonenkarte interpoliert. Das entspricht einer intermediären Rekombination auf der Elterngeneration. Die Mutation hingegen extrapoliert mit einer bestimmten Schrittlänge auf Vektoren von Entscheidungsvariablen, die sich außerhalb der Kohonenkarte abbilden würden. Die Schrittlänge wiederum hängt sowohl von einem Lernfaktor als auch einer Zufallskomponente ab.

Büche et al geben als Vorteil ihrer Methode an, daß es mit Hilfe der SOM gelingt, zwei „ähnliche“ Individuen für die Rekombination auszuwählen. Diese Herangehensweise erscheint jedoch nur zum Ende der Evolution hin sinnvoll. Gerade die Rekombination von unterschiedlichen Individuen führt i.d.R. zu großen „Evolutionssprüngen“, die so verhindert werden. Ein weiterer Kritikpunkt ist, daß für die Belehrung der SOM lediglich die Entscheidungsvariablen, nicht jedoch die Zielfunktionen verwendet werden. Eine Ähnlichkeitsbeziehung im Raum der Entscheidungsvariablen, die bei diesem Ansatz in einer Nachbarschaftsbeziehung auf der Kohonenkarte umgesetzt wird, impliziert hier auch eine Ähnlichkeit der durch sie bestimmten Vektoren im Zielraum, was nicht zwangsläufig der Fall ist. Benachbarte Individuen auf der Kohonenkarte entsprechen also nicht auch zwangsläufig Individuen mit ähnlichen Zielfunktionswerten. Das Ziel der multikriteriellen Optimierung besteht darin, eine Approximation PF^* der Pareto-Front zu generieren, die sich möglichst dicht an die globale Pareto-Front PF_{true} anschmiegt. Gerade die Homogenität der PF^* kann mit diesem Ansatz jedoch nicht zwangsläufig erhöht werden. Die SOM übernimmt hier lediglich die Funktion des Sortierens der Individuen bezüglich der Entscheidungsvariablen, d.h. dem Auffinden von Nachbarschaftsrelationen, was nicht ihrer Leistungsfähigkeit gerecht wird.

Der Einsatz von NN, die mit der aktuellen Population belehrt werden und es dem EA gestatten, dieses Wissen für die evolutionäre Suche auszunutzen, wird aus Sicht des Autors als sehr sinnvoll eingeschätzt. In [Meyer, 2000] wurde diese Idee bereits formuliert und mit dem in dieser Arbeit vorgestellten Ansatz MOMBES weiterentwickelt (siehe Abschnitt 6.4.7). Statt wie bei dem SOM-MOEA von Bücher et al „ähnliche“ Individuen zu rekombinieren, erscheint es effektiver, das durch die NN repräsentierte Wissen über die aktuelle Population, d.h. der aktuellen Approximation $PF^*(t)$ der Pareto-Front, dazu zu nutzen, um Regionen geringer Dichte auf $PF^*(t)$ systematisch zu verringern, was mit fortschreitender Iteration des MOEA zu einer homogeneren Approximation PF^* der Pareto-Front führt. Genau dieser Gedanke wurde mit dem MOMBES-System versucht zu realisieren.

6 MOMBES - Ein Verfahren zur modellbasierten Mehrzieloptimierung

In diesem Kapitel wird das System MOMBES: **M**ulti **O**bjective **M**odell **B**ased **E**volution **S**trategy detailliert beschrieben. Dabei werden zunächst die Anforderungen an das Verfahren formuliert. Anschließend wird das generelle Lösungskonzept der Methode, der Optimierungskern und das Decision-Making-Modul von MOMBES sowie seine Abgrenzung zu den in Abschnitt 5.4 beschriebenen MOEA dargestellt.

6.1 Anforderungen an das Verfahren

Folgende Anforderungen sollen durch das zu entwickelnde Verfahren zur Lösung multikriterieller Optimierungsprobleme erfüllt werden, die sich aus der praktischen Arbeit bei der Modellierung und Optimierung höherdimensionaler Prozesse (siehe Abschnitt 8.2) ergeben haben.

Es ist ein Verfahren zu entwickeln, mit dessen Hilfe kontinuierliche MOP´s gelöst werden können. Es soll aus einem Optimierungskern zur Generierung einer qualitativ hochwertigen Approximation PF^* von PF_{true} des MOP´s und einem Decision-Making-Modul bestehen, das im Anschluß den Zugriff auf die nicht-dominierten Lösungen realisiert. Eine Approximation PF^* einer Pareto-Front PF_{true} wird dann als qualitativ hochwertig eingeschätzt, wenn sie sich möglichst eng an PF_{true} des MOP´s anschmiegt und ihre Elemente möglichst breit und homogen im Zielraum verteilt sind[62]. Das zu erarbeitende System soll sich durch Robustheit auszeichnen und leicht auf neue, multikriterielle Optimierungsprobleme anpaßbar sein.

Die zu minimierenden Zielfunktionen und zu berücksichtigen Randbedingungen müssen nicht als mathematisch formulierbare Zusammenhänge vorgegeben sein, sondern können auch in Form von Beispielsituationen vorliegen. Es soll dann mit Hilfe von geeigneten Modellen die Abbildung des Prozesseingangs auf den Prozessausgang realisiert werden.

Das System soll sowohl lineare und nichtlineare Zielfunktionen minimieren als auch lineare und nichtlineare Restriktionen verarbeiten können. Es sind Lösungen, die den Randbedingungen genügen, denjenigen Lösungen vorzuziehen, die die Zielfunktionen minimieren. Die Anzahl der Randbedingungen und Zielfunktionen soll dabei nicht beschränkt sein. Es muß keine exakte Gewichtung der Zielfunktionen zu Beginn der Optimierung angegeben werden, die u.U. den Zielraum durch eine nicht geeignete Wahl dieser Gewichtungen zu stark einschränken würden. Statt dessen ist eine begrenzte Menge P^* von nicht-dominierten Lösungen (Kompromißlösungen) zu generieren, die eine Menge PF^* von nicht-dominierten Zielfunktionswertevektoren bestimmt, die möglichst homogen und breit im Zielraum verteilt sind und sich möglichst eng an die PF_{true} des Problems anschmiegen.

Nach Abschluß der Optimierung soll der Anwender interaktiv unter Berücksichtigung von konkreten Zielvorgaben Lösungen aus der durch das System generierten Menge nicht-dominierter Lösungen auswählen können. Es soll zusätzlich die Möglichkeit bestehen, auch auf Lösungen zuzugreifen, die Zielfunktionswertevektoren bestimmen, die sich zwischen den Elementen der Approximation der Pareto-Front befinden. Dieser kontinuierliche Zugriff auf P^*

[62] In Abschnitt 7.1 werden die gebräuchlichsten Kriterien zur Charakterisierung der Qualität einer Approximation einer Pareto-Front beschrieben.

und PF^* sei durch ein durch den Anwender parametrierbares Vertrauensmaß beschränkbar. Auf diese Weise kann die für den Anwender „ideale Lösung“ interaktiv nach Abschluß der Optimierung ermittelt werden. Es soll darüber hinaus möglich sein, interaktiv qualitative Aussagen über den Einfluß einzelner Entscheidungsvariablen auf einzelne Zielfunktionen sowie über eventuelle Abhängigkeiten der Zielfunktionen und Entscheidungsvariablen untereinander zu extrahieren. Eine geeignete Visualisierungstechnik soll die höherdimensionalen Zusammenhänge darstellbar machen. Durch diese Herangehensweise wird die interaktive Rolle des Anwenders unterstrichen. Während der Iteration soll die Optimierung möglichst zielgerichtet in Richtung homogen verteilter PF^* erfolgen. Dazu sollen ausschließlich Informationen aus denen zum aktuellen Iterationsschritt vorliegenden nicht-dominierten Lösungen verwendet werden.

6.2 Lösungskonzept

Im folgenden Abschnitt wird die in dieser Arbeit verwendete Methode zur Lösung von Problemen der Mehrzieloptimierung dargestellt. Um die in Abschnitt 6.1 beschriebenen Anforderungen erfüllen zu können, wurde ein mehrstufiges System entwickelt, das aus einem Modul zur Prozessmodellbildung, einem Optimierungskern und einem Decision-Making-Modul besteht. Während sich die Prozessmodellbildungskomponente am Verfahren der Modellierung mittels Neuronaler Netze (siehe Abschnitt 3.2.2) orientiert, wurde der Optimierungskern in Form eines modellbasierten, multi-hybriden MOEA realisiert, der sich insbesondere auf die Kombination von Evolutionsstrategien und Neuronalen Netzen stützt. Das Decision-Making-Modul, das die durch den Optimierungskern generierte Menge P^* und die duch sie bestimmte Menge PF^* modelliert, wurde auf Basis von Neuronalen Netzen zur Interpolation und qualitativen Datenanalyse realisiert. Optimierungskern und Decision-Making-Modul werden zum System MOMBES zusammengefaßt und stellen in der vorliegenden Form eine Neuentwicklung zur Lösung von kontinuierlichen MOP´s dar. Abbildung 6.1 zeigt das allgemeine Lösungskonzept zur modellbasierten Mehrzieloptimierung in der Prozessindustrie und die Einordnung von MOMBES in dieses Schema. Das System MOMBES und die Prozessmodellbildungskomponente sind zwei voneinander unabhängige Module.

Im Folgenden werden die einzelnen in der Abbildung dargestellten Module kurz vorgestellt und in den folgenden Abschnitten das System MOMBES näher beschrieben.

Gegeben sei ein Prozess, der sich durch eine Menge von Tupeln aus Eingangs- und Ausgangsgrößen (Beispielsituationen) oder spezielles Wissen in Form von Regeln und Berechnungsvorschriften über den Zusammenhang zwischen Eingangs- und Ausgangsgrößen darstellen kann (A). In Abhängigkeit davon, in welcher Form das Wissen über die Zusammenhänge zwischen Eingangs- und Ausgangsgrößen vorliegt, kann die Berechnung der Zielfunktionen und Randbedingungen mit Hilfe verschiedenster Modelle erfolgen. Im Rahmen der Gesamtlösungsmethode liegt die Motivation einer möglichst guten Modellierung der Zielfunktionen und Randbedingungen darin, diese im Verlaufe der evolutionären Suche stabil und mit hoher Modellsicherheit bei der Berechnung der Fitness der Individuen zu verwenden (B). Liegt das Prozessmodell vor, kann der Anwender flexibler Modellversuche durchführen, als dies am realen Prozess möglich ist (C). So können begrenzt Erkenntnisse über das Prozessverhalten in unbekannten Betriebspunkten gewonnen und verschiedenste Prozesssituationen simuliert werden (Prozess-Simulation), deren Durchführung in der Realität unter Umständen zu gefährlich oder zu komplex ist (D). Durch Erkenntnisse, die durch das Prozessmodell gewonnen werden, können gezielt spezielle Arbeitspunkte des Modells validiert (Versuchsplanung) und durch abermaliges Belehren des Modells seine sukzessive Verfeinerung erreicht werden. Durch Anwenden einer selbstorganisierenden Merkmalskarte können qualitative Zusammenhänge zwischen den an der Modellierung des Prozesses beteiligten

Größen gewonnen werden, indem Signalähnlichkeiten zwischen den Trainingsvektoren in Lageähnlichkeiten auf einer topologischen Karte umgesetzt werden (Data-Mining der Prozessdaten).

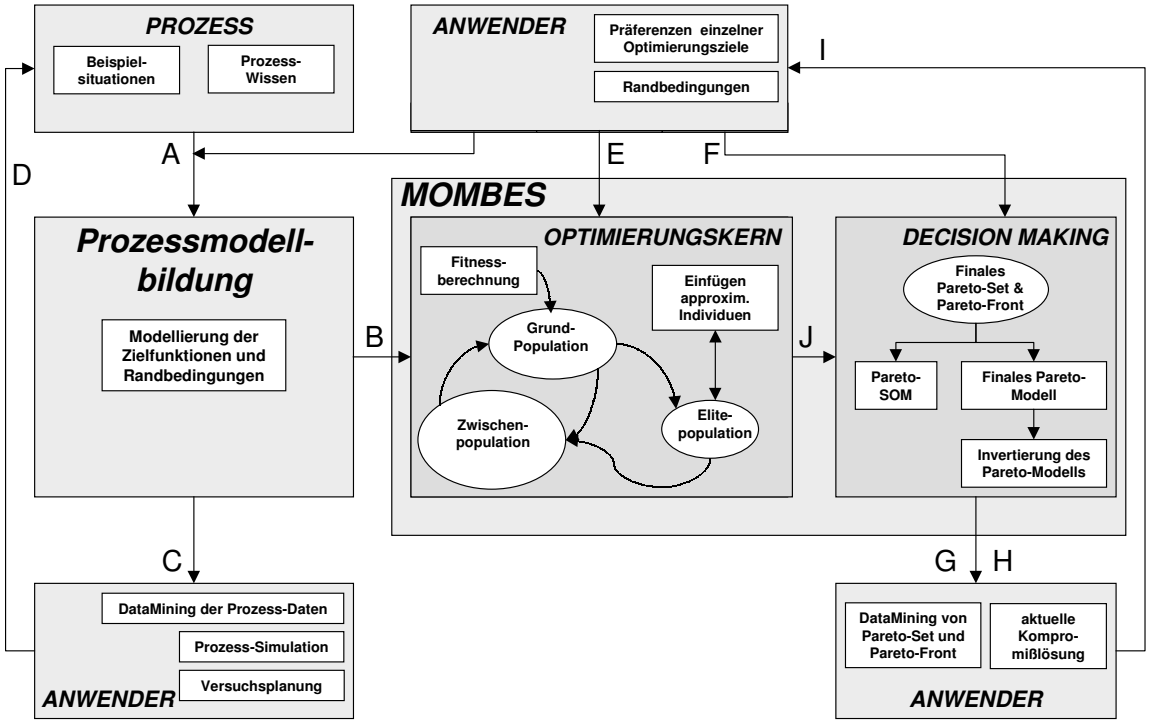


Abbildung 6.1: Methode der modellbasierten Mehrzieloptimierung

Die Modellbeschreibung der Zielfunktionen und Randbedingungen stellt die Grundlage für die Berechnung der Fitness der Individuen im Optimierungskern von MOMBES dar (B). Die multikriterielle Optimierung im System MOMBES selbst generiert über eine Anzahl von Iterationszyklen P^* und die duch sie bestimmte Approximation PF^* der Pareto-Front. Hierbei kann der Anwender einzuhaltende Randbedingungen z.B. in Form einer nicht zu unterschreitenden Vorhersagesicherheit des Prozessmodells oder bestimmter, einzuhaltender Wertebereiche der Prozessgrößen vorgeben (E).

In der Phase des Decision-Makings kann der Anwender unter Berücksichtigung von Präferenzen (F) seine Kompromißlösung aus der Menge der generierten nicht-dominierten Lösungen auswählen (H). Im einfachsten Fall erfolgt dies direkt durch die Auswahl einer Lösung aus P^*. Die durch den Optimierungskern generierte Menge P^* und die duch sie bestimmte Front PF^* werden jedoch darüber hinaus als Menge von Beispielsituationen interpretiert, die denjenigen Ausschnitt des Gesamtprozesses beschreiben, der grundsätzlich zur Ausprägung pareto-optimaler Lösungen führt. Für diesen Ausschnitt des Gesamtprozesses wird mit Hilfe eines NN, das mit den Elementen aus P^* als Modelleingang und den Elementen aus PF^* als Modellausgang belehrt wird, ein finales Pareto-Modell erzeugt. Durch Anwenden dieses Modells können unter Berücksichtigung von Vorgaben interpolierend Lösungen vorgeschlagen werden, deren Zielfuntionswertevektoren sich zwischen den Elementen von PF^* befinden. Zusätzlich können qualitative Zusammenhänge zwischen den Prozessgrößen durch den Einsatz einer SOM interaktiv durch den Anwender gewonnen (G) und so das Wissen des Anwenders über den Prozess verfeinert werden (I).

6.3 Berechnung der Zielfunktionen und Randbedingungen

Mit Hilfe der Modellbildungskomponente erfolgt die Abbildung der Prozesseingangsgrößen auf die Prozessausgangsgrößen[63]. Wie in Kapitel 1 dargestellt, müssen in Abhängigkeit davon, in welcher Form das Wissen über den Prozess vorliegt, verschiedene Modelle für diese Abbildung gewählt werden.

Liegt dieses Wissen wie bei den Beispielen aus Kapitel 7 und Abschnitt 8.1 in Form mathematischer Funktionen vor, ist also exakt analytisch beschreibbar, ist die Ermittlung eines speziellen Prozessmodells nicht notwendig. Hier werden die mathematischen Ausdrücke direkt für die Berechnung der Prozessausgangsgrößen (Zielfunktionswertevektoren) genutzt.

Wird das Prozessverhalten jedoch wie bei der in Abschnitt 8.2 dargestellten Anwendung nur durch Beispielsituationen, d.h. eine Menge von Arbeitspunkten des Prozesses beschrieben, muß ein datengetriebenes Modell ermittelt werden (Abbildung 6.2).

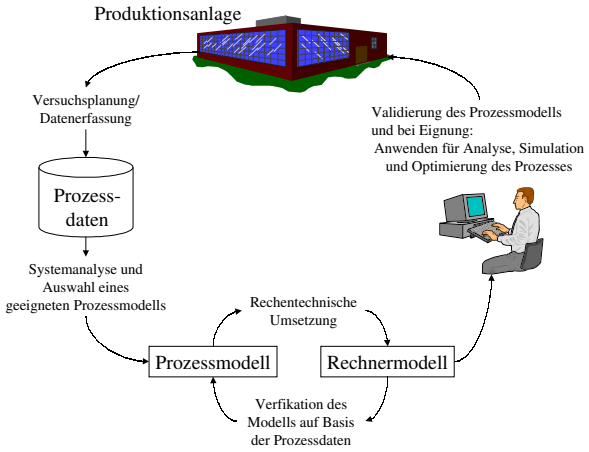


Abbildung 6.2: Erstellen eines Prozessmodells

Gerade in der chemischen und verfahrenstechnischen Industrie sind die zu modellierenden Prozesse meist nichtlineare, hochkomplexe Mehrgrößensysteme. Häufig ist das zur Verfügung stehende Datenmaterial ungleichmäßig über den hochdimensionalen Datenraum verteilt und unvollständig. Eine vollständige mathematische Beschreibung ist aus diesen Gründen für den Anwender vor Ort nicht mehr handhabbar oder sogar unmöglich. Ebenso ist das Aufstellen einer den Prozess vollständig beschreibenden Regelbasis sehr schwierig, stark vom know-how der Experten abhängig und führt bei sich ändernden Prozessbedingungen zu einem großen Aktualisierungsaufwand. Aus diesen Gründen ist die datengetriebenen Modellschätzung auf Basis belehrbarer Neuronale Netze u.U. die einzige Möglichkeit, eine Abbildungsvorschrift der Prozesseingangs- auf die Prozessausgangsgrößen zu erhalten und hat sich seit Mitte der 1980er Jahre als praxistaugliche Methode etabliert ([Butz, 1997], [Meyer und Eder, 1999] und [Sturm, 2000]). Da es sich um Standardverfahren handelt, wird hier auf die Darstellung der Grundlagen der Modellierung mittels Neuronaler Netze in Kapitel 3 und die ausführliche Darstellung einer industriellen Beispielanwendung zur datengetriebenen Modellbildung und Optimierung eines konkreten verfahrenstechnischen Prozesses in Abschnitt 8.2 verwiesen.

[63] Im Sinne der in Kapitel 2 vorgenommenen Definitionen werden die Prozesseingangsgrößen als Entscheidungsvariablen und die Prozessausgangsgrößen als Zielfunktionswerte interpretiert.

6.4 Beschreibung des Optimierungskerns

Die eigentliche multikriterielle Optimierung durch den Optimierungskern stellt den Hauptteil von MOMBES dar, für den ein spezieller MOEA entwickelt wurde. Dieser entspricht in seiner Grundstruktur dem generischen EA und wird in seinem Ablauf durch Abbildung 6.3 und Algorithmus 2 S. 75 illustriert.

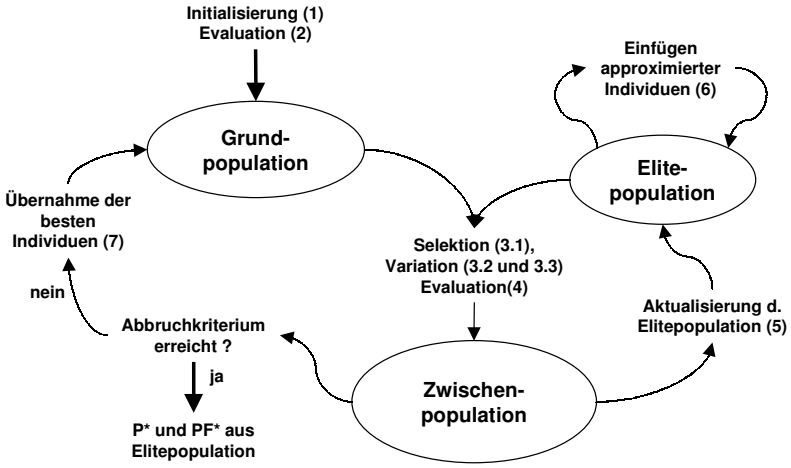


Abbildung 6.3: MOMBES-Optimierungskern, die eingezeichneten Nummern entsprechen den Schritten in Algorithmus 2.

Zunächst wird der prinzipielle Ablauf der Optimierung beschrieben und dann detailliert in den folgenden Abschnitten seine einzelnen Elemente dargestellt.

Initialisierung (Schritt 1 in Algorithmus 2 und Abschnitt 6.4.1): MOMBES arbeitet auf einer Grundpopulation $P(t)$ von N Individuen. Am Anfang der Iteration werden alle Individuen dieser Grundpopulation initialisiert, indem die Vektoren ihrer Entscheidungsvariablen mit zufälligen Werten belegt werden, die gleichverteilt über ihren Wertebereich sind. Die Elitepopulation $\overline{P}(t)$ ist leer.

Evaluation (Schritt 2 in Algorithmus 2 und Abschnitt 6.4.2): Anschließend werden alle Individuen aus $P(t)$ evaluiert, indem ihre Fitness berechnet wird. Der Optimierungskern von MOMBES ist ein MOEA, d.h. realisiert die simultane Minimierung mehrerer Zielfunktionen. Die Berechnung der Fitness der Individuen erfolgt auf Basis ihres Pareto-Rangs. Gleichzeitig wird mittels einer Penalty-Funktion die Einhaltung von Randbedingungen berücksichtigt. Zur Fitnessberechnung werden für jedes Individuum zunächst der Zielfunktionswertevektor und der Grad der Verletzung etwaiger Randbedingungen berechnet, die durch seine Entscheidungsvariablen bestimmt werden. Anschließend wird für jedes Individuum der Pareto-Rang festgelegt und die Fitnessfunktion angewandt. Da die Berechnung der Fitness auf Basis des Pareto-Ranges der Individuen erfolgt, kann diese immer nur für die gesamte Population und nicht für ein einzelnes Individuum durchgeführt werden.

Erzeugen neuer Individuen (Schritt 3 in Algorithmus 2 und Abschnitt 6.4.3 bis 6.4.5): In dieser Phase der Iteration erfolgt die Erzeugung neuer Individuen durch Selektion und Variation. Mit gleicher Wahrscheinlichkeit werden aus der Grund- und der Elitepopulation durch Wettkampfselektion Eltern bestimmt und durch Rekombination neue Individuen erzeugt, die

dann durch Mutation variiert und in die Zwischenpopulation abgelegt werden. Haben mehrere Individuen die gleiche Fitness, wird dasjenige selektiert, dessen euklidische Distanz im Zielraum zum nächsten Nachbarn der aktuellen Approximation der Pareto-Front am größten ist, um die Diversität der Population zu erhöhen. MOMBES ist als Evolutionäre Strategie mit Selbstanpassung der Muatationsschrittweiten für jede Entscheidungsvariable realisiert. Rekombination und Mutation der Entscheidungsvariablen der Individuen erfolgen nach dem Grundschema einer Standard-ES. Sind für alle Eltern alle Nachkommen erzeugt, werden die Individuen der Zwischenpopulation wie oben evaluiert.

Aktualisierung der Elitepopulation (Schritt 5 in Algorithmus 2 und Abschnitt 6.4.6): Es existiert neben der Grundpopulation $P(t)$ als Elite eine zweite Population $\overline{P}(t)$, die eine Auswahl aller bis zum Zeitpunkt t der Iteration erzeugten nicht-dominierten Individuen enthält. Die Elitepopulation $\overline{P}(t)$ wird nach jedem Generationszyklus aktualisiert, so daß in ihr immer nur maximal $\overline{N}$ Individuen enthalten sind, deren korrespondierende Zielfunktionswertevektoren möglichst breit und homogen im Zielraum verteilt sind. Die Individuen der Elitepopulation repräsentieren die aktuelle Approximation $P^*(t)$ der Pareto-Menge, die die aktuelle Approximation $PF^*(t)$ der Pareto-Front bestimmt.

Einfügen approximierter Individuen (Schritt 6 in Algorithmus 2 und Abschnitt 6.4.7): Eine Besonderheit von MOMBES besteht in seiner Fähigkeit, die Evolutionsrichtung gezielt in Richtung homogen verteilter Approximation der Pareto-Front zu beeinflussen. Dies erfolgt, indem nicht wie bei bisherigen Ansätzen ausschließlich über die Selektionsoperatoren, sondern aktiv in Regionen geringer Dichte und an den Eckpunkten von $PF^*(t)$ nach Lösungen gesucht wird. Dazu wird nach jeder Generationen mit einer bestimmten Wahrscheinlichkeit ein sogenanntes „approximiertes Individuum“ erzeugt. Dies geschieht, indem ein gradientenbasierter Optimierungsalgorithmus angestoßen wird, der eine Lösung generieren soll, die einen Zielfunktionswertevektor in einer Region geringer Dichte bzw. einer der Ecken der aktuellen Approximation der Pareto-Front bestimmt. Ist das approximierte Individuum gültig, werden die Zwischen- und die Elitepopulation mit ihm aktualisiert.

Übernahme der besten Individuen in die Grundpopulation (Schritt 7 in Algorithmus 2): Ist die Zwischenpopulation gefüllt, die Elitepopulation aktualisiert und das Einfügen approximierter Individuen erfolgt, ist der aktuelle Iterationszyklus abgeschlossen. In Abhängigkeit davon, ob das Abbruchkriterium erfüllt ist, wird die Optimierung beendet oder beginnt ein neuer Iterationsszyklus. Im Falle der Terminierung können die Ergebnisse in Form der Approximation der Pareto-Menge P^* und der durch sie bestimmten Approximation PF^* der Pareto-Front aus den Individuen der Elitepopulation extrahiert werden. Ist das Abbruchkriterium jedoch noch nicht erfüllt, wird die Grundpopulation gelöscht, werden die N Individuen mit der besten Qualität im Sinne des Pareto-Rangs und Einhaltung der Randbedingungen aus der Zwischenpopulation in die Grundpopulation übernommen und anschließend die Zwischenpopulation gelöscht. Der Iterationszyklus setzt sich dann in Schritt 3 von Algorithmus 2 – also dem Erzeugen einer neuen Zwischenpopulation fort.

In Algorithmus 2 wird der oben dargestellte, prinzipielle Ablauf der multikriteriellen Optimierung mit MOMBES in Form von Pseudo-Code beschrieben:

INPUT N : Größe der Grundpopulation
$\overline{N}$: Größe der Elitepopulation
$t_{\max}$: maximale Anzahl von Generationen
μ : Anzahl Eltern
λ : Anzahl Nachkommen
ρ : Anzahl Wettkämpfer
$\sigma^{(0)}$: Initialwert für alle Mutationsschrittweiten
p_a : Parameter, der die Wahrscheinlichkeit für das Einfügen approximierter Individuen bestimmt

OUTPUT P^* : Approximation der Pareto-Menge
PF^*: durch P^* bestimmte Approximation der Pareto-Front

1 Initialisierung der Grund- und Elitepopulation:
$t = 0$; $P(0) = \varnothing$; $\overline{P}(0) = \varnothing$
$for(i = 0; i < N; i++)$
$P(0) = P(0) + \{a_i\}$ mit $\vec{a}_i = \left(\vec{x}_i, \sigma_i^{(0)}\right)$ und $\vec{x}_i$ zufällig gleichverteilt in X

2 Fitnessberechnung $\forall\ \vec{a}_i \in P(t) : \Phi(\vec{a}_i) = FIT\left(\vec{a}_i, P(t)\right)$ nach Gleichung [45]

3 Erzeugen neuer Individuen und Ergänzen der Zwischen-Population:
$P' = \varnothing$
$for(i = 0; i < \lambda;\ i++)$
{

3.1 Selektion: siehe Algorithmus 3
$\vec{a}^{E1} = SELECT\left(\rho, (P(t) \cup \overline{P}(t)\right)$ und $\vec{a}^{E2} = SELECT\left(\rho, (P(t) \cup \overline{P}(t)\right)$

3.2 Rekombination: $\vec{a}^K = REKOM\left(\vec{a}^{E1}, \vec{a}^{E2}\right)$ siehe Algorithmus 4

3.3. Mutation: $\vec{a}^M = MUTAT(\vec{a}^K)$ siehe Abschnitt 6.4.5

3.4 Ergänzen der Zwischenpopulation: $P'(t) = P'(t) \cup \left\{\vec{a}^M\right\}$

}

4 Fitnessberechnung: $\forall\ \vec{a}_i \in P'(t) : \Phi(\vec{a}_i) = FIT\left(\vec{a}_i, P'(t)\right)$ siehe Gleichung [45]

5 Aktualisierung der Elitepopulation: $ACT\left(P'(t), \overline{P}(t)\right)$ siehe Algorithmus 6

6 Einfügen approximierter Individuen: $APPROX\left(p_a, \overline{P}(t)\right)$ siehe Algorithmus 7

7 Übernahme der λ besten Individuen aus $P'(t)$ in $P(t)$: $P'(t+1) = BEST\left(\lambda, P'(t)\right)$

8 $t = t + 1$

9 Wenn $t > t_{\max}$ oder anderes Abbruchkriterium erreicht ist, gehe zu Schritt 10 sonst gehe zu Schritt 3

10 Ausgabe der Ergebnisse: P^* und PF^* aus $\overline{P}(t)$

11 Stop

Algorithmus 2: Grundalgorithmus des MOMBES-Optimierungskerns

In den folgenden Abschnitten werden alle Elemente des Optimierungskerns detailliert beschrieben.

6.4.1 Initialisierung

In der Initialisierungsphase erfolgt die Vorbelegung der Startpopulation. Dabei werden bei MOMBES folgende Strategien verwendet:

Zufallsinitialisierung: Hier wird jede Entscheidungsvariable jedes zu initialisierenden Individuums mit einer über den Wertebereich der Entscheidungsvariable gleichverteilten Zufallszahl belegt. Ist der Wertebereich der Entscheidungsvariable nicht bekannt, sollte von einem gültigen Individuum ausgegangen werden und neue Individuen durch Addition einer normalverteilten Zufallszahl auf jede Entscheidungsvariable gewonnen werden. Typischerweise sind jedoch die Ober- und Untergrenzen der Entscheidungsvariablen bekannt, so daß sich die oben beschriebene Vorgehensweise empfiehlt.

Berücksichtigung von Präferenzen des Anwenders: Verfügt der Anwender bereits über Vorwissen über die Lage einzelner pareto-optimaler Lösungen im Entscheidungsraum, sollte dieses Wissen bei der Initialisierung der Population berücksichtigt werden z.B. indem bestimmte Bereiche von $X_{feasible}$ bei der Zufallsauswahl von $\vec{x}$ mit einer höheren Auswahlwahrscheinlichkeit belegt werden als andere, weniger relevante Bereiche des Wertebereichs.

Einerseits beschleunigt eine durch viel Vorwissen initialisierte Population die Optimierung, andererseits sollte aber der Anteil von Individuen, deren Entscheidungsvariablen zufällig gesetzt wurden, nicht zu gering sein, um die Verschiedenartigkeit der Population zu erhöhen und die Optimierung nicht künstlich nur in Richtung einer Lösungspräferenz und damit vorzeitiger Konvergenz zu lenken.

6.4.2 Berechnung der Fitness

Die Fitness eines Individuums stellt sein Selektionskriterium für den Evolutionsprozess dar. MOMBES verwendet bei der Berechnung der Fitness eines Individuums einen hierarchischen Ansatz. Dieser ist so gestaltet, daß der Grad der Verletzung der Randbedingungen und der Vektor der Zielfunktionswerte des MOP auf zwei verschiedenen Hierarchiestufen berücksichtigt werden: es wird der Einhaltung der Randbedingungen eine höhere Priorität zugeordnet als dem Auffinden der nicht-dominierter Lösungen und der durch sie bestimmten Approximation der Pareto-Front. Die Optimierung verläuft also zunächst in Richtung $Y_{feasible}$ und erst anschließend in Richtung PF_{true}. Eine solche Herangehensweise ist nur dann sinnvoll, wenn Lösungen, die den Randbedingungen genügen, höher zu bewerten sind, als Individuen, die sie verletzen – unabhängig vom Wert ihrer Zielfunktionen. Diese Aussage trifft auf harte Randbedingungen zu, die auf diese Weise in MOMBES integriert werden. Dagegen wird die Berücksichtigung weicher Randbedingungen bei MOMBES in Form der Erweiterung des Zielfunktionsvektors um diese Funktionen realisiert.

Bei der Berechnung der Fitness Φ eines Individuums $\vec{a}_i$ der Population P wird zunächst für alle Individuen der Population der (erweiterte) Zielfunktionswertevektor und die Verletzung der harten Randbedingungen berechnet. Anschließend wird für jedes Individuum der Population der Pareto-Rang nach Goldberg (siehe Abschnitt 5.2.2.4) und der dabei maximal auftretende Rang bestimmt (Gleichung [44]).

Individuen, die den Randbedingungen genügen, wird so unanhängig von ihrem Rang eine Fitness zugeordnet, deren Wert in jedem Falle über dem Fitnesswert desjenigen Individuums liegt, das den geringsten Rang besitzt und den Randbedingungen nicht genügt.

Nach Goldberg wird den nicht-dominierten Individuen einer Population der Rang 1 zugeordnet. Damit beträgt der nach Gleichung [44] maximal zu erreichende Fitnesswert ebenfalls 1 und wird

allen nicht-dominierten Individuen der Population, die den Randbedingungen entsprechen zugewiesen.

$$\Phi(\vec{a}_i) = FIT(\vec{a}_i, P) = \begin{cases} \dfrac{1}{rank(\vec{a}_i)}; & \text{für } \vec{x}_i \in X_{feasible} \\ \dfrac{1}{|P| + p(\vec{a}_i) + rank(\vec{a}_i)}; & \text{für } \vec{x}_i \notin X_{feasible} \end{cases} \qquad [44]$$

$$\text{mit:} \quad \vec{x} = \Gamma(\vec{a}) \quad \text{und} \quad p(\vec{a}) = \sum_{i=1}^{m} \delta(g_i(\vec{x})) + \sum_{i=1}^{l} |h_i(\vec{x})|$$

$$\text{sowie } \delta(g_i(\vec{x})) = \begin{cases} |g_i(\vec{x})| & \text{für } g_i(\vec{x}) < 0 \\ 0 & \text{für } g_i(\vec{x}) \geq 0 \end{cases} \qquad [45]$$

Wie in Abschnitt 5.2 (S. 55) beschrieben, garantiert allein pareto-basiertes Ranking nicht die für eine multikriterielle Optimierung notwendige Diversität einer Population. Vielmehr müssen zusätzlich diversitätserhaltende Techniken wie z.B. Nischentechniken (u.a. Fitness sharing und mating restriction) oder spezielle Selektionstechniken eingesetzt werden.

Aufgrund der Sensitivität der Parametrierung von Nischentechniken wurde im Optimierungskern von MOMBES als diversitätserhaltende Technik eine spezielle, die Verteilung der Individuen im Zielraum berücksichtigende Selektion und keine die Fitness beeinflussende Technik (wie z.B. Fitness-sharing) eingesetzt.

6.4.3 Selektion

Die Selektion erfolgt bei MOMBES nach dem Prinzip der Wettkampfselektion, die auf der Vereinigungsmenge der aktuellen Grundpopulation $P(t)$ und der Elitepopulation $\overline{P}(t)$ aufsetzt. Dabei werden zufällig ρ Individuen $\vec{a}_i$ ohne Zurücklegen in jedem Schritt gezogen und verglichen. Es gewinnt dasjenige Individuum, dessen Fitness im Vergleich zu den anderen ρ Wettkämpfern am größten ist. Ist die Fitness des besten und zweitbesten Individuums gleich, wird das Individuum selektiert, dessen euklidischer Abstand $dist$ im Zielraum zum nächsten Element der aktuellen Elitepopulation $\overline{P}(t)$ größer ist (Gleichung [45]).

$$dist(\vec{a}) = \min_{p=1}^{|PF^*|} \left\| \vec{f}(\vec{x}) - \vec{y}^{\,p} \right\| \quad \text{mit: } \vec{x} = \Gamma(\vec{a}) \qquad [46]$$

$$\text{sowie:} \quad \vec{y}^{\,p} = \vec{f}(\vec{x}^{\,p}) \in PF^* \;;\; \vec{x}^{\,p} = \Gamma(\vec{a}^{\,p}) \text{ und: } \vec{a}^{\,p} \in \overline{P}(t)$$

Diese Berücksichtigung der Verteilung der Individuen im Zielraum in Zusammenhang mit der speziellen Archivierungsfunktion der Elitepopulation ist eine einfache, jedoch sehr wirkungsvolle diversitäterhaltende Maßnahme. In den in Abschnitt 7.4 dargestellten Benchmarks wird gezeigt, daß mit MOMBES Approximationen einer Pareto-Front generiert werden können, die sich gerade hinsichtlich ihrer Diversität und Konvergenz durch eine hohe Qualität auszeichnen. Algorithmus 3 stellt den verwendeten Selektionsmechanismus schematisch dar.

$\vec{a}^E = SELECT(\rho, P)$

1 Wähle zufällig ρ Individuen $\vec{a}_i$ aus P aus

2 Finde das beste $\vec{a}^1$ und zweitbeste $\vec{a}^2$ Individuum aus $\{\vec{a}_1, \vec{a}_2, .., \vec{a}_\rho\}$ bezüglich Φ

3 Vergleiche die Fitness von $\vec{a}^1$ und $\vec{a}^2$:
if $(\Phi(\vec{a}^1) = \Phi(\vec{a}^2))$ { $\Phi(\vec{a}^1) = dist(\vec{a}^1)$ und $\Phi(\vec{a}^2) = dist(\vec{a}^2)$ }

4 *if* $(\Phi(\vec{a}^1) > \Phi(\vec{a}^2))$ $\vec{a}^E = \vec{a}^1$ *else* $\vec{a}^E = \vec{a}^2$

Algorithmus 3: Wettkampfselektion $\vec{a}^E = SELECT(\rho, P)$

6.4.4 Rekombination

Alle Nachkommen werden bei MOMBES durch Rekombination erzeugt. Dabei wird diskrete Rekombination für die Entscheidungsvariablen und intermediäre Rekombination für die Mutationsschrittweiten verwendet. Bei der diskreten Rekombination wird jede Komponente des Vektors der Entscheidungsvariablen mit einer bestimmten Wahrscheinlichkeit von einem der Eltern übernommen. Je größer der Fitnesswert des Elternindividuums ist, um so größer ist die Auswahlwahrscheinlichkeit für seine Komponente, da kleine Fitnesswerte i.d.R. bedeuten, daß das Individuum dominiert wird oder den Randbedingungen nicht entspricht. Dadurch soll die Wahrscheinlichkeit erhöht werden, daß der durch die Rekombination entstehende Nachkomme dem Elter mit großer Fitness ähnlicher ist als dem Elter mit geringerer Fitness.

Algorithmus 4 beschreibt die bei MOMBES eingesetzte Form der Rekombination in Pseudo-Code.

$\vec{a}^K = REKOM(\vec{a}^{E1}, \vec{a}^{E2})$

1 $p^{E1} = \dfrac{\Phi(\vec{a}^{E1})}{\Phi(\vec{a}^{E1}) + \Phi(\vec{a}^{E2})}$; $p^{E2} = \dfrac{\Phi(\vec{a}^{E2})}{\Phi(\vec{a}^{E1}) + \Phi(\vec{a}^{E2})}$

2 $for(i = 0; i < n;\ i++)\{$

$\sigma_i^K = \dfrac{\sigma_i^{E1} + \sigma_i^{E2}}{2}$

Erzeuge eine gleichverteilte Zufallszahl $rand \in [0,1]$

$if\,(rand < p^{E1})\ \ x_i^K = x_i^{E1}\ \ else\ \ x_i^K = x_i^{E2}$

}

Algorithmus 4: Rekombination zweier Individuen $REKOM(\vec{a}^{E1}, \vec{a}^{E2})$

6.4.5 Mutation

Alle rekombinierten Nachkommen werden bei MOMBES nach dem Schema einer Evolutionsstrategie mit Selbstanpassung der Mutationsschrittweiten mutiert. Die Mutation der Entscheidungsvariablen und Mutationsschrittweiten erfolgt daher nach Gleichung [28] und [29], die Parameter τ und τ' werden entsprechend den Empfehlungen aus Gleichung [28] gesetzt. Algorithmus 5 beschreibt die bei MOMBES eingesetzte Form der Mutation in Pseudo-Code. Im Unterschied zur Standardform einer ES wird hier jedoch jeweils nur eine zufällig ausgewählte Entscheidungsvariable variiert. Damit verringert sich gerade bei MOP´s mit einer sehr großen

Anzahl von Entscheidungsvariablen die Wahrscheinlichkeit, durch Mutation lethale Nachkommen zu erzeugen.

$\vec{a}^{\,M} = MUTAT(\vec{a})$

1 $\tau = \frac{1}{\sqrt{2 \cdot n}}$; $\tau' = \frac{1}{\sqrt{2 \cdot \sqrt{n}}}$

2 Wähle gleich verteilt zufällig eine Komponente i aus dem Vektor der n Entscheidungsvariablen aus

3 Erzeuge 3 reellwertige, $N(0,1)$ normal verteilte Zufallszahlen $rand_i$ mit $i = 1,2,3$

4 Variiere die Mutationsschrittweiten: $\sigma_i^M = \sigma_i \cdot \exp(\tau \cdot rand_1 + \tau' \cdot rand_2)$

5 Variiere den Wert der Entscheidungsvariable: $x_i^M = x_i + \sigma_i^M \cdot rand_3$

Algorithmus 5: Mutation eines Individuums $\vec{a}^{\,M} = MUTAT(\vec{a})$

6.4.6 Aktualisierung der Elitepopulation

Bei der Aktualisierung der Elitepopulation und damit der aktuellen Approximation $PF^*(t)$ der Pareto-Front, werden diejenigen Nachkommen der Zwischenpopulation, die nicht von Individuen der Elitepopulation dominiert werden, in diese übernommen. Anschließend erfolgt unter Berücksichtigung der Verteilung der Elemente im Zielraum eine Ausdünnung der Elitepopulation auf eine nicht zu überschreitende Anzahl von Elementen. Der verwendete Mechanismus wird in Algorithmus 6 dargestellt. Eine gezielte Einschränkung auf eine bestimmte Anzahl von Elementen der Elitepopulation ist notwendig, da sonst bei fortschreitender Anzahl von Generationen die Anzahl der Elemente von $\overline{P}(t)$ gegen unendlich streben könnte. Der Aufwand zur Archivierung würde die Effizienz des Algorithmus dann so stark einschränken, daß er nicht mehr handhabbar wäre. Durch das Ausdünnen von $\overline{P}(t)$ unter Berücksichtigung der Verteilung ihrer Elemente im Zielraum wird die Ausprägung einer Approximation von PF_{true} erreicht, deren Elemente im Zielraum breit und homogen verteilt sind.

1 Übernehme alle nicht-dominierten Individuen aus $P'(t)$ in $\overline{P}(t)$:
$\overline{P}(t) = \overline{P}(t) \cup \left\{ \vec{a}_i \in P'(t) \mid \neg \exists\, \vec{a}_j \in \overline{P}(t) : \vec{a}_j \prec \vec{a}_i \right\}$

2 Entferne alle dominierten Individuen aus $\overline{P}(t)$:
$\overline{P}(t) = \overline{P}(t) \setminus \left\{ \vec{a}_i \in \overline{P}(t) \mid \exists\, \vec{a}_j \in \overline{P}(t) : \vec{a}_j \prec \vec{a}_i \right\}$

3 $if\left(\left|\overline{P}(t)\right| > \overline{N} \right)$ reduziere $\overline{P}(t)$:
Entferne solange dasjenige Element mit der jeweils geringsten euklidischen Distanz zu einem anderen Element aus $\overline{P}(t)$ bis $\left|\overline{P}(t)\right| \le \overline{N}$

Algorithmus 6: Aktualisierung der Elitepopulation $ACT\left(P'(t), \overline{P}(t)\right)$

6.4.7 Einfügen approximierter Individuen

In diesem Schritt des Optimierungszyklus von MOMBES erfolgt das gezielte Erzeugen von Individuen, die sich im Bereich geringster Dichte und an den Ecken der aktuellen Approximation $PF^*(t)$ der Pareto-Front befinden, was die Herausbildung einer Menge PF^* mit homogen und

breit über den Zielraum verteilten nicht-dominierten, zulässigen Zielfunktionswertevektoren unterstützen soll. Dazu ist es zunächst notwendig zu klären, was unter den Begriffen „Bereich geringster Dichte“ und „Ecke“ einer Pareto-Front verstanden wird, wie diese identifiziert werden und wie es gelingen kann, in diese Regionen Individuen zu plazieren.

Der „Bereich geringster Dichte“ oder „größtes Loch“ sei der Bereich auf der $(k-1)$-Dimensionen hyperflächigen Pareto-Front, der sich zwischen denjenigen Nachbarelementen der Pareto-Front befindet, zwischen denen die größte Hyperfläche aufgespannt wird.

Unter einer Ecke einer Pareto-Front wird das Element der Pareto-Front verstanden, bei dem der Wert des Zielfunktionswertevektors in einer Dimension maximal und in allen anderen Dimensionen minimal ist. Ein MOP zur Minimierung von k Zielfunktionen erzeugt eine $(k-1)$-dimensionale Pareto-Front (siehe dazu auch Abschnitt 2.2) mit k Ecken. Der Rand einer Pareto-Front wird durch die Minimierung nur einer Zielfunktion des MOP erreicht. Die Begriffe „Loch“ und „Ecke“ einer Pareto-Front werden im folgenden als „Zielbereich“ bezeichnet. Abbildung 6.4 illustriert die oben erklärten Begriffe für ein MOP mit drei Zielfunktionen.

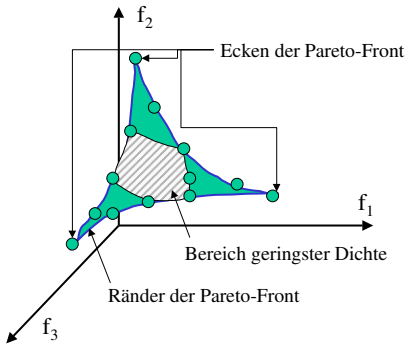


Abbildung 6.4: Ecken, Ränder und Bereich geringster Dichte einer 3-dim. Pareto-Front

Um Individuen zu generieren, die sich im Zielbereich (hier entweder im Bereich geringster Dichte oder an den Ecken der aktuellen Approximation $PF^*(t)$ der Pareto-Front) befinden, wird ein schnelles Gradientenabstiegsverfahren nach Levenberg/Marquardt (siehe Abschnitt 4.2.2, Schritt 6 in Algorithmus 7) eingesetzt. Dies ist sinnvoll, da an dieser Stelle das MOP in ein Problem mit nur einem Optimierungsziel, nämlich dem Erreichen des Zielbereichs, überführt werden kann. Der Startpunkt der Optimierung kann durch den Vektor der Entscheidungsvariablen desjenigen Individuums der Elitepopulation initialisiert werden, dessen Zielfunktionswertevektor die geringste euklidische Distanz zum Zielbereich aufweist.

Wie bereits in Abschnitt 4.2.4 und 5.3.1 dargestellt, ist es sehr effizient EA mit lokalen Suchstrategien wie z.B. Gradientenabstiegsverfahren zu kombinieren, um die Vorteile der jeweiligen Methoden zu nutzen. Dazu ist jedoch notwendig, das Optimierungsziel für das Gradientenabstiegsverfahren vorzugeben. In diesem Fall wird ein Ansatz zur Minimierung der gewichteten Summe der Zielfunktionswerte (siehe Abschnitt 5.2.1.2, S. 49) verwendet. Voraussetzung für eine solche Aggregation ist die Normierung der beteiligten Zielfunktionswerte, die hier auf den durch $PF^*(t)$ definierten Wertebereich erfolgt. Es sind also zunächst diejenigen Vektoren $\vec{w}^{Ziel}$ von Gewichtungsfaktoren (Zielgewichten) zu ermitteln, die bei Minimierung der in [47] angegebenen Summe solche Lösungen generieren, die Zielfunktionswertevektoren im Zielbereich bestimmen.

$$S = \sum_{i=1}^{k} w_i^{Ziel} f_i(\vec{x}) \quad \textit{mit:} \quad \vec{x} \in X_{feasible} \quad \textit{und} \quad \sum_{i=1}^{k} w_i^{Ziel} = 1 \qquad [47]$$

Zielgewichte für die Optimierung in Richtung „Ecken“: Das Auffinden neuer Elemente an den k Ecken von $PF^*(t)$ kann realisiert werden, indem jeweils nur eine Zielfunktion f_i minimiert wird. Dies ist zulässig, da bei MOP mit Zielfunktionskonflikten die Minimierung einer Zielfunktion i.d.R. die Maximierung der anderen Zielfunktionen bewirkt, was einer Optimierung in Richtung „Ecke“ der Pareto-Front entspricht. Stehen nicht alle f_i in Konflikt, wird mit der Minimierung einer einzelnen Zielfunktion der Rand der Pareto-Front (siehe Abbildung 6.4) angestrebt, was ebenso eine Ausdehnung der $PF^*(t)$ im Zielraum bewirkt und damit erwünscht ist. Der Vektor $\vec{w}^{Ziel}$ wird daher nach Gleichung [48] bestimmt, wobei der Index der zu minimierenden Zielfunktion zufällig gleichverteilt ausgewählt wird (Schritt 4.1 in Algorithmus 7).

$$\vec{w}^{Ziel} = \left[w_1^{Ziel},..,w_k^{Ziel}\right] \text{ mit } w_i = 1 \text{ und } w_j = 0 \ \forall \ j = 1,..,k \ \wedge \ j \neq i \qquad [48]$$

Zielgewichte für die Optimierung in Richtung „Bereich geringster Dichte“: Schwieriger als die Beschreibung einer Optimierung in Richtung „Ecke“ der Pareto-Front ist die Definition der Zielgewichte, die eine Optimierung in Richtung des „Bereichs geringster Dichte“ der aktuellen Approximation der Pareto-Front realisieren. Dazu muß zunächst geklärt werden, wie dieser Bereich beschrieben und lokalisiert werden kann.

Dazu gibt es verschiedene Möglichkeiten: Die einfachste und zugleich rechenaufwendigste Methode besteht darin, für alle Elemente alle Nachbarschaftsdistanzen im Zielraum zu berechnen und dann die k Nachbarelemente zu identifizieren, die die größte Hyperfläche zwischen sich aufspannen. Das einzufügende Element müßte dann genau auf den Schwerpunkt dieser Hyperfläche plaziert werden. Da diese Methode mit zunehmender Anzahl von Zielfunktionen, d.h. mit zunehmender Dimensionalität der Pareto-Front, und mit zunehmender Größe von $PF^*(t)$ ineffizient und sehr rechenaufwendig wird, wurde ein anderer Ansatz entwickelt.

Die Idee des hier entwickelten Ansatzes besteht darin, ein Clusterverfahren auf die aktuelle Approximation $PF^*(t)$ der Pareto-Front anzuwenden, das nicht nur die Clusterzentren der Elemente von $PF^*(t)$ sondern auch ihre topologische Beziehung zueinander ermittelt, um so die Nachbarschaftsbeziehungen zwischen den Clustern beschreiben zu können. Umgekehrt sollten auch diejenigen Bereiche im Zielraum und ihre Lage zu den Clusterzentren identifizierbar sein, die nicht durch die Elemente von $PF^*(t)$ repräsentiert werden. Ist eine solche Abbildung der Elemente von $PF^*(t)$ vorhanden, könnten „Löcher“ auf $PF^*(t)$ ermittelt werden, indem diejenigen nicht repräsentierten Bereiche identifiziert und beschrieben werden, die am weitesten von ihren benachbarten Clusterzentren entfernt sind.

Um diesen Gedanken umzusetzen, wurde eine topologische Karte in Form eines Neuronalen Netzes vom Typ selbstorganisierender Merkmalskarte (SOM - siehe Abschnitt 3.2.1) verwendet, die genau eine solche geforderte Abbildung realisiert. Mit Hilfe kompetetiven Lernens wird hier eine Menge von hochdimensionalen Trainingsvektoren aufgrund von Ähnlichkeiten zueinander derart auf die SOM projiziert, daß ähnliche Vektoren in einer gewissen Nachbarschaft auf der Karte abgebildet werden, d.h. Vektorähnlichkeiten in Lagenachbarschaften auf der topologischen Karte umgesetzt werden.

Die Neurone auf der belehrten SOM, die bei der Verarbeitung aller Trainingsvektoren mindestens einmal die maximale Aktivität besitzen, also jeweils Winner-Neurone sind, werden im folgenden als *Cluster-Neurone* bezeichnet. Umgekehrt werden alle diejenigen Neurone der belehrten SOM, die bei Anbieten aller Trainingsvektoren nie ein Winner-Neuron sind, als *No-Cluster-Neurone* bezeichnet.

SOM adaptieren während des Trainings die Kopplungsvektoren ihrer Neurone derart, daß diese möglichst genau auf die Zentren der Cluster der Trainingsvektoren positioniert werden. Die Kopplungsvektoren der anderen Neurone sind dabei aufgrund der Nachbarschaftserregung in den Adaptationsprozess einbezogen und werden daher durch die umliegenden Cluster-Neurone beeinflußt. Bei ungleichmäßig verteilten Daten, d.h. weit entfernten Clustern, sind auch auf der topologischen Karte die sie repräsentierenden Cluster-Neurone weit entfernt. Zwischen ihnen befinden sich No-Cluster-Neurone, die durch sie erregt werden. Die Kopplungsvektoren dieser No-Cluster-Neurone werden entsprechend der Nachbarschaftsfunktion (siehe Abschnitt 3.2.1) in Richtung der Kopplungsvektoren ihrer benachbarten Cluster-Neuronen adaptiert und positionieren sich in deren Zwischenräumen, in der Nähe der sie am stärksten beeinflussenden Cluster-Neurone [Kohonen, 1995].

Dieser Umstand wird bei dem hier verwendeten Ansatz zur Beschreibung des Zielbereichs ausgenutzt: es wird dasjenige No-Cluster-Neuron ermittelt, das sich möglichst weit entfernt von seinen benachbarten Cluster-Neuronen auf der Karte befindet. Zur Beschreibung der Entfernung der No-Cluster-Neurone bezüglich seiner auf dem SOM-Gitter benachbarten Cluster-Neurone wird ein Dichtemaß *density* eingeführt. Es berechnet sich für jedes No-Cluster-Neuron aus der Summe der Manhattan-Distanz[64] im SOM-Gitter zum jeweils nächsten Cluster-Neuron entlang der Gitterlinien abzüglich der Anzahl seiner auf dem Gitter unmittelbar benachbarten Cluster-Neurone. Zur Bestimmung des Zielbereichs wird dasjenige No-Cluster-Neuron ausgewählt, dessen density maximal ist. Bei mehr als einem No-Cluster-Neuron mit maximaler density wird zufällig eines dieser Neurone ausgewählt.

Der Kopplungsvektor des ausgewählten No-Cluster-Neurons repräsentiert von allen No-Cluster-Neuronen am besten den „Bereich geringster Dichte“ im Raum der Trainingsvektoren und wird daher zu dessen Beschreibung verwendet. Dieser Ansatz ist zulässig, da aufgrund der topologieerhaltenden Abbildung von der Lage der Neurone des SOM-Gitters auf die Lage der durch sie (in Form ihrer Kopplungsvektoren) repräsentierten Cluster im Raum der Trainingsvektoren geschlossen werden kann.

Abbildung 6.5 illustriert die oben beschriebene Vorgehensweise am Beispiel der Clusterung einer Menge von ungleichmäßig verteilten 3-dimensionalen Vektoren. In Abbildung 6.5 (links) sind die Koordinaten der Trainingsvektoren sowie die Kopplungsvektoren der Cluster- und No-Cluster-Neurone eingezeichnet.

In Abbildung 6.5 (rechts) wird die Gitter-Struktur der dabei verwendeten 4x4 SOM mit der density ihrer No-Cluster-Neurone dargestellt. Es ist zu erkennen, daß das Neuron mit der größten density (blau) den Vektor im Signalraum bestimmt, der die größte euklidische Distanz zu allen drei vorhandenen Clusterzentren (Ecken des Dreiecks) besitzt und sich damit von allen

[64] Unter der Manhattan-Distanz wird der durch die Metrik $d = |x-u| + |y-v|$ definierte Abstand zweier Punkte (x, y) und (u, v) verstanden [Skiena, 1990]. Bei diesem Ansatz wurde die Manhattan-Distanz aufgrund der Einfachheit ihrer Berechnung für die Identifikation des isoliertesten No-Cluster-Neurons innerhalb des SOM-Gitters eingesetzt; (x, y) bezeichnen hier die Gitterkoordinaten des betrachteten No-Cluster-Neurons und (u, v) des entlang der Gitterlinien nächsten benachbarten Cluster-Neurons.

No-Cluster-Neuronen dem „Bereich geringster Dichte“, der dem Schwerpunkt der drei Clusterzentren entspricht, am besten nähert; siehe Abbildung 6.5 (links).

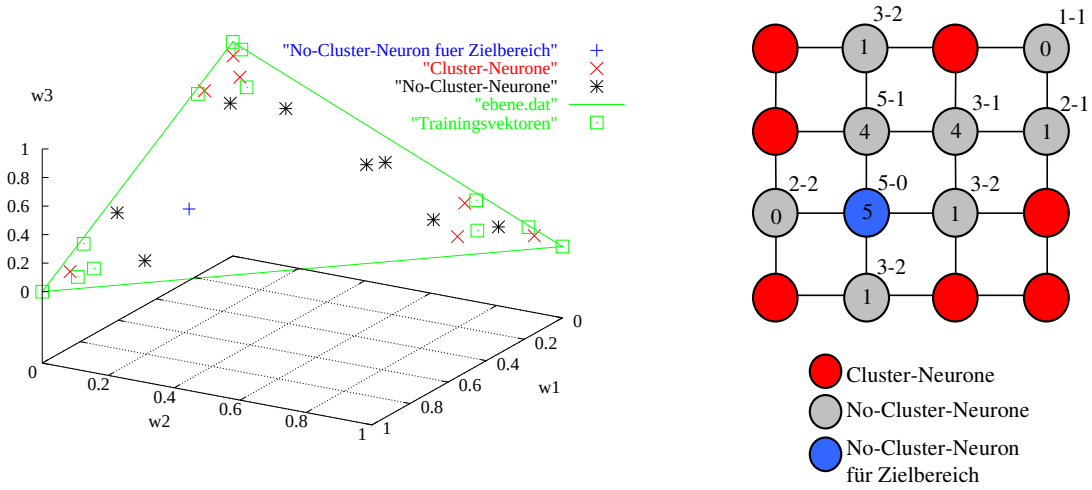


Abbildung 6.5: Trainings- und Kopplungsvektoren der Cluster- und No-Cluster-Neurone einer 4x4-SOM sowie die density der No-Cluster-Neurone

Abbildung 6.6 zeigt die Kopplungsvektoren der belehrten SOM in der Sammon-Projektion[65]. Man erkennt, daß die Kopplungsvektoren der Neurone der Karte bestrebt sind, sich in einer gleichmäßigen Gitterstruktur über den durch die Trainingsvektoren repräsentierten Datenraum anzuordnen, jedoch die Knoten an den Ecken der Karte dichter angeordnet sind als im mittleren Bereich. Daraus kann man schließen, daß es auch im Raum der Trainingsvektoren einen Bereich gibt, der sich zwischen den Clustern befindet und der schwach durch Datenpunkte repräsentiert wird.

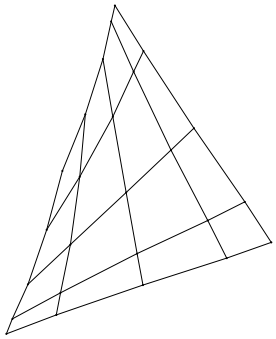


Abbildung 6.6 4x4 SOM in Sammon-Projektion

Die Idee, den „Bereich geringster Dichte“ durch Schwerpunktbildung der Kopplungsvektoren der Cluster-Neurone, die die meisten No-Cluster-Neurone einschließen, zu ermitteln, wurde als zu aufwendig verworfen. Der Grund dafür liegt darin, daß die Bestimmung des größten

[65] Mit Hilfe der Sammon-Projektion [Sammon, 1969] kann eine Menge höherdimensionaler Vektor auf eine zweidimensionale Ebene projiziert werden, wobei sich der Abstand zwischen den Knoten im dargestellten Netz der euklidischen Distanz der Vektoren nähert. Sie wird i.d.R. zur Visualisierung der Kopplungsvektoren von SOM eingesetzt und vermittelt einen Eindruck der Lage der durch die Neurone repräsentierten Cluster im Raum der Trainingsvektoren.

zusammenhängenden Gebietes von No-Cluster-Neuronen und der dieses Gebiet umgebenden Cluster-Neurone deutlich aufwendiger ist, als die oben dargestellte Methode auf Basis der Manhattan-Distanz der No-Cluster-Neurone.

Das hier beschriebene Verfahren zum Auffinden von Bereichen geringer Datendichte im mehrdimensionalen Signalraum soll nun angewandt werden, um den „Bereich geringster Dichte" auf der aktuellen Approximation der Pareto-Front zu identifizieren und die Zielgewichte vorzuschlagen, die eine Optimierung in Richtung dieses Bereiches wahrscheinlich machen.

Bei der Anwendung der SOM zur Clusterung von $PF^*(t)$ wird sie als Menge der Trainingsvektoren interpretiert. Da zwar $PF^*(t)$ geclustert werden soll, zur Zielbeschreibung für den Einsatz des Gradientenabstiegsverfahrens jedoch eine Kombination von Gewichtungsfaktoren $\vec{w}^{Ziel}$ gefordert ist, wird die $(k-1)$-dimensionale Menge $PF^*(t)$ zunächst in eine Menge von $(k-1)$-dimensionalen Pseudo-Gewichtsvektoren ihrer Elemente konvertiert. Dies erfolgt, indem aus jedem Element $\vec{y}^p$ aus $PF^*(t)$ nach der Vorschrift [49] ein Pseudo-Gewichtsvektor[66] $\vec{w}^p$ berechnet wird [Deb und Goel, 2001].

$$w_i^p = \frac{\dfrac{y_i^{\max} - y_i^p}{y_i^{\max} - y_i^{\min}}}{\sum_{k=1}^{M} \dfrac{y_k^{\max} - y_k^p}{y_k^{\max} - y_k^{\min}}} \quad \textit{bei } y_i^{\max} = \max_{k=1}^{M} y_i^k \textit{ und } y_i^{\min} = \min_{k=1}^{M} y_i^k \qquad [49]$$

$$\textit{mit} \quad \sum_{i=1}^{k} w_i^p = 1; \ \forall \ p \ \textit{und} \ \vec{y}^p \in PF^* \mid p = 1,..,M \le \overline{N} \ \textit{sowie} \ M = \left|PF^*\right|$$

Anschließend wird die Menge der so gewonnenen Pseudo-Gewichtsvektoren $\{\vec{w}^p\}$ auf eine topologische Karte (SOM) durch das Standardlernverfahren dieses Netzwerktyps (siehe Abschnitt 3.2.1) abgebildet. Ähnliche Kombinationen von Pseudo-Gewichtsvektoren werden nun durch auf der Karte benachbarte Cluster-Neurone repräsentiert. Danach werden alle Cluster- und No-Cluster-Neurone, für jedes No-Cluster-Neurone jeweils die density bestimmt und das No-Cluster-Neuron mit der maximalen density ermittelt. Sein Kopplungsvektor bestimmt den gesuchten Gewichtsvektor $\vec{w}^{Ziel}$.

Im folgenden werden 1-dimensionale SOM[67] für MOP mit $k = 2$ Zielfunktionen und SOM mit einer 2-dimensionalen Gitterstruktur für MOP mit $k > 2$ Zielfunktionen verwendet. Die Anzahl der Neurone der SOM sollte dabei etwa der Anzahl der Elemente von $PF^*(t)$ entsprechen. Die Wahl einer 1-dimensionalen SOM bei der Minimierung von nur zwei Zielfunktionen ist insofern sinnvoll, da die Abhängigkeit der Gewichte über alle Elemente p aus $PF^*(t)$ linear ist und die Kohonenkarte bestrebt ist, dieser linearen Funktion zu folgen, wohingegen ab $k \ge 3$ eine Projektion in die Ebene erfolgt. Die Elemente von $PF^*(t)$ können dann als homogen verteilt eingeschätzt werden, wenn bei gleicher Anzahl von Neuronen und Elementen von $PF^*(t)$

[66] Als Pseudo-Gewichtsvektor $\vec{w}^i$ kann der Vektor $\vec{w}$ interpretiert werden, mit Hilfe dessen man bei Minimierung der gewicheten Summe der Zielfunktionswerte $\sum w_i \cdot f_i(\vec{x})$ des MOP´s genau den Zielfunktionswertevektor $\vec{y}^i$ auf der Pareto-Front gewinnt, der seiner Berechnung zugunde liegt.

[67] Die Manhatten-Distanz wird dann nur noch in einer Dimension berechnet.

keine No-Cluster-Neurone identifizierbar sind. Mit der hier vorgeschlagenen Methode sind auch bei MOP mit vielen Zielfunktionen nur Nachbarschaftsbeziehungen der Cluster auf einem maximal zweidimensionalen Gitter und nicht auf der höherdimensionalen $PF^*(t)$ zu prüfen, was eine erhebliche Vereinfachung des Problems darstellt.

Den „Bereich geringster Dichte“ durch $\vec{w}^{Ziel}$ und nicht durch einen geschätzten Vektor von Funktionswerten $\vec{y}^{Ziel}$ auf der aktuellen $PF^*(t)$ zu beschreiben ist aus folgenden Gründen vorteilhaft: Das lokale Suchverfahren wird nicht gezwungen, in der Nähe der aktuellen und u.U. noch weit von PF_{true} entfernten aktuellen Approximation von PF_{true} zu verharren, sondern kann weiter in Richtung globaler Pareto-Front PF_{true} voranschreiten. Damit erhöht sich die Chance deutlich, daß durch die lokale Optimierung eine nicht-dominierte Lösung ermittelt wird. Dies gilt vor allem zu Beginn der Iteration des MOEA, da die aktuelle Approximation $PF^*(t)$ der Pareto-Front zu diesem Zeitpunkt i.d.R. noch relativ weit von der globalen Pareto-Front PF_{true} entfernt ist (siehe Abbildung 6.7).

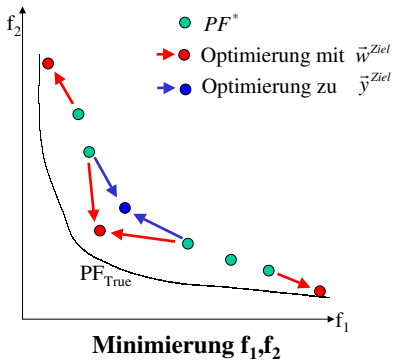


Abbildung 6.7 Optimierungsrichtung der lokalen Suche

Darüber hinaus erfolgt durch die Konvertierung der Elemente aus $PF^*(t)$ in die Menge ihrer Pseudo-Gewichtsvektoren eine Linearisierung der Vektoren[68], was die Modellierung der Pareto-Front durch die SOM erleichtert. Vorteilhaft ist weiterhin, daß $\vec{w}^{Ziel}$ direkt durch Zuordnung eines ausgewählten Kopplungsvektors aus dem Modell (SOM) der Pseudo-Gewichtsvektoren gewonnen werden kann und in jedem Falle eine Optimierung in Richtung Minimierung der Zielfunktionen und damit mit großer Wahrscheinlichkeit in Richtung globaler Pareto-Front bewirkt. Dagegen kann die Optimierung bei ungünstig gewählten $\vec{y}^{Ziel}$ durchaus in Bereiche streben, die nicht mehr pareto-optimal sind. Aus diesen Gründen ist die Verwendung der Pseudo-Gewichtsvektoren nicht nur vorteilhaft, sondern für eine Optimierung in Richtung globaler Pareto-Front auch notwendig.

Lokale Suche auf Basis der ermittelten Zielgewichte: Sind die Kombination von Zielgewichten $\vec{w}^{Ziel}$ für das Optimieren in Richtung der „Ecken“ und den „Bereich geringster Dichte“ nach der hier dargestellten Methode bestimmt, wird ein schnelles gradientenbasiertes

[68] Dies wird durch die Bedingung $\sum w_i = 1$ verursacht.

Suchverfahren nach Levenberg/Marquardt (siehe Abschnitt 4.2.2) zur Minimierung der in [47] angegebenen Summe angestoßen.

Im Ergebnis dieser lokalen Suche wird eine Lösung $\vec{x}^{Opt}$ [50] generiert, die einen Zielfunktionswertevektor $\vec{y}^{Opt}$ bestimmt, der sich möglichst dicht am Zielbereich befindet.

$$\vec{x}^{Opt} := OPT\_LM\left(\vec{w}^{Ziel}, \vec{x}^{Start}, \vec{f}(\vec{x})\right) \qquad [50]$$

Hierbei erfolgt der Start der lokalen Suche ausgehend von der Lösung $\vec{x}^{Start}$ die das Element aus $PF^*(t)$ bestimmt, für das durch [47] angegebene Optimierungskriterium der lokalen Suche minimal ist [51].

$$i := \arg\min_{p=1}^{|PF^*|}\left(\vec{w}^{Ziel} \cdot \vec{y}^{\,p}\right)$$
$$\vec{x}^{Start} = \vec{x}^{\,i} \in P^* \quad mit: \quad \vec{y}^{\,i} = \vec{f}(\vec{x}^{\,i}) \in PF^* \qquad [51]$$

Die aktuelle Approximation P^* der Pareto-Menge wird dabei durch die Entscheidungsvariablen der Individuen der aktuellen Elitepopulation, $\vec{x}^{Start}$ durch das Individuum $\vec{a}^{Start} \in \overline{P}(t)$ repräsentiert.

Nach Abschluß der lokalen Suche wird das einzufügende Individuum $\vec{a}^{Approx}$ generiert, indem die Werte der Entscheidungsvariablen von $\vec{a}^{Start}$ aus $\vec{x}^{Opt}$ übernommen werden. In Abhängigkeit von seiner Fitness wird entsprechend Algorithmus 6 entschieden, ob $\vec{a}^{Approx}$ der aktuellen Elitepopulation zugefügt wird.

In Abbildung 6.8 wird das Einfügen eines approximierten Individuums nach dieser Idee in den „Bereich geringster Dichte“ für ein MOP mit 2 Zielfunktionen schematisch dargestellt.

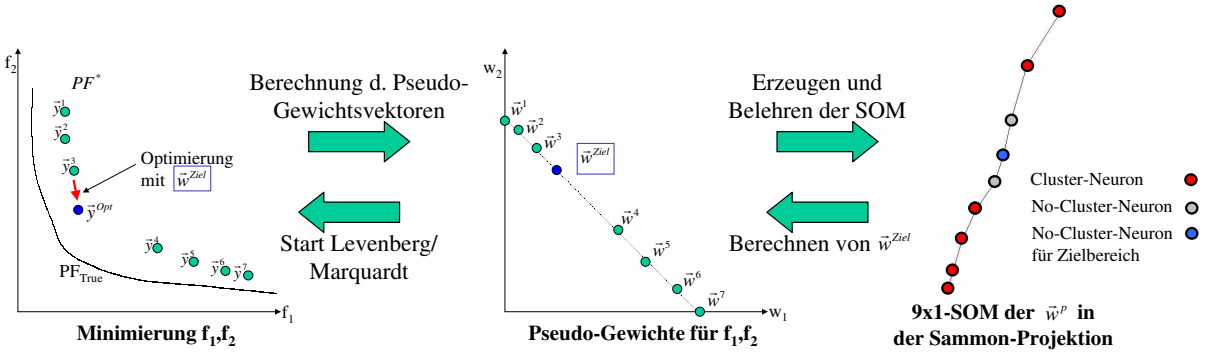


Abbildung 6.8: Schematische Darstellung der Generierung approximierter Individuen

Es kann nicht garantiert werden, daß das durch die lokalen Suche bestimmte Individuum $\vec{a}^{Approx}$ auch ein nicht-dominiertes Individuum bezüglich der aktuellen Elitepopulation ist. Die Ursachen dafür liegen in der fehlenden Optimalitätsgarantie der Optimierungsansätze auf Basis numerischer Aggregation der Zielfunktionen (siehe Abschnitt 5.2.1.2). Daß der hier beschriebene Ansatz dennoch zu einer signifikanten Verbesserung der Fortschrittsgeschwindigkeit des MOEA führt, zeigen die in Kapitel 7 dargestellten ausführlichen Benchmarks. Die Fortschrittsgeschwindigkeit wird dabei durch die Verbesserung der Qualität

der generierten Approximation PF^* der Pareto-Front bezüglich der Anzahl durchgeführter Berechnungen des Zielfunktionswertevektors (function calls) charakterisiert.

Es ist also davon auszugehen, daß der hier beschriebene Ansatz des Einfügens approximierter Individuen die Herausbildung einer qualitativ hochwertigen Approximation PF^* der globalen Pareto-Front PF_{true} unterstützt.

1	Erzeuge zwei gleich verteilte, reelle Zufallszahlen $rand_1 \in [0,1]$ und $rand_2 \in [0,1]$
2	If ($rand_1 < p_a$) {
3	If ($rand_2 > 0.5$) {
4	*// Gewichtungen für Optimierung in Richtung Ecken von* $PF^*(t)$
4.1	Wähle gleich verteilt, zufällig einen Index j aus $\{1,2,..k\}$
4.2	$\vec{w}^{Ziel} = [w_1^{Ziel},...,w_k^{Ziel}]$ mit: $w_i^{Ziel} = 0$ für $\forall i = 0,..,k \wedge i \neq j$ und $w_j^{Ziel} = 1$
	}
	Else {
5	*// Gewichtungen für Optimierung in Richtung geringster Dichte von* $PF^*(t)$
5.1	Berechne nach Gleichung [49] für jedes Element p aus $\overline{P}$ den Vektor seiner Pseudo-Gewichte: $\{\vec{y}^p\} \Rightarrow \{\vec{w}^p\}$
5.2	Belehre mit der Menge der $\{\vec{w}^p\}$ eine SOM
5.3	Identifiziere die Cluster- und No-Cluster-Neurone auf der SOM
5.4	Berechne für jedes No-Cluster-Neuron die density
5.5	Wähle zufällig ein No-Cluster-Neuron i maximaler density und setze $\vec{w}^{Ziel} = \vec{v}^i$, wobei $\vec{v}^i$ der Kopplungsvektor des Neurons i ist.
	}
6	Ermittele nach Gleichung [51] die Startlösung $\vec{x}^{Start}$ und das durch sie repräsentierte Startindividuum $\vec{a}^{Start}$ und erzeuge ein neues Individuum $\vec{a}^{Approx} = \vec{a}^{Start}$
7	Wende das Gradientenabstiegsverfahren nach Levenberg/Marquardt zur Minimierung der gewichteten Summe entsprechend Gleichung [47] an: $\vec{x}^{Opt} := OPT\_LM(\vec{w}^{LM}, \vec{x}^{Start}, \vec{f}(\vec{x}))$
8	Übernehme die durch LM berechneten Entscheidungsvariablen $\Gamma(\vec{a}^{Approx}) = \vec{x}^{Opt}$
9	Berechne die Fitness $\Phi(\vec{a}^{Approx})$
10	Ist $\vec{a}^{Approx}$ gültig, ergänze die Zwischen-Population und aktualisiere nach Algorithmus 6 die Elitepopulation mit $\vec{a}^{Approx}$
	}

Algorithmus 7: Einfügen approximierter Individuen $APPROX(p_a, \overline{P}(t))$

Im folgenden wird das Einfügen approximierter Individuen unter Bezugnahme der Darstellung der Methode in Algorithmus 7 in den einzelnen Schritten beschrieben:

Das Einfügen approximierter Individuen erfolgt nicht nach jedem Generationszyklus, sondern nur mit einer durch den Parameter p_a bestimmten Wahrscheinlichkeit (Schritt 2 in Algorithmus 7). Wird entschieden, daß approximierte Individuen eingefügt werden sollen, wird zunächst

festgelegt (Schritt 3 in Algorithmus 7), ob die Optimierung in Richtung Ecken (Schritt 4 in Algorithmus 7) oder Bereich geringster Dichte (Schritt 5 in Algorithmus 7) von $PF^*(t)$ erfolgen soll. Bei der Optimierung in Richtung Ecken wird zufällig eine Ecke ausgewählt und der Gewichtungsvektor $\vec{w}^{Ziel}$ gesetzt (Schritt 4.1 in Algorithmus 7).

Bei der Optimierung in Richtung „Bereich geringster Dichte" werden entsprechend der oben beschriebenen Methode zunächst für die Elemente aus $PF^*(t)$ deren Pseudo-Gewichtsvektoren berechnet (Schritt 5.1 in Algorithmus 7), mit ihnen eine SOM belehrt (Schritt 5.2 in Algorithmus 7), die Cluster und No-Cluster-Neurone identifiziert, das No-Cluster-Neuron mit der maximalen density ermittelt sowie $\vec{w}^{Ziel}$ mit den Kopplungsvektor dieses Neurons belegt (Schritt 5.3 bis 5.5 in Algorithmus 7).

Anschließend wird die Startlösung bestimmt (Schritt 6 in Algorithmus 7) und ein schneller gradientenbasierter Optimierungsalgorithmus nach Levenberg/Marquardt (LM) (Schritt 7 in Algorithmus 7, siehe Abschnitt: 4.2.2) angestoßen, der die Entscheidungsvariablen für das approximierte Individuum liefert (Schritt 8 in Algorithmus 7). Für das so entstandene approximierte Individuum wird seine Fitness berechnet (Schritt 9 in Algorithmus 7). Wurde ein gültiges Individuum erzeugt, wird es der Zwischen-Population zugefügt und mit ihm die Elitepopulation aktualisiert (Schritt 10 in Algorithmus 7).

Einfügen approximierter Individuen - Zusammenfassung: Der Optimierungskern von MOMBES besitzt die besondere Eigenschaft, die Verteilung der Elemente der aktuellen Approximation der Pareto-Front bereits während der evolutionären Suche mittels einer selbstorganisierenden Merkmalskarte zu modellieren. Durch das Anwenden dieses Modells kann die Evolution gezielt in die Richtung gesteuert werden, die für die Ausprägung einer qualitativ hochwertigen Approximation von PF_{true} erfolgversprechend ist. Es ist festzustellen, daß das gezielte Einbringen approximierter Individuen eine deutliche Erhöhung der Fortschrittsgeschwindigkeit der Optimierung bewirkte, d.h. sich bei gleicher Anzahl von function calls die Qualität[69] von $PF^*(t)$ erhöhte (siehe Abbildung 6.9 und Abschnitt 7.4). Es konnte beobachtet werden, daß gerade bei schwierigen multikriteriellen Problemen die gezielte Steuerung der Evolution durch den hier beschriebenen Ansatz vorteilhaft ist (siehe Abschnitt 7.4.2, 7.4.3 und 7.4.4). Diese Beschleunigung läßt sich allein durch eine Rekombination zweier dominanter Individuen pro Generation nicht erklären, sondern ist Resultat der gezielten Approximation.

In Abbildung 6.9 ist beispielhaft für das Testproblem QV-1 (siehe Abschnitt 7.4.3) die Abhängigkeit der Qualität der aktuellen Approximation der Pareto-Front (hier beschrieben durch die Größe des von den Elementen von $PF^*(t)$ dominierten Bereichs S im Zielraum[70]) von der durch den Parameter $p_a \in [0,1]$ bestimmten Wahrscheinlichkeit für das Einfügen approximierter Individuen dargestellt. Die Graphiken zeigen, daß sich bei diesem Beispiel das Einfügen approximierter Individuen generell günstig auswirkt: die Werte für S sind für $p_a > 0$ größer als die für S ohne das Einfügen approximierter Individuen, d.h. für $p_a = 0$. Jedoch existiert auch für den Wert des Parameters p_a ein Optimum: ist er zu groß gewählt, wird das

[69] Die betrachteten Qualitätskriterien zur Charakterisierung der Güte einer Pareto-Front werden in Abschnitt 7.1.2. beschrieben.

[70] Zur Definition von S siehe auch Abschnitt 7.1.2.

Gradientenverfahren zu häufig angestoßen – und damit eine Anzahl von function calls ausgeführt, ohne daß es zu einer Verbesserung der Güte von $PF^*(t)$ kommt. Ist der Wert für p_a jedoch zu gering gewählt, wird das Gradientenverfahren nicht häufig genug ausgeführt und kann sein Potential nicht entfalten und damit der Population nicht zu den in Abbildung 6.9 (links) gut sichtbaren Evolutionssprüngen verhelfen. Abbildung 6.9 (rechts) zeigt, daß sich für dieses MOP Werte für $p_a \in [0{,}05;0{,}3]$ günstig auf die Qualität der nach Abschluß der Iteration durch MOMBES generierten PF^* auswirken.

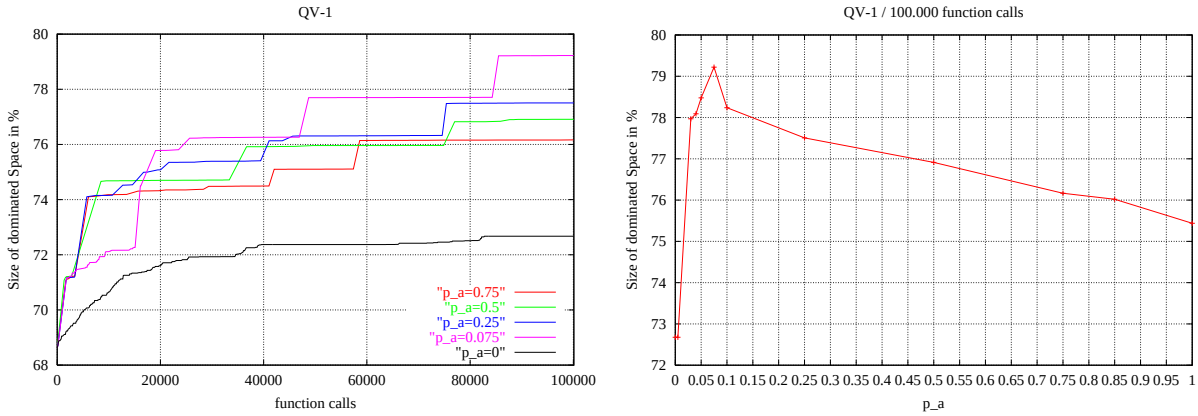


Abbildung 6.9: S für verschiedene Werte des Parameters p_a für das MOP QV-1

Wie gut sich die Anwendung des hier verwendeten gradientenbasierten Optimierungsverfahren bei der Lösung des MOP auswirken kann, hängt vom Problem einerseits und vom Startpunkt der Optimierung andererseits ab. Es kann daher von dem hier beschriebenen Verhalten bezüglich des Problems QV-1 nicht auf alle MOP verallgemeinert werden. Jedoch konnten für die in den Kapitel 7 und 8 beschriebenen multikriteriellen Probleme die in Abbildung 6.9 dargestellten Ergebnisse bestätigt werden und MOMBES mit einem Wert für $p_a = 0{,}2$ sehr gute Ergebnisse bei den vergleichenden Benchmarks erzielen (siehe Abschnitt 7.4). Der Wert für $p_a = 0{,}2$ wird daher als Empfehlung angegeben.

6.5 Decision Making

Das Decision-Making (DM) ist der zweite Hauptbestandteil von MOMBES. Hier wird dem Benutzer ermöglicht, Wissen über die Zusammenhänge der Elemente der durch MOMBES generierten Approximation P^* der Pareto-Menge und der duch sie bestimmten Approximation PF^* der Pareto-Front sowie Zusammenhänge innerhalb von P^* und innerhalb von PF^* zu extrahieren.

Weiter ist es möglich, interaktiv auf nicht-dominierte Lösungen zuzugreifen, ohne den Optimierungskern abermals zu durchlaufen. Das Decision-Making-Modul basiert dabei auf der durch den Optimierungskern generierten Lösungsmenge P^* und der durch sie bestimmten Menge PF^*. Es kann daher mit Hilfe seiner Modelle nur jenen Teil des Prozesses abbilden, der durch die näherungsweise pareto-optimalen Lösungen determiniert wird.

Das Decision-Making erfolgt in den 2 Schritten: der qualitativen Analyse (Abschnitt 6.5.1) der Mengen P^* und PF^* und dem eigentlichen Zugriff auf die näherungsweisen pareto-optimalen

Lösungen (Abschnitt 6.5.2). Beide Elemente des Decision-Makings basieren auf einer Modellierung der Elemente von P^* und PF^* mittels Neuronaler Netze und werden im Folgenden näher vorgestellt.

6.5.1 Qualitative Analyse der Approximation der Pareto-Menge und Pareto-Front

In der Phase der qualitativen Analyse des DM werden die durch den Optimierungskern generierten Ergebnisse interaktiv durch den Anwender analysiert. Ziel ist es dabei, in Form von Korrelationen diejenigen Zusammenhänge zwischen den Prozessgrößen zu finden, die für die Pareto-Menge und die Pareto-Front typisch sind, also den Teil des Prozesses beschreiben, der zur Ausprägung pareto-optimaler Lösungen führt (siehe Abbildung 6.10).

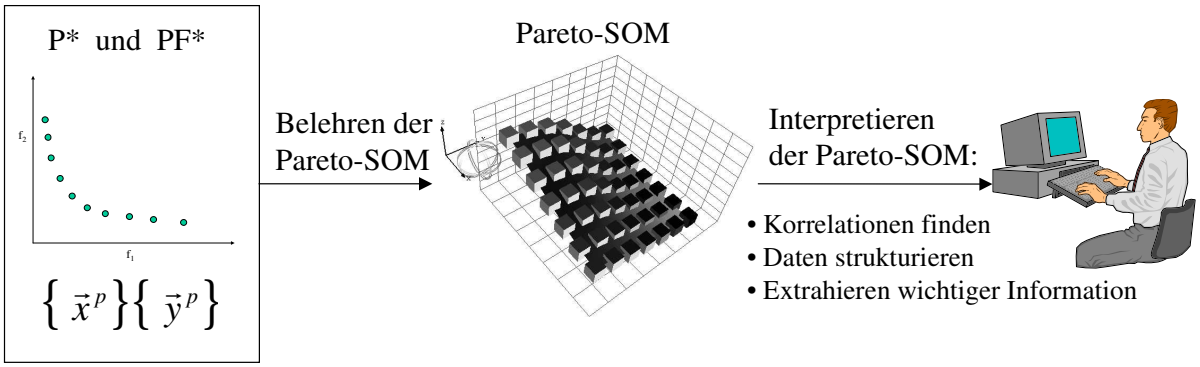


Abbildung 6.10: Qualitative Analyse der Pareto-Menge und der Pareto-Front

Dazu ist es notwendig, die Daten so zu strukturieren, daß diese Korrelationen extrahiert werden können. Diese Strukturierung wird durch ein Clusterverfahren mit Hilfe eines Neuronalen Netzes in Form einer SOM realisiert. SOM eignen sich wie oben dargestellt hervorragend zur Abbildung hochdimensionaler Zusammenhänge auf 2-dimensionale Projektionen und sollen aus diesem Grund auch an dieser Stelle des Systems MOMBES angewandt werden. Ihr Wirkprinzip und der hier verwendete Lernalgorithmus sind in Abschnitt 3.2.1 beschrieben.

Es wird zunächst eine 2-dimensionale, quadratische Kohonen-Karte mit rechteckiger Gitteranordnung der Neurone und gaussförmiger Nachbarschaftsaktivierung erzeugt und zufällig initialisiert. Anschließend wird diese Karte mit der Menge der Vektoren $\{\vec{t}^{\,p}_{SOM}\}$ belehrt, die sich aus der Menge der Vektoren $\vec{x}^p \in P^*$ und aus der Menge der Vektoren $\vec{y}^p \in PF^*$ zusammensetzen, die aus der aktuellen Elitepopulation extrahiert werden. Jeder Vektor $\vec{t}^{\,p}_{SOM}$ setzt sich dabei aus den Komponenten des Vektors $\vec{x}^p$ und den Komponenten des durch ihn bestimmten Vektors $\vec{y}^p$ zusammen (siehe Gleichung [52]).

$$\begin{aligned} \vec{t}^{\,p}_{SOM} &= \left[x_1^p, x_2^p, .., x_n^p, y_1^p, .y_2^p, .., y_k^p\right] \\ \text{mit: } \{\ \vec{x}^p &= \Gamma(\vec{a})\ \} = P^* \ \ \forall\ \vec{a} \in \overline{P}(t) \\ \text{und: } \{\ \vec{y}^p &= \vec{f}(\vec{x}^p)\} = PF^* \ \ \forall\ \vec{x}^p \in P^* \end{aligned} \qquad [52]$$

Die Initialisierung und Belehrung der Karte erfolgt nach dem in Abschnitt 3.2.1 dargestellten Standard-Verfahren nach Kohonen. Die Neurone der Karte sind in Form eines rechteckigen Gitters angeordnet. Die Anzahl der Neurone wird dabei so gewählt, daß sie geringer als die

Anzahl der Elemente von PF^* ist, um eine Abbildung von mehreren Elementen aus $\left\{ \vec{t}_{SOM}^{\,p} \right\}$ auf ein Clusterzentrum (Neuron der SOM) zuzulassen.

Um die qualitativen Zusammenhänge in möglichst vielen Dimensionen simultan zu beschreiben, wurde eine sogenannte Hyper-Cube-Visualisierung eingesetzt [Meyer und Eder, 1999], die auf der Darstellung der Kopplungsvektoren der SOM beruht. Die Neurone der 2-dimensionalen Kohonenkarte werden dabei als auf den Knoten eines Gitters positionierte Würfel dargestellt. Die Werte der Kopplungsvektoren können durch verschiedene Eigenschaften des Diagramms dargestellt werden: Größe der Würfel, Farbe der Würfelseiten, Höhe der Gitterpunkte sowie Hintergrundfarbe des Gitters. Bei der interaktiven Visualisierung wählt der Anwender jeweils eine Komponente des Signalraums aus und belegt diese mit einem der oben genannten Attribute der Karte z.B. der Höhe der Gitterpunkte oder der Würfelgröße. Entsprechend der Werte in der gewählten Dimension des Kopplungsvektors werden die visuellen Eigenschaften der Karte verändert. Dabei gilt: je stärker die Färbung, je größer Höhe des Gitters oder die Größe der Würfel, um so größer sind die Werte der Kopplungen in der ausgewählten Dimension. Werden nun nicht nur eine sondern mehrere Komponenten des Signalraumes mit verschiedenen Karteneigenschaften belegt, können aufgrund der Ausprägung dieser Eigenschaften auf der Karte Korrelationen erkannt werden. Auf diese Weise ist es möglich, Bereiche im Signalraum zu identifizieren, in denen sich Prozessgrößen ähnlich verhalten und diese Prozessgrößen zu identifizieren.

Die Seitenflächen der Würfel können mit 6 unterschiedlichen Farben eingefärbt werden. Die Hintergrundfarbe des Gitters kann mit drei verschiedenen Farben gefärbt werden, wobei Doppelbelegungen möglich sind, die Farben sich also mischen können.

Wie oben beschrieben repräsentiert jedes Neuron der SOM (und damit jeder Würfel im Gitterdiagramm) eine bestimmte Anzahl von Trainingsvektoren als deren Clusterzentrum. Durch die Zuordnung zwischen Würfeln und die sie repräsentierenden Trainingsvektoren $\left\{ \vec{t}_{SOM}^{\,p} \right\}$ sind diejenigen Trainingsvektoren identifizierbar, die sich auf ein Clusterzentrum abbilden und über gemeinsame Eigenschaften verfügen.

Das oben beschriebene Visualisierungskonzept ist im Rahmen der in Abschnitt 8.2 dargestellten industriellen Applikation mit den Programmen „OpenInventor[71]“ und „Visual Basic[72]“ realisiert worden (siehe Abbildung 6.12 und Abbildung 6.11). Das Programm OpenInventor gestattet mit Hilfe der Vektorgraphik, die Karte dreidimensional darzustellen und frei im Raum zu bewegen. Sowohl die Karte als auch die einzelnen Würfel können vom Anwender gedreht werden. Somit besteht die Möglichkeit, 11 Eigenschaften der Karte[73] und damit 11 Korrelationen[74] im Signalraum gleichzeitig darzustellen. Es ist nicht nur die Darstellung der Kopplungsmatrix realisiert, sondern es existiert darüber hinaus auch die Möglichkeit, diejenigen Datensätzen zu identifizieren, die durch einen oder mehrere auszuwählende Würfel repräsentiert werden. Damit ist es möglich, diejenigen Datensätze zu identifizieren, die ähnliche Eigenschaften besitzen und zu einem Cluster zusammengefaßt werden.

---

[71] Template Graphics Software Incorporation

[72] Microsoft Corporation

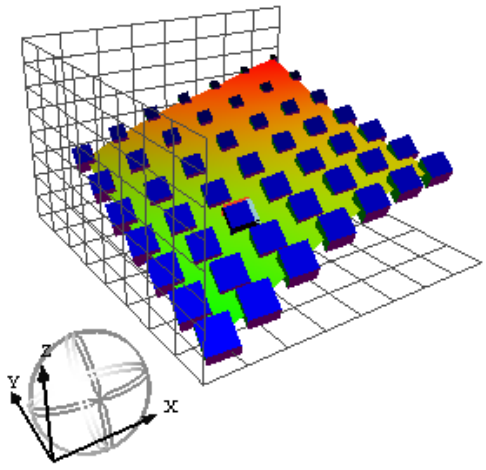
[73] Die Zahl 11 ergibt sich aus der Anzahl der darzustellenden Eigenschaften der Karte: Kartenhöhe, Würfelgröße, 3 Oberflächenfarben und 6 gefärbte Würfelflächen.

[74] Im Gegensatz zu der Beschreibung der Zusammenhänge zwischen den Prozessgrößen mittels der klassischen Korrelationsanalyse [Sachs, 1982] durch Korrelationskoeffizienten, können mit der hier beschriebenen Methode Cluster identifiziert und innerhalb dieser Cluster die Korrelationen einzeln dargestellt werden.

Durch die Anwendung der Visualisierungstechnik kann der Anwender folgende Information aus der belehrten SOM extrahieren: Identifikation von Clustern innerhalb von $\left\{ \vec{t}_{SOM}^{\,P} \right\}$ und Identifikation von Korrelationen zwischen den einzelnen Entscheidungsvariablen aus P^* und aus PF^* sowie den Größen untereinander.

Im folgenden wird die oben beschriebene Vorgehensweise am Beispiel des in Gleichung [6] (Seite 7) dargestellten MOP nach Fonseca und Fleming illustriert. Bei diesem MOP handelt es sich um ein leicht verständliches und übersichtliches Problem, so daß die Arbeitsweise dieses Teils von MOMBES klar dargestellt werden kann.

Nachdem mit dem Optimierungskern von MOMBES eine Approximation PF^* der Pareto-Front mit 100 Elementen generiert wurde, wurde eine Kohonenkarte mit 7x7=49 Neuronen in 300 Lernschritten mit einer Lernrate von 0,05 mit den Daten aus P^* und aus PF^* belehrt. Abbildung 6.11 zeigt die Darstellung der Kohonenkarte mittels der oben beschriebenen Visualisierungstechnik. Die Ausprägung der Attribute der Karte wurden den einzelnen Größen zugeordnet, die das MOP beschreiben, d.h. den Ausprägungen seiner zwei Entscheidungsvariablen sowie der beiden Zielfunktionen.



x_1: Würfelgröße

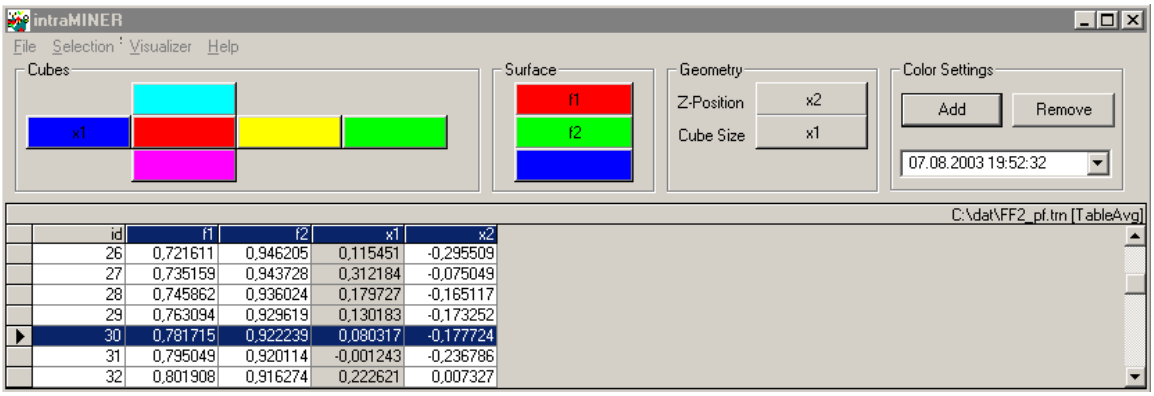
x_2: Höhe der Gitterpunkte

f_1: Rote Färbung der Kartenoberfläche

f_2: Grüne Färbung der Kartenoberfläche

Abbildung 6.11: Visualisierung der Kohonenkarte

Abbildung 6.12 zeigt durch die Zeilenmarkierung im Tabellenfeld denjenigen Datensatz an, der durch den in der Kohonenkarten selektierten Würfel repräsentiert wird.



id	f1	f2	x1	x2
26	0,721611	0,946205	0,115451	-0,295509
27	0,735159	0,943728	0,312184	-0,075049
28	0,745862	0,936024	0,179727	-0,165117
29	0,763094	0,929619	0,130183	-0,173252
30	0,781715	0,922239	0,080317	-0,177724
31	0,795049	0,920114	-0,001243	-0,236786
32	0,801908	0,916274	0,222621	0,007327

Abbildung 6.12: Zuordnung der Visualisierungskomponenten

Durch die Interpretation der Ausprägung der Attribute auf der visualisierten Kohonenkarte können folgende Schlüsse bezüglich der Korrelation der einzelnen Größen des MOP´s gezogen werden:

- Die rote Färbung der Kartenoberfläche ist dort stark ausgeprägt, wo eine grüne Färbung der Kartenoberfläche nicht auftritt und umgekehrt. Daraus kann interpretiert werden: *große Werte der Zielfunktion* f_1 *korrelieren mit kleinen Werte der Zielfunktion* f_2 *und umgekehrt*, d.h. die beiden Zielfunktionen des Problems stehen (erwartungsgemäß) in Konflikt.
- Die rote Färbung der Kartenoberfläche ist dort stark ausgeprägt, wo auch die Höhe der Karte stark ausgeprägt ist und umgekehrt: *große Werte der Zielfunktion* f_1 *korrelieren mit großen Werte der Entscheidungsvariable* x_2 *und umgekehrt*, d.h. zwischen f_1 und x_2 besteht eine positive Korrelation.
- Die rote Färbung der Kartenoberfläche ist dort stark ausgeprägt, wo die Größe der Würfel und ihre blaue Färbung schwach ausgeprägt ist und umgekehrt: *große Werte der Zielfunktion* f_1 *korrelieren mit kleinen Werte der Entscheidungsvariable* x_1 *und umgekehrt*, d.h. zwischen f_1 und x_1 besteht eine negative Korrelation.
- Die grüne Färbung der Kartenoberfläche ist dort stark ausgeprägt, wo die Höhe der Karte schwach ausgeprägt ist und umgekehrt: *große Werte der Zielfunktion* f_2 *korrelieren mit kleinen Werte der Entscheidungsvariable* x_2 *und umgekehrt*, d.h. zwischen f_2 und x_2 besteht eine positive Korrelation.
- Die grüne Färbung der Kartenoberfläche ist dort stark ausgeprägt, wo die Größe der Würfel und ihre blaue Färbung stark ausgeprägt ist und umgekehrt: *große Werte der Zielfunktion* f_2 *korrelieren mit großen Werte der Entscheidungsvariable* x_1 *und umgekehrt*, d.h. zwischen f_2 und x_1 besteht eine negative Korrelation.

Anhand dieses Beispiel konnte illustriert werden, wie einfach allein auf Basis der Auswertung der Kohonenkarte mit der vorgeschlagenen Methode die prinzipiellen Zusammenhänge zwischen den Entscheidungsvariablen und Zielfunktionswerte sowie der Größen untereinander beschrieben werden können und der Anwender qualitatives Wissen über den Bereich des Ziel- und Lösungsraums extrahieren kann, der die pareto-optimalen Lösungen und die Pareto-Front bestimmt.

6.5.2 Zugriff auf die näherungsweise pareto-optimalen Lösungen

In dieser Phase des Decision-Makings von MOMBES erfolgt die eigentliche Auswahl der näherungsweise pareto-optimalen Lösungen durch den Anwender. Da durch den Optimierungskern nicht einzelne Lösungen, sondern eine Menge von nicht-dominierten Lösungen zu Verfügung gestellt werden, muß der Anwender aufgrund von Präferenzen entscheiden, welches der Elemente aus PF^*, und damit auch welches Element aus P^*, er auswählt. Der durch den Anwender auszuwählende Ziel-Punkt auf PF^* soll im Folgenden als Target-Vektor $\vec{y}^{target}$, seine ihn bestimmende Lösung als Target-Lösung $\vec{x}^{target}$ bezeichnet werden. Bei der Charakterisierung des Target-Vektors wird der Anwender durch verschiedene Visualisierungstechniken unterstützt. Ist der Target-Vektor durch den Anwender beschrieben, wird durch das Decision-Making-Modul von MOMBES eine Lösung und ein Zielfunktionswertevektor vorgeschlagen, der Vorgaben des Anwenders möglichst gut entspricht. Abbildung 6.13 zeigt den Zugriff auf die nicht-dominierten Lösungen in seinen einzelnen Schritten.

Die Schwierigkeiten eines multikriteriellen Decision-Makings bestehen darin, zum einen den Target-Vektor derart zu definieren, daß er tatsächlich näherungsweise auf der Pareto-Front liegt, und zum anderen Lösungen zu generieren, die nicht nur durch die Elemente der Approximation P^* der Pareto-Menge repräsentiert werden, sondern auch im Zielraum zwischen ihnen positioniert sein können. Beide Probleme können durch den im Folgenden beschriebenen Ansatz gelöst werden.

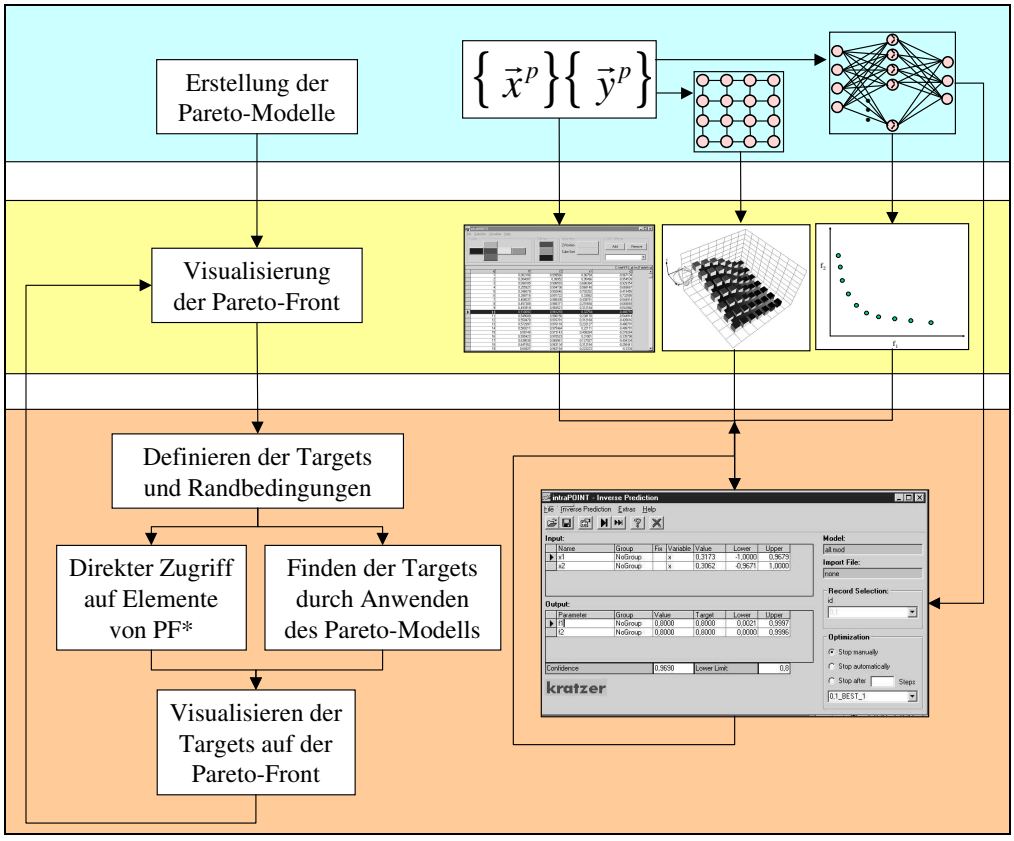


Abbildung 6.13: Zugriff auf die näherungsweise pareto-optimalen Lösungen bei MOMBES

Zunächst werden die Vektoren aus P^* und PF^* visualisiert, um für den Anwender Zusammenhänge zwischen den Zielfunktionen und Entscheidungsvariablen im näherungsweise pareto-optimalen Bereich des Lösungs- und Zielraumes sichtbar zu machen. Damit diese Darstellung nicht auf die Vektoren von P^* und der PF^* beschränkt bleibt, werden bei MOMBES sogenannte Pareto-Modelle verwendet, die die Abhängigkeiten zwischen den einzelnen Entscheidungsvariablen und Zielfunktionswerten im näherungsweise pareto-optimalen Bereich modellieren und so eine interpolierende Darstellung der Elemente von PF^* und P^* möglich machen. Ziel der Visualisierung ist es, den Target-Vektor in jeder Komponente möglichst genau zu definieren.

6.5.2.1 Schätzen des Target-Vektors durch Pareto-Modelle

Im einfachsten Fall des Decision-Makings, greift der Anwender direkt auf eines der durch den Optimierungskern generierten Elemente der Approximation PF^* der Pareto-Front und damit auf eine Lösung aus P^* zu. Sucht er jedoch nach einem Target-Vektor, der nicht direkt durch die

Elemente aus PF^* repräsentiert wird, wird der Entscheidungsprozess schwieriger. In diesem Fall ist es notwendig, die Elemente des Target-Vektors zu schätzen, was ein Wissen über die Zusammenhänge zwischen den Zielfunktionen in dem Bereich von $Y_{feasible}$ voraussetzt, der von den nicht-dominierten Lösungen bestimmt wird. Gibt der Anwender beispielsweise lediglich nur eine Komponente des Target-Vektors vor, kann u.U. nicht eindeutig entschieden werden, welche anderen Komponenten notwendig sind, um nicht nur gültige Lösungen, sondern eben auch pareto-optimale Lösungen zu generieren.

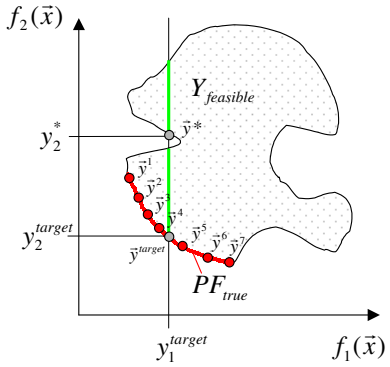


Abbildung 6.14: MOP mit zwei Zielfunktionen

Abbildung 6.14 zeigt ein Beispiel, bei dem nur allein auf Basis des MOP´s – also der Abbildung von $X_{feasible}$ auf $Y_{feasible}$ – nicht eindeutig entschieden werden kann, welche Werte die Komponenten des Target-Vektor enthalten sollen, wenn allein y_1^{target} durch den Anwender vorgegeben wird. Die Graphik zeigt für ein definiertes MOP die globale Pareto-Front PF_{true} und ihre durch MOMBES generierten Approximation PF^* mit den Elementen $\vec{y}^p \in PF^*$ mit $p = 1,2,..,7$. Man erkennt, daß der Target-Vektor $\vec{y}^{target}$ nicht durch eines der Elemente von PF^* repräsentiert wird, sondern „zwischen“ $\vec{y}^4$ und $\vec{y}^5$ lokalisiert ist. Die grüne Linie symbolisiert alle diejenigen gültigen Zielfunktionswertevektoren, deren Komponenten in der ersten Dimension mit dem Wert von y_1^{target} übereinstimmen. Ein Optimierungsverfahren, das als alleiniges Zielkriterium das Erreichen von y_1^{target} bei gleichzeitiger Minimierung von f_2 hätte, würde Gefahr laufen, den Punkt $\vec{y}^*$ und nicht $\vec{y}^{target}$ zu erreichen. Um das zu verhindern, müßte auch der Wert für y_2^{target} geeignet geschätzt werden und zwar derart, daß er sich in der Nähe der durch die $\vec{y}^p$ vorgegebenen PF^* befindet. Dies soll durch das Anwenden eines Modells (rote Linie) erreicht werden, das die funktionelle Abhängigkeit zwischen f_1 und f_2 modelliert, um bei Vorgabe von y_1^{target} durch Anwenden des Modells y_2^{target} zu liefern.

Der hier dargestellte Ansatz geht davon aus, daß die Verteilung der Elemente aus PF^* in geeigneten Modellen abgebildet und die Schätzung zur Lokalisierung des Target-Vektors durch diese Modelle unterstützt werden kann.

Dazu wird durch die Pareto-Modelle zunächst die Approximation PF^* der Pareto-Front ohne Berücksichtigung der sie bestimmenden nicht-dominierten Lösungen modelliert, d.h. nur der durch die Menge der Zielfunktionswertevektoren $y^p \in PF^*$ beschriebene Zusammenhang zwischen den Zielfunktionen ohne Berücksichtigung der entsprechenden Vektoren der

Entscheidungsvariablen $\vec{x}^{\,p} \in P^*$ abgebildet[75]. Dies erfolgt, indem bei k Zielfunktionen des MOP´s k Teilmodelle MOD_i erzeugt werden, die jeweils den Zusammenhang nur einer der k Zielfunktionen in Abhängigkeit der komplementären $(k-1)$ Zielfunktionen abbildet (siehe Abbildung 6.15 und Gleichung [53]).

$$\vec{y}_{Ni} = [y_1,..,y_{i-1},y_{i+1},..,y_k] \qquad [53]$$

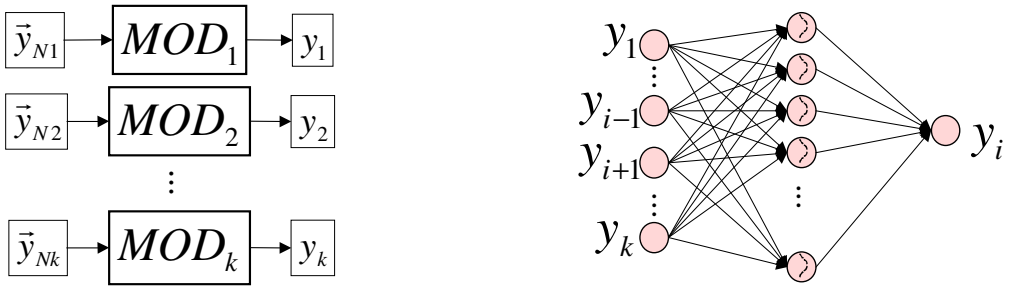


Abbildung 6.15: Modellierung der Zusammenhänge innerhalb der Pareto-Front durch Teilmodelle MOD_i der *i*-ten Komponente von $\vec{y}$

Als Modelltyp wird für jedes MOD_i ein dreilagiges vollvernetztes Feed-Forward-RBF-Netz mit exponentiellen radialsymmetrischen Basisfunktionen eingesetzt (siehe Abschnitt 3.2.2), das sich besonders gut zur Funktionsapproximation eignet. Das Netz verfügt jeweils über $(k-1)$ Eingabeneurone und ein Ausgabeneuron. Die Anzahl der Zwischenneurone wird der Anzahl der Trainingsdatensätze, d.h. der Größe von PF^* gleichgesetzt. Dies ist sinnvoll, da die Anzahl der Trainingsdatensätze relativ gering ist (i.d.R. ist $|PF^*| \leq 100$) und damit jedes Zwischenneuron auf einen Trainingsvektor positioniert werden kann. Das Modell wird mit den durch die Elemente von PF^* vorgegebenen Stützstellen im Zielraum belehrt, wobei das implementierte Lernverfahren lediglich die Kopplungsmatrix zwischen den Neuronen der Hidden- und der Ausgabeschicht modifiziert. Sowohl die Positionierung der Basisfunktionen, als auch ihre Größe werden durch die Verteilung der Elemente aus PF^* im Zielraum ermittelt: die Positionsvektoren der Zwischenneurone werden mit den Komponenten der Trainingsvektoren $\{y^p\}$ und die Größe der Basisfunktionen mit dem mittleren euklidischen Abstand zweier benachbarter Elemente aus PF^* initialisiert. Die in der Trainingsphase entstehenden Gleichungssysteme zur Minimierung des Fehlverhaltens der Modelle MOD_i, also jeweils der Minimierung der Abweichung zwischen

[75] Es wäre natürlich bereits hier möglich, wie in Abschnitt 6.5.2.3 dargestellt, die Abbildung der Vektoren aus P^* auf die Vektoren aus PF^* zu modellieren und das so entstandene finale Pareto-Modell für die Schätzung des vollständigen Target-Vektors zu nutzen. Da jedoch Simulationen (siehe Abschnitt 7.5) gezeigt haben, daß die Modellgüte des finalen Pareto-Modell geringer als die der Pareto-Modelle MOD_i ist, erscheint es an dieser Stelle sinnvoller, den i.d.R. hochdimensionalen Raum der Entscheidungsvariablen bei der Abbildung des Zusammenhanges der Zielfunktionen untereinander nicht zu berücksichtigen und statt dessen in Form der MOD_i kleine und schnell zu invertierende Teilmodelle zu erzeugen. Darüberhinaus ist der in der PF^* repräsentierte funktionelle Zusammenhang zwischen den Zielfunktionen i.d.R. weniger komplex als der zwischen den Entscheidungsvariablen und Zielfunktionen, was die zu erwartende Modellgüte erhöht.

der Aktivität des Ausgabeneurons $\hat{y}_i$ und $y_i^p \in PF^*$, wird mittels einer Singulär-Werte-Zerlegung (siehe [Numerical Recipies, 1994]) gelöst.

Durch die Pareto-Modelle werden die Zusammenhänge zwischen den Zielfunktionen innerhalb der durch MOMBES generierten Approximation PF^* der Pareto-Front modelliert. Daher ist es durch ihre Anwendung möglich, bei Vorgabe einer Komponente des Target-Vektors die jeweiligen komplementären Komponenten derart zu ermitteln, daß der resultierende Target-Vektor näherungsweise auf PF^* positioniert ist.

Für den Sonderfall eines MOP mit 2 Zielfunktionen kann dies erfolgen, indem bei Vorgabe von y_1^{target} der Wert der komplementäre Komponente y_2^{target} durch Propagierung von y_1^{target} durch das Modell MOD_2 berechnet wird: $y_2^{target} = MOD_2(y_1^{target})$. Analog kann y_1^{target} bei Vorgabe von y_2^{target} als Netzoutput des Modells MOD_1 bestimmt werden: $y_1^{target} = MOD_1(y_2^{target})$.

Für MOP mit $k > 2$ Zielfunktionen kann dieser Ansatz nicht mehr verwendet werden, da bei Vorgabe von y_i^{target} nicht gewährleistet ist, daß sich der durch die Teilnetze MOD_i berechnete vollständige Vektor $\vec{y}^{target}$ auf der interpolierten PF^* befindet. Daher werden bei $k > 2$ Zielfunktionen bei Vorgabe von einer Komponente y_i^{target} die Werte des komplementären Vektors $\vec{y}_{Ni}^{target}$ ermittelt, indem das entsprechende MOD_i invertiert wird. Dies geschieht mit Hilfe eines schnellen Gradientenabstiegsverfahren nach Levenberg/Marquardt, das ausgehend von einem Startpunkt $\vec{y}^{start} \in PF^*$ die Werte der Komponenten des Vektors $\vec{y}_{Ni}$ solange modifiziert, bis bei Propagierung von $\vec{y}_{Ni}$ durch MOD_i die absolute Abweichung zwischen Modelloutput und y_i^{target} minimal wird: $\vec{y}_{Ni}^{target} = INV(MOD_i, y_i^{target})$. Als Startpunkt der Invertierung wird derjenige Vektor $\vec{y}^{start} \in PF^*$ gewählt, für den $\left\| y_i^{start} - y_i^{target} \right\|$ minimal ist. Die Bestimmung von $\vec{y}^{target}$ auf Basis der Vorgabe nur einer Komponente y_i^{target} ist bei $k > 2$ nicht eindeutig. Das hier beschriebene Verfahren ermittelt denjenigen komplementären Vektor $\vec{y}_{Ni}^{target}$, der dem Startpunkt $\vec{y}^{start}$ der Invertierung am nächsten liegt.

Die oben beschriebene Anwendung der Modelle MOD_i zur Schätzung des Target-Vektors bilden neben der in Abschnitt 6.5.1 beschriebenen selbstorganisierenden Merkmalskarte, die mit den Elementen aus P^* und PF^* der belehrt wird, die Basis für die Visualisierung der durch MOMBES generierten Mengen P^* und PF^*.

6.5.2.2 Visualisierung der Approximation der Pareto-Front

Um die Auswahl des Target-Vektors zu unterstützen, sind in MOMBES verschiedene Module zur Visualisierung der Elemente aus PF^* integriert worden. Jede Darstellung zeigt synchron den aktuell durch den Anwender ausgewählten bzw. modifizierten Punkt auf PF^* an.

- *Darstellung als Tabelle:* Die Vektoren aus PF^* werden in einer Tabelle mit $\overline{N} = \left| PF^* \right|$ Zeilen und k Spalten angezeigt. Jedes Element der PF^* wird in einer Zeile und jede Komponente des Vektors aus PF^* in einer Spalte dargestellt. In einer zusätzlichen Zeile kann der Anwender – ausgehend von einem Element aus PF^* – den Target-Vektor komponentenweise editieren. Wird eine Komponente des Vektors modifiziert, schlägt das

System auf Basis der oben dargestellten Methode die komplementären Elemente des Target-Vektors vor.

- *Darstellung als Zeigerdiagramm:* Jeder Wert der Komponenten y_i^{target} des aktuellen Vektors $\vec{y}^{target}$ der Zielfunktionswerte wird in einem Zeigerdiagramm angezeigt. Wird durch Verschieben der Zeiger der Wert einer Komponente y_i^{target} modifiziert, werden entsprechend der obigen Darstellung die komplementären Werte aus $\vec{y}_{Ni}^{target}$ geschätzt und in den entsprechenden Zeigerdiagrammen angezeigt. Auf diese Weise ist es dem Anwender möglich, die Reaktion auf die Änderung des Wertes einer Zielfunktion auf die Werte der anderen Zielfunktionen im näherungsweise pareto-optimalen Bereich zu beobachten.
- *Darstellung als scatter-plot-matrix:* Für die Darstellung einer Pareto-Front wird i.d.R. die scatter-plot-matrix-Methode verwendet[76] [Deb, 2002], d.h. die 2-dimensionale Darstellung der Zielfunktionswertevektoren als Projektion jeweils zweier ihrer Komponenten in die Ebene. Für die vollständige Darstellung der Pareto-Front eines MOP´s mit k Zielfunktionen sind mit dieser Methode auch k Diagramme notwendig. Die Analyse einer mit dieser Methode dargestellten Pareto-Front ist für MOP´s mit $k < 4$ Zielfunktionen durchaus übersichtlich, für MOP´s mit mehr als 3 Zielfunktionen jedoch nicht mehr praktikabel. Daher wird diese Methode zur graphischen Darstellung der durch MOMBES generierten Approximation einer Pareto-Front hier ergänzend eingesetzt.
- *Darstellung auf der topologischen Karte:* Eine weitere Möglichkeit, Zusammenhänge zwischen den einzelnen Komponenten höherdimensionaler Vektoren darzustellen, bietet sich mit der Anwendung selbstorganisierender Merkmalskarten. Entsprechend der Darstellung aus Abschnitt 6.5.1 wird eine Pareto-SOM mit den Elementen aus PF^* belehrt. Mittels der ebenfalls in Abschnitt 6.5.1 beschriebenen Hyper-Cube-Visualisierungstechnik können qualitative Zusammenhänge in bis zu 11 Dimensionen gleichzeitig visualisiert werden. Durch Selektion eines Würfels auf der Karte erhält der Anwender die Möglichkeit, die Vektorkomponenten des durch den ausgewählten Würfel repräsentierten Clusterschwerpunktes in den Target-Vektor zu übernehmen und ihn gleichzeitig auf dem scatter-plot anzuzeigen. Dies gilt sowohl für Cluster- als auch für No-Cluster-Neurone. Dabei wird die Kopplungsmatrix der Neurone der Karte in den Raum der Trainingsvektoren zur Belegung der Vektorkomponenten verwendet. Andererseits wird jeder in der Tabelle, den Zeigerdiagrammen und dem scatter-plot ausgewählte Punkt auf PF^* ($\vec{y}^{target}$) auf der Pareto-SOM angezeigt, indem derjenige Würfel auf der Karte markiert wird, der der Schwerpunkt desjenigen Clusters ist, indem sich der aktuelle Target-Vektor befindet.

Jede Modifikation des Target-Vektors in der Tabelle, den Zeigerdiagrammen, seine Auswahl als selektierter Punkt auf dem scatter-plot und der Pareto-SOM werden in jedem der Visualisierungsmodule synchron angezeigt.

Abbildung 6.16 zeigt beispielhaft die für das MOP [6] (S. 7) durch MOMBES generierte Approximation PF^* der Pareto-Front PF_{true}, PF_{true} selbst sowie die durch die Anwendung des Pareto-Modells MOD_1 abgebildete Abhängigkeit zwischen f_1 und f_2 als scatter-plot.

[76] Weniger gebräuchliche Methoden zur graphischen Darstellung einer Pareto-Front sind die „Value Path"-[Geoffrion et al, 1972] und die „Star-Coordinate"-Methode [Manas, 1982], die jedoch bei der Darstellung von Pareto-Fronten mit vielen Elementen (größer 30) unübersichtlich werden und aus diesem Grunde in dieser Arbeit nicht verwendet wurden.

Neben dem Schätzen der Komponenten des Target-Vektors kann der Anwender weitere einzuhaltende Randbedingungen in Form von bestimmten Ober- und Untergrenzen des Wertebereichs jeder Entscheidungsvariable und jedes Zielfunktionswertes für die zu generierende Lösung angeben. Nach der interaktiven Definition des Target-Vektors und eventueller einzuhaltender Randbedingungen durch den Anwender erfolgt der eigentliche Zugriff auf PF^* im Bereich des Target-Vektors sowie die Ausgabe der Target-Lösung.

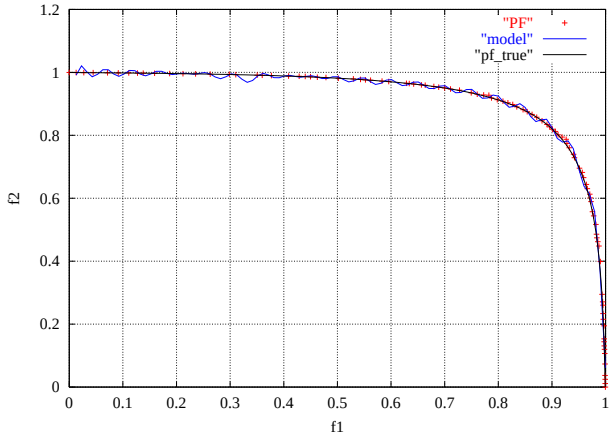


Abbildung 6.16: scatter-plot für die Pareto-Front für MOP [6]

6.5.2.3 Ermitteln der Target-Lösung durch Invertierung des finalen Pareto-Modells

Eine wichtige Eigenschaft von MOMBES ist die Fähigkeit des interaktiven und kontinuierlichen Zugriffs auf PF^* und der sie bestimmenden Lösungsmenge P^* unter Berücksichtigung von Zielvorgaben und ohne abermaliges Durchlaufen des Optimierungskerns. Dabei kann sich der durch den Anwender definierte Target-Vektor von den Elementen aus PF^* unterscheiden, muß sich jedoch innerhalb des durch sie begrenzten Raums befinden. Mit Hilfe von MOMBES ist es also möglich, eine Target-Lösung $\vec{x}^{target}$ zu generieren, die einen Zielfunktionswertevektor bestimmt, dessen euklidische Distanz zum durch den Anwender definierten Target-Vektor möglichst gering ist. Dabei befindet sich $\vec{y}^{target}$ auf der durch die MOD_i modellierten Approximation PF^* der Pareto-Front PF_{true}.

Zur Lösung dieses Problems ist eine Vorschrift zu finden, die die Zielfunktionswerte auf die Entscheidungsvariablen abbildet, um so durch Vorgabe eines Tupels der Zielfunktionswerte im Bereich von PF^* durch Anwenden einer solchen Abbildung die sie bestimmenden Vektoren der Entscheidungsvariablen zu erhalten. Eine solche Abbildung zu modellieren ist jedoch aus folgenden Gründen sehr schwierig: Zwischen den Zielfunktionswertevektoren aus PF^* und den einzelnen Komponenten der Vektoren der Entscheidungsvariablen aus P^* kann u.U. eine mehrdeutige Beziehung (siehe 3.2.3) bestehen. In diesem Fall existiert keine eindeutige Zuordnung zwischen den Vektoren $\vec{y}^{\,p} \in PF^*$ und einzelnen Komponenten x_i^p der Vektoren $\vec{x}^{\,p} \in P^*$ und damit keine eindeutige Abbildung von $\vec{y}^{\,p}$ auf x_i^p (siehe dazu auch Abschnitt 3.2.3). Damit ist eine Modellierung einzelner Entscheidungsvariablen aus P^* in Abhängigkeit der Zielfunktionswertevektoren aus PF^* nicht möglich.

Um die oben beschriebenen Schwierigkeiten zu umgehen, wurde in MOMBES folgende Lösungsstrategie zum Auffinden der Target-Lösung unter Vorgabe des Target-Vektors verwendet (siehe Abbildung 6.18). Zur Berechnung der Zielfunktionen wurde ein sogenanntes finales Pareto-Modell (FPM) erzeugt, das in Form eines Approximations-Netzwerkes (RBF) die Abbildung der Elemente aus P^* auf die Elemente von PF^* modelliert. Zum Auffinden der Target-Lösung ist dieses Modell zu invertieren, indem mittels einer Optimierungsroutine der Modelleingang solange modifiziert wird, bis die Abweichung (euklidische Distanz) zwischen Modellausgang und Target-Vektor minimal ist. Dies entspricht dem in Abschnitt 3.2.3 beschriebenen Verfahren der indirekten Modellinvertierung.

Ist der Target-Vektor nicht mit einem Element aus PF^* identisch, ist mit dem Auffinden der Target-Lösung wiederum ein Optimierungsproblem zu lösen, dessen Ziel jedoch nicht mehr in dem Auffinden der Pareto-Menge, sondern einer einzelnen Lösung besteht (siehe Abbildung 6.17). Die Lösung des MOP´s gestaltet sich in diesem Fall in drei Stufen:

- dem Generieren einer möglichst guten Approximation PF^* der Pareto-Front PF_{true},
- dem Schätzen des Target-Vektors und schließlich
- der Optimierung in Richtung des Target-Vektors auf PF^*.

Durch die Anwendung des finalen Pareto-Modells zur Berechnung der Zielfunktionen wird der durch die Optimierungsroutine zu durchsuchende gültige Zielraum $Y_{feasible}$ vor Beginn der Optimierung auf den Bereich im Zielraum eingeschränkt, der PF^* beschreibt. Da der zu durchsuchende Zielraum so bereits beschränkt und der Startpunkt der Optimierung durch ein Element aus P^* geeignet belegt werden kann, kann als Optimierungsroutine ein schnelles gradientenbasiertes Verfahren nach Levenberg/Marquardt (siehe Abschnitt 4.2.2) eingesetzt werden.

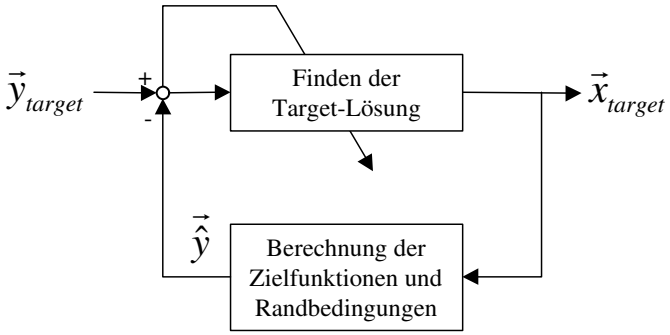


Abbildung 6.17: Generieren der Target-Lösung

Die bekannten Nachteile dieser Klasse von Verfahren (siehe Abschnitt 4.2.4) können durch das Einschränken des Zielraumes und die günstige Initialisierung ausgeglichen werden, so daß ein schnelles Verfahren zum Auffinden der Target-Lösung zur Verfügung steht. Der Startpunkt der Optimierung $\vec{x}^{start}$ wird mit dem Element aus P^* initialisiert, das denjenigen Zielfunktionswertevektor auf PF^* bestimmt, dessen euklidischer Abstand zum Target-Vektor minimal ist (Gleichung [54]).

$$\vec{x}^{\,start} = \vec{x}^{\,i} \;\; mit \;\; i = \arg\min_{p=1}^{|P^*|} \left\| \vec{f}(\vec{x}^{\,p}) - \vec{y}^{\,target} \right\| \qquad [54]$$

$$und: \quad \vec{x}^{\,p} \in P^* \quad sowie: \quad \vec{y}^{\,p} = \vec{f}(\vec{x}^{\,p}) \in PF^*$$

Das finale Pareto-Modell ist ebenso wie die Pareto-Modelle MOD_i als dreilagiges vollvernetztes Feed-Forward-RBF-Netz mit exponentiellen radialsymmetrischen Basisfunktionen strukturiert (siehe Abschnitt 3.2.2). Es besitzt n Eingangs und k Ausgangs- sowie $|PF^*|$ Hidden-Neurone.

Als Trainingsvektoren werden die Vektoren aus P^* (Modelleingang) und aus PF^* (Modellausgang) verwendet. Auch für dieses Modell paßt das implementierte Lernverfahren mittels Singulär-Werte-Zerlegung nur die Kopplungsmatrix zwischen den Neuronen der Hidden- und der Ausgabeschicht an und positioniert die Basisfunktionen auf die Trainingsvektoren.

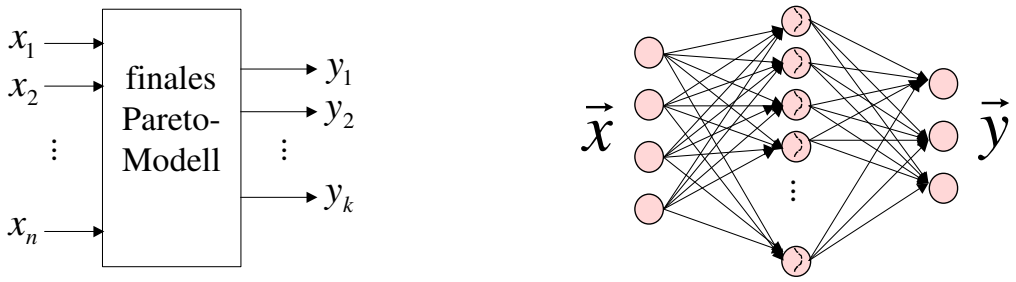


Abbildung 6.18: Finales Pareto-Modell zur Abbildung von P^* auf PF^*

Durch das Anwenden der Optimierungsroutine zum Auffinden der Target-Lösung wird eine Modellinvertierung des finalen Pareto-Modells realisiert. Diese Vorgehensweise des Auffindens der Target-Lösung birgt Vor- und Nachteile: Vorteilhaft ist, daß der Zielraum bereits vor der Optimierung auf eine „region of interest“ (PF^*) eingeschränkt wird, was die Suche beschleunigt und ein Einfangen in lokalen Minima verhindert – nachteilig hingegen ist, daß sich Fehler bei der Modellierung der Zusammenhänge zwischen den Elementen der Pareto-Menge und der Pareto-Front auch auf das Auffinden der Target-Lösung fortpflanzen. D.h. die Target-Lösung kann nur so gut sein, so gut auch das finale Pareto-Modell in der Lage ist, die Vektoren aus P^* auf die Vektoren aus PF^* abzubilden.

Um die Gefahr unsicherer Modellvorhersagen zu verringern, kann der Anwender ein Vertrauensmaß (Konfidenz) angeben, dessen Unterschreiten als zusätzliche Randbedingung bei der Optimierungsroutine zum Auffinden der Target-Lösung berücksichtigt wird. Diese Konfidenz wird gewonnen, indem ein Maximum-Operator auf die Aktivität der Zwischenneurone angewandt wird. Die Konfidenz repräsentiert die Aktivität des maximal erregten Zwischenneurons bei Verarbeiten eines Eingangsvektors durch das RBF-Netz und kann als Maß dafür interpretiert werden, welche minimale euklidische Distanz dieser Eingangsvektor zu den Trainingsvektoren aufweist. Sie gibt also Aufschluß darüber, wie „bekannt“ ein Datensatz für das Modell d.h. wie gut er durch Trainingsvektoren abgesichert ist bzw. kann als Ähnlichkeitsmaß des aktuellen Arbeitspunktes zu den trainierten Stützstellen des Modells interpretiert werden.

Liegt eine analytische Beziehung zwischen den Entscheidungsvariablen und Zielfunktionen vor, kann auf die Anwendung des finalen Pareto-Modells verzichtet werden und die Berechnung des Zielfunktionswertevektors direkt unter Verwendung dieser Beziehung erfolgen.

Mit Hilfe des finalen Modells und seiner Invertierung kann der Anwender nun nicht nur auf einzelne Lösungspunkte der P^* zugreifen, sondern für beliebige Kombination der einzelnen Zielfunktionswerte innerhalb der PF^* generieren. Dabei können Einschränkungen des Wertebereichs für jedes x_i aus $\vec{x}$, jedes y_i aus $\vec{y}$ sowie der Sicherheit der Approximation berücksichtigt werden. Im Ergebnis des Decision-Makings erhält der Anwender eine Kompromißlösung, die seinen speziellen Anforderungen entspricht (siehe z.B. Abschnitt 8.1.3).

6.6 Abgrenzung von MOMBES zu anderen MOEA

Mit MOMBES ist ein zweistufiges System zur Mehrzieloptimierung entwickelt worden, das aus einem multi-hybriden MOEA als Optimierungskern und einem modellgestützten Decision-Making-Modul besteht. In beiden Stufen wurden Neuronale Netze verwendet, um die Zusammenhänge zwischen den Entscheidungsvariablen und Zielfunktionswerten für die Bereiche zu finden, die durch pareto-optimale Lösungen bestimmt werden. Im folgenden sollen die Gemeinsamkeiten und Unterschiede zwischen dem Optimierungskern von MOMBES und bekannten Vertretern moderner MOEA dargestellt werden.

Ähnlichkeiten von MOMBES mit bekannten MOEA[77]

- MOMBES verwendet ähnlich wie die MOEA: SPEA2, NSGA2, MOGA, PESA und NPGA das Prinzip der Pareto-Dominanz als Basis der Fitnessberechnung der Individuen.
- Wie bei den MOEA: SPEA2, NSGA2 und PESA wird bei MOMBES das Eliteprinzip in Form einer externen Population für nicht-dominierte Individuen angewandt.
- Die externe Elitepopulation enthält bei MOMBES wie auch bei den MOEA: SPEA2, NSGA2 und PESA nur eine Auswahl von nicht-dominierten Individuen. Als Auswahlkriterium wird bei MOMBES die Nachbarschaft zweier Individuen in Form der euklidischen Distanz im Zielraum benutzt. Dieser Ansatz ist der bei NSGA2 verwendeten Methode ähnlich, wogegen bei SPEA2 die Entscheidung, ob ein Individuum aus der Elitepopulation entfernt wird, aufgrund der Anzahl der von ihm dominierten Individuen und bei PESA auf Basis der Anzahl der sich im selben Hypergittersegment befindlichen Individuen getroffen wird.
- Im System MOMBES wird ein Evolutionärer Algorithmus mit einem gradientenbasierten Optimierungsverfahren kombiniert. Im Unterschied zu anderen Kombinationen von MOEA mit gradientenbasierten Optimierungsverfahren wie z.B. dem hybriden MOEA von Quagliarella und Vicini (siehe Abschnitt 5.4.3.1), der von einem Element der durch den MOEA generierten Approximation der Pareto-Front ausgeht und dieses in Richtung globaler Pareto-Front optimiert, wird bei MOMBES zunächst ein Bereich geringster Dichte oder Ecke auf der aktuellen Approximation von PF_{true} bestimmt und von dort aus in Richtung globaler Pareto-Front optimiert (siehe Abschnitt 6.4.7).
- MOMBES verwendet ebenso wie der SOM-MOEA von Büche et al (siehe Abschnitt 5.4.3.2) eine selbstorganisierende Merkmalskarte innerhalb des MOEA. Während jedoch Büche et al die SOM als Rekombinationsoperator nutzen, indem benachbarte Individuen auf der SOM für die Rekombination ausgewählt werden, verwendet MOMBES die SOM zum Auffinden von Regionen geringer Dichte auf der aktuell generierten Approximation der Pareto-Front (siehe Abschnitt 6.4.7).

[77] Die im folgenden zitierten MOEA sind in Abschnitt 5.4 ausführlich dargestellt worden.

Unterschiede von MOMBES gegenüber bekannten MOEA:

- MOMBES ist als MOEA in Form einer Evolutionsstrategie mit Selbstanpassung der Mutationsschrittweiten und nicht wie die MOEA: SPEA2, NSGA2, MOGA, und NPGA als Genetischer Algorithmus umgesetzt. Damit werden die in Abschnitt 4.1.4 dargestellten Vorteile von ES gegenüber GA genutzt.
- MOMBES modelliert während der Optimierung die aktuelle Approximation $PF^*(t)$ der Pareto-Front durch ein selbstorganisierendes Neuronales Netz und vermag durch Interpretation dieses Modells die Evolution in Richtung homogener und breit verteilter Pareto-Front zu beeinflussen, indem gezielt sogenannte approximierte Individuen in Regionen „geringster Dichte“ und den Ecken von $PF^*(t)$ eingebracht werden. Dadurch kann eine Verbesserung sowohl der Fortschrittsgeschwindigkeit des MOEA als auch der Qualität von $PF^*(t)$ erreicht werden.
- MOMBES beschränkt sich in der Präsentation der Ergebnisse für den Anwender nicht auf die Elemente der P^* und der PF^* sondern verfügt darüber hinaus über ein interaktives Decision-Making-Modul, das nach Abschluß der Optimierung einen kontinuierlichen Zugriff auf PF^* und der P^* ohne abermaliges Durchlaufen des Optimierungskerns gestattet. Dazu werden Modelle erzeugt, die die Abhängigkeiten zwischen den konkurrierenden Zielfunktionen und die Zuordnung zwischen den Elementen aus P^* und den Elementen der aus PF^* abbilden. Durch Invertierung dieser Modelle ist es möglich, für einen durch den Anwender definierten Zielfunktionswertevektor innerhalb von PF^* die passende Lösung zu generieren (siehe Abschnitt 6.5.2).
- MOMBES gestattet innerhalb des Decision-Making-Moduls eine interaktive, qualitative Analyse der Zusammenhänge zwischen den Komponenten der Approximation der Pareto-Front und der sie bestimmenden Approximation der Pareto-Menge. Dazu wird eine Clusteranalyse mittels einer selbstorganisierenden Merkmalskarte (SOM) durchgeführt und dem Anwender mit der Hyper-Cube-Visualisierungstechnik die Möglichkeit gegeben, Zusammenhänge zwischen den Entscheidungsvariablen und Zielfunktionswerten in mehren Dimensionen simultan zu visualisieren. Der Anwender kann so gezielt Wissen über den Bereich des zu optimierenden Prozesses extrahieren, der die pareto-optimalen Lösungen bestimmt (siehe Abschnitt 6.5.1).

Die beschriebenen Unterschiede – insbesondere die Mechanismen der Selbstanpassung – machen MOMBES zu einem leistungsfähigen MOEA, der sein Potential bei den in Kapitel 7 beschriebenen Benchmarks und den in Kapitel 8 beschriebenen technischen Anwendungen demonstriert.

7 Simulationen

Das beschriebene Verfahren MOMBES wird in diesem Kapitel auf beschränkte kontinuierliche Optimierungsprobleme mit $k \geq 2$ Zielfunktionen angewandt und in seiner Leistungsfähigkeit mit den bekannten und in Abschnitt 5.4 beschriebenen Ansätzen verglichen. Dabei werden die in Abschnitt 7.1 beschriebenen Gütekriterien zur Charakterisierung von Approximationen einer Pareto-Front verwendet. Der Vergleich erfolgt dabei bezüglich einer Anzahl von Berechnungen des Zielfunktionswertevektors (function call).

Neben der Charakterisierung der durch den Algorithmus generierten Ergebnisse (hier die Approximation der Pareto-Menge und der Pareto-Front) spielen jedoch zur vollständigen Beurteilung der Güte eines Optimierungsverfahrens Kriterien wie Allgemeinheit und Leistungsfähigkeit des Verfahrens, Effizienz bezüglich des Rechen- und Speicheraufwandes, Einfachheit der Implementierung sowie Stabilität des Algorithmus´ eine wichtige Rolle. Nur Verfahren, die auch hinsichtlich dieser Kriterien als gut bewertet werden können, werden in der ingenieurtechnischen Praxis eine breite Anwendung finden. An dieser Stelle sollen jedoch diese Kriterien nicht diskutiert, sondern auf die Literatur [Battiti und Tecchilio, 1995] verwiesen werden.

7.1 Darstellung und Diskussion bekannter Leistungskriterien für MOEA

In diesem Abschnitt werden Leistungskriterien beschrieben, die für ein definiertes MOP die Güte der Approximation(en) PF^* seiner Pareto-Front PF_{true} hinsichtlich:

1.) ihres Abstandes zur (theoretisch möglichen) Pareto-Front PF_{true},

2.) ihrer Ausbreitung und Homogenität im Zielraum und

3.) der Dominanz ihrer Elemente bezüglich einer weiteren Approximation der Pareto-Front

charakterisieren. Während sich Kriterien der Gruppen 1 und 2 auf die Charakterisierung einer einzelnen Approximation einer Pareto-Front beziehen, beschreiben Kriterien der Gruppe 3 das Verhältnis der Leistungsfähigkeit bezüglich der Dominanz der Elemente von zwei Approximationen einer Pareto-Front. Nachfolgend werden einige der gebräuchlichsten Vertreter dieser 3 Gruppen dargestellt und ihre Stärken und Schwächen beschrieben.

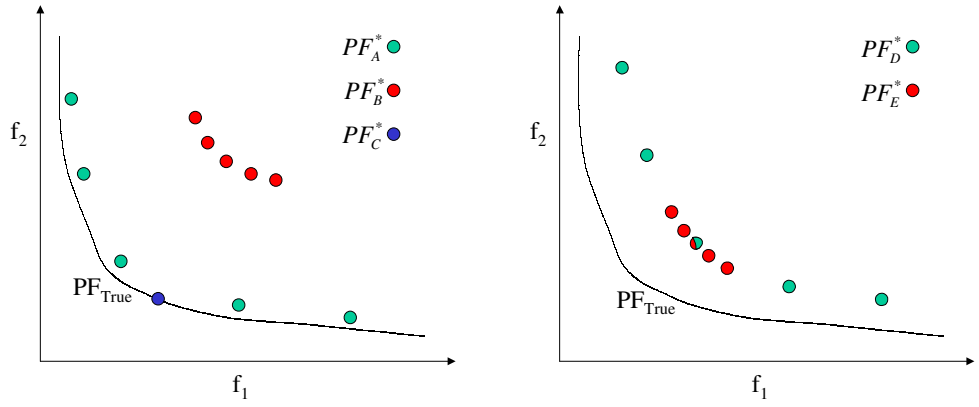


Abbildung 7.1: verschiedene Approximationen der Pareto-Front eines MOP´s

Dabei dienen die in Abbildung 7.1 dargestellten Approximationen PF_A^*, PF_B^*, PF_C^*, PF_D^* und PF_E^* der Pareto-Front PF_{true} der Illustration der Aussagekraft der einzelnen Leistungskriterien.

7.1.1 Charakterisierung des Abstandes zur theoretisch möglichen Pareto-Front

Häufig wird zur Evaluierung der Güte der Approximation PF^* einer Pareto-Front PF_{true} eine Maßzahl verwendet, die ihre Nähe zur (theoretisch möglichen) Pareto-Front PF_{true} oder den kleinsten euklidischen Abstand ihrer Elemente zum Idealvektor $\vec{y}_{ideal}$ beschreibt. Um diese Maßzahlen zu verwenden, muß jedoch PF_{true} oder $\vec{y}_{ideal}$ bekannt[78] sein, was i.d.R. bei Optimierungsproblemen aus der Praxis nicht der Fall ist. Trotzdem haben Gütemaße dieser Kategorie gerade für vergleichende Tests auf Basis künstlicher MOP´s ihre Bedeutung, da hier i.d.R. das Optimierungsziel in Form von PF_{true} analytisch berechenbar vorliegt. Die in diesem Abschnitt beschriebenen Gütemaße sind Referenzmetriken (d.h. Maßzahlen, die eine Vergleichsmetrik erfordern) bezüglich PF_{true} oder $\vec{y}_{ideal}$.

Error Ratio: Der Parameter $ER(PF^*)$ gibt das prozentuale Verhältnis zwischen der Anzahl der Elemente aus PF^*, die nicht in PF_{true} enthalten sind und der Größe von PF^* an (siehe Gleichung [55] nach [Veldhuizen und Lamont, 1999]).

$$ER(PF^*) = \frac{1}{\left|PF^*\right|} \cdot \sum_{p=1}^{\left|PF^*\right|} e_p \quad \text{mit} \quad e_p = \begin{cases} 0 \;\; if \;\; \vec{y}^{\,p} \in PF_{true} \\ 1 \;\; if \;\; \vec{y}^{\,p} \notin PF_{true} \end{cases} \quad \text{für} \;\; \forall \;\; \vec{y}^{\,p} \in PF^* \qquad [55]$$

Im Sinne der Error Ratio ist PF^* dann besonders leistungsfähig, wenn $ER(PF^*)$ möglichst gering ist. Der Einsatz dieser Maßzahl als Vergleichskriterium ist jedoch nur dann sinnvoll, wenn die theoretisch erreichbare Pareto-Front PF_{true} bekannt ist. Es können keine Aussagen über den Grad der Entfernung von PF^* zu PF_{true}, die Dominanz ihrer Elemente gegenüber den Elementen einer Vergleichsfront, ihre Ausbreitung im Zielraum sowie ihre Homogenität gemacht werden. Wird $ER(PF^*)$ beispielsweise zum Vergleich von PF_A^*, PF_B^* und PF_C^* in Abbildung 7.1 herangezogen, werden PF_A^* und PF_B^* gleich ($ER(PF_A^*) = ER(PF_B^*) = 100\%$), PF_C^* jedoch als leistungsstärkste Approximation der Pareto-Front mit $ER(PF_C^*) = 0\%$ bewertet. Eine sinnvolle Erweiterung für $ER(PF^*)$ besteht darin, für jedes $\vec{y}^{\,p} \in PF^*$ die euklidische Distanz d_p zum nächsten Punkt auf PF_{true} zu berechnen und in Gleichung [55] dann $e_p = 1$ zu setzen, wenn d_p einen Schwellenwert δ überschreitet. Auf diese Weise werden alle $\vec{y}^{\,p}$ der PF_{true} zugehörig gewertet, die sich innerhalb eines durch δ bestimmten Bandes um PF_{true} befinden [Deb, 2002].

Generational Distance: Der Parameter $GD(PF^*)$ [Veldhuizen und Lamont, 1999] entspricht dem Mittelwert der jeweils kleinsten euklidischen Abstände der Elemente von PF^* zu PF_{true} (Gleichung [56]).

---

[78] PF_{true} sollte hier für das entsprechende MOP entweder als endliche Menge von pareto-optimalen Zielfunktionswertevektoren oder als Berechnungsvorschrift vorliegen.

$$GD(PF^*) = \frac{1}{\left|PF^*\right|} \sum_{p=1}^{\left|PF^*\right|} \min_{i=1}^{\left|PF_{True}\right|} \left\| \vec{y}^{\,p} - \vec{y}^{\,i}_{true} \right\| \quad mit \quad \vec{y}^{\,p} \in PF^* \quad und \quad \vec{y}^{\,i}_{true} \in PF_{True} \qquad [56]$$

Eine Approximation PF^* einer Pareto-Front PF_{true} wird bezüglich der Generational Distance dann als besonders leistungsfähig eingeschätzt, wenn $GD(PF^*)$ sich dem Wert 0 nähert. Da der Abstand der einzelnen Elemente von PF^* zum jeweils nächsten Element von PF_{true} stark variieren kann, ist es notwendig, bei der Verwendung des Parameters $GD(PF^*)$ auch die Standardabweichung der minimalen Abstände anzugeben. Ist diese Standardabweichung gering, kann $GD(PF^*)$ zur Charakterisierung der Nähe der einzelnen Elemente von PF^* zu PF_{true} akzeptiert werden. Mit Hilfe von $GD(PF^*)$ ist jedoch nur der mittlere Abstand der einzelnen Elemente von PF^* zu PF_{true} meßbar, nicht jedoch die Homogenität von PF^* sowie die Variabilität oder die Dominanz ihrer Elemente gegenüber den Elementen einer Vergleichsfront. Als alleiniges Maß zur Charakterisierung einer Approximation PF^* einer Pareto-Front ist $GD(PF^*)$ daher ebenfalls ungeeignet. Für das in Abbildung 7.1 dargestellte Beispiel gilt: $GD(PF_C^*) < GD(PF_A^*) < GD(PF_B^*)$ d.h. PF_C^* wird bezüglich des Parameters $GD(PF^*)$ als leistungsfähiger als PF_A^* und PF_A^* wiederum bezüglich $GD(PF^*)$ als leistungsfähiger als PF_B^* charakterisiert.

Distance to Ideal Vector: Coello verwendet in [Coello, 1996] zur Charakterisierung der Güte eines MOEA u.a. den geringsten euklidischen Abstand $L_p(PF^*)$ der Elemente $\vec{y}^{\,p}$ aus PF^* zum Idealvektor $\vec{y}^{\,ideal}$ (siehe Definition 4, S.5):

$$L_p(PF^*) = \min_{p=1}^{\left|PF^*\right|} \left\| \vec{y}^{\,p} - \vec{y}^{\,ideal} \right\| \quad mit: \quad \vec{y}^{\,p} \in PF^* \qquad [57]$$

$L_p(PF^*)$ läßt keinerlei Aussagen über die Ausbreitung, die Dominanz, den Abstand der Elemente aus PF^* zu PF_{true} sowie die Homogenität ihrer Verteilung im Zielraum zu und ist daher lediglich für den Vergleich der Qualität von zwei einzelnen Lösungen, nicht jedoch für den Vergleich der Güte zweier Approximationen einer Pareto-Front geeignet. Für das in Abbildung 7.1 dargestellte Beispiel mit $\vec{y}_{ideal} = [0,0]$ gilt: $L_p(PF_C^*) = L_p(PF_A^*) < L_p(PF_B^*)$ d.h. PF_C^* und PF_A^* werden als gleichwertig und leistungsfähiger als PF_B^* bezüglich ihrer Werte für $L_p(PF^*)$ charakterisiert.

7.1.2 Beschreibung der Ausbreitung und Homogenität von Approximationen einer Pareto-Front

Die zweite Gruppe von Bewertungskriterien für die Approximation PF^* einer Pareto-Front PF_{true} beinhaltet diejenigen Gütemaße, die die Ausbreitung der Elemente aus PF^* im Zielraum und ihre Homogenität beschreiben. Die Gütemaße dieser Gruppe sind Kardinalmaße und unabhängige Metriken, da sie keine Referenzmengen benötigen.

Deviation Measure: Deb charakterisiert in [Deb, 2000][79] die Homogenität der Verteilung der Elemente einer Approximation PF^* einer Pareto-Front mit dem Parameter $\Delta(PF^*)$ (Gleichung [58]) mit d_i als euklidischer Distanz zweier benachbarter Elemente von PF^* und $\bar{d}$ als Mittelwert aller d_i.

$$\Delta(PF^*) = \sqrt{\frac{1}{\left|PF^*\right|}\sum_{i=1}^{\left|PF^*\right|}\left(d_i - \bar{d}\right)^2} \qquad [58]$$

Benachbart sind diejenigen beiden Elemente aus PF^*, deren euklidische Distanz im Zielraum minimal ist. Eine Approximation PF^* einer Pareto-Front PF_{true} ist bezüglich dieses Kriteriums dann besonders leistungsfähig, wenn sich $\Delta(PF^*)$ dem Wert 0 nähert. Als alleiniges Maß zur Charakterisierung der Güte von PF^* ist dieser Parameter ebenfalls ungeeignet, da er weder die Ausbreitung der Elemente von PF^* im Zielraum, noch ihren Abstand zu PF_{true} oder ihre Dominanz gegenüber den Elementen einer Vergleichsfront berücksichtigt. Dieses Leistungskriterium besitzt nur dann Aussagekraft, wenn die PF_{true} des betrachteten MOP´s zusammenhängend ist (ein Gegenbeispiel ist in Abschnitt 7.4.1.3 angegeben). Die in Abbildung 7.1 dargestellten Approximationen einer Pareto-Front werden bezüglich $\Delta(PF^*)$ als gleichwertig eingeschätzt: $\Delta(PF_A^*) = \Delta(PF_B^*) = \Delta(PF_C^*) = 0$.

Maximum Spread: Zitzler definiert in [Zitzler, 1999] den Parameter $D(PF^*)$ als Länge der Diagonalen desjenigen Hyperquaders, der durch die maximale Ausdehnung aller Vektoren $\vec{y}^p$ aus PF^* in jeder der k Dimensionen von $\vec{y}^p$ aufgespannt wird (Gleichung [59]). Eine Approximation PF^* einer Pareto-Front PF_{true} wird dann als besonders leistungsfähig im Sinne der Maximum Spread eingeschätzt, wenn $D(PF^*)$ möglichst groß ist.

$$D(PF^*) = \sqrt{\sum_{i=1}^{k}\left(\max_{p=1}^{\left|PF^*\right|} y_i^p - \min_{p=1}^{\left|PF^*\right|} y_i^p\right)^2} \qquad [59]$$

Mit diesem Parameter sind keine Aussagen über die tatsächliche Verteilung der Elemente von PF^*, ihren Abstand zu PF_{true} oder die Dominanz einzelner Elemente gegenüber den Elementen einer Vergleichsfront möglich. Einen Algoritmus zur Lösung eines MOP´s mit guter Performance zeichnet aus, daß die einzelnen k Zielfunktionen $f_i(\vec{x})$ nicht separat, sondern simultan minimiert (bzw. maximiert) werden. Für den Praktiker sind i.d.R. Lösungen interessant, die sich in der Nähe der best-over-all-Lösung (siehe Definition 6, S.6) und nicht an den Rändern[80] der Pareto-Front befinden. Mit dem Leistungskriterium $D(PF^*)$ werden nun jedoch genau diejenigen Approximationen einer Pareto-Front als hochwertig eingeschätzt, deren Ränder möglichst weit in Richtung Minimierung der einzelnen $f_i(\vec{x})$ verschoben sind,

[79] Ähnliche, auf der Streuung der Elemente innerhalb einer Pareto-Front basierende Maßzahlen finden sich in [Horn und Nafpliotis, 1993], [Srinivas und Deb, 1994] und [Fonseca und Fleming, 1998].

[80] Die Begriffe „Ecke“ und „Rand“ einer Pareto-Front werden in Abschnitt 6.4.7 geklärt.

unabhängig davon, wie ihre Elemente im Zielraum verteilt oder wie nahe sie PF_{true} sind. Als alleiniges Maß zur Bewertung der Güte einer Approximation einer Pareto-Front ist $D(PF^*)$ daher nicht geeignet, kann jedoch in Verbindung mit anderen Maßzahlen einen Hinweis über die Ausbreitung der Elemente von PF^* im Zielraum geben. Für das in Abbildung 7.1 dargestellte Beispiel gilt: $D(PF_A^*) > D(PF_B^*) > D(PF_C^*)$, d.h. bezüglich $D(PF^*)$ wird PF_A^* mit größter und PF_C^* mit geringster Güte eingeschätzt.

Size of the Dominated Space: Ebenfalls von Zitzler wird in [Zitzler und Thiele, 1998] der Parameter S als absoluter, in beschränktem Maße die Ausdehnung einer Approximation PF^* einer Pareto-Front PF_{true} beschreibender Parameter eingeführt. Dabei berechnet sich S aus dem Volumen der Vereinigungsmenge der Hyperquader, die von den Elementen aus PF^* zu einem Referenzpunkt im Zielraum aufgespannt werden. Für Maximierungsprobleme ist dieser Referenzpunkt i.d.R. der Koordinatenursprung im Zielraum. Für Minimierungsprobleme dagegen werden die Koordinaten des Referenzpunktes i.d.R. so gewählt, daß ihr Wert deutlich über dem Maximalwert liegt, der in der entsprechenden Dimension des Zielfunktionswertevektors durch die Elemente aus PF^* definiert wird. Die Idee, die dieser Maßzahl zu Grunde liegt, besteht darin, denjenigen Raum zu bemessen, der durch die Elemente der betrachteten Menge nicht-dominierter Zielfunktionswertevektoren bezüglich des Referenzpunktes dominiert wird.

Eine Approximation PF^* einer Pareto-Front PF_{true} wird im Sinne des Hypervolumens S dann als besonders leistungsfähig eingeschätzt, wenn S einen möglichst großen Wert annimmt. Der Vergleich der Werte für S zweier Approximationen PF_1^* und PF_2^* einer Pareto-Front läßt keine direkten Schlüsse auf die Dominanz der einzelnen Elemente aus PF_1^* und PF_2^* zueinander zu. Auch läßt sich weder der Abstand von PF^* zu PF_{true} noch die Homogenität der Verteilung seiner Elemente mit S beschreiben. Besonders ungünstig ist es, daß die Größe von S von der Wahl des Referenzpunktes abhängt. Gerade für Minimierungsprobleme läßt sich zeigen, daß zwei Approximationen PF_A^* und PF_B^* einer Pareto-Front existieren können, die bei zwei verschiedenen Referenzpunkten I und II folgende Eigenschaft besitzen: $S(PF_A^*)^{\mathrm{I}} < S(PF_B^*)^{\mathrm{I}}$ und $S(PF_A^*)^{\mathrm{II}} > S(PF_B^*)^{\mathrm{II}}$ (siehe S.76 aus [Knowles, 2002]).

7.1.3 Vergleich der Dominanz der Elemente zweier Approximationen einer Pareto-Front

Coverage: Wiederum Zitzler führt in [Zitzler und Thiele, 1998] einen Parameter $C(PF_A^*, PF_B^*)$ ein, der das Verhältnis der Dominanz der Elemente zweier Mengen PF_A^* und PF_B^* nicht-dominierter Vektoren charakterisiert: $C(PF_A^*, PF_B^*)$ bestimmt den prozentualen Anteil der Elemente von PF_B^*, die durch mindestens ein Element aus PF_A^* schwach dominiert werden (Gleichung [60]).

$$C(PF_A^*, PF_B^*) := \frac{\left| \left\{ b \in PF_B^* \mid \exists\, a \in PF_A^* \; a \preceq b \right\} \right|}{\left| PF_B^* \right|} \qquad [60]$$

Je größer der Wert für $C(PF_A^*, PF_B^*)$ ist, desto leistungsfähiger wird PF_A^* bezüglich der Coverage gegenüber PF_B^* eingeschätzt. Ein Wert von $C(PF_A^*, PF_B^*) = 0\%$ bedeutet, daß kein Element aus PF_B^* von Elementen aus PF_A^*, $C(PF_A^*, PF_B^*) = 100\%$ hingegen, daß jedes Element aus PF_B^* von mindestens einem Element aus PF_A^* schwach dominiert wird. Unter Verwendung von $C(PF_A^*, PF_B^*)$ ist es auch ohne die Kenntnis von PF_{true} möglich, das Verhältnis der Güte zweier Approximationen einer Pareto-Front bezüglich der Pareto-Dominanz ihrer Elemente zu bewerten. Keine Aussagen können jedoch hinsichtlich ihrer geometrischen Ausdehnung im Raum der Zielfunktionen und die Homogenität der Verteilung ihrer Elemente gemacht werden. So gilt für das rechts in Abbildung 7.1 dargestellte Beispiel bei dem alle Elemente von PF_D^* und PF_E^* den gleichen Pareto-Rang besitzen: $C(PF_D^*, PF_E^*) = C(PF_E^*, PF_D^*) = 0\%$, d.h. beide Approximationen einer Pareto-Front werden als gleichwertig im Sinne der Coverage eingeschätzt.

Zusammenfassend wird festgestellt, daß für den Vergleich zweier Approximationen einer Pareto-Front keines der oben aufgelisteten Kriterien in der Lage ist, gleichzeitig die Dominanz der Elemente und ihre geometrische Ausdehnung und Homogenität im Zielraum zufriedenstellend zu beschreiben. Aus diesem Grunde soll in Abschnitt 7.2 ein neues Leistungskriterium KDA als **K**ombiniertes-**D**ominanz-**A**usdehnungskriterium eingeführt werden, das dieser Anforderung genügt.

7.2 Einführung des Kombinierten-Dominanz-Ausdehnungskriteriums

Für ein MOP nach Definition 1 seien zwei Approximationen PF_A^* und PF_B^* der Pareto-Front PF_{true} als endliche Mengen nicht-dominierter, zulässiger Zielfunktionswertevektoren gegeben. Es ist eine Maßzahl zu ermitteln, die sowohl die Eigenschaft der Dominanz der einzelnen Vektoren aus PF_A^* hinsichtlich der Vektoren aus PF_B^* und umgekehrt, als auch ihre geometrische Anordnung widerspiegelt.

Um diese Maßzahl zu berechnen, wird zunächst die Menge PF_A^* auf die Menge PF_A^{*D} reduziert, indem nur diejenigen Vektoren aus PF_A^* in PF_A^{*D} übernommen werden, für die es keinen Vektor aus PF_B^* gibt, der sie streng dominiert (Gleichung [61]). Analog setzt sich die Menge PF_B^{*D} aus allen denjenigen Vektoren aus PF_B^* zusammen, die von keinem Vektor aus PF_A^* stark dominiert werden (Gleichung [62]).

$$PF_A^{*D} = \left\{ \vec{a} \in PF_A^* \middle| \neg\exists\ \vec{b} \in PF_B^* : \vec{b} \prec \vec{a} \right\} \quad [61]$$

$$PF_B^{*D} = \left\{ \vec{b} \in PF_B^* \middle| \neg\exists\ \vec{a} \in PF_A^* : \vec{a} \prec \vec{b} \right\} \quad [62]$$

Anschließend werden für jede Dimension $i = 1,..,k$ die minimalen Werte ug_i und maximalen Werte og_i der Komponenten der Vektoren aus $PF_A^{*D} \cup PF_B^{*D}$ ermittelt:

$$\begin{gathered} ug_i = \min\left(\min\left(a_i^D\right), \min\left(b_i^D\right)\right) \\ og_i = \max\left(\max\left(a_i^D\right), \max\left(b_i^D\right)\right) \\ \text{für: } i = 1,..,k \text{ und } \vec{a}^D = \left[a_1^D,..,a_k^D\right] \in PF_A^{*D} \text{ und } \vec{b}^D = \left[b_1^D,..,b_k^D\right] \in PF_B^{*D} \end{gathered} \quad [63]$$

Es wird nun in jeder Dimension $i = 1,..,k$ der Bereich $[ug_i, og_i]$ in n^{grid} gleich große Abschnitte unterteilt. Dabei ist $n^{grid} = \max(|PF_A^*|, |PF_B^*|)$. Auf diese Weise entsteht ein k-dimensionales Hypergitter, dessen Gittersegmente in der Dimension i die Länge l_i^{grid} besitzen (Gleichung [64]).

$$l_i^{grid} = \frac{og_i - ug_i}{n^{grid}} \qquad [64]$$

Weiter sei $count_A$ die Anzahl der Gittersegmente, die mindestens ein Element aus PF_A^{*D} enthalten. Analog ist $count_B$ die Anzahl derjenigen Gittersegmente, die mindestens ein Element aus PF_B^{*D} enthalten. Das Verhältnis dieser beiden Maßzahlen $count_A$ und $count_B$ definiert das Kombinierte-Dominanz-Ausdehnungskriterium, das mit $KDA(PF_A^*, PF_B^*)$ bezeichnet wird. Ist $count_B = 0$, wird $KDA(PF_A^*, PF_B^*) = |PF_A^{*D}|$ gesetzt. Es gilt also:

$$KDA(PF_A^*, PF_B^*) = \begin{cases} \frac{count_A}{count_B} & \text{für} \quad count_B > 0 \\ |PF_A^{*D}| & \text{für} \quad count_B = 0 \end{cases} \qquad [65]$$

Für $count_A > 0$ und $count_B > 0$ gilt: $KDA(PF_A^*, PF_B^*) = 1/KDA(PF_B^*, PF_A^*)$.

Die Approximation PF_A^* einer Pareto-Front wird gegenüber einer Approximation PF_B^* dieser Pareto-Front als leistungsfähiger bezüglich des KDA eingeschätzt, wenn $KDA(PF_A^*, PF_B^*) > 1$ ist. Dann besitzt PF_A^* eine größere Anzahl nicht-dominierter Vektoren bezüglich $PF_A^* \cup PF_B^*$, die breit und homogen im Zielraum verteilt sind, als PF_B^*. Je größer der Wert für $KDA(PF_A^*, PF_B^*)$ ist, desto besser ist das Dominanzverhalten der Elemente aus PF_A^* und breiter und homogener ihre geometrische Verteilung gegenüber den Elementen aus PF_B^*.

Da bei der Berechnung des *KDA* zunächst die Dominanz und erst anschließend die geometrische Anordnung der Vektoren der Mengen berücksichtigt wird, besitzt die Charakterisierung des Dominanz-Verhaltens der Elemente eine höhere Priorität gegenüber der Charakterisierung ihrer geometrischen Anordnung im Zielraum.

Für das in Abbildung 7.2 dargestellte Beispiel mit den Approximationen PF_A^* und PF_B^* einer Pareto-Front werden nach der oben beschriebenen Vorgehensweise folgende Werte für das KDA ermittelt:

$KDA(PF_A^*, PF_B^*) = 6/3 = 2$ und $KDA(PF_B^*, PF_A^*) = 3/6 = 0.5$ da $count_A = 6$ und $count_B = 3$ ist.

Interpretiert man diese Werte, so ist die Approximation PF_A^* einer Pareto-Front bezüglich der Pareto-Dominanz und geometrischen Anordnung ihrer Elemente um den Faktor 2,5 leistungsfähiger als die Approximation PF_B^* dieser Pareto-Front.

Für die in Abbildung 7.1 illustrierten Beispiele liefert das KDA folgende Ergebnisse:

- Abbildung 7.1 linkes Bild: $KDA(PF_A^*, PF_B^*) > KDA(PF_B^*, PF_A^*)$, $KDA(PF_A^*, PF_C^*) > KDA(PF_C^*, PF_A^*)$ und $KDA(PF_C^*, PF_B^*) > KDA(PF_B^*, PF_C^*)$; damit wird nach dem KDA-Kriterium PF_A^* als die leistungsstärkste Approximation der Pareto-Front, gefolgt von PF_C^* eingeschätzt. Entsprechend des KDA ist die leistungsschwächste Approximation dieser Pareto-Front: PF_B^*.
- Abbildung 7.1 rechtes Bild: $KDA(PF_D^*, PF_E^*) > KDA(PF_E^*, PF_D^*)$: obwohl alle Elemente in $PF_D^* \cup PF_E^*$ bezüglich der Pareto-Dominanz gleichwertig sind, wird PF_D^* aufgrund der geometrischen Verteilung seiner Elemente als leistungsstärker bezüglich des KDA als PF_E^* eingeschätzt.

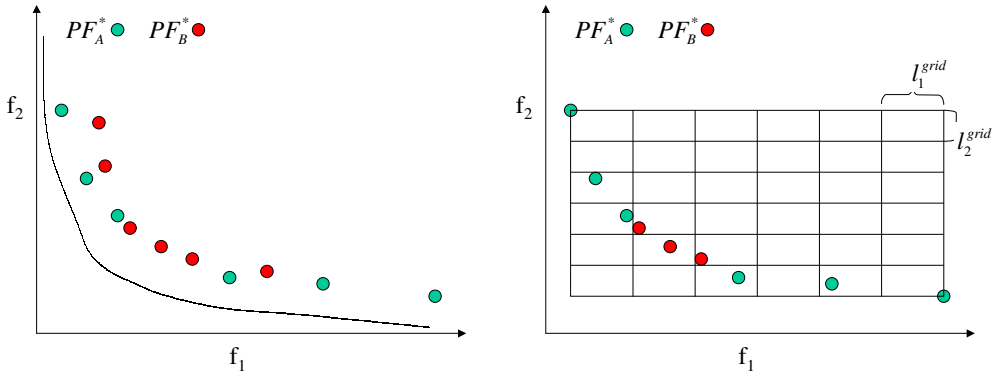


Abbildung 7.2: Berechnung des Kombinierten-Dominanz-Ausdehnungskriteriums

Dies veranschaulicht, daß es mit Hilfe des Kombinierten-Dominanz-Ausdehnungskriteriums möglich ist, die Leistungsfähigkeit zweier Approximationen einer Pareto-Fronten bezüglich ihres Dominanz-Verhaltens *und* ihrer geometrischen Verteilung zu beschreiben.

Da das KDA auf der Dominanz-Relation basiert, die reflexiv, antisymmetrisch und transitiv ist, erfüllt auch das KDA die Bedingungen der Transitivität, Antisymmetrie und Reflexivität und ist somit als Quasiordnungsrelation geeignet.

7.3 Auswertemethode

7.3.1 Charakterisierung der Güte des Optimierungskerns von MOMBES

Um die Leistungsfähigkeit des Optimierungskerns von MOMBES mit den in Abschnitt: 5.4 dargestellten existierenden MOEA objektiv vergleichen zu können, wurden Benchmarks zur Einschätzung der durch die MOEA generierten Approximationen der entsprechenden Pareto-Fronten für die aus der Literatur bekannten Testprobleme durchgeführt (Abschnitt 7.4).

MOMBES wurde mit den MOEA der ersten Generation: VEGA, HLGA, NPGA, NSGA, MOGA und SPEA in Abschnitt 7.4.1 und den modernen MOEA der zweiten Generation NSGA2, SPEA2 und PESA in den Abschnitten 7.4.2 bis 7.4.4 verglichen.

Für jeden der untersuchten Algorithmen lagen für die in den Abschnitten 7.4.1 bis 7.4.4 dargestellten Testprobleme Daten[81] für 30 voneinander unabhängig durchgeführte Testläufe,

[81] Diese Daten wurden der homepage von E.Zitzler: http://www.tik.ee.ethz.ch/~zitzler/ entnommen.

d.h. eine Testgruppe von jeweils 30 Approximationen der entsprechenden Pareto-Front pro MOEA und Testproblem, vor.

Zum Vergleich der Qualität der generierten Fronten wurden folgende Leistungskriterien verwendet: $\left|PF^*\right|$, L_p, S und Δ sowie KDA. Dabei wurden für jedes Testproblem die Mittelwerte und Standardabweichungen für L_p, S, Δ und $\left|PF^*\right|$ pro Testgruppe ermittelt und hier angegeben. Die Werte für die Größe des durch die Elemente der jeweiligen Fronten dominierten Hypervolumens S wird prozentual angegeben und bezieht sich auf die Größe des Hypervolumens, daß sich durch den Koordinatenursprung und den Punkt $\vec{y}'_{\text{ref}}$ im Zielraum aufspannt wird, wobei $\vec{y}'_{\text{ref}} = \vec{y}_{\text{ref}} - \vec{y}^{ideal}$ ist.

Es wurde weiterhin aus den bei den 30 Testläufen pro MOEA erzeugten Approximationen der Pareto-Front des jeweiligen Testproblems diejenige Front PF^*_{MOEA} ausgewählt, die über den größten KDA-Wert innerhalb ihrer Testgruppe verfügte und für den KDA-Vergleich der MOEA untereinander verwendet.

Damit war es möglich, das Dominanzverhalten der einzelnen Elemente der Approximationen einer Pareto-Front, ihre Ausdehnung und Homogenität der Verteilung im Zielraum sowie ihre Nähe zur jeweiligen globalen Pareto-Front zu beschreiben. Um einen Hinweis auf das Konvergenzverhalten der einzelnen Algorithmen zu erhalten, wurden die Auswertungen in den Abschnitten 7.4.2 bis 7.4.4 nach jeweils 100.000, 500.000 und 1.000.000 Berechnungen des Zielfunktionswertevektors durchgeführt.

In Abschnitt 7.4.1 wurden die Testprobleme von Zitzler und Thiele [Zitzler und Thiele, 1999] sowie Deb [Deb, 1999] verwendet. Bei den verwendeten Testproblemen handelt es sich um konstruierte multikriterielle Optimierungsprobleme, die die Fähigkeit der verwendeten Algorithmen untersuchen sollen, MOP´s mit konvexen (Abschnitt 7.4.1.1), konkaven (Abschnitt 7.4.1.2), diskreten (Abschnitt 7.4.1.3) und multimodalen (Abschnitt 7.4.1.4) Pareto-Fronten zu lösen. Bei den in den Abschnitten 7.4.2, 7.4.3 und 7.4.4 dargestellten Problemen handelt es sich um anerkannte künstliche Testprobleme, die jedoch schwieriger zu lösen sind und hauptsächlich dem Vergleich zwischen den moderneren Varianten von MOEA inklusive MOMBES dienen sollen. Es wurden MOP´s mit höherdimensionalen multimodalen Zielfunktionsvektor (Abschnitt 7.4.2), multimodalen Zielfunktionen, die eine extrem konkave Pareto-Front bestimmen (Abschnitt 7.4.3), und ein Optimierungsproblem mit Epistatsis der Entscheidungsvariablen (Abschnitt 7.4.4) betrachtet.

7.3.2 Charakterisierung der Güte des Decision-Making-Moduls von MOMBES

Zur Einschätzung der Güte des Decision-Making-Moduls von MOMBES soll die Güte der durch MOMBES erzeugten Modelle MOD_i, des finalen Pareto-Modells (FPM) sowie der Güte der Invertierung der Modelle charakterisiert werden. Dabei beziehen sich alle im folgenden beschriebenen Auswertungen auf jeweils nur eine durch MOMBES generierte Approximation PF^* der PF_{true} des entsprechenden MOP´s: es wird auch hier für die Auswertungen, wie oben, diejenige durch MOMBES generierte Approximation PF^*_{MOMBES} von PF_{true} ausgewählt, die über den größten KDA-Wert innerhalb ihrer Testgruppe verfügt.

Die für jedes Modell angegebenen Werte der Größe der radialen Basisfunktionen ($Sigma$) wurde heuristisch ermittelt, indem pro Modell jeweils ca. 20 Validierungsdurchläufe mit verschiedenen Werten für $Sigma$ durchgeführt wurden, und der Wert für $Sigma$ ausgewählt wurde, der den geringsten Vorhersagefehler auf der Testfraktion bestimmte. Dabei wurden die

Modelle für jeden Validierungsdurchlauf mit jeweils 80% zufällig entnommenen Trainingsvektoren trainiert und anhand der verbliebenen 20% getestet.

Charakterisierung der Güte der Modelle MOD_i: Die MOD_i modellieren innerhalb einer Approximation PF^* der Pareto-Front den Zusammenhang zwischen jeweils einer der k Zielfunktionen in Abhängigkeit von den komplementären $(k-1)$ Zielfunktionen (siehe Abschnitt 6.5.2.1). Um die Güte jedes der k Modelle MOD_i pro PF^* zu testen, wurden die Elemente aus PF^* jeweils 10-mal zufällig in eine Trainings- und eine Testmenge aufgeteilt, die MOD_i mit den Daten der Trainingsmenge trainiert und auf die Daten beider Mengen angewandt. In der Trainingsmenge befinden sich jeweils 80% und in der Testmenge 20% der Vektoren aus PF^*. Es wird für jeden Datensatz die absolute Abweichung zwischen dem durch die MOD_i prognostizierten Modelloutput $\hat{y}$ und dem entsprechenden, durch die Trainingsdaten vorgegebenen Wert $y_i^p \in PF^*$ als Vorhersagefehler δ^{mod_i} ermittelt sowie die mittleren Werte $\overline{\delta}^{mod\_i}$ aller Datensätze für alle 10 Test- und Trainingsmengen pro MOP angegeben. Dabei beziehen sich die Angaben des Vorhersagefehlers in Prozent auf die durch die Elemente aus PF^* bestimmten Wertebereiche der einzelnen Dimensionen der Zielfunktionswertevektoren.

$$\delta^{mod\_i} = \left| \hat{y} - y_i^p \right| \qquad [66]$$

Charakterisierung der Güte der finalen Pareto-Modelle: Das finale Pareto-Modell modelliert die Abbildung der Vektoren aus P^* auf die Vektoren aus PF^* (siehe Abschnitt 6.5.2.3). Zur Charakterisierung der Güte des finalen Pareto-Modells pro PF^* werden auch hier jeweils die Vektoren aus P^* und PF^* 10-mal zufällig in eine Trainings- und eine Testfraktion unterteilt, das finale Pareto-Modell mit den Daten der Trainingsfraktion trainiert und auf die Daten der Trainings- und Testfraktion angewandt. Wiederum werden 80% der Vektoren der jeweiligen Menge als Trainings- und 20% als Testdaten verwendet. Anschließend wird für jede Komponente i des Zielfunktionswertevektors die absolute Abweichung δ_i^{FPM} zwischen jedem Modelloutput $\hat{y}_i$ und dem entsprechenden Wert $y_i^p \in PF^*$ bestimmt und ihr Mittelwert $\overline{\delta}_i^{FPM}$ für alle 10 Test- und Trainingsmengen angegeben. Wie oben beziehen sich die prozentualen Angaben auch hier auf den durch die Elemente aus PF^* bestimmten Wertebereich für jedes y_i.

$$\delta_i^{FPM} = \left| \hat{y}_i - y_i^p \right| \qquad [67]$$

Charakterisierung der Güte der Modellinvertierung: Für die Charakterisierung der Güte der Invertierung der Teilmodelle MOD_i und der finalen Pareto-Modelle FPM wurden die jeweiligen Modelle zunächst mit allen Vektoren aus PF^* und bei den finalen Pareto-Modellen aus P^* und PF^* der betrachteten Approximationen der Pareto-Menge und Pareto-Front traininiert.

Anschließend wurde jeweils eine Komponente i des Zielfunktionswertevektors als Target y_i^{target} vorgegeben, indem der durch die Elemente aus PF^* bekannte Wertebereich der entsprechenden Größe y_i in 500 äquidistante Abschnitte unterteilt wurde.

Zunächst wurde dann für jedes y_i^{target} durch die Invertierung des finalen Pareto-Modells eine Target-Lösung $\vec{x}^{target}$ berechnet, die bei Propagierung durch das FPM einen Modelloutput $\hat{\vec{y}}$ bestimmt, dessen Komponente $\hat{y}_i$ einen möglichst geringen Abstand zu y_i^{target} aufweist. Diese absolute Abweichung zwischen y_i^{target} und $\hat{y}_i$ wird als Invertierungsfehler $\delta_i^{INV\_FPM}$ des finalen Pareto-Modells bezeichnet, dessen prozentuale Angaben sich wie oben auf den Wertebereich der entsprechenden Komponente der Vektoren aus PF^* beziehen.

$$\delta_i^{INV\_FPM} = \left| \hat{y}_i - y_i^{target} \right| \qquad [68]$$

Damit die Güte der Modellinvertierung des FPM ohne Einfluß der Güte der Teilmodelle MOD_i untersucht werden konnte, wurde auf die Berechnung des vollständigen Target-Vektors $\vec{y}^{target}$ (siehe Abschnitt 6.5.2.1) verzichtet und bei der Invertierung des finalen Pareto-Modells nur die Annäherung von $\hat{y}_i$ an y_i^{target} berücksichtigt. Damit $\vec{y}^{target}$ bei Vorgabe nur einer Komponente eindeutig bestimmt war, wurden nur MOP mit $k = 2$ Zielfunktionen betrachtet.

Analog der obigen Vorgehensweise für die Charakterisierung der Invertierung der FPM erfolgt die Beschreibung der Güte der Invertierung der Teilmodelle MOD_i durch den Invertierungsfehler $\delta^{INV\_mod\_i}$. Hier wird durch die Modellinvertierung bei Vorgabe von y_i^{target} für MOD_i ein Vektor $\vec{y}_{Ni}^{target}$ bestimmt, der bei Propagierung durch das Modell MOD_i einen Modelloutput $\hat{y}$ liefert, dessen Wert eine möglichst geringe Abweichung zu y_i^{target} besitzt. Der absolute Abstand zwischen $\hat{y}$ und y_i^{target} wird als Invertierungsfehler $\delta^{INV\_mod\_i}$ der Teilmodelle MOD_i bezeichnet, dessen prozentuale Angaben sich auch hier auf den Wertebereich der entsprechenden Komponente der Vektoren aus PF^* beziehen.

$$\delta^{INV\_mod\_i} = \left| \hat{y} - y_i^{target} \right| \qquad [69]$$

Zur Charakterisierung der Güte der Modellinvertierung werden die Mittelwerte für alle Invertierungsfehler für alle 500 Angaben von y_i^{target} der entsprechenden Simulationen angegeben. Die mindestens einzuhaltende Konfidenz bei der Invertierung der Modelle wurde hierbei für alle Simulationen auf 0,9 festgesetzt.

7.4 Benchmarks zur Charakterisierung der Güte des Optimierungskerns von MOMBES

7.4.1 Testprobleme nach Zitzler, Deb und Thiele (ZTD-1 bis ZTD-4)

Nach Deb [Deb, 1999] werden Optimierungsprobleme mit lediglich zwei Zielfunktionen betrachtet, die dem Ausdruck [70] folgen.

Dabei ist f_1 nur vom Wert der Entscheidungsvariable x_1 abhängig. Durch Modifikation von $g(\vec{x})$ und $h(\vec{x})$ ist die Realisierung von Testproblemen möglich, die verschiedenen Problemgruppen genügen: es wurden MOP´s mit konvexen (ZTD-1), konkaven (ZTD-2) und diskreten (ZTD-3) sowie multimodalen (ZTD-4) globalen Pareto-Fronten untersucht.

$$\begin{aligned} &\textit{min: } \vec{f}(\vec{x}) = [f_1(x_1), f_2(\vec{x})] \\ &\textit{mit: } f_2(\vec{x}) = g(x_2,...,x_n) \cdot h(f_1(x_1), g(x_2,...,x_n)) \\ &\textit{und: } \vec{x} = [x_1,...,x_n] \end{aligned} \qquad [70]$$

Die globale Pareto-Front wird bei den Testproblemen ZTD-1, ZTD-2 und ZTD-3 bei $g = 1$ und bei dem Testproblem ZTD-4 bei $g = 1{,}25$ herausgebildet. Für das Problem ZTD-4 existieren 21^9 lokale Pareto-Fronten. Dieses Problem ist daher besonders für den Vergleich der Leistungsfähigkeit der MOEA geeignet, multimodale Optimierungsprobleme zu lösen ([Zitzler, 1999] S.59).

Es wurden folgende MOEA für die Vergleiche verwendet: VEGA, HLGA, NPGA, NSGA, MOGA, SPEA. Dabei wurden folgende Parametereinstellungen auf die Vergleichs-MOEA angewandt:

Anzahl der Generationen:	$t_{max} = 250$
Größe der Population bei VEGA, HLGA, NPGA, NSGA und MOGA:	$N = 100$
Größe der Elitepopulation bei SPEA:	$\lvert\overline{P}_t\rvert = 20$
Größe der Population bei SPEA:	$N = 80$
Wahrscheinlichkeit für 1-Punkt-crossover:	$p_c = 0{,}8$
Mutationsrate pro Bit:	$p_m = 0{,}01$
Nischenradius:	$\sigma_{share} = 0{,}4886$
Größe der Vergleichsmenge beim NPGA:	$t_{dom} = 10\%$
Anzahl der Bits pro kodierter Entscheidungsvariable:	$n_{Bits} = 30$

Für MOMBES wurden folgende Einstellungen verwendet:

Anzahl der function calls:	$n_{function\ call} = N + (2 \cdot N \cdot t_{max}) =$ $100 + (2 \cdot 100 \cdot 250) = 50\ 100$
Anzahl der Eltern:	$\mu = 15$
Anzahl der Nachkommen:	$\lambda = 100$
Anzahl der Wettkämpfer:	$\rho = 2$
Größe der Elitepopulation:	$\lvert\overline{P}_t\rvert = 100$
Wahrscheinlichkeit für das Einfügen approximierter Individuen:	$p_a = 0{,}2$

Um einen fairen Vergleich zwischen MOMBES und den zitierten MOEA durchzuführen, wurde MOMBES nach einer festen Anzahl von Berechnungen des Zielfunktionswertevektors (function calls), die den oben genannten Parametereinstellungen der Genetischen Algorithmen entsprechen, abgebrochen.

7.4.1.1 Konvexe Pareto-Front (ZTD-1)

Der in den Termen [71] beschriebene Ausdruck definiert das als ZTD-1 aus der Literatur [Zitzler, 1999] bekannte Beispiel eines MOP´s mit konvexer Pareto-Front, das für den in diesem Abschnitt beschriebenen Leistungsvergleich herangezogen wurde.

$$\begin{gathered} \textit{min}:\ \vec{f}(\vec{x}) = [f_1(x_1), f_2(\vec{x})]; \\ f_1(x_1) = x_1; \quad f_2(\vec{x}) = g(x_2,..,x_n)\cdot h(f_1,g) \quad \textit{mit}: \\ g(x_2,...,x_n) = 1 + 9\cdot\frac{\sum_{i=2}^{n} x_i}{n-1}; \quad h(f_1,g) = 1 - \sqrt{\frac{f_1}{g}}; \quad \textit{und } n = 30 \textit{ sowie } x_i \in [0;1] \end{gathered} \qquad [71]$$

In Tabelle 7.1 wird die Leistungsfähigkeit der durch die untersuchten MOEA generierten Approximationen der Pareto-Front bezüglich der Anzahl ihrer Elemente, deren geringste euklidische Distanz zum Idealvektor (L_p), ihrer Homogenität (Δ) sowie der Größe S des von ihnen dominierten Raums bezüglich des Referenzpunktes [10;10] dargestellt.

	$\left\|PF^*_{MOEA}\right\|$	L_p *mit* $\vec{y}^{ideal} = [0;0]$	Δ	S *in % bezüglich* $\vec{y}_{ref} = [10;10]$
MOGA	$25{,}8 \pm 5{,}8$	$1{,}28 \pm 0{,}06$	$0{,}016 \pm 0{,}009$	$90{,}2 \pm 0{,}86$
NPGA	$21{,}7 \pm 4{,}6$	$1{,}3 \pm 0{,}07$	$0{,}013 \pm 0{,}011$	$89{,}9 \pm 0{,}70$
VEGA	$22{,}3 \pm 4{,}7$	$1{,}18 \pm 0{,}08$	$0{,}018 \pm 0{,}014$	$90{,}9 \pm 1{,}01$
HLGA	$21{,}6 \pm 4{,}15$	$1{,}17 \pm 0{,}08$	$0{,}016 \pm 0{,}013$	$89{,}89 \pm 0{,}96$
NSGA	$43{,}36 \pm 5{,}97$	$0{,}75 \pm 0{,}02$	$0{,}003 \pm 0{,}002$	$96{,}6 \pm 0{,}25$
SPEA	$72{,}4 \pm 8{,}49$	$0{,}57 \pm 0{,}005$	$0{,}001 \pm 0{,}0008$	$99{,}16 \pm 0{,}089$
MOMBES	$99{,}93 \pm 0{,}25$	$0{,}57 \pm 0{,}008$	$0{,}0007 \pm 0{,}0002$	$99{,}2 \pm 0{,}086$

Tabelle 7.1: Performancevergleich bezüglich $\left|PF^*\right|$, L_p, Δ und S für ZTD-1

Es ist zu erkennen, daß MOMBES in jeder Kategorie die günstigsten Werte aufweist. So wurden von MOMBES bei fast jedem der Testdurchläufe ca. 100, d.h. die maximal zulässige Anzahl, nicht-dominierter Lösungen generiert. Dagegen erreichten die durch NPGA, HLGA, VEGA und MOGA generierten Fronten durchschnittlich nur ca. 25% der maximal zu erreichenden Größe. SPEA und NSGA generierten durchschnittlich ca. 50% der maximal zulässigen Anzahl von Elementen von PF^*, wobei bei SPEA dabei die höchste Streuung zu beobachten war. Der Abstand zum Idealvektor war sowohl bei SPEA als auch bei MOMBES am geringsten und betrug z.T. weniger als die Hälfte der entsprechenden Werte bei MOGA, VEGA, NPGA oder HLGA. Betrachtet man die Homogenität der Verteilung der Elemente von PF^*, so ist für die die Homogenität beschreibende Größe Δ bei MOMBES sowohl der Mittelwert als auch die Streuung am geringsten, gefolgt von den Werten für SPEA und NSGA. Schließlich sind auch die mittleren Werte für S, die den durch die Elemente von PF^* dominierten Raum bezüglich des Referenzpunktes [10,10] beschreiben, für MOMBES am größten, unterscheiden sich jedoch nur geringfügig von denen für SPEA. Sowohl bezüglich der Mittelwerte als auch der Streuung der Größe S sind SPEA und MOMBES den betrachteten Konkurrenz-MOEA deutlich überlegen. Der durch die obige Auswertung gewonnene Eindruck der Überlegenheit von MOMBES gegenüber den untersuchten Vergleichs-MOEA wird durch seine KDA-Werte gestützt (siehe Tabelle 7.2). Es ist zu erkennen, daß MOMBES bezüglich des KDA-Kriteriums bei keinem der betrachteten MOEA einen Wert kleiner als 1 aufweist, d.h. jedem der betrachteten MOEA überlegen ist. Die Werte von 100 sagen aus, daß jedes der 100 Elemente der durch

MOMBES generierten Front PF^*_{MOMBES} jedes Element der durch den Vergleichs-MOEA generierten Approximationen PF^*_{MOGA}, PF^*_{NPGA}, PF^*_{VEGA}, PF^*_{HLGA} und PF^*_{NSGA} streng dominiert.

KDA(A,B)		B							
		MOGA	**NPGA**	**VEGA**	**HLGA**	**NSGA**	**SPEA**	**MOMBES**	**Mittelwert**
A	**MOGA**		0,66	0,07	0,00	0,00	0,00	0,00	0,01
	NPGA	1,50		0,15	0,25	0,00	0,00	0,00	0,08
	VEGA	13,00	6,50		0,09	0,00	0,00	0,00	0,02
	HLGA	22,00	4,00	11,00		0,00	0,00	0,00	2,75
	NSGA	44,00	44,00	44,00	44,00		0,00	0,00	22,00
	SPEA	81,00	81,00	81,00	81,00	81,00		0,0028	60,75
	MOMBES	100,00	100,00	100,00	100,00	100,00	35,00		83,75

Tabelle 7.2: Performancevergleich bezüglich $KDA(A,B)$ für ZTD-1

Allein in der durch SPEA generierten Front PF^*_{SPEA} existieren Elemente, die Elemente aus PF^*_{MOMBES} streng dominieren. Jedoch ist bei PF^*_{MOMBES} sowohl die Anzahl der nicht-dominierten Elemente, als auch ihre Verteilung um das 35-fache günstiger als bei SPEA. Betrachtet man die Mittelwerte für das KDA der einzelnen MOEA, so wird MOMBES als am leistungsstärksten eingeschätzt, gefolgt von SPEA und NSGA. MOGA, NPGA und VEGA sind ihnen hingegen deutlich unterlegen.

Die durch die Werte der objektiven Leistungskriterien beschriebene Überlegenheit von MOMBES wird durch den visuellen Eindruck ihrer PF^*_{MOEA} bestätigt (Abbildung 7.3).

Man erkennt, daß die von MOMBES generierte Approximation der Pareto-Front sich am besten der theoretisch möglichen PF_{true}, die durch die funktionale Abhängigkeit: $f_2 = 1 - \sqrt{f_1}$ beschrieben wird, anschmiegt und ihre Elemente sowohl über eine hohe Homogenität der Verteilung, als auch breite Abdeckung im Zielraum verfügen.

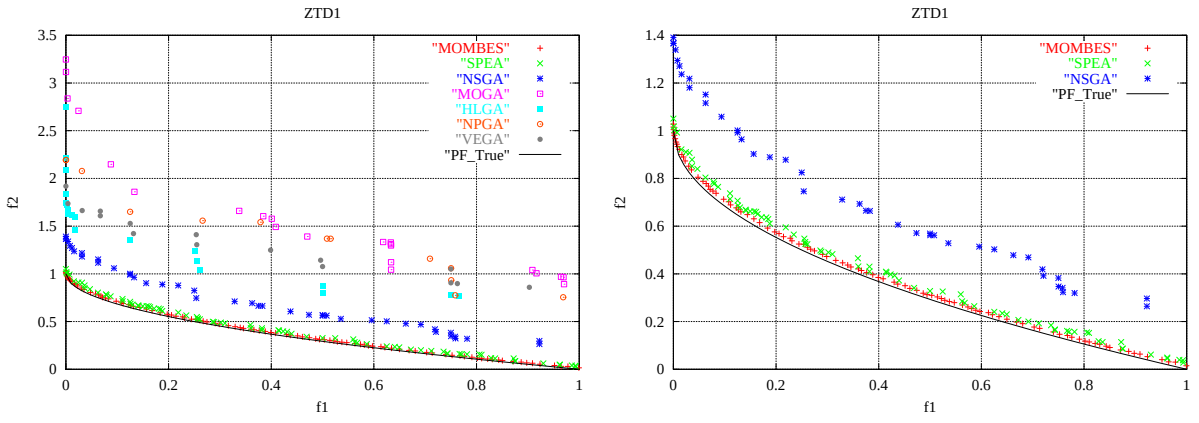


Abbildung 7.3: PF^*_{MOEA} für ZTD-1

7.4.1.2 Konkave Pareto-Front (ZTD-2)

In diesem Abschnitt werden analog zu Abschnitt 7.4.1.1 die Ergebnisse des Leistungsvergleichs zwischen MOMBES und den Vergleichs-MOEA auf der Basis des in den Termen [72] beschriebenen MOP´s mit konkaver Pareto-Front dargestellt.

$$\textit{min}: \ \vec{f}(\vec{x}) = [f_1(x_1), f_2(\vec{x})]$$

$$f_1(x_1) = x_1; \qquad g(x_2,...,x_n) = 1 + 9 \cdot \frac{\sum_{i=2}^{n} x_i}{n-1}; \qquad h(f_1, g) = 1 - \left(\frac{f_1}{g}\right)^2 \qquad [72]$$

$$\textit{mit } n = 30 \textit{ und } x_i \in [0;1]$$

Auch hier wird aus Tabelle 7.3 und Tabelle 7.4 die Leistungsfähigkeit von MOMBES im Vergleich mit MOGA, NPGA, VEGA, HLGA, NSGA und SPEA hinsichtlich der Generierung einer homogen, breit über den Zielraum verteilten und sich gut an PF_{true} anschmiegenden PF^* deutlich. Die theoretisch möglichen Pareto-Front PF_{true} definiert sich hier durch die Menge der Paare $\{[f_1(x), f_2(x)]\}$ mit $f_2 = 1 - f_1^2$.

	$\lvert PF^* \rvert$	L_p *mit* $\vec{y}^{ideal} = [0;0]$	Δ	S *in % bezüglich* $\vec{y}_{ref} = [10;10]$
MOGA	15,16 ± 4,80	2,19 ± 0,19	0,026 ± 0,015	79,51 ± 2,12
NPGA	9,83 ± 2,43	1,78 ± 0,108	0,036 ± 0,024	84,42 ± 1,35
VEGA	2,93 ± 1,33	1,65 ± 0,061	0,043 ± 0,052	84,12 ± 1,06
HLGA	6,06 ± 2,42	1,83 ± 0,08	0,065 ± 0,059	81,87 ± 0,86
NSGA	19,86 ± 3,26	1,15 ± 0,028	0,008 ± 0,005	92,77 ± 0,55
SPEA	43,53 ± 9,10	0,92 ± 0,014	0,002 ± 0,001	97,65 ± 0,38
MOMBES	100,0 ± 0,0	0,88 ± 0,05	0,0005 ± 0,0003	98,70 ± 0,15

Tabelle 7.3: Performancevergleich bezüglich $\lvert PF^* \rvert$, L_p, Δ und S für ZTD-2

Bei gleicher Anzahl von function calls ist MOMBES als einziger Algorithmus bei jedem Testlauf in der Lage, die maximale Anzahl von nicht-dominierten Lösungen zu generieren und zeichnet sich bei jedem der 30 Testläufe durch eine hohe Diversität der Population bezüglich der Pareto-Dominanz aus. Gleichzeitig ist auch hier über alle Testläufe der geringste Mittelwert des jeweils kleinsten euklidischen Abstand zwischen dem Idealvektor (hier Koordinatenursprung im Zielraum) und den Elementen der durch MOMBES generierten Approximationen der Pareto-Front zu finden. Die Homogenität der generierten Fronten ist bei MOMBES und SPEA am größten. Bezüglich des Referenzpunktes [10;10] dominieren die durch MOMBES und SPEA generierten Elemente von PF^* den größten Anteil im Zielraum, wobei die Mittelwerte von S bei MOMBES die von SPEA geringfügig überschreiten.

Deutlich wird die Überlegenheit von MOMBES gegenüber den betrachteten Vergleichs-MOEA auch bei der Betrachtung der KDA-Werte der jeweils besten Fronten bezüglich des KDA, die bei den 30 Testdurchläufen pro MOEA generiert wurden. Man erkennt, daß MOGA, HLGA, NPGA und VEGA diejenigen MOEA sind, die sich am wenigsten für das verwendete Testproblem eignen. Ihre mittleren KDA-Werte liegen weit unter denen von NSGA, SPEA und MOMBES. Alle von MOMBES generierten Elemente von PF^* dominieren alle Elemente aller Vergleichs-MOEA streng. Hingegen werden alle Elemente der durch MOGA generierten PF^* von allen Elementen aller anderen Fronten streng dominiert.

KDA(A,B)		B							
		MOGA	**NPGA**	**VEGA**	**HLGA**	**NSGA**	**SPEA**	**MOMBES**	**Mittelwert**
A	**MOGA**		0,00	0,00	0,00	0,00	0,00	0,00	0,00
	NPGA	10,00		0,33	3,00	0,00	0,00	0,00	2,22
	VEGA	6,00	3,00		6,00	0,00	0,00	0,00	2,50
	HLGA	5,00	0,33	0,00		0,00	0,00	0,00	0,89
	NSGA	22,00	22,00	22,00	22,00		0,00	0,00	14,67
	SPEA	54,00	54,00	54,00	54,00	54,00		0,00	45,00
	MOMBES	100,00	100,00	100,00	100,00	100,00	100,00		100,00

Tabelle 7.4: Performancevergleich bezüglich $KDA(A,B)$ für ZTD-2

In Abbildung 7.4 wird für jeden MOEA die Approximation von PF_{true} mit dem größten KDA-Wert innerhalb ihrer Testgruppe graphisch dargestellt. Man erkennt, daß die von SPEA und MOMBES generierten Approximationen sich etwa gleich gut an die PF_{true} anschmiegen, die von MOMBES erzeugte Front jedoch gerade im Bereich $f_1 \in [0;0,2]$ über eine dichtere Verteilung ihrer Elemente verfügt.

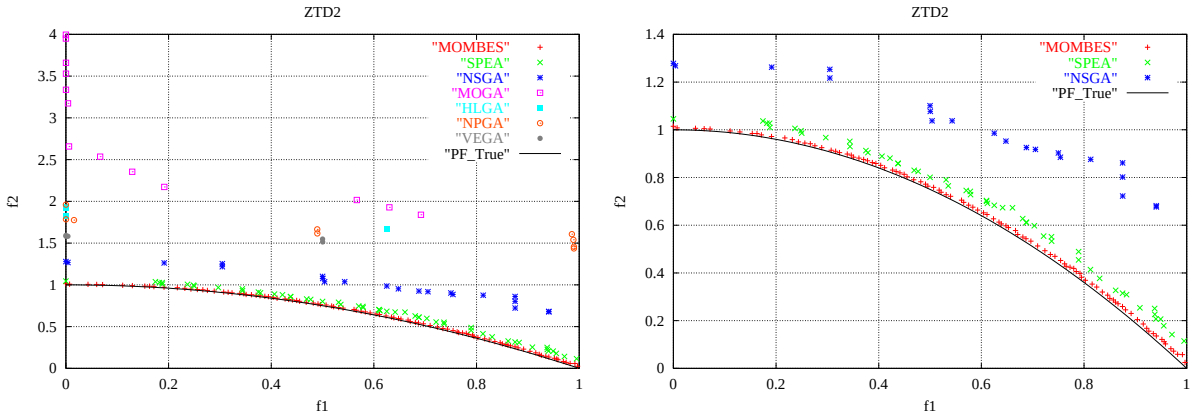


Abbildung 7.4: PF^{*}_{MOEA} für ZTD-2

Wie auch schon aus den Werten der Leistungskriterien aus Tabelle 7.3 und Tabelle 7.4 ersichtlich, kann MOMBES als am leistungsstärksten, gefolgt von SPEA und NSGA eingeschätzt werden. MOGA, NPGA, HLGA und VEGA sind hingegen, wie bereits auch in Abschnitt 5.4 beschrieben, nicht in der Lage, ebenso hochwertige Approximationen von konkaven Pareto-Fronten zu generieren. MOMBES verfügt bei diesem Testproblem über ein sehr gutes Konvergenzverhalten, d.h. hat nach einer vergleichsweise geringen Anzahl von function calls eine qualitativ hochwertige Approximation der Pareto-Front generiert.

7.4.1.3 Diskrete Pareto-Front (ZTD-3)

In den Termen [73] wird ein MOP beschrieben, daß über eine Pareto-Front verfügt, die aus mehreren, nicht miteinander verbundenen Teilstücken besteht. Da die Pareto-Front nicht stetig ist, ist es in diesem Fall nicht sinnvoll, das Qualitätsmaß Δ zur Charakterisierung der Homogenität der Verteilung der Elemente der generierten Pareto-Fronten zu verwenden.

$$\textit{min: } \vec{f}(\vec{x}) = [f_1(x_1), f_2(\vec{x})]$$

$$f_1(x_1) = x_1; \qquad g(x_2,...,x_n) = 1 + 9 \cdot \frac{\sum_{i=2}^{n} x_i}{n-1}\ ; \qquad h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}} - \frac{f_1}{g} \cdot \sin(10 \cdot \pi \cdot f_1) \quad [73]$$

$$\textit{mit } n = 30 \textit{ und } x_i \in [0;1]$$

Daher werden in Tabelle 7.5 die durch die unterschiedlichen MOEA generierten Approximationen der Pareto-Front lediglich bezüglich ihrer mittleren Größe, ihrer mittleren geringsten euklidische Distanz zum Idealvektor und der mittleren durch ihre Elemente dominierten Größe im Zielraum gegenübergestellt.

	$\lvert PF^* \rvert$	L_p *mit* $\vec{y}^{ideal} = [0;-1]$	S *in % bezüglich* $\vec{y}_{ref} = [10;10]$
MOGA	28,0 ± 6,01	1,75 ± 0,12	84,95 ± 1,38
NPGA	28,1 ± 4,49	1,60 ± 0,09	86,59 ± 1,09
VEGA	23,1 ± 4,97	1,65 ± 0,11	86,3 ± 1,43
HLGA	19,1 ± 4,12	1,78 ± 0,18	84,57 ± 1,96
NSGA	46,13 ± 6,77	1,02 ± 0,04	93,9 ± 0,97
SPEA	69,66 ± 0,01	0,88 ± 0,01	96,51 ± 0,68
MOMBES	98,10 ± 3,60	0,87 ± 0,11	90,71 ± 0,02

Tabelle 7.5: Performancevergleich bezüglich $\lvert PF^* \rvert$, L_p und S für ZTD-3

Auch bei diesen Simulationen zeigt sich, daß MOMBES derjenige MOEA ist, der für dieses MOP Approximationen der Pareto-Front generierte, die durchschnittlich über die meisten Elemente mit der geringsten euklidischen Distanz zum Idealvektor verfügen. SPEA und NSGA stellten auch für dieses Problem die stärksten Konkurrenten für MOMBES dar. Beide MOEA konnten Approximationen der Pareto-Front mit einer großen Anzahl von Elementen generieren, die jedoch unter der von MOMBES lag. MOGA, NPGA, VEGA und HLGA konnten auch bei diesem Testproblem keine Approximationen der Pareto-Front generieren, die ihrem tatsächlichen Verlauf folgen. Eine deutlich bessere Qualität als alle anderen betrachteten MOEA, inklusive MOMBES, erreichen SPEA und NSGA bezüglich des Parameters S. Die Elemente der von ihnen generierten Fronten dominieren einen deutlich größeren Bereich im Zielraum, als die Elemente der durch die anderen MOEA generierten Fronten. Im Mittel liegen die Werte von S bei SPEA um ca. 10% über denen von MOGA, NPGA, VEGA, HLGA und MOMBES. Betrachtet man die graphische Darstellung in Abbildung 7.5, so ist erkennbar, daß hier sowohl SPEA als auch NSGA in der Lage sind, eine aus 5 Teilstücken bestehende Approximation der Pareto-Front zu generieren, wogegen MOMBES lediglich eine nur aus 3 Teilstücken bestehende Front erzeugte, deren Elemente jedoch eine geringe Distanz zur PF_{true} aufweisen. Die Besonderheit des Testproblems [73] besteht in seiner sinusoidalen Abhängigkeit zwischen den beiden Zielfunktionen, was dazu führt, daß die PF_{true} für dieses Testproblem nicht zusammenhängend ausgeprägt ist. Aufgrund dieser Abhängigkeit existieren eine Anzahl von lokalen Minima im Verlauf der Funktion $f_2(f_1)$ (siehe Abbildung 7.5). Gerade die Kombination mit einem Gradientenverfahren, die MOMBES zu seiner guten Konvergenzeigenschaft verhilft, wirkt sich hier negativ aus: zum einen, weil Gradientenverfahren schlecht einmal gefundene lokale Minima zu Gunsten eines globalen Minimums verlassen können, und zum anderen, weil durch die rasche Konvergenz die Elemente in den bereits generierten Stücken der Front solche Elemente dominieren, die Bereiche im Zielraum mit

$f_1 > 0{,}6$ repräsentieren. MOMBES generierte daher im Vergleich zu allen anderen MOEA diejenigen Approximationen der Pareto-Front, deren Elemente die Fronten der Vergleichs-MOEA dominierten, verfügt aber andererseits über eine geringere Diversität als die von SPEA und NSGA generierten Approximationen. Der Vorteil des Einfügens approximierter Individuen geht bei MOMBES verloren, da in Regionen geringer Dichte der aktuellen Approximation der Pareto-Front zwar eine lokale Optimierung angestoßen wird, die jedoch zu keiner Verbesserung der Qualität von PF^*_{MOMBES} führen kann, da die Pareto-Front in genau diesen Regionen nicht existiert.

KDA(A,B)		B							
		MOGA	**NPGA**	**VEGA**	**HLGA**	**NSGA**	**SPEA**	**MOMBES**	**Mittelwert**
A	**MOGA**		0,31	0,04	0,15	0,00	0,00	0,00	0,08
	NPGA	3,20		0,12	0,25	0,00	0,00	0,00	0,60
	VEGA	23,00	8,33		0,92	0,00	0,00	0,00	5,37
	HLGA	6,50	4,00	1,09		0,00	0,00	0,00	1,93
	NSGA	42,00	42,00	42,00	42,00		0,00	0,27	28,05
	SPEA	85,00	85,00	85,00	85,00	85,00		0,74	70,96
	MOMBES	100,00	100,00	100,00	100,00	3,60	1,34		67,49

Tabelle 7.6: Performancevergleich bezüglich $KDA(A,B)$ für ZTD-3

In Tabelle 7.6 werden die Werte für das KDA der unterschiedlichen MOEA dargestellt. Auch hier wird deutlich, daß SPEA diejenige Approximation der Pareto-Front erzeugte, die im Durchschnitt über den höchsten KDA-Wert verfügt. Zwar ist lt. KDA MOMBES jedem der betrachteten MOEA überlegen (d.h. besitzt einen KDA-Wert größer 1), jedoch ist der Wert bezüglich SPEA und NSGA nur geringfügig größer als 1. Die Gründe dafür liegen darin, daß die Elemente aus PF^*_{MOMBES} zwar dominant gegenüber den Elementen aus PF^*_{SPEA} und PF^*_{NSGA} sind, jedoch über eine geringere Variabilität im Zielraum verfügen.

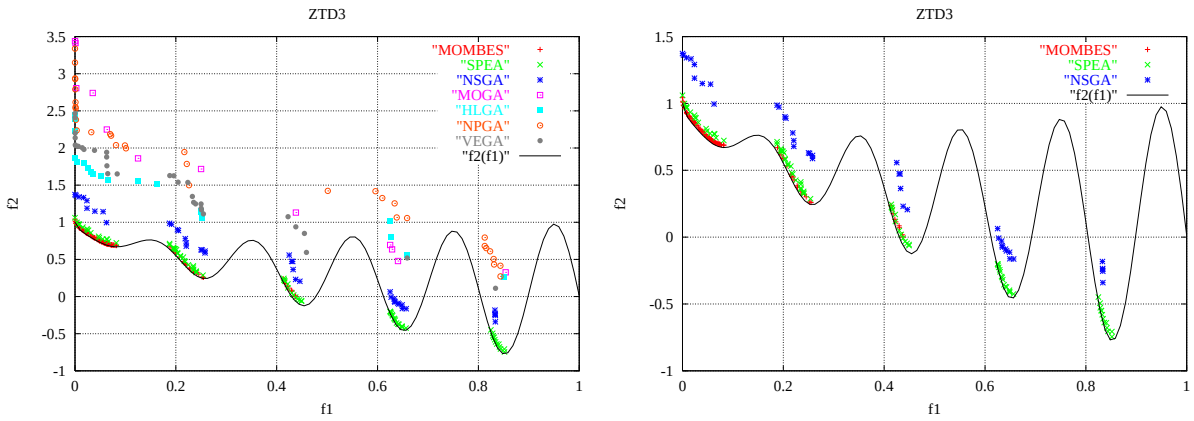


Abbildung 7.5: PF^*_{MOEA} für ZTD-3

Hingegen dominieren alle 85 Elemente aus PF^*_{SPEA} alle Elemente aus PF^*_{MOGA}, PF^*_{NPGA}, PF^*_{VEGA}, PF^*_{HLGA} und PF^*_{NSGA} streng. Lediglich gegenüber MOMBES ist SPEA bezüglich des KDA-Wertes unterlegen ($KDA(PF^*_{MOMBES}, PF^*_{SPEA}) = 0{,}74$). Der KDA-Wert für SPEA ist daher im Mittel sehr hoch. Zusammenfassend wird festgestellt, daß für MOP´s mit diskontinuierlicher

Pareto-Front MOMBES weniger gut geeignet ist als SPEA und NSGA, jedoch den MOEA der ersten Generation überlegen ist.

7.4.1.4 Multimodale Pareto-Fronten (ZTD-4)

In diesem Abschnitt wird die Leistungsfähigkeit der verschiedenen MOEA untersucht, ein MOP zu lösen, das über eine große Anzahl von lokalen Pareto-Fronten verfügt.

$$\begin{gathered} \textit{min: } \vec{f}(\vec{x}) = [f_1(x_1), f_2(\vec{x})] \\ f_1(x_1) = x_1; \qquad g(x_2,...,x_n) = 1 + 10 \cdot (n-1) + \sum_{i=2}^{n} \left(x_i^2 - 10 \cdot \cos(4 \cdot \pi \cdot x_i)\right); \\ h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}} \\ \textit{mit } n = 10 \textit{ und } x_1 \in [0;1] \textit{ sowie } x_i \in [-5;+5]; \; i = 2,..,n \end{gathered} \qquad [74]$$

Die funktionale Abhängigkeit zwischen den Zielfunktionen innerhalb der globalen Pareto-Front wird durch Gleichung [75] beschrieben.

$$f_2 = 1{,}25 \cdot \left(1 - \sqrt{\frac{f_1}{1{,}25}}\right) \qquad [75]$$

Aus den in Tabelle 7.7 und Tabelle 7.8 dargestellten Ergebnissen der Simulation geht hervor, daß auch für dieses Testproblem MOMBES, SPEA und NSGA sehr gute Ergebnisse gegenüber MOGA, NPGA, VEGA und HLGA erzielt haben. Die Anzahl der generierten näherungsweise pareto-optimalen Lösungen ist bei MOMBES und SPEA durchschnittlich um ca. das 10-fache größer als bei den betrachteten Vergleichs-MOEA. Bezüglich der Distanz zum Idealvektor übertrifft MOMBES alle anderen MOEA (inklusive SPEA) deutlich. Der mittlere geringste euklidische Abstand der Elemente von der von MOMBES generierten Approximationen der Pareto-Front zum Idealpunkt liegt bei nur 0,76. Dieser Wert ist ca. 5-fach geringer als der entsprechende Wert von SPEA und ca. 40-fach geringer als der von MOGA. Die Homogenität der Verteilung der Elemente der von MOMBES und SPEA generierten Fronten ist etwa gleichwertig und mit einem Wert von ca. 0,01 gegenüber den anderen Fronten vergleichsweise gering. Klar erkennbar wird die gute Performance von MOMBES, SPEA und NSGA bei der Betrachtung der mittleren Größe des durch die Elemente ihrer Fronten dominierten Bereichs des Zielraums. Während die durch MOGA generierten Fronten bezüglich des Referenzpunktes [10;10] keinen Bereich im Zielraum dominieren, besitzt das Hypervolumen bei SPEA eine mittlere Größe von 57,86% und bei MOMBES eine mittlere Größe von 97,24%.

	$\lvert PF^* \rvert$	L_p *mit* $\vec{y}^{ideal} = [0;0]$	Δ	S *in % bezüglich* $\vec{y}_{ref} = [10;10]$
MOGA	$8{,}73 \pm 3{,}36$	$32{,}51 \pm 6{,}5$	$1{,}61 \pm 1{,}29$	$0{,}0 \pm 0{,}0$
NPGA	$4{,}73 \pm 1{,}83$	$14{,}93 \pm 4{,}58$	$0{,}72 \pm 0{,}74$	$1{,}78 \pm 4{,}37$
VEGA	$2{,}23 \pm 1{,}13$	$11{,}48 \pm 2{,}38$	$0{,}16 \pm 0{,}28$	$3{,}13 \pm 9{,}92$
HLGA	$7{,}2 \pm 2{,}56$	$9{,}78 \pm 2{,}70$	$0{,}63 \pm 0{,}66$	$11{,}77 \pm 13{,}68$
NSGA	$7{,}63 \pm 2{,}67$	$5{,}05 \pm 1{,}62$	$0{,}09 \pm 0{,}07$	$49{,}29 \pm 16{,}58$
SPEA	$84{,}8 \pm 50{,}5$	$4{,}25 \pm 1{,}72$	$0{,}01 \pm 0{,}02$	$57{,}86 \pm 17{,}98$
MOMBES	$54{,}0 \pm 17{,}38$	$0{,}76 \pm 0{,}1$	$0{,}01 \pm 0{,}01$	$97{,}24 \pm 1{,}43$

Tabelle 7.7: Performancevergleich bezüglich $\lvert PF^* \rvert$, L_p, Δ und S für ZTD-4

Betrachtet man das Dominanzverhalten der Elemente der verschiedenen Fronten, so ist erkennbar, daß alle Elemente von PF^*_{MOMBES} alle Elemente aller betrachteten Vergleichsfronten streng dominieren. Deutlich wird dies durch die großen Werte für das KDA, was bei den betrachteten Approximationen der Anzahl ihrer Elemente entspricht.

Die durch SPEA generierte Approximation PF^*_{SPEA} mit dem größten KDA-Wert innerhalb der Testgruppe enthält mehr Elemente als die entsprechende Front von PF^*_{MOMBES} ($\left|PF^*_{SPEA}\right| = 100$, $\left|PF^*_{MOMBES}\right| = 70$).

KDA(A,B)		B							
		MOGA	**NPGA**	**VEGA**	**HLGA**	**NSGA**	**SPEA**	**MOMBES**	**Mittelwert**
A	**MOGA**		0,00	0,00		0,00	0,00	0,00	0,00
	NPGA	5,00		2,00	0,50	0,00	0,00	0,00	1,25
	VEGA	4,00	0,50		0,25	0,00	0,00	0,00	0,79
	HLGA	7,00	2,00	4,00		0,00	0,22	0,00	2,20
	NSGA	9,00	9,00	9,00	9,00		9,00	0,00	7,50
	SPEA	100,00	100,00	100,00	4,50	0,00		0,00	50,75
	MOMBES	70,00	70,00	70,00	70,00	70,00	70,00		70,00

Tabelle 7.8: Performancevergleich bezüglich $KDA(A,B)$ für ZTD-4

Sowohl aus Tabelle 7.8 als auch aus Abbildung 7.6 ist erkennbar, das für dieses Testproblem alle 9 Elemente von PF^*_{NSGA} alle Elemente aus PF^*_{SPEA} streng dominieren. Dafür besitzen die Elemente der durch SPEA generierten Approximationen der Pareto-Front eine deutlich größere Anzahl gut über den Zielraum verteilter Elemente, was sich in einem mittleren Wert für das Hypervolumen von ca. 58% widerspiegelt, der über dem entsprechenden Wert bei NSGA von ca. 49% liegt.

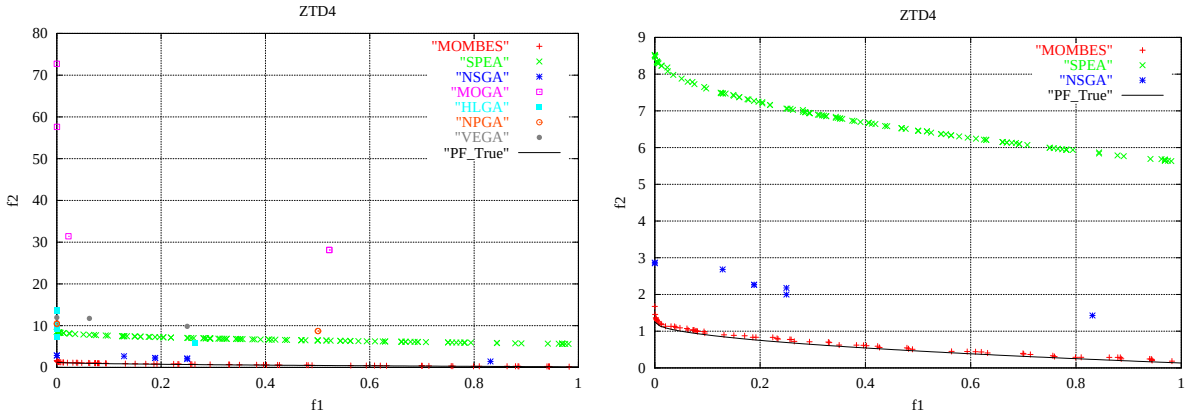


Abbildung 7.6: PF^*_{MOEA} für ZTD-4

MOMBES besitzt im Vergleich zu den betrachteten MOEA nicht nur die günstigsten Werte bezüglich der Größe des dominierten Hypervolumens, der Distanz zum Idealvektor und der Homogenität der Verteilung seiner Elemente, sondern verfügt darüber hinaus auch über die geringste Streuung dieser Werte, was auf eine stabile Wirkungsweise des Algorithmus´ schließen läßt.

SPEA konnte ebenfalls zahlenmäßig große und homogen verteilte Approximationen der Pareto-Fronten generieren, deren Elemente jedoch insgesamt eine größere Distanz zur globale Pareto-Front aufweisen als die Elemente der durch MOMBES generierten Fronten.

7.4.2 Testproblem nach Schaffer, Laumanns, Rudolph und Schwefel (SPH-3)

In diesem Abschnitt wird ein spärisches Optimierungsproblem [76] mit 3 Zielfunktionen betrachtet, das aus [Zitzler et al, 2002] entnommen und ursprünglich in [Schaffer, 1985] definiert wurde.

$$\textit{min: } \vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), f_3(\vec{x})]$$

$$f_j(\vec{x}) = \sum_{i=1; i\neq j}^{n} x_i^2 + (x_j - 1)^2;\ 1 \le j \le 3 \qquad [76]$$

$$\textit{mit } n = 100 \textit{ und } x_i \in [-10^3; +10^3];\ i = 1,..,n$$

Für die Benchmarks in diesem Abschnitt wurden folgende MOEA verwendet: NSGA2, SPEA2, PESA. Dabei wurden folgende Parametereinstellungen auf die Vergleichs-MOEA angewandt:

Größe der Population bei NSGA2 und der des Pareto-Archivs bei PESA:	$N = 100$
Größe der Elitepopulation bei SPEA	$\lvert\overline{P}_t\rvert = 20$
Größe der Population bei SPEA	$N = 80$
Wahrscheinlichkeit für 1-Punkt-crossover:	$p_c = 0{,}8$
Mutationsrate pro Bit:	$p_m = 0{,}01$
Nischenradius:	$\sigma_{share} = 0{,}4886$

Alle Entscheidungsvariablen wurden nicht binärcodiert. sondern als Fließkommazahlen dargestellt. Der Operator für Mutation und Rekombination wurde mittels des SBX-20 Operator realisiert [Zitzler et al, 2001]. Die Größe der Population und der maximal zu speichernden nicht-dominierten Lösungen wurde auf 100 festgesetzt. Für MOMBES gelten die in Abschnitt 7.4.1 beschriebenen Parametereinstellungen. Die Auswertemethode folgt analog zu den bisher beschriebenen Benchmarks. Zur besseren Beurteilung des Konvergenzverhaltens der einzelnen MOEA wurden die Werte der Gütekriterien jeweils nach 100.000, 500.000 und $1.000.000$ function calls berechnet.

In Tabelle 7.9 sind dementsprechend die Mittelwerte und Standardabweichungen der verwendeten Leistungskriterien dargestellt. Es ist erkennbar, daß für das Testproblem [76] NSGA2 und SPEA2 nach 100.000 function calls Approximationen der Pareto-Front mit jeweils nur einem Element generierten, das sich jenseits des Referenzpunktes befand, d.h. einen Wert für das Hypervolumen von $S = 0$ lieferten. Sie weisen darüber hinaus einen ca. 10^4-fach größeren durchschnittlichen Wert für L_p auf als die Elemente der durch MOMBES generierten Fronten. MOMBES und PESA konnten dagegen bereits nach 100.000 function calls Fronten mit einer vergleichsweise großen Anzahl von Elementen und relativ kleinen Werten für L_p generieren, die bereits einen deutlich größeren Anteil des Hypervolumens bezüglich $\vec{y}_{ref}$ dominierten. So liegt der durchschnittliche Wert für S bei MOMBES bereits nach 100.000 function calls bei 92% und der von SPEA bei immerhin schon 21%. Die durchschnittliche Anzahl der Elemente der generierten Fronten ist bei MOMBES mit ca. 56 um das ca. 7-fache größer als

bei PESA, jedoch besteht für beide MOEA bei diesem Parameter eine große Standardabweichung, was auf noch kein stabiles Verhalten schließen läßt.

function calls	MOEA	$\left\|PF^*\right\|$	L_p *mit* $\vec{y}^{ideal} = [0;0;0]$	S *in % bezüglich* $\vec{y}_{ref} = [100;100;100]$
100.000	**PESA**	$8{,}73 \pm 12{,}2$	$72{,}42 \pm 16{,}50$	$21{,}20 \pm 9{,}82$
	NSGA2	$1{,}1 \pm 0{,}3$	$4948{,}1 \pm 953{,}1$	$0{,}0 \pm 0{,}0$
	SPEA2	$1{,}0 \pm 0{,}0$	$5094{,}2 \pm 1131{,}2$	$0{,}0 \pm 0{,}0$
	MOMBES	$56{,}93 \pm 41{,}8$	$6{,}66 \pm 14{,}20$	$91{,}86 \pm 17{,}77$
500.000	**PESA**	$99{,}96 \pm 0{,}18$	$2{,}98 \pm 0{,}35$	$96{,}59 \pm 0{,}59$
	NSGA2	$100{,}0 \pm 0{,}0$	$3{,}48 \pm 0{,}43$	$96{,}22 \pm 0{,}75$
	SPEA2	$100{,}0 \pm 0{,}0$	$3.36 \pm 0{,}35$	$96{,}34 \pm 0{,}58$
	MOMBES	$98{,}16 \pm 8{,}03$	$1{,}40 \pm 0{,}11$	$99{,}76 \pm 0{,}18$
1.000.000	**PESA**	$100{,}0 \pm 0{,}0$	$1{,}93 \pm 0{,}11$	$98{,}40 \pm 0{,}21$
	NSGA2	$100{,}0 \pm 0{,}0$	$1{,}62 \pm 0{,}10$	$99{,}40 \pm 0{,}19$
	SPEA2	$100{,}0 \pm 0{,}0$	$1{,}71 \pm 0{,}09$	$99{,}15 \pm 0{,}18$
	MOMBES	$100{,}0 \pm 0{,}0$	$1{,}34 \pm 0{,}09$	$99{,}90 \pm 0{,}07$

Tabelle 7.9: Performancevergleich bezüglich $\left|PF^*\right|$, L_p und S für SPH-3

Trotz dieser großen Schwankungen können MOMBES und PESA nach 100.000 function calls als leistungsstärkste MOEA identifiziert werden.

Die relativ schwache Performance von SPEA2 und NSGA2 nach 100.000 function calls ist jedoch nach 500.000 function calls fast aufgehoben. Hier liegen die Werte aller MOEA bezüglich der mittleren Größe der Front, der mittleren geringsten Distanz ihrer Elemente zum Idealvektor und der mittleren Größe des dominierten Hypervolumens in der jeweils gleichen Größenordnung, wobei SPEA2, NSGA2 und PESA stabil Fronten mit jeweils 100 Elementen generieren konnten, deren mittlere Werte für L_p jedoch noch doppelt so groß wie die von MOMBES sind. Die Größe des dominierten Hypervolumens ist bei PESA, NSGA2 und SPEA2 etwa gleich groß und liegt mit ca. 96% um fast 4% unter der von MOMBES. MOMBES besitzt die günstigsten Werte für L_p und S, weist jedoch auch mit fast 10% die größte Standardabweichung bezüglich der mittleren Anzahl der nicht-dominierten Lösungen und damit $\left|PF^*\right|$ auf. Dieser vergleichsweise große Wert der Standardabweichung bei MOMBES beschränkt sich jedoch nur auf den Parameter $\left|PF^*\right|$ und betrifft nicht die Standardabweichungen für L_p und S, d.h. MOMBES konnte bei den durchgeführten 30 Testläufen nach 500.000 function calls stabil Approximationen der Pareto-Front generieren, die sich in ihrer Distanz zum Idealvektor und dem durch ihre Elemente dominierten Hypervolumen nicht erheblich unterschieden und bezüglich dieser Parameter leistungsfähiger waren als die durch NSGA2, SPEA2 und PESA generierten Fronten.

Alle MOEA konvergierten nach 1.000.000 function call auf Fronten etwa gleicher Qualität bezüglich der Parameter $\left|PF^*\right|$, L_p und S und konnten bei allen Testläufen die maximal zu erreichende Anzahl von nicht-dominierten Lösungen generieren. Die mittlere kleinste euklidische Distanz zum Idealvektor ist bei MOMBES ca. 20% geringer als bei SPEA2, NSGA2 und PESA, wobei PESA dabei den größten Wert aufweist. Die Größe des durchschnittlich dominierten Hypervolumens ist bei den durch MOMBES erzeugten Fronten mit 99,9% bezüglich des Referenzpunktes größer als die entsprechenden Werte bei SPEA2, NSGA2 und PESA. Auch bei diesem Parameter weist PESA die geringste Performance auf. Bemerkenswert ist,

daß die Werte der von MOMBES generierten Approximationen der Pareto-Front bezüglich S bereits nach 500.000 function calls größer sind als die entsprechenden Werte aller anderen betrachteten MOEA nach 1.000.000 function calls.

MOMBES konnte Approximationen der Pareto-Front generieren, die im Mittel zu jedem betrachteten Zeitpunkt der Iteration bezüglich der Parameter $\left|PF^*\right|$, L_p und S eine höhere Qualität aufwiesen als die durch alle anderen betrachteten MOEA generierten Fronten. PESA verfügt zwar über ein ähnlich schnelles Konvergenzverhalten wie MOMBES, das jedoch am Ende der Iteration zu Einbußen der Qualität der generierten Fronten führte.

Function calls	**KDA(A,B)**		**B**				
			PESA	**NSGA2**	**SPEA2**	**MOMBES**	**Mittelwert**
100.000	**A**	**MOMBES**	100,0	100,0	100,0	-	100,0
500.000	**A**	**MOMBES**	100,0	100,0	100,0	-	100,0
1.000.000	**A**	**PESA**	-	0,02	0,00	0,00	0,01
		NSGA2	49,00	-	2,55	0,06	17,20
		SPEA2	100,00	0,39	-	0,01	33,47
		MOMBES	100,00	16,33	97,00	-	71,11

Tabelle 7.10: Performancevergleich bezüglich $KDA(A,B)$ für SPH-3

Die Ergebnisse des Leistungsvergleichs bezüglich der Parameter $\left|PF^*\right|$, L_p und S konnten durch die KDA-Vergleiche zwischen den Approximationen (PF^*_{MOEA}) mit dem größten KDA-Wert jeder Testgruppe bestätigt werden (Tabelle 7.10). Es ist zu erkennen, daß sowohl nach 100.000 als auch nach 500.000 function calls alle 100 Elemente aus PF^*_{MOMBES} alle Elemente aus PF^*_{MOGA}, PF^*_{NPGA}, PF^*_{VEGA}, PF^*_{HLGA}, PF^*_{NSGA} und PF^*_{SPEA} streng dominieren. Auch nach 1.000.000 function calls ist PF^*_{MOMBES} den anderen PF^*_{MOEA} überlegen, wobei sich der Vorteil gegenüber PESA und SPEA2 noch deutlicher als gegenüber NSGA2 darstellt. PESA kann auch bei diesem Vergleich nach 1.000.000 function calls als der bezüglich KDA leistungsschwächste MOEA identifiziert werden.

Die in Abbildung 7.7 dargestellten Approximationen der 3-dimensionalen Pareto-Front und ihre in Abbildung 7.8 gezeigten 2-dimensionalen Projektionen illustrieren die oben genannte Leistungsfähigkeit von MOMBES: relativ schnell und stabil auf Approximationen der Pareto-Front hoher Qualität zu konvergieren. Es ist aus den Graphiken sehr gut zu erkennen, wie groß der qualitative Unterschied der Fronten bezüglich der Konvergenz zum Idealpunkt bereits schon nach 100.000 function calls ist, wie die anderen MOEA diesen dann im Verlaufe der Iteration aufholen, MOMBES jedoch auch nach 1.000.000 eine Approximation der Pareto-Front generierte, die sowohl bezüglich der Dominanz ihrer Elemente als auch ihrer breiten Verteilung im Zielraum als am leistungsfähigsten eingeschätzt werden kann.

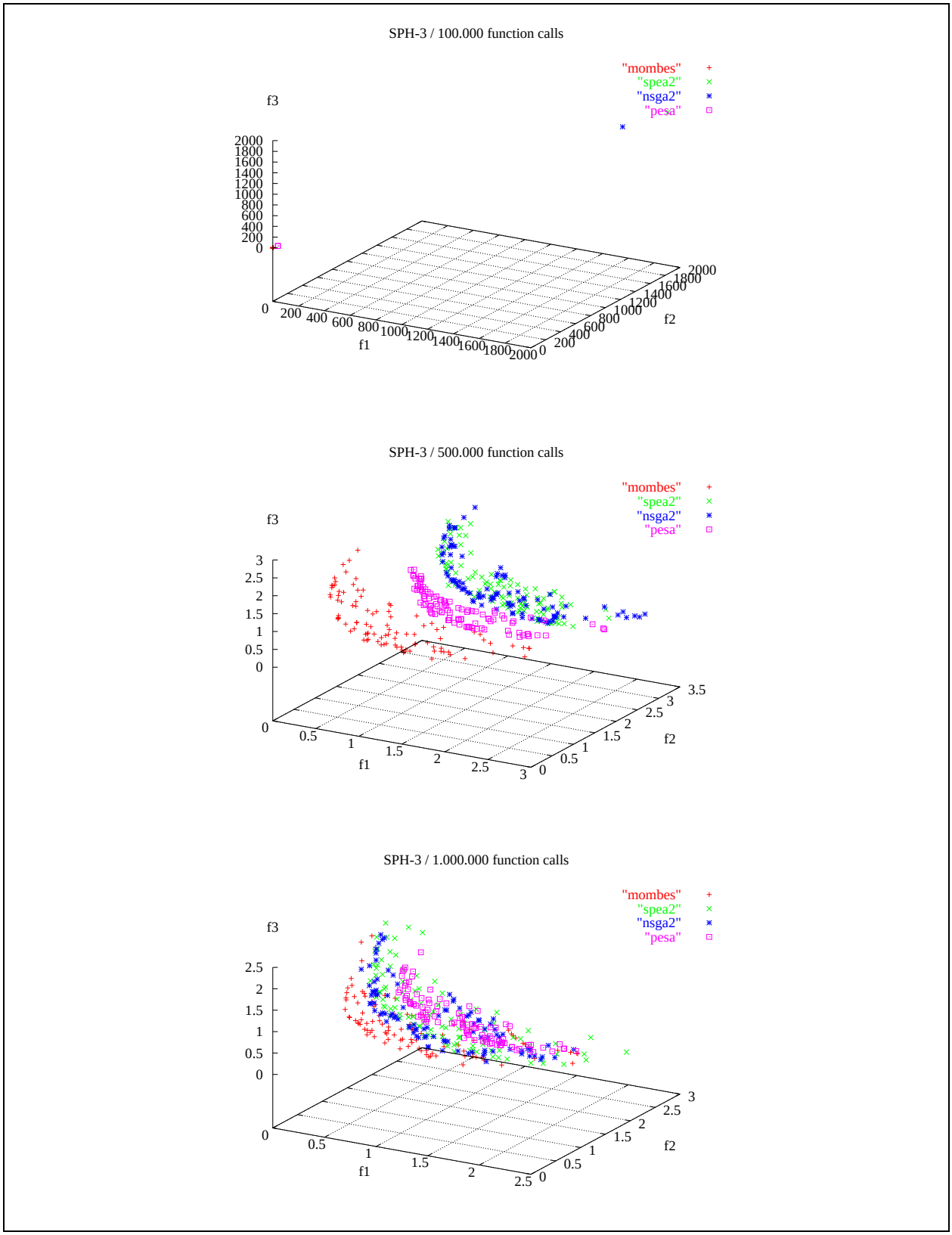


Abbildung 7.7: PF^{*}_{MOEA} für SPH-3

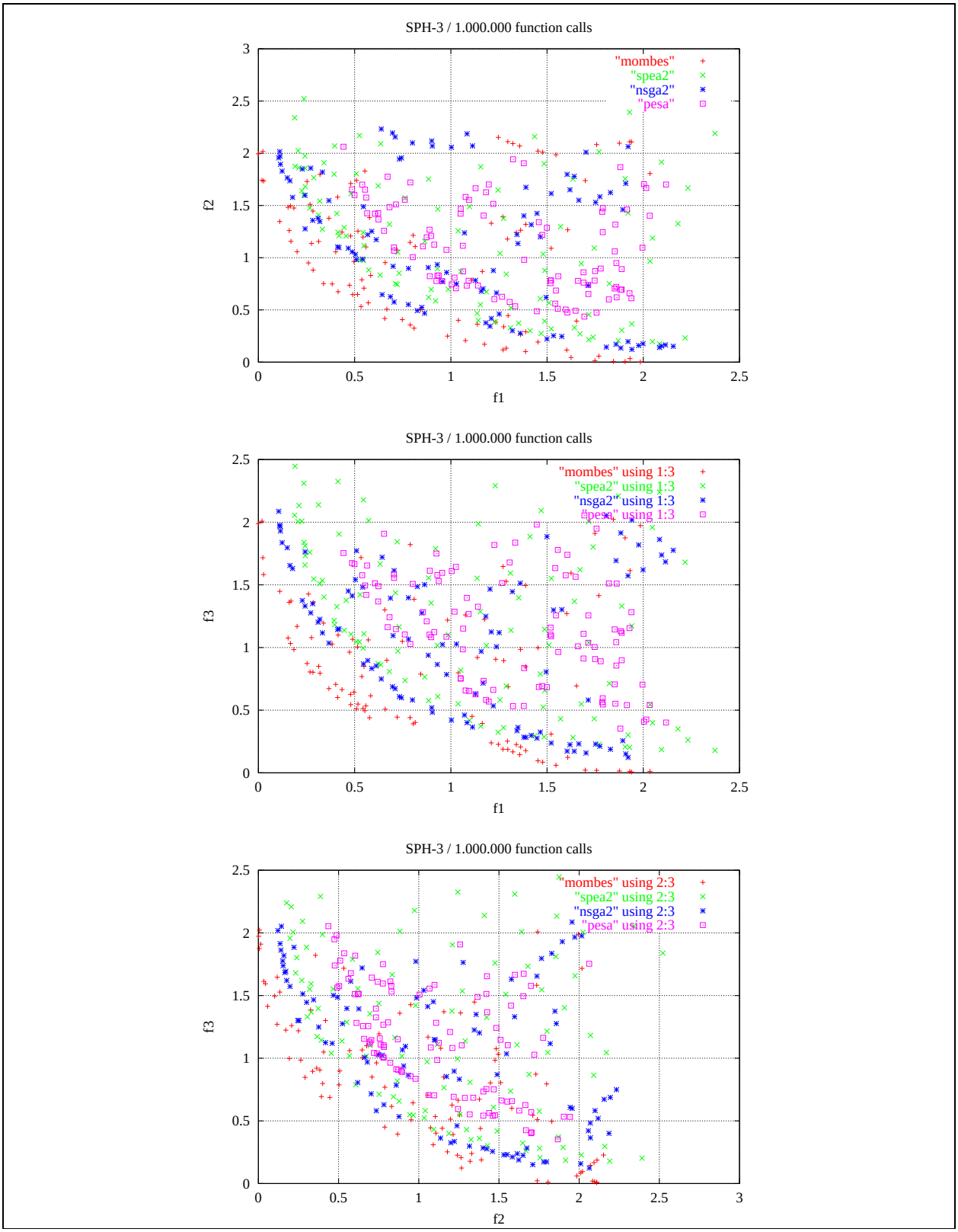


Abbildung 7.8: Projektionen der PF^{*}_{MOEA} für SPH-3

7.4.3 Testproblem nach Quagliarella und Vicini (QV-1)

Das in diesem Abschnitt betrachtete Testproblem QV-1 wurde ebenso aus [Zitzler et al, 2002] entnommen und in [Quagliarella und Vicini, 1997] erstmals definiert.

$$\begin{gathered} \textit{min: } \vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x})] \\ f_1(\vec{x}) = \left(\frac{1}{n} \cdot \sum_{i=1}^{n} \left(x_i^2 - 10 \cdot \cos(2 \cdot \pi \cdot x_i) + 10 \right) \right)^{\frac{1}{4}} \\ f_2(\vec{x}) = \left(\frac{1}{n} \cdot \sum_{i=1}^{n} \left((x_i - 1{,}5)^2 - 10 \cdot \cos(2 \cdot \pi \cdot (x_i - 1{,}5)) + 10 \right) \right)^{\frac{1}{4}} \\ \textit{mit } n = 100 \textit{ und } x_i \in [-5;+5] \; ; \; i = 1,..,n \end{gathered} \qquad [77]$$

Die beiden zu minimierenden Zielfunktionen $f_1(\vec{x})$ und $f_2(\vec{x})$ sind multimodale Rastrigin-Funktionen. Die Schwierigkeit bei der Lösung dieses Testproblems besteht darin, die extrem konkave Pareto-Front trotz der vorhandenen Multimodalität zu generieren. Die Parametrierung der verwendeten MOEA entspricht den im Abschnitt 7.4.2 diesbezüglich gemachten Angaben. Analog zu Abschnitt 7.4.2 wurde auch für dieses Testproblem die Leistungsfähigkeit der MOEA: NSGA2, SPEA2, PESA und MOMBES nach jeweils 100.000. 500.000 und 1.000.000 function calls untersucht. In Tabelle 7.11 sind die Ergebnisse der getesteten MOEA bezüglich der Mittelwerte und Standardabweichungen der Leistungskriterien $|PF^*|$, Δ, L_p und S dargestellt.

function calls	**MOEA**	$\lvert PF^* \rvert$	Δ	L_p *mit* $\vec{y}^{ideal} = [0;0]$	S *in % bezüglich* $\vec{y}_{ref} = [10;10]$
100.000	PESA	$99{,}8 \pm 0{,}40$	$0{,}0005 \pm 0{,}0003$	$2{,}54 \pm 0{,}009$	$70{,}65 \pm 0{,}35$
	NSGA2	$100{,}0 \pm 0{,}0$	$0{,}0003 \pm 0{,}0002$	$2{,}50 \pm 0{,}009$	$73{,}09 \pm 0{,}33$
	SPEA2	$100{,}0 \pm 0{,}0$	$0{,}0003 \pm 0{,}0007$	$2{,}51 \pm 0{,}009$	$72{,}87 \pm 0{,}35$
	MOMBES	$100{,}0 \pm 0{,}0$	$0{,}0013 \pm 0{,}0004$	$2{,}33 \pm 0{,}02$	$80{,}24 \pm 0{,}70$
500.000	PESA	$99{,}93 \pm 0{,}25$	$0{,}0003 \pm 0{,}0002$	$2{,}53 \pm 0{,}007$	$70{,}69 \pm 0{,}35$
	NSGA2	$100{,}0 \pm 0{,}0$	$0{,}0007 \pm 0{,}0007$	$2{,}38 \pm 0{,}01$	$78{,}61 \pm 0{,}37$
	SPEA2	$100{,}0 \pm 0{,}0$	$0{,}0003 \pm 0{,}0004$	$2{,}38 \pm 0{,}01$	$78{,}56 \pm 0{,}46$
	MOMBES	$100{,}0 \pm 0{,}0$	$0{,}0012 \pm 0{,}0004$	$2{,}33 \pm 0{,}02$	$80{,}38 \pm 0{,}87$
1.000.000	PESA	$99{,}76 \pm 0{,}43$	$0{,}0005 \pm 0{,}0003$	$2{,}53 \pm 0{,}007$	$70{,}76 \pm 0{,}38$
	NSGA2	$99{,}66 \pm 0{,}18$	$0{,}0008 \pm 0{,}0007$	$2{,}35 \pm 0{,}01$	$79{,}71 \pm 0{,}39$
	SPEA2	$100{,}0 \pm 0{,}0$	$0{,}0003 \pm 0{,}0002$	$2{,}45 \pm 0{,}01$	$80{,}09 \pm 0{,}38$
	MOMBES	$100{,}0 \pm 0{,}0$	$0{,}0011 \pm 0{,}0003$	$2{,}33 \pm 0{,}02$	$80{,}44 \pm 0{,}97$

Tabelle 7.11: Performancevergleich bezüglich $|PF^*|$, Δ, L_p und S für QV-1

Es ist zu erkennen, daß alle durch die betrachteten MOEA erzeugten Approximationen der Pareto-Front relativ schnell auf eine vergleichbare Qualität konvergieren: die Werte für L_p und S liegen bei allen MOEA bereits nach 100.000 function calls in der selben Größenordnung und unterscheiden sich nur geringfügig. Im Verlaufe der Iterationen entwickeln sich die PF^* jedoch in Hinblick auf ihre Ausbreitung und Homogenität im Zielraum. Besonders auffällig ist diese Entwicklung bei MOMBES zu beobachten: die durch MOMBES generierten Fronten besitzen durchschnittlich die größte Ausdehnung im Zielraum, weisen jedoch auch zunächst große Lücken auf, die dann im Verlaufe der Iteration geschlossen werden (siehe dazu Abbildung 7.9). Davon zeugen die vergleichsweise großen Werte für Δ bei MOMBES. Die große Ausdehnung

der Elemente der durch MOMBES generierten PF^* ist mit der Fähigkeit von MOMBES zu erklären, mittels seiner gradientenbasierten Optimierungskomponente schnell die Extrembereiche der Pareto-Front und damit die f_i^* des MOP´s zu finden.

Function calls	KDA(A,B)		B				
			PESA	NSGA2	SPEA2	MOMBES	Mittelwert
100.000	A	MOMBES	0,86	0,62	0,53	-	0,67
500.000	A	MOMBES	2,34	2,17	1,78	-	2,10
1.000.000	A	PESA	-	0,53	0,53	0,31	0,46
		NSGA2	1,86	-	0,77	0,56	1,06
		SPEA2	1,86	1,29	-	0,33	1,16
		MOMBES	3,25	1,77	3,00	-	2,67

Tabelle 7.12: Performancevergleich bezüglich $KDA(A,B)$ für QV-1

Tabelle 7.12 zeigt die Ergebnisse des KDA-Vergleiches der PF^*_{MOEA}. Hier wird deutlich, daß MOMBES nach 100.000 function calls zunächst noch NSAG2, SPEA2 und PESA bezüglich der Dominanz und Ausbreitung der Pareto-Elemente unterlegen ist. Jedoch weist MOMBES ab 500.000 function calls bereits eine bessere Performance bezüglich KDA und S auf als die Vergleichs-MOEA.

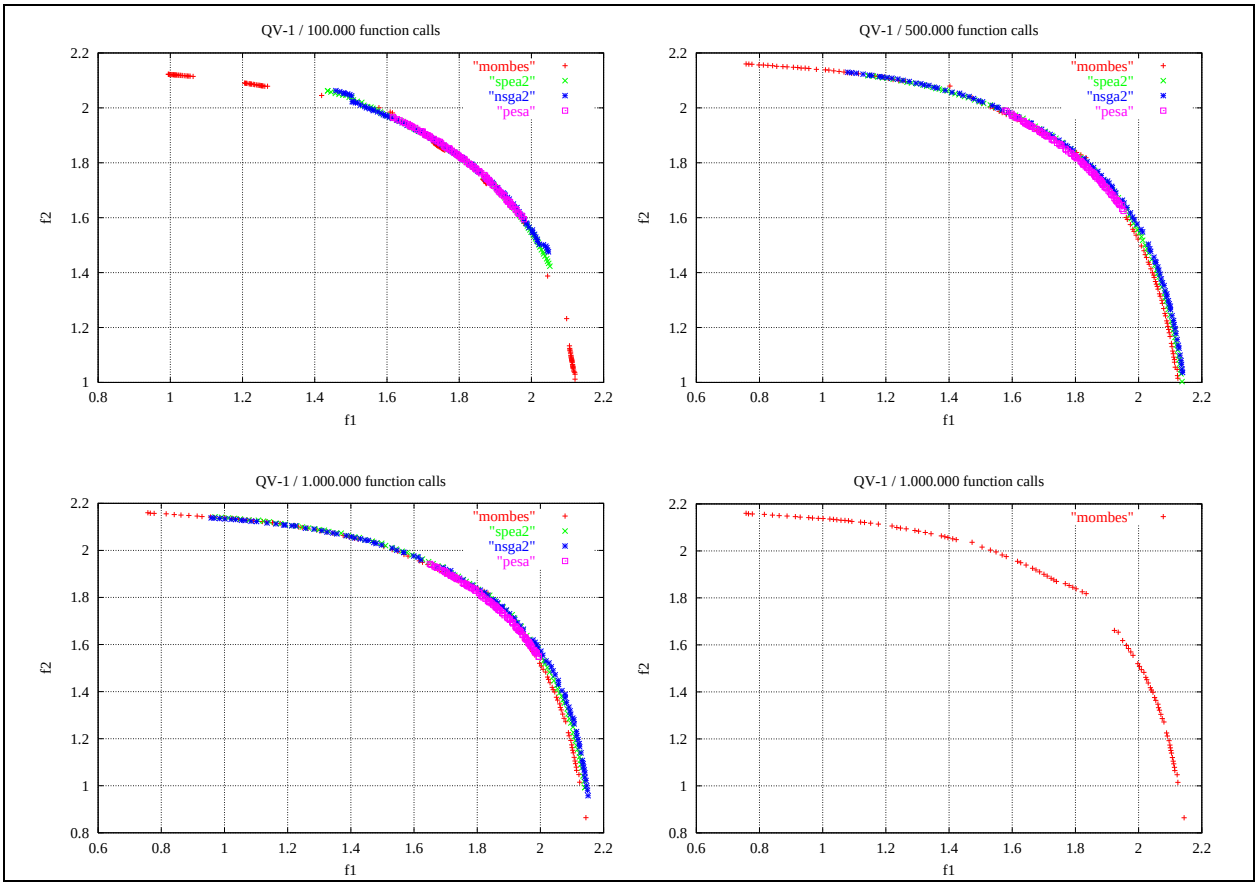


Abbildung 7.9: PF^*_{MOEA} für QV-1

7.4.4 Testproblem nach Kursawe (KUR-1)

Das Testproblem KUR-1 in der in [78] beschriebenen Form wurde aus [Zitzler et al, 2002] entnommen und stellt eine Variante des in [Kursawe, 1991] definierten MOP´s [79] dar. Da es sich um ein weit verbreitetes Testproblem handelt (es findet sich und weitere Modifizierungen u.a. in [Veldhuizen und Lamont, 1999], [Zitzler et al, 2001] und [Deb, 2002]) soll auf beide Varianten eingegangen werden.

$$\textit{min: } \vec{f}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x})]$$

$$f_1(\vec{x}) = \sum_{i=1}^{n} \left(|x_i|^{0,8} + 5 \cdot \sin^3(x_i) + 3{,}5828 \right); \quad f_2(\vec{x}) = \sum_{i=1}^{n-1} \left(1 - e^{-0,2 \cdot \sqrt{x_i^2 + x_{i+1}^2}} \right) \qquad [78]$$

$$\textit{mit } n = 100 \textit{ und } x_i \in [-10^3; +10^3]\ ;\ i = 1,..,n$$

Zunächst wird auf Basis des MOP´s [78] die Leistungsfähigkeit von MOMBES analog zu der in den Abschnitten 7.4.2 und 7.4.3 beschriebenen Vorgehensweise mit der von NSGA2, SPEA2 und PESA verglichen (Tabelle 7.13 und Tabelle 7.14). Anschließend wird eine Approximation der Pareto-Front präsentiert, die mit MOMBES nach 100.000 function calls für das nach [Deb, 2002] vereinfachte MOP [79] mit nur 3 Entscheidungsvariablen erzielt wurde. Die Parametrierung der verwendeten MOEA ist bei allen Tests identisch mit den diesbezüglich im Abschnitt 7.4.2 gemachten Angaben.

function calls	MOEA	$\lvert PF^* \rvert$	Δ	L_p *mit* $\vec{y}^{ideal} = [0;0]$	S *in % bezüglich* $\vec{y}_{ref} = [500;50]$
100.000	PESA	61,93 ± 22,98	0,21 ± 0,39	640,58 ± 351,3	0,0 ± 0,0
	NSGA2	24,93 ± 8,08	2,54 ± 3,03	1667,8 ± 486,5	0,0 ± 0,0
	SPEA2	27,23 ± 9,23	12,68 ± 49,18	2433,3 ± 603,7	0,0 ± 0,0
	MOMBES	17,36 ± 8,58	4,92 ± 9,96	361,88 ± 36,28	24,48 ± 8,41
500.000	PESA	99,96 ± 0,18	0,08 ± 0,07	443,56 ± 295,1	21,52 ± 22,36
	NSGA2	99,96 ± 0,18	0,07 ± 0,05	283,08 ± 189,1	35,55 ± 21,98
	SPEA2	95,23 ± 16,03	0,11 ± 0,15	339,87 ± 233,0	28,16 ± 23,31
	MOMBES	98,83 ± 2,78	0,24 ± 0,34	200,44 ± 51,41	51,11 ± 6,39
1.000.000	PESA	99,76 ± 0,50	0,16 ± 0,28	420,17 ± 295,34	23,92 ± 23,59
	NSGA2	100,0 ± 0,0	0,09 ± 0,05	221,48 ± 145,15	41,44 ± 25,53
	SPEA2	100,0 ± 0,0	0,21 ± 0,24	258,33 ± 225,96	38,8 ± 24,85
	MOMBES	100,0 ± 0,0	0,18 ± 0,13	90.69 ± 16.43	60,27 ± 2,33

Tabelle 7.13: Performancevergleich bezüglich $\lvert PF^* \rvert$, Δ, L_p und S für KUR-1

Aus den Daten der Tabelle 7.13 ist erkennbar, daß MOMBES bezüglich der Leistungskriterien L_p und S nach jedem der angegebenen Iterationsschritte gegenüber den Vergleichs-MOEA die günstigeren Werte aufweist. So dominieren bereits nach 500.000 function calls die Elemente der von MOMBES generierten Fronten durchschnittlich ein größeres Hypervolumen als alle Elemente der von den Vergleichs-MOEA nach 1.000.000 function calls generierten Fronten. Zu beachten ist auch die vergleichsweise geringe Streuung für die Werte des Hypervolumens bei MOMBES, die nur ca. 10% der Streuungen bei den entsprechenden Werten der Vergleichs-MOEA beträgt. Diese geringen Werte der Streuungen lassen darauf schließen, daß MOMBES der Algorithmus ist, der die Fronten am stabilsten generiert hat. Der durchschnittliche Wert für den geringsten Abstand zum Idealvektor beträgt bei MOMBES nach 100.000 function calls ca. 50% des entsprechenden Wertes von PESA, ca. 20% des von NSGA2 und sogar nur ca. 15% des von SPEA2. Während PESA im Vergleich mit NSGA2 und SPEA2 bereits nach 100.000

function calls deutlich geringere durchschnittliche Werte für L_p erreicht hat, konnte dieser Trend nach 500.000 und 1.000.000 function calls nicht fortgesetzt werden, was auf eine vorzeitige Konvergenz von PESA schließen läßt.

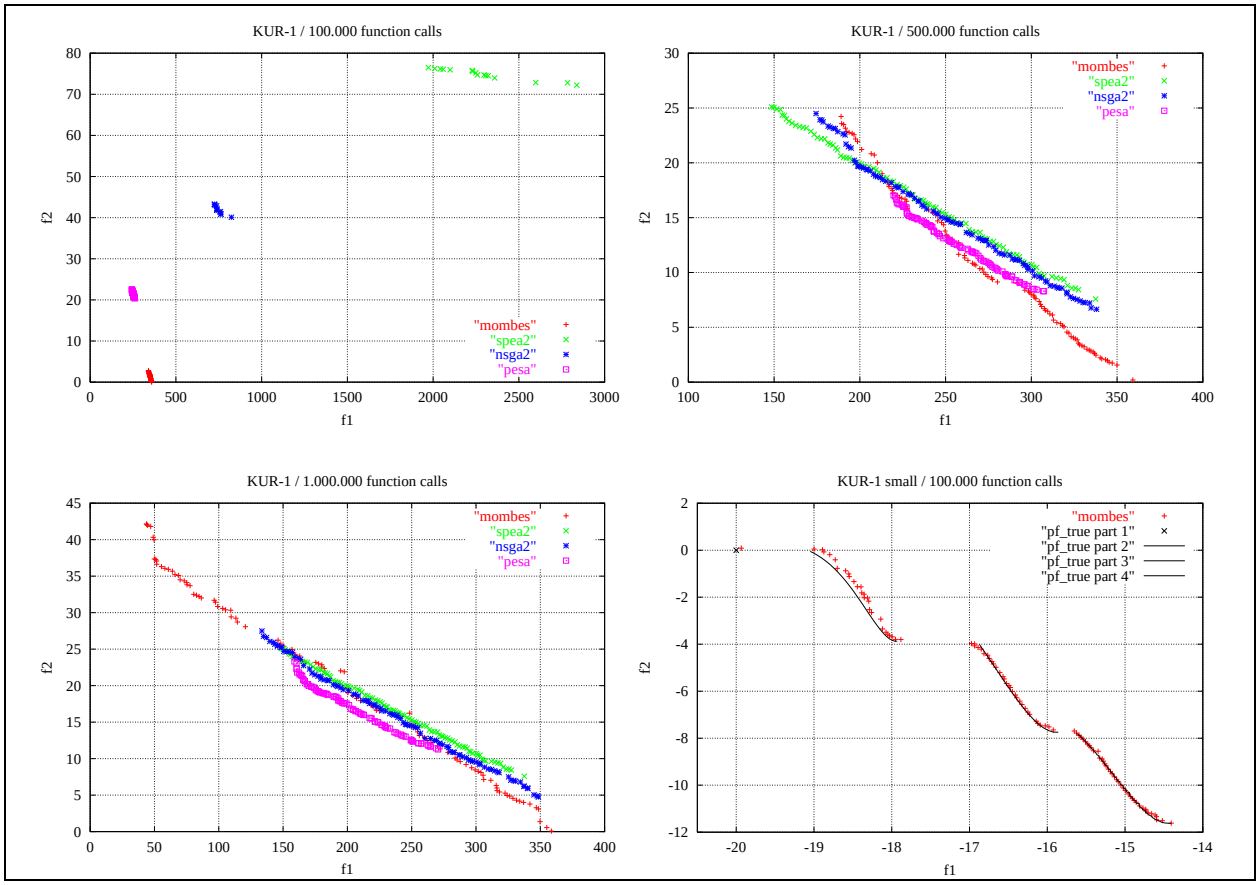


Abbildung 7.10: PF^*_{MOEA} für KUR-1 und KUR-1 small

MOMBES hingegen weist sowohl für die durchschnittlichen Werte der geringsten Distanz zum Idealvektor als auch des Hypervolumens die deutlich günstigsten Werte auf und verfügt damit über das beste Konvergenzverhalten innerhalb der betrachteten MOEA (siehe dazu auch die Approximationen der Pareto-Front in Abbildung 7.10). Zu beachten ist, daß in Abbildung 7.10 die jeweils leistungsstärkste Approximation der Pareto-Front des MOEA bezüglich des KDA innerhalb seiner Testgruppe dargestellt und in Tabelle 7.13 jedoch die Durchschnittswerte des entsprechenden Leistungskriteriums für die gesamte Testgruppe angegeben wurden. So ist zwar die beste von PESA nach 1.000.000 function calls generierte und in Abbildung 7.10 dargestellte Front sehr leistungsstark – ihre Elemente dominieren die der von SPEA2, NSGA2 und MOMBES generierten Fronten vollständig streng – jedoch ist die Leistungsfähigkeit von PESA bezüglich L_p und S über die gesamte Testgruppe geringer als die diesbezüglichen Werte von NSGA2, SPEA2 und MOMBES.

Die etwas größeren Werte bei MOMBES bezüglich des Parameters Δ sind damit zu erklären, daß die von MOMBES generierten Approximationen der Pareto-Front i.d.R. eine deutlich größere Ausdehnung im Zielraum aufweisen als die von den Konkurrenz-MOEA erzeugten. Damit wirkt sich eine prozentual gleiche Abweichung der einzelnen Δ_i von $\overline{\Delta}$ bei stark

ausgedehnten Fronten ungünstiger als die prozentual gleiche Abweichung bei weniger stark im Zielraum ausgedehnten Fronten aus. Die durchschnittlich geringere Anzahl der Elemente der durch MOMBES erzeugten Approximationen der Pareto-Front gegenüber den der Konkurrenz-MOEA nach 100.000 function calls ist mit der schnellen Konvergenz des MOMBES-Algorithmus zu erklären: wird z.B. durch den Einsatz des Gradientenverfahren von einem Individuum ein sehr großer Evolutionssprung vollzogen, d.h. verbessert es sich deutlich in einer oder mehrerer Dimensionen des Zielfunktionswertevektors, werden u.U. eine große Anzahl von Individuen von ihm dominiert und müssen daher aus der Elitepopulation entfernt werden, womit sich die Größe der aktuellen Approximation der Pareto-Front verringert. Mit zunehmender Anzahl von Iterationsschritten wird jedoch die Wahrscheinlichkeit für solche großen Evolutionssprünge geringer, was sich in der geringen Streuung für $\left|PF^*\right|$ zeigt.

Function calls	KDA(A,B)		B				
			PESA	NSGA2	SPEA2	MOMBES	Mittelwert
100.000	A	MOMBES	1,27	57,00	57,0	-	38,42
500.000	A	MOMBES	2,88	2,70	2,30	-	2,63
1.000.000	A	PESA	-	1,46	2,93	0,67	1,68
		NSGA2	0,68	-	8,00	0,46	3,04
		SPEA2	0,34	0,125	-	0,29	0,25
		MOMBES	1,48	2,15	3,41	-	2,34

Tabelle 7.14: Performancevergleich bezüglich $KDA(A,B)$ für KUR-1

Die in Tabelle 7.14 dargestellten Werte für den Performancevergleich zwischen den einzelnen MOEA bezüglich des KDA bestätigen die oben gemachten Aussagen noch einmal objektiv. MOMBES erreicht gegenüber PESA, NSGA2 und SPEA2 nach jedem untersuchten Iterationsschritt einen KDA-Wert, der größer als 1 ist, d.h. MOMBES ist im Sinne des KDA den betrachteten Vergleichs-MOEA überlegen.

Das in [Kursawe, 1991] angegebene Testproblem [79] besitzt eine relativ schwer zu generierende Pareto-Front, die aus 4 nicht miteinander verbundenen Teilstücken besteht und konkave und konvexe Elemente sowie einen Einzelpunkt im Zielraum enthält. Neben der Multimodalität der Zielfunktionen $f_1(\vec{x})$ und $f_2(\vec{x})$ besteht eine Bedingung der Pareto-Optimalität für dieses MOP darin, daß die Werte der Entscheidungsvariablen paarweise ($\forall x_i : x_i = x_{i+2}$) identisch sein müssen, was dem Verhalten einer Epistasis entspricht.

$$\textit{min: } \vec{f}(\vec{x}) = \left[f_1(\vec{x}), f_2(\vec{x})\right]$$

$$f_1(\vec{x}) = \sum_{i=1}^{n-1}\left(-10 \cdot e^{-0,2\cdot\sqrt{x_i^2+x_{i+1}^2}}\right); \quad f_2(\vec{x}) = \sum_{i=1}^{n}\left(\left|x_i\right|^{0,8} + 5\cdot\sin\left(x_i{}^3\right)\right) \qquad [79]$$

$$\textit{mit } n = 100 \textit{ und } x_i \in \left[-10^3;+10^3\right];\ i = 1,..,n$$

Abbildung 7.10 rechts unten zeigt eine mit MOMBES nach 100.000 function calls generierte Approximation PF^* der Pareto-Front für die von Veldhiuzen [Veldhuizen, 1999] angegebene Vereinfachung des in [79] definierten MOP´s : $n = 3$ mit $x_i \in [-5;+5]$ für $i = 1,2,3$. Es ist zu erkennen, daß MOMBES nach 100.000 function calls in der Lage ist, alle Teilstücke der Pareto-Front zu generieren. Die Elemente aus PF^* weisen darüber hinaus eine geringe Distanz zur theoretisch möglichen PF_{true} auf und folgen ihrem Verlauf sehr gut. Dies läßt den Schluß zu, daß MOMBES auch für ein multimodales MOP mit Epistasis der Entscheidungsvariablen in der Lage ist, nicht zusammenhängende, konkave und konvexe Teilstücke der Pareto-Front zu generieren.

7.5 Simulationen zur Charakterisierung der Güte des Decision-Making-Moduls von MOMBES

Entsprechend der Darstellungen in Abschnitt 7.3.2 wird hier die Güte des Decision-Making-Moduls von MOMBES anhand der Güte der Modelle MOD_i und der finalen Pareto-Modelle (FPM) sowie der Güte der Invertierung der MOD_i und der FPM untersucht.

7.5.1 Güte der Modelle MOD$_i$

Tabelle 7.15 zeigt die mittleren Abweichungen $\overline{\delta}^{mod\_i}$ (siehe Gleichung [66], S. 110) zwischen den durch die Modelle MOD_i prognostizierten Werten $\hat{y}$ und den Werten y_i^p der i-ten Komponente der Vektoren $\vec{y}^p \in PF^*_{MOMBES}$ entsprechend der Trainings- und Testmenge.

MOP	**Modell**		**Vorhersagefehler** $\overline{\delta}^{mod\_i}$	
	MOD$_i$	**Sigma**	**Training**	**Test**
ZDT-1	MOD_1	0,25	1,59%	1,66%
	MOD_2	0,25	0,84%	0,88%
ZDT-2	MOD_1	0,25	0,58%	0,60%
	MOD_2	0,35	1,60%	1,85%
ZDT-3	MOD_1	0,25	1,89%	1,93%
	MOD_2	0,20	2,67%	2,97%
ZDT-4	MOD_1	0,15	1,79%	2,24%
	MOD_2	0,15	1,50%	2,17%
QV-1	MOD_1	0,25	1,39%	1,78%
	MOD_2	0,15	1,58%	1,96%
KUR-1	MOD_1	0,20	1,46%	1,99%
	MOD_2	0,20	1,64%	1,71%
SPH-3	MOD_1	0,75	2,48%	2,87%
	MOD_2	0,80	2,13%	2,57%
	MOD_3	0,60	2,30%	2,34%

Tabelle 7.15: Güte der Modelle MOD$_i$

Die Resultate zeigen, daß der mittlere Vorhersagefehler für die MOD_i sowohl bezüglich der Trainings- als auch der Testfraktion für alle untersuchten Testprobleme relativ gering ist: er überschreitet für kein Modell einen Wert von 3% des Wertebereichs der entsprechenden Komponente der Vektoren $\vec{y}^p \in PF^*_{MOMBES}$ und beträgt für die Testfraktionen aller MOP´s durchschnittlich ca. 2%. Damit ist die Annahme bestätigt worden, daß es möglich ist, den funktionalen Zusammenhang zwischen den Zielfunktionswerten in dem durch PF^*_{MOMBES} beschriebenen, näherungsweise pareto-optimalen Bereich zu modellieren.

7.5.2 Güte der finalen Pareto-Modelle

Die durch MOMBES erzeugten finalen Pareto-Modelle zur Abbildung der Vektoren aus P^*_{MOMBES} auf die Vektoren von PF^*_{MOMBES} haben für alle untersuchten Testprobleme Prognoseleistungen erbracht, die unter denen der durch die MOD_i erzielten Resultate lagen. Tabelle 7.16 zeigt für

jede Komponente der Zielfunktionswertevektoren die mittleren Abweichungen $\overline{\delta}_i^{FPM}$ (siehe Gleichung [67], S.110) zwischen den durch die finalen Pareto-Modelle prognostizierten Werten $\hat{y}_i$ und den Werten y_i^p der i-ten Komponente der Vektoren $\vec{y}^p \in PF^*_{MOMBES}$ entsprechend der Trainings- und Testmenge.

MOP		Vorhersagefehler $\overline{\delta}_1^{FPM}$		Vorhersagefehler $\overline{\delta}_2^{FPM}$		Vorhersagefehler $\overline{\delta}_3^{FPM}$	
	Sigma	Training	Test	Training	Test	Training	Test
ZDT-1	3,0	3,09%	5,05%	6,45%	3,80%	-	-
ZDT-2	2,0	4,07%	4,77%	6,46%	5,37%	-	-
ZDT-3	1,5	2,57%	3,42%	3,99%	2,65%	-	-
ZDT-4	2,5	2,55%	3,13%	3,70%	2,86%	-	-
QV-1	3,5	2,86%	2,25%	4,08%	3,21%	-	-
KUR-1	1,5	4,01%	3,45%	5,87%	4,79%	-	-
SPH-3	1,5	5,97%	5,76%	5,73%	6,52%	5,77%	5,81%

Tabelle 7.16: Güte der finalen Pareto-Modelle

Es ist erkennbar, daß es mit Hilfe der finalen Pareto-Modelle gelungen ist, den durch die entsprechenden MOP definierten funktionalen Zusammenhang zwischen den Entscheidungsvariablen und Zielfunktionen abzubilden. Der mittlere Vorhersagefehler für die jeweiligen Testmengen liegt hier bei ca. 4%, was die Aussage zuläßt, daß die finalen Pareto-Modelle eine Abbildung der Entscheidungsvariablen auf die Zielfunktionswerte im näherungsweise pareto-optimalen Bereich in zufriedenstellender Güte realisieren.

Es ist zu erkennen, daß die mittlere Abweichung zwischen den durch die finalen Pareto-Modelle prognostizierten und durch die Trainingsdaten vorgegebenen Zielfunktionswerten bei ca. 4,5% liegt, einen Wert von 2,25% nicht unterschreitet und maximal ca. 6,52% beträgt. Die deutlich höheren Werte der Vorhersagefehler im Vergleich zu den Vorhersagefehlern der MOD_i sind darin begründet, daß bei den finalen Pareto-Modellen bei gleicher Anzahl von Trainingsdaten sowohl der Modelleingang als auch der Modellausgang eine höhere Dimension aufweist als bei den MOD_i. Bei den finalen Pareto-Modellen werden n-dimensionale auf k-dimensionale Vektoren abgebildet, wobei i.d.R. n deutlich größer als k ist. Hingegen erfolgt bei den MOD_i immer nur eine Abbildung von $(k-1)$-dimensionalen auf eindimensionale Vektoren, was das Modellierungsproblem vereinfacht.

Trotzdem kann festgestellt werden, daß es mit Hilfe der finalen Pareto-Modelle prinzipiell möglich ist, die Vektoren aus P^*_{MOMBES} auf die Vektoren aus PF^*_{MOMBES} abzubilden, also die in Form von Approximationen der Pareto-Menge und Pareto-Front vorliegenden Ergebnisse des Optimierungskerns von MOMBES zu generalisieren.

7.5.3 Güte der Modellinvertierung

In Tabelle 7.17 sind die mittleren Invertierungsfehler (siehe [68], S.111) sowohl der finalen Pareto-Modelle als auch der Teilmodelle MOD_i angegeben.

Es ist erkennbar, daß es bei beiden Modellgruppen durch das hier vorgeschlagene Verfahren der Modellinvertierung gelungen ist, bei Vorgabe einer Komponente des Target-Vektors einen Modellinput zu generieren, die einen Modelloutput bestimmt, der sich in geringer Abweichung zur angegebenen Komponente des Target-Vektors befindet.

MOP	Target	Variations-intervall	Finale Pareto-Modelle		Teilmodelle MOD_i	
			Invertierungs-fehler $\overline{\delta}_1^{INV\_FPM}$	Invertierungs-fehler $\overline{\delta}_2^{INV\_FPM}$	Modell	Invertierungs-fehler $\overline{\delta}^{INV\_mod\_i}$
ZDT-1	y_1^{target}	[0,000;0,999]	0,033%	-	MOD_1	0,0004%
	y_2^{target}	[0,014;1,017]	-	0,91%	MOD_2	0,67%
ZDT-2	y_1^{target}	[0,001;0,999]	0,24%	-	MOD_1	0,13%
	y_2^{target}	[0,062;1,103]	-	0,19%	MOD_2	0,02%
ZDT-3	y_1^{target}	[0,001;0,437]	4,56%	-	MOD_1	3,45%
	y_2^{target}	[0,000;1,016]	-	0,73%	MOD_2	0,34%
ZDT-4	y_1^{target}	[0,000;1,000]	0,055%	-	MOD_1	0,07%
	y_2^{target}	[0,001;1,453]	-	0,39%	MOD_2	0,18%
QV-1	y_1^{target}	[0,780;2,190]	0,41%	-	MOD_1	0,33%
	y_2^{target}	[0,865;2,198]	-	0,86%	MOD_2	0,82%
KUR-1	y_1^{target}	[43,736;358,619]	0,94%	-	MOD_1	0,76%
	y_2^{target}	[0,074;42,140]	-	1,01%	MOD_2	0,91%

Tabelle 7.17: Güte der Modellinvertierung

Es ist erkennbar, daß die mittlere Abweichung zur vorgegebene Komponente y_i^{target} des Target-Vektors i.d.R. geringer als 1% des durch die Elemente von PF^* definierten Wertebereichs beträgt, was einer sehr hohen Güte der Modellinvertierung entspricht.

Eine Ausnahme stellt hier jedoch das Testproblem ZTD-3 dar. Dieses MOP besitzt eine diskrete Pareto-Front, d.h. es existieren auch innerhalb ihrer Approximation Bereiche, für die keine Paare $[y_1, y_2]$ von Zielfunktionswertevektoren definiert und damit für die Modelle unbekannt sind. Wird nun y_i^{target} genau in diese Bereiche plaziert, gelingt es durch die Modellinvertierung nicht, sich diesen Target-Werten zu nähern, ohne daß die Modell-Konfidenz unter den erlaubten Wert von 0,9 fällt. Da dies jedoch eine harte Randbedingung bei der Modellinvertierung darstellt, bietet die Modellinvertierung nur Lösungen, die sich im durch die Elemente von P^* und PF^* beschriebenen Bereich befinden und eine relativ große Abweichung zum vorgegebenen y_i^{target} und damit einen großen Invertierungsfehler bestimmen.

7.6 Beurteilung

Die Resultate der durchgeführten Simulationen haben gezeigt, daß MOMBES in der Lage ist, qualitativ hochwertige Approximationen von konvexen (Abschnitt 7.4.1.1), konkaven (Abschnitt 7.4.1.2), diskreten (Abschnitt 7.4.1.3) und multimodalen (Abschnitt 7.4.1.4) Pareto-Fronten zu generieren sowie, MOP´s mit Epistasis der Entscheidungsvariablen zu lösen.

Die Güte der untersuchten MOEA wurde aufgrund der Qualität der von ihnen erzeugten Approximationen der Pareto-Front des jeweiligen Problems bezüglich der Kriterien: $\left|PF^*\right|$, L_p, S und Δ sowie KDA untersucht.

Es kann festgestellt werden, daß bezüglich der untersuchten Testprobleme MOMBES den MOEA der ersten Generation: VEGA, HLGA, NPGA, MOGA, NSGA und SPEA deutlich überlegen und gegenüber den MOEA der zweiten Generation: NSGA2, SPEA2 und PESA nach 100.000 function calls überlegen und nach 1.000.000 function calls gleichwertig ist. Es wurden

von MOMBES für alle Testprobleme, mit Ausnahme des MOP´s mit diskreter Pareto-Front, Approximationen der Pareto-Front erzeugt, die den durch die Vergleichs-MOEA erzeugten Fronten bezüglich der Anzahl der Elemente, ihrer Variablität und Nähe zur globalen Pareto-Front sowie Dominanz überlegen waren. Lediglich für die Lösung des MOP´s mit diskreter Pareto-Front ist MOMBES als weniger leistungsfähig als SPEA und NSGA einzuschätzen, jedoch VEGA, HLGA, NPGA, MOGA überlegen. Bei MOMBES wird gezielt versucht, Lösungen zu generieren, die Bereiche geringer Dichte auf der aktuellen Approximation der Pareto-Front verringern. Da Bereiche geringer Dichte auf der Pareto-Front hier problem-immanent sind, führt die Anwendung des Verfahren zum Einfügen approximierter Individuen hier zu einer Einbuße der Effizienz von MOMBES, d.h. die durch MOMBES generierten Resultate weisen bei gleicher Anzahl von function calls bei einem MOP mit diskreter Pareto-Front eine geringere Qualität auf, als bei einem MOP mit der gleichen, jedoch kontinuierlichen Pareto-Front.

Besonders hervorzuheben sind die guten Konvergenzeigenschaften und die Stabilität von MOMBES. Die guten Resultate von MOMBES wurden bei den durchgeführten Simulationen mit geringer Streuung bereits nach einer vergleichsweise geringen Anzahl von function calls erzeugt, während die Vergleichs-MOEA nach einer deutlich größeren Anzahl von function calls ähnlich gute Ergebnisse erzielten.

Diese Ergebnisse zeigen, daß der hier vorgeschlagene Optimierungskern von MOMBES in seiner Leistungsfähigkeit mit den modernen MOEA der zweiten Generation konkurrieren kann und ihnen z.T. überlegen ist. Dies ist vor allem auf die gezielte Beeinflussung der Evolution in Richtung homogen und breit verteilter Pareto-Front durch das Einfügen approximierter Individuen, den gezielten Einsatz des gradientenbasierten Optimierungsverfahren nach Levenberg/Marquardt sowie der Anpassung der Mutationsschrittweiten zurückzuführen.

Die durchgeführten Simulationen zur Charakterisierung der Güte des Decision-Making-Moduls von MOMBES erlauben den Schluß, daß es mit der hier vorgeschlagenen Strategie für die untersuchten MOP möglich ist, diejenigen Zusammenhänge zwischen den Vektoren aus P^* und PF^* zu modellieren, die zur Herausbildung näherungsweise pareto-optimaler Lösungen führten. Dies wurde durch die geringen Vorhersage- und Invertierungsfehler der durch MOMBES erzeugten Pareto-Modelle bestätigt. Damit ist illustriert worden, daß es möglich ist, zum einen die wichtigsten Zusammenhänge zwischen den Zielfunktionswerten innerhalb von PF^* sowie zwischen den Vektoren aus P^* und den Vektoren aus PF^* zu generalisieren und zum anderen auch ohne erneutes Durchlaufen des Optimierungskerns von MOMBES durch Invertierung des finalen Pareto-Modells bei Vorgabe eines Target-Vektors näherungsweise pareto-optimale Lösungen zu generieren.

8 Beispielanwendungen

In diesem Kapitel sollen beispielhaft zwei Anwendungen des beschriebenen Lösungsansatzes MOMBES dargestellt werden. Dabei handelt es sich zum einen um die Darstellung der Berechnung eines optimalen T-Trägers und zum anderen um die Beschreibung eines Systems zur modellgestützten Rezepturoptimierung aus der Tierfutterherstellung. Obwohl die dargestellten Optimierungsprobleme aus ganz unterschiedlichen Bereichen der ingenieurtechnischen Praxis stammen, konnten beide mit Hilfe der dargestellten Methode MOMBES gelöst werden. Dabei umfaßt die Lösung der multikriteriellen Optimierungsprobleme nicht nur die Generierung einer qualitativ hochwertigen Approximation der Pareto-Front, sondern schließt auch das Decision-Making in Form der invertierten Modellbildung und qualitativen Datenanalyse der Entscheidungsvariablen und Zielfunktionswertevektoren im näherungsweise pareto-optimalen Bereich mit ein.

8.1 Optimierung eines T-Trägers

Bei dem folgenden Anwendungsbeispiel handelt es sich um ein technisches Konstruktionsproblem zur Optimierung eines T-Stücks mit 2 Zielfunktionen und einer harten, nichtlinearen Randbedingung, das aus [Ocyczka, 1985] entnommen wurde und sich ebenso in [Coello, 1996] findet. Es wird gezeigt, daß das System MOMBES ohne spezielle Anpassungen für dieses technische Problem anwendbar ist. Da es sich um ein MOP mit einer geringen Anzahl von Zielfunktionen und Entscheidungsvariablen handelt, sind die Zusammenhänge zwischen den Entscheidungsvariablen und Zielfunktionen transparent darstellbar und kann die Wirkungsweise von MOMBES gut illustriert werden.

8.1.1 Problemstellung

Es sei ein T-Träger zu konstruieren, der über eine hohe Festigkeit und damit eine geringe statische Durchbiegung bei Krafteinwirkung verfügt *und* mit möglichst hoher Wirtschaftlichkeit, d.h. geringem Materialverbrauch und damit geringem Querschnitt, hergestellt werden kann. Diese beide Ziele widersprechen sich, da ein Träger mit großer Querschnittfläche zwar über eine hohe Festigkeit verfügt, jedoch auch hohen Materialaufwand in seiner Herstellung erfordert. Andererseits biegt sich ein günstig herzustellenden Träger mit sehr kleiner Querschnittsfläche bei Krafteinwirkung stärker durch und besitzt damit einen geringen Durchbiegungswiderstand.

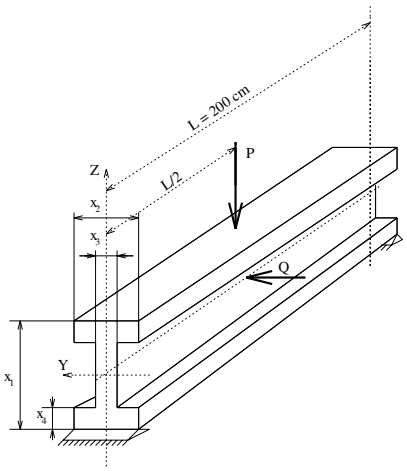


Abbildung 8.1: Konstruktionsoptimierung eines T-Stücks, aus [Coello, 1996] S.145

In Abbildung 8.1 ist die Geometrie des zu optimierenden T-Trägers dargestellt. Ziel der Optimierung ist es, solche geometrischen Maße $\vec{x} = [x_1, x_2, x_3, x_4]$ zu finden, die sowohl eine hohe Festigkeit als auch einen geringen Querschnitt des Trägers bestimmen.

Zu minimieren seien daher die zwei folgenden, miteinander konkurrierenden Zielfunktionen:

- Querschnitt des Trägers, und damit (bei einer definierten Länge L) sein Volumen sowie
- Statische Durchbiegung des Trägers unter Einwirkung der Kraft P.

Die Entscheidungsvariablen der Optimierung seien die geometrischen Abmaße des Trägers d.h. der Vektor: $\vec{x} = [x_1, x_2, x_3, x_4]$.

Es seien zusätzlich folgende Angaben bekannt, die in Form von Randbedingungen bei der Optimierung zu berücksichtigen sind:

1. Für die geometrischen Abmaße des Trägers (in cm) sollen folgende Einschränkungen gelten:

$$10 \le x_1 \le 80;\ 10 \le x_2 \le 50;\ 0{,}9 \le x_3 \le 5;\ 0{,}9 \le x_4 \le 5 \qquad [80]$$

2. Für die zulässige Biegespannung k_g des Trägers, sein Elastizitätsmodul E sowie die maximal auftretenden Biegekräfte P und Q gelten folgende Angaben:

$$k_g = 16\frac{kN}{cm^2};\quad E = 2 \cdot 10^4 \frac{kN}{cm^2};\quad P = 600kN;\quad Q = 50kN \qquad [81]$$

3. Für die angreifenden Kräfte M_Y, M_Z und die Widerstandsmomente W_Y, W_Z gelte folgende Beziehung:

$$\frac{M_Y}{W_Y} + \frac{M_Z}{W_Z} \le k_g \qquad [82]$$

Wobei für die Kräfte M_Y, M_Z folgende Werte angenommen werden: $M_Y = 30000kN/cm$ und $M_Z = 2500kN/cm$.

Durch Umformung der Terme ergeben sich für die Zielfunktionen und die Randbedingung folgende Gleichungen[82]:

Querschnitt des Balkens:

$$f_1(\vec{x}) = 2x_2x_4 + x_3(x_1 - 2x_4) \qquad [83]$$

Statische Durchbiegung:

$$f_2(\vec{x}) = \frac{60000}{x_3(x_1 - 2x_4)^3 + 2x_2x_4(4x_4^2 + 3x_1(x_1 - 2x_4))} \qquad [84]$$

sowie als einzuhaltende, harte Randbedingung:

[82] Auf die exakte Herleitung der Gleichungen soll an dieser Stelle verzichtet und auf die Arbeit von [Ocyczka, 1985] verwiesen werden.

$\vec{g}(\vec{x}) \geq 0$, wobei gilt:

$$\vec{g}(\vec{x}) = 16 - \frac{180000x_1}{x_3(x_1 - 2x_4)^3 + 2x_2x_4(4x_4^2 + 3x_1(x_1 - 2x_4))} - \frac{15000x_2}{(x_1 - 2x_4)x_3^3 + 2x_4x_2^3} \quad [85]$$

Aus der harten Randbedingung [85] ergibt sich für dieses MOP die in Abbildung 8.2 visualisierte Menge der zulässigen Zielfunktionswerte $Y_{feasible}$ als Bild von $X_{feasible}$ in den Zielraum.

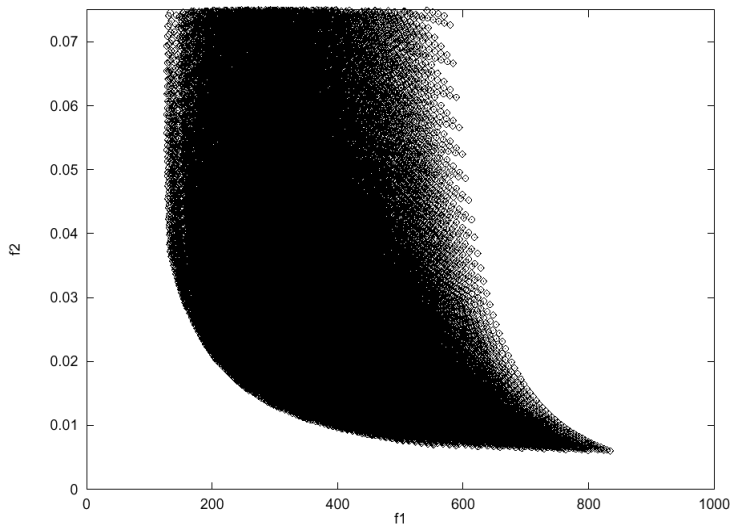


Abbildung 8.2: Menge der zulässigen Zielfunktionswerte, aus [Coello, 1996]

Der Idealvektor ergibt sich aus $X_{feasible}$ näherungsweise mit: $\vec{y}^{ideal} = \left[f_1(\vec{x}^{*1}), f_2(\vec{x}^{*2})\right] = [127{,}46; 0{,}0059]$.

8.1.2 Generierung näherungsweise pareto-optimaler Lösungen

Zunächst wurde mit dem Optimierungskern von MOMBES eine Approximation PF^* der Pareto-Front und der sie bestimmenden Lösungsmenge P^* ermittelt. Da zwischen den Zielfunktionen und Randbedingungen und den Entscheidungsvariablen eine analytische Beziehung besteht (siehe Terme [83] bis [85]) ist es nicht notwendig, ein Prozessmodell zu erstellen. Statt dessen können die oben beschriebenen Gleichungen direkt zu Berechnung der Zielfunktionswerte und der Verletzung der Randbedingung während der Optimierung verwendet werden. Dabei wurden folgende Parametereinstellungen auf den Optimierungskern von MOMBES angewandt:

Anzahl der Generationen:	$t_{max} = 250$
Anzahl der Eltern:	$\mu = 15$
Anzahl der Nachkommen:	$\lambda = 100$
Anzahl der Wettkämpfer:	$\rho = 2$
Größe der Elitepopulation:	$\left\|\overline{P}_t\right\| = 20$
Parameter, der die Wahrscheinlichkeit für das Einfügen approximierter Individuen bestimmt:	$p_a = 0{,}2$

In Abbildung 8.3 ist die von MOMBES generierte Front PF^* mit den durch den Idealvektor definierten Untergrenzen der einzelnen Zielfunktionswerte („min_f1“ und „min_f2“ aus $\vec{y}_{ideal}$) dargestellt. Zusätzlich ist der Zielfunktionswertevektor („Best over all“) eingezeichnet, der durch $\vec{x}^B$ (siehe Gleichung [5] S. 6) bestimmt wird sowie die geometrische Form des Balkens, die sich durch die einzelnen Lösungen aus P^* definiert.

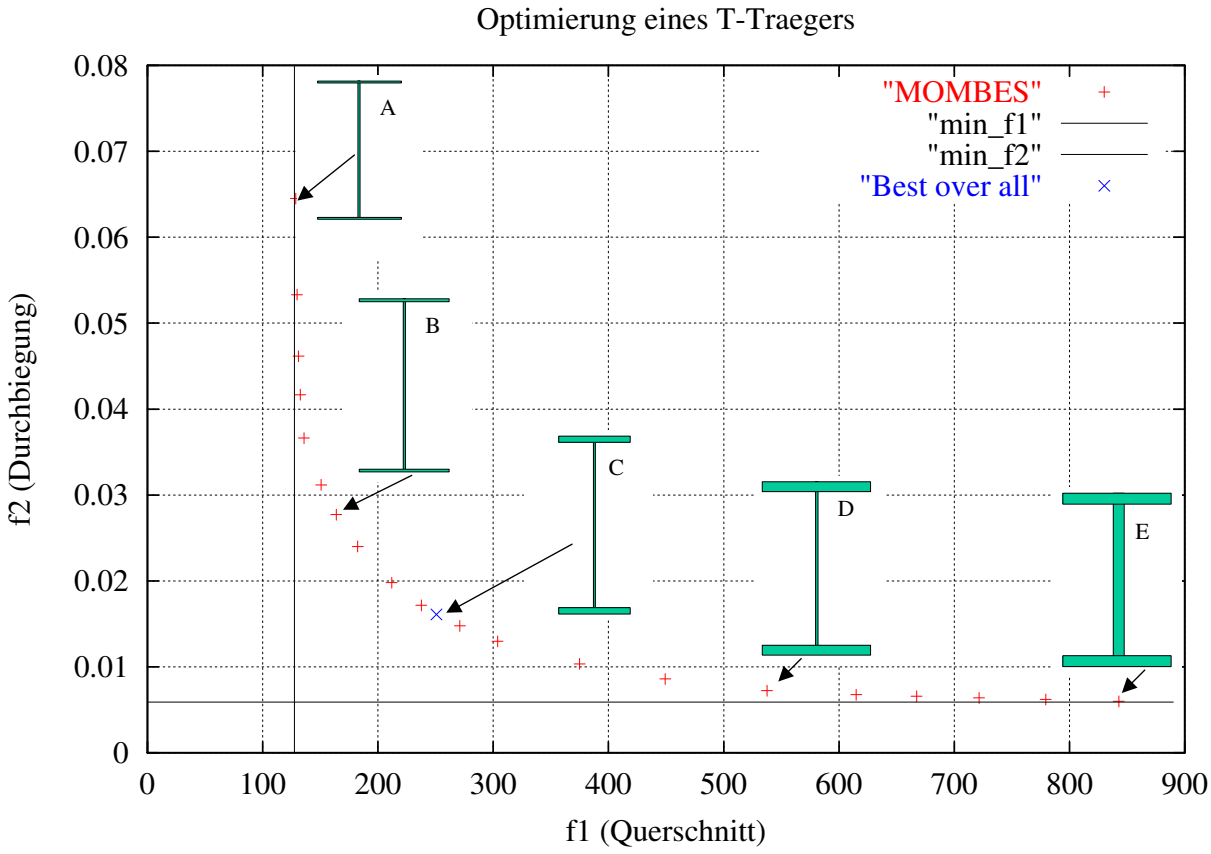


Abbildung 8.3: scatter-plot für die durch MOMBES erzeugte Pareto-Front

Es ist erkennbar, daß MOMBES eine Approximation der Pareto-Front generiert hat, deren Elemente homogen und breit über den Zielraum verteilt sind und an den Ecken gegen die Werte der Komponenten des Ideal-Vektors in der entsprechenden Dimension konvergieren. Der Balken mit der größten Querschnittsfläche (E) weist bei Einwirkung der vorausgesetzten Kräfte auch die geringste Durchbiegung auf, während der Balken mit der geringsten Querschnittsfläche (A) sich bei Einwirkung der gleichen Kräfte am stärksten verformt. Ein Kompromiß zwischen Querschnittsfläche (und damit Wirtschaftlichkeit) und Verformung bei Krafteinwirkung (und damit Widerstandsfähigkeit) kann nur durch den Anwender gefunden werden, indem dieser spezielle Präferenzen (z.B. aktuelle Materialkosten) bei seiner Entscheidung berücksichtigt.

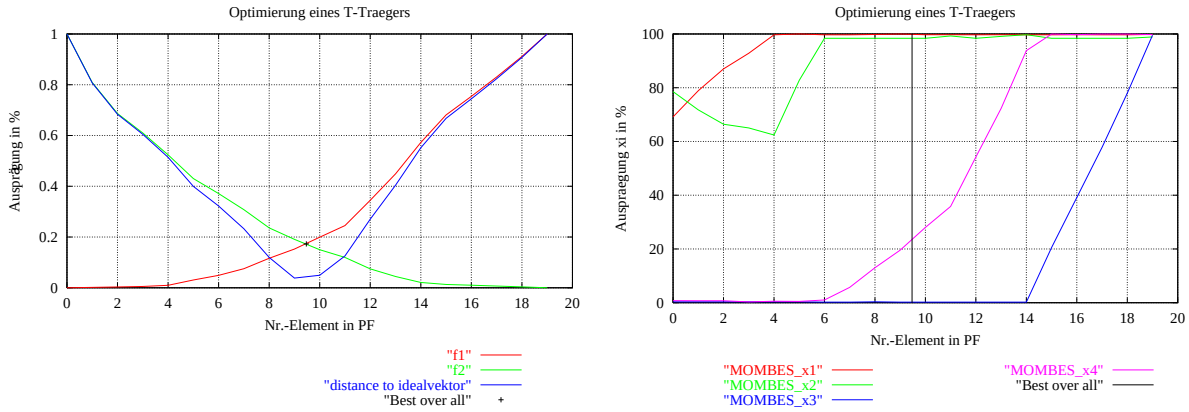


Abbildung 8.4: Normierte Zielfunktionswerte und Entscheidungsvariablen entsprechend der durch MOMBES generierten Approximation der Pareto-Front

Eine Möglichkeit des Kompromisses zwischen den Zielfunktionen liefert die best-over-all-Lösung $\vec{x}^B$. Sie wird vom Decision-Making-Modul von MOMBES in erster Näherung geschätzt, indem für alle Elemente aus PF^* der Abstand zum Idealvektor ausgewertet wird (Abbildung 8.4 links). Dabei werden die Komponenten der Zielfunktionswertevektoren der Approximation PF^* der Pareto-Front auf den durch die Elemente von PF^* definierten Bereich normiert. In Abbildung 8.4 rechts sind die Werte der einzelnen Entscheidungsvariablen für die gesamte Lösungsmenge P^* dargestellt (die Numerierung der Elemente erfolgt aufsteigend beginnend mit dem geringsten Wert für f_1). Die durch MOMBES für den Anwender bereitgestellten Diagramme zur Darstellung des Verlaufs der normierten Werte (entsprechend der durch [80] definierten Ober- und Untergrenzen) der Entscheidungsvariablen aus P^* gestatten eine erste Interpretation des Einflusses der einzelnen Entscheidungsvariablen auf die Werte der Zielfunktionen (siehe auch Abschnitt 8.1.4). Diese Abhängigkeiten werden in der Phase des Decision-Makings durch MOMBES modelliert und gestatten einen interpolierenden Zugriff auf PF^* (siehe Abschnitt 8.1.3).

8.1.3 Decision Making

Nach der Generierung der Approximation PF^* der Pareto-Front, wurde (wie in Abschnitt 6.5.2 dargestellt) ein finales Pareto-Modell in Form eines dreilagigen RBF-Netzes mit 4 Input- und 2 Output-Neuronen erstellt und mit den Vektoren aus P^* und aus PF^* belehrt. Alle Komponenten wurden mit denen durch P^* und PF^* definierten Ober- und Untergrenzen der entsprechenden Dimension normiert. Die Anzahl der Hidden-Neurone wurde der Anzahl der Elemente von PF^* gleichgesetzt und die Positionsvektoren auf die Elemente der (normierten) Menge PF^* positioniert. Für die radialen Basisfunktionen wurde hier entsprechend Gleichung [23] (S.17) ein Wert von $\sigma^{RBF} = 0{,}15$ gewählt.

Nach dem Belehren des Modells kann der Anwender in einer Eingabemaske (siehe Abbildung 8.5) den Modellinput variieren und so die Wirkungen auf den Modelloutput beobachten. Darüber hinaus können einzelne Komponenten des Target-Vektors sowie Randbedingungen definiert werden und MOMBES ermittelt durch Invertierung des finalen Pareto-Modells (siehe Abschnitt 6.5.2) die Target-Lösung, die bei Propagierung durch das Modell den Target-Vektor möglichst genau bestimmt. Folgende Randbedingungen können dabei bei der Modellinvertierung berücksichtigt werden:

- Wertebereiche in Form von parametrierbaren Ober- und Untergrenzen für jede Entscheidungsvariable und jeden Zielfunktionswert,
- Fixierung auf einzelne, parametrierbare Werte für jede Entscheidungsvariable und
- Modellsicherheit in Form einer nicht zu unterschreitenden Konfidenz.

Zusätzlich besteht die Möglichkeit, eine Näherung der best-over-all-Lösung durch MOMBES zu generieren. Dies geschieht, indem diejenige Lösung ermittelt wird, die bei Propagierung durch das FPM einen Zielfunktionswertevektor bestimmt, dessen euklidische Distanz zum Idealvektor minimal ist. Die Komponenten des Idelvektors hingegen werden geschätzt, indem sie den Minimalwerten in der entsprechenden Dimension der Vektoren aus PF^* zugewiesen werden.

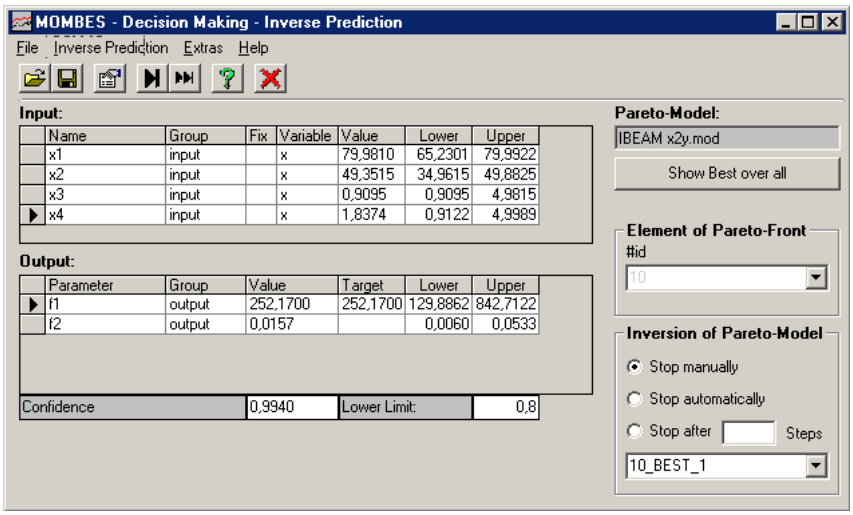


Abbildung 8.5: Eingabemaske zur Definition des Target-Vektors

In Abbildung 8.6 und Tabelle 8.1 sind die Ergebnisse dargestellt, die bei Anwendung des Decision-Making-Moduls von MOMBES erzielt wurden. Bei den durchgeführten Tests wurde jeweils eine Komponente des Target-Vektors y_1^{target} vorgegeben und dann durch das System die entsprechende Target-Lösung ($\vec{x}^{target}$ in Tabelle 8.1) und der durch sie bestimmte Modelloutput ($\hat{\vec{y}}$ in Tabelle 8.1 und „net“ in Abbildung 8.6) ermittelt. Da für dieses MOP die analytische Beziehung zwischen den Entscheidungsvariablen und Zielfunktionen vorliegt, wurde zusätzlich der Zielfunktionswertevektor $\vec{y}^{calc} = \vec{f}(\vec{x}^{target})$ bestimmt. In Tabelle 8.1 ist neben der Modellsicherheit (Konfidenz) jeweils für alle untersuchten Target-Vektoren auf PF^* die prozentuale Abweichung zwischen dem normierten Modelloutput und dem durch $\vec{x}^{target}$ bestimmten und normierten Zielfunktionswertevektor angegeben. Die Normierung erfolgte hierbei auf den Bereich im Zielraum, der durch die Elemente der durch MOMBES generierten Front PF^* definiert wurde.

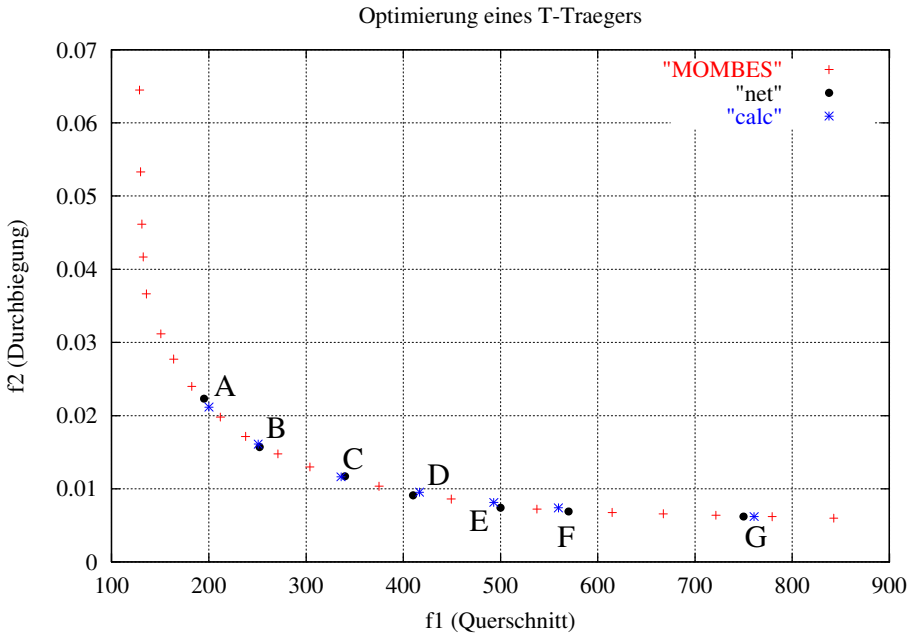


Abbildung 8.6: Zugriff auf das finale Pareto-Modell

	A	B	C	D	E	F	G
y_1^{target}	195,000	252,170	340,000	410,000	500,000	570,000	750,000
Confidence	90,10%	98,00%	92,10%	94,30%	95,10%	98,00%	94,10%
$\hat{y}_1$	195,000	252,170	340,000	410,000	500,000	570,034	750,000
$\hat{y}_2$	0,022	0,016	0,012	0,009	0,007	0,007	0,006
x_1^{target}	79,900	79,981	79,792	78,761	79,407	79,072	79,792
x_2^{target}	37,400	49,352	49,352	49,352	49,378	49,421	49,873
x_3^{target}	0,901	0,910	0,910	0,910	1,184	1,458	3,755
x_4^{target}	1,763	1,837	2,720	3,562	4,140	4,631	4,999
y_1^{calc}	200,621	250,757	336,057	416,715	493,019	559,538	760,723
y_2^{calc}	0,021	0,016	0,012	0,010	0,008	0,007	0,006
$\left\|\vec{y}^{calc} - \vec{\hat{y}}\right\|$	1,88%	0,71%	0,57%	1,18%	1,58%	1,69%	1,50%

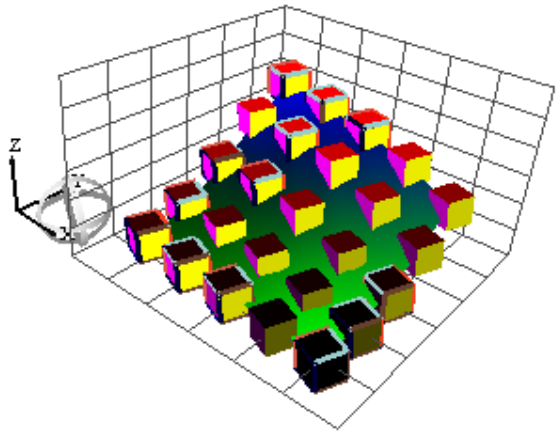
Tabelle 8.1: Target-Lösung und Güte der Approximation entsprechend Abbildung 8.6

Die Ergebnisse zeigen, daß MOMBES in der Lage ist, Target-Lösungen zu generieren, die bei Propagierung durch das finale Pareto-Modell einen Modelloutput erzeugen, der eine geringe Distanz zum vorgegebenen Target-Vektor aufweist. Werden die so gewonnenen Target-Lösungen zur Berechnung der Zielfunktionen auf Basis der Gleichungen [83] und [84] verwendet, ist die Abweichung zwischen Modelloutput und analytisch berechneten Zielfunktionswertevektor gering: sie beträgt nicht mehr als 2%. Dies illustriert die Fähigkeit von

MOMBES die Zusammenhänge zwischen den Entscheidungsvariablen und Zielfunktionen zu modellieren und einen sinnvollen Zugriff auf die Approximation der Pareto-Front zu ermöglichen, der nicht ausschließlich auf die Elemente aus PF^* beschränkt ist. Der Punkt „B“ entspricht der durch das System vorgeschlagenen best-over-all-Lösung.

8.1.4 Qualitative Analyse der Approximation der Pareto-Menge und Pareto-Front

Abschließend sollen die qualitative Zusammenhänge zwischen den Entscheidungsvariablen und Zielfunktionswerten im näherungsweise pareto-optimalen Bereich dargestellt werden. Dazu wird nach der in Abschnitt 6.5.1 dargestellten Methode eine qualitative Clusteranalyse mit Hilfe einer selbstorganisierenden Kohonenkarte durchgeführt.



f_1: Höhe der Oberfläche

f_2: Oberfläche grün

x_1: gelbe Würfelfläche

x_2: lila Würfelfläche

x_3: Oberfläche blau

x_4: rote Würfelfläche

Markierung aller Winner-Neurone bezüglich der Trainingsvektoren

Abbildung 8.7: Mit der Pareto-Menge und der Pareto-Front trainierte Kohonenkarte

Abbildung 8.7 stellt die Visualisierung der verwendeten Kohonenkarte, bestehend aus $5 \times 5 = 25$ Neuronen, die über eine gaussförmige Erregungsfunktion in einer Nachbarschaft von 5 Neuronen verfügen und mit einer von Lernrate $\alpha_{START} = 0{,}05$ in 500 Schritten mit den Vektoren P^* und PF^* belehrt wurde. Folgende Information kann auf Basis der Interpretation der Kohonenkarte gewonnen werden (Vergleiche dazu auch Abbildung 8.4):

- Die Höhe der Kartenoberfläche ist dort stark ausgeprägt, wo keine grüne Färbung der Kartenoberfläche auftritt und umgekehrt. Daraus folgt: *große Werte der Zielfunktion* f_1 *korrelieren mit kleinen Werte der Zielfunktion* f_2 *und umgekehrt*.
- Es existieren Bereiche auf der Karte, in denen die Höhe der Kartenoberfläche stark ausgeprägt und sie nicht grün gefärbt ist. In diesen Bereichen ist die Oberfläche blau gefärbt und die Würfel besitzen eine rote Seitenfläche, d.h. *große Werte für* x_3 *und* x_4 *korrelieren mit großen Werten für* f_1 *und kleinen Werten für* f_2. Der Anteil der Oberfläche die blau gefärbt ist beträgt nur ca. 5%, der Anteil der Würfel, die eine rote Seitenfläche besitzen ist mit ca. 20% etwas größer – jedoch auch gering, d.h. *nur ein geringer Anteil der Elemente aus* PF^* *wird durch große Werte von* x_3 *und* x_4 *bestimmt*.
- Sehr viele Würfel (ca. 75%) der Karte besitzen eine gelbe Seitenfläche, d.h. *die Mehrzahl der Elemente aus* PF^* *wird durch große Werte von* x_1 *bestimmt*. Es existieren Bereiche auf der Karte in denen die Würfel stark gelb gefärbt sind und die Höhe der Oberfläche stark ausgeprägt ist sowie Bereiche, bei denen die Würfel nicht gelb gefärbt sind und die grüne

Färbung der Oberfläche stark ausgeprägt ist. D.h. *große Werte von x_1 korrelieren mit großen Werten für f_1 und kleinen Werten für f_2.*

- Die Mehrheit der Würfel (ca. 60%) besitzen eine lila gefärbte Seitenfläche d.h. *ein großer Anteil der Elemente aus PF^* wird durch große Werte von x_2 bestimmt*. Die grüne Färbung der Kartenoberfläche ist dort stark ausgeprägt, wo die lila Färbung der Würfelseite schwach und die Höhe der Oberfläche stark ausgeprägt ist: *große Werte von x_2 korrelieren mit großen Werten für f_1 und kleinen Werten für f_2.*

Natürlich kann diese Kurzanalyse nicht umfassend sein und lediglich die groben qualitativen Zusammenhänge zwischen den Größen innerhalb der generierten Approximation der Pareto-Menge und der durch sie bestimmten Approximation der Pareto-Front widerspiegeln. Sie trägt jedoch dazu bei, das Wissen des Anwenders über die Beziehungen der Zielfunktionen und Entscheidungsvariablen zu erweitern.

Die hier dargestellte Beispielanwendung zur Lösung eines MOP´s illustriert die Arbeitsweise von MOMBES, von der Generierung näherungsweise pareto-optimaler Lösungen bis zum interaktiven Decision-Making. Mittels der hier vorgeschlagenen Methode war es möglich, in Form von PF^* den nicht-dominierten Bereich des Zielraums zu bestimmen und zu modellieren. Durch die Modellierung mittels SOM wurde dem Anwender die Möglichkeit gegeben, qualitative Zusammenhänge zwischen den Entscheidungsvariablen und Zielfunktionen im näherungsweise pareto-optimalen Bereich des Lösungs- und Zielraums zu extrahieren. Darüber hinaus wurde durch die Modellierung mittels eines RBF-Netzes eine Abbildung der Elemente aus P^* auf die Elemente von PF^* realisiert, die dem Anwender einen kontinuierlichen Zugriff auf die näherungsweise pareto-optimale Lösungsmenge gestattet.

8.2 Datengetriebene Rezepturoptimierung in der Tierfutterindustrie

In diesem Abschnitt wird die Anwendung von MOMBES in der in der verfahrenstechnischen Industrie am Beispiel der Rezepturoptimierung in der Tierfutterherstellung beschrieben. Das dieser Beschreibung zugrunde liegende Pilotprojekt wurde durch die Firma Masterfoods GmbH/Verden[83] beauftragt und durch die Firma Kratzer Automation AG/München[84] in Zusammenarbeit mit der Firma Masterfoods durchgeführt. Es führte zur Entwicklung des „Neural Forecast System“ (NFCS), einer Anpassung von MOMBES auf die speziellen Anforderungen der Prozessindustrie. Aus Gründen der Datensicherheit kann an dieser Stelle nicht im Detail auf die internen Prozessparameter und Rezepturbestandteile des zu modellierenden und optimierenden Prozesses eingegangen werden. Um jedoch dennoch einen Überblick über die Probleme und die hier vorgeschlagene Lösungsstrategie zu geben, wird das System anhand einer synthetischen Rezepturdatenbank erläutert.

Wie in Abschnitt 6.3 dargestellt, spielen Simulationsmodelle in der Automatisierung verfahrenstechnischer und chemischer Prozesse eine immer größere Rolle. Ihre rechentechnische Umsetzung gestattet eine flexiblere Durchführung von Modellversuchen, als dies am realen Prozess möglich ist. So können Erkenntnisse über das Prozessverhalten in unbekannten Betriebspunkten gewonnen und verschiedenste Prozesssituationen simuliert werden, deren Durchführung in der Realität unter Umständen zu gefährlich, teuer oder zu komplex ist. Andererseits werden durch den Einsatz eines geeigneten Prozessmodells und entsprechender Optimierungsverfahren die Entwicklungszeiten für das Finden optimaler Prozessparameter drastisch reduziert.

8.2.1 Problemstellung

Die Produktionsanlage der Firma Masterfoods ist eine moderne, hochautomatisierte Fabrik in der täglich ca. 800t Rohmaterialien verarbeitet werden. Die dabei zugrunde liegenden Rezepturen, d.h. die Beschreibung der Mischungsverhältnisse der einzelnen Bestandteile des Produkts und verschiedener Prozessparameter, sind sehr komplex. Sie werden durch ca. 150 Eingangsgrößen (Rohmaterialien, ihre chemischen Analysewerte sowie verschiedene Prozessparameter) und ca. 25 Ausgangsgrößen (Qualitätskriterien) des Produkts charakterisiert. Die Rohmaterialien durchlaufen in der Anlage verschiedene verfahrenstechnische Phasen so daß die „Just-in-Time“ angelieferten Zutaten über die Zerkleinerung, das Emulsieren, Mischen, Kochen, Zerschneiden, Abkühlen, in Büchsen Abfüllen und Sterilisieren, schließlich als Produkt nach wenigen Stunden in Paletten abgepackt ihren Weg in die Verkaufslager antreten. Durch die hohen Qualitätsstandards an das Produkt sowie die Anforderungen am Markt, kostengünstig zu produzieren, ist es notwendig, die Produkte ständig zu verbessern und neue Entwicklungen voranzutreiben. Diese Entwicklung neuer oder verfeinerter Produkte wird nicht an der Produktionsanlage selbst, sondern in einer sogenannten Pilotanlage, die diese modelliert, durchgeführt.

---

[83] Die Firma Masterfoods (http://www.masterfoods.de/) ist ein Unternehmen der internationalen Mars Incorporated, die mit ca. 30.000 Mitarbeiter in 60 Ländern eine der weltweit größten Hersteller von Süßwaren und Tierfutter ist. Mit ca. 70% Marktanteil und einem Jahresumsatz von ca. 1 Mrd. € pro Jahr ist Masterfoods der wichtigste Hersteller von Heimtiernahrung in Deutschland. In Verden findet neben der eigentlichen Produktion der Tiernahrung auch die Entwicklung neuer Produkte statt, in deren Rahmen das hier beschriebene System entwickelt wurde.

[84] Die Firma Kratzer Automation AG (http://www.kratzer-ag.com/) ist mit ca. 150 Mitarbeitern ein mittelständisches Software-Unternehmen für industrielle Anwendungen aus den Bereichen Automatisierung und Integrierter Fertigung. Im F&E-Bereich des Unternehmens wird u.a. an der Entwicklung intelligenter, lernfähiger Systeme zur Modellbildung und Optimierung gearbeitet.

Mit dem NFCS sollte ein System erarbeitet werden, mit dessen Hilfe:

- die Abhängigkeiten zwischen den Rezepturkomponenten und Qualitätsgrößen modelliert und Korrelationen zwischen ihnen extrahiert,
- unter Berücksichtigung der täglich aktualisierten Preise der Rohmaterialien die kostengünstigsten Rezepturen unter Einhaltung bestimmter Qualitätsgrößen vorgeschlagen,
- die Charakteristiken neuer Produkte bei veränderten Rezepturkomponenten und Prozessparametern vorhergesagt,
- neue Arbeitspunkte für das Aufstellen einer umfassenden Rezepturdatenbank vorgeschlagen

werden können. Darüber hinaus sollte es möglich sein, die gewonnen Erfahrungen in der Rezepturentwicklung in Form von neuen Rezepturen in das System einfließen zu lassen und es so ständig zu aktualisieren. Das Wissen über den zu modellierenden und optimierenden Prozess stand dabei zunächst ausschließlich in Form von ca. 500 Beispielrezepturen, d.h. gemessenen Prozessgrößen konkreter Mischungen zur Verfügung, welche an der Pilotanlage gemessen wurden (siehe Abbildung 8.8).

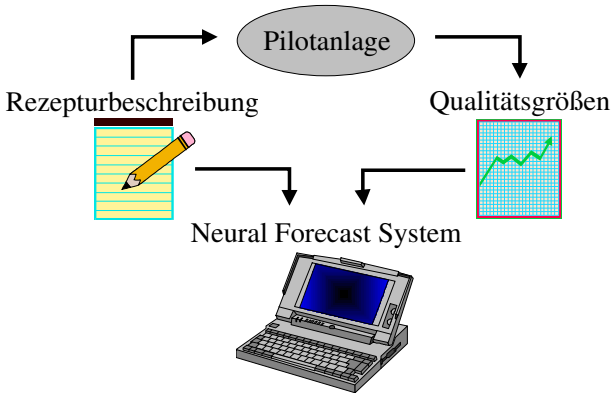


Abbildung 8.8: Datenmaterial für das Neural Forecast System

Der erhoffte Nutzen einer solchen Modellierungs- und Optimierungssoftware bestand für das Unternehmen in der Kostenreduzierung durch

- das Senken der Anzahl der Simulationen in der Fabrik,
- das Auffinden kostenminimaler Rezepturen unter gleichzeitiger Maximierung der durch verschiedene Charakteristiken beschriebenen Produktqualität sowie
- dem Verkürzen der Entwicklungszeit für neue oder variierte Rezepturen mit gewünschten Produkteigenschaften

Das NFCS folgt in seinem Aufbau den in Kapitel 6 beschriebenen System MOMBES. Es besteht aus folgenden aufeinander aufgebauten Stufen zur datengetriebenen Prozessmodellbildung und –Optimierung:

- Prozessmodellbildung mit einem RBF-Netz (siehe Abschnitt 6.3),
- multikriterielle Optimierung auf Basis des Prozessmodells (siehe Abschnitt 6.4) sowie einem
- interaktiven Decision-Making-Modul (siehe Abschnitt 6.5) auf Basis der Ergebnisse der multikriteriellen Optimierung in den Schritten qualitative Datenanalyse (siehe Abschnitt 6.5.1) und Lösungszugriff (siehe Abschnitt 6.5.2)

8.2.2 Aufbau der synthetischen Rezepturdatenbank

Die Rezepturbeschreibungen liegen in Form einer Daten-Tabelle im Rezeptur-Management-System vor und werden ebenso wie die tages-aktuelle Liste der Rohmaterialienpreise durch das System gelesen. Dabei stellt jede Zeile der Rezeptur-Tabelle eine Rezeptur und jede Spalte eine Prozessgröße dar. Zu beachten ist, daß die Angaben der Rohmaterialien Prozentwerte sind, deren Summe über alle Rohmaterialien 100% ergeben muß. An dieser Stelle sei die Zusammensetzung der im Folgenden verwendeten synthetischen Rezepturen[85] kurz beschrieben (siehe Tabelle 8.2). Es handelt sich dabei um 500 Rezepturen mit jeweils 8 verschiedenen Rohmaterialien und 5 Qualitätsgrößen zur Beschreibung des Endprodukts. Auf die Berücksichtigung von (verfahrenstechnischen) Prozessparametern wie z.B. der Sterilisationstemperatur oder Abkühldauer wurde verzichtet, da sie i.d.R. im praktischen Anlagebetrieb für alle Rezepturen konstant sind. Dagegen unterliegen die Rohmaterialien erheblichen Schwankungen bezüglich ihrer Qualität sowie hinsichtlich ihres Preises, was im vorliegenden Beispiel durch die zwei verschiedenen Preise demonstriert wird.

	Name	**≙**	**Beschreibung**	**Preis-1 in € / Kg**	**Preis-2 in € / Kg**	**Optimierungsziel/ Randbedingung**
Prozess-eingangs-größen	Zucker	x_1	Karamel-Zucker	1.03	1.03	$\sum_{i=1}^{8} x_i = 100\% \hat{=} 1\mathrm{Kg}$
	S-Oel	x_2	Soja-Öl	2.94	2.94	
	Wasser	x_3	Wasser	0	0	
	In_Schwein	x_4	Diverse Innereien vom Schwein	0.83	0.83	
	W-Mehl	x_5	Weizenmehl	0.404	0.23	
	M-Mehl	x_6	Maismehl	0.38	0.74	
	Emulgatoren	x_7	Emulgatoren	2.01	2.01	
	Vitamine	x_8	Vitaminmix	14.26	14.26	
Prozess-ausgangs-größen	Preis	f_1	Preis aller verwendeten Rohmaterialien			Minimieren
	Festigkeit	f_2	Festigkeit			Maximieren
	Fettbindung	f_3	Bindung der Fette			Maximieren
	ProtVit	f_4	Anteile der Proteine und Vitamine			Maximieren
	Textur	f_5	Farbe und Textur			Maximieren

Tabelle 8.2: Zusammensetzung der synthetischen Beispielrezepturen

[85] Die Abhängigkeiten zwischen den Rezepturkomponenten und Qualitätsgrößen wurden hier stark vereinfacht und sollen lediglich die Wirkungsweise der entwickelten Methode illustrieren und keinesfalls eine reale Rezepturoptimierung leisten. Die hier verwendete Beispiel-Rezepturtabelle ist unter http://www.wzw.tum.de/~dmeyer/ zugänglich.

Ziel der Modellierung und Optimierung sei das Auffinden von Rezepturen, die bei möglichst geringem Preis aller verwendeten Rohmaterialien die angegebenen Qualitätsgrößen maximieren d.h. einen Kompromiß zwischen kostenminimaler Produktion und hoher Produktqualität realisieren. Darüber hinaus sollen die Abhängigkeiten zwischen den verschiedenen Prozessgrößen extrahiert und ihr Einfluß auf die Qualität des Produktes dargestellt werden.

8.2.3 Prozessmodellbildung

Grundlage der datengetriebenen Modellierung ist ausschließlich die Menge der Arbeitspunkte des zu modellierenden Prozesses, d.h. die verschiedenen Kombinationen von Prozesseingangsgrößen sowie die entsprechenden Prozessausgangsgrößen. Daher kommt der Erfassung dieser Daten eine große Bedeutung zu und sollte unter Berücksichtigung moderner Methoden der Versuchsplanung [Kleppmann, 1998] erfolgen. Nur wenn auch die Beispieldaten den Prozess in den relevanten Arbeitsbereichen beschreiben, kann auch das Prozessmodell eine relevante Beschreibung des Prozesses leisten.

Häufig sind die in der Praxis gemessenen Daten jedoch unvollständig, d.h. es liegen beispielsweise für einen Arbeitspunkt nicht für alle Prozessgrößen Werte vor. Um jedoch auch in diesen Fällen den Arbeitspunkt als Trainingsvektor für die Belehrung des Modells verwenden zu können, ist es notwendig, die fehlenden Werte mittels einer geeigneten Strategie zu belegen. Eine häufig verwendete Methode für die Ersetzung fehlender Daten ist, diese durch die Mittelwerte der jeweiligen Prozessgröße zu ersetzen. Diese Methode hat jedoch den Nachteil, daß der unvollständige Vektor u.U. durch die zu starke Mittelung in einen Punkt im Datenraum verschoben wird, der nicht seiner tatsächlichen Repräsentation entspricht.

Eine erfolgversprechendere und hier verwendete Strategie ist es dagegen, die fehlende Komponente nicht durch den Mittelwert der jeweiligen Komponente aller Trainingsvektoren, sondern nur seiner nächsten Nachbarn zu ersetzen. Eine Realisierung dieser Idee besteht darin, die Trainingsvektoren zu clustern und für die Ersetzung die Komponenten der entsprechenden Clusterzentren zu verwenden. Aufgrund der einfachen aber wirkungsvollen Arbeitsweise wurde in dieser Anwendung die Clusterananlyse mittels Learning Vector Quantization (LVQ: siehe Abschnitt 3.2.1) durchgeführt. Die LVQ gestattet, eine Menge von Codebook-Vektoren im Datenraum so zu positionieren, daß die Vektoren die Schwerpunkte der vorhandenen Cluster möglichst gut repräsentieren. Fehlende Meßwerte in der Datenbank können so durch die entsprechende Komponente des für diesen Datensatz repräsentativen Codebook-Vektors eingesetzt werden. Die Auswahl, welcher Codebook-Vektor mit dem Vektor korrespondiert, dessen fehlende Komponente ermittelt werden soll, erfolgt ohne Berücksichtigung dieser Komponente.

Die eigentliche Prozessmodellbildung, d.h. Abbildung der Prozesseingangsgrößen auf die Prozessausgangsgrößen wird wie in Abschnitt 6.3 beschrieben mit Hilfe eines Feed-Forward-RBF-Netzes mit exponentiellen radialsymmetrischen Basisfunktionen realisiert. Wie in Abschnitt 3.2.2 dargestellt, gestattet dieser Netztyp durch eine Linearkombination seiner Basisfunktionen eine nichtlineare Funktionsapproximation.

Die Anzahl der Eingabe- bzw. Ausgabeneurone entspricht der Anzahl der Prozesseingangs- bzw. Prozessausgangsparametern, die Anzahl der Zwischenneurone ist abhängig von der Anzahl der Prozessparameter und Datensätze sowie der Güte des LVQ. Zusätzlich zu den Aktivitäten der Output-Neurone wird ein intuitives Vertrautheitsmaß (Konfidenz; siehe Abschnitt 6.5.2 S. 98) als Modell-Output geliefert. Um die Zwischenneurone im RBF-Netz zu initialisieren, werden die Codebook-Vektoren des LVQ als Positionsvektoren interpretiert. Damit ist eine möglichst repräsentative Positionierung der Zwischenneurone möglich, die die Struktur im Datenraum widerspiegelt. Ein Gütemaß des LVQ, das aus der Streuung der Datensätze um die

Codebook-Vektoren ermittelt wird, gestattet eine sinnvolle Vorbelegung der Größe der radialen Basisfunktionen im RBF-Netz [Meyer und Eder, 1999].

Das implementierte Lernverfahren modifiziert im Gegensatz zum Standardlernverfahren (Abschnitt 3.2.2) lediglich die Kopplungsmatrix zwischen den Neuronen der Zwischen- und der Ausgabeschicht. Das entstehende Gleichungssystem zur Minimierung des Fehlverhaltens des Prozessmodells wird mittels einer Singulär-Werte-Zerlegung gelöst. Auf diese Weise wird eine sehr rasche Trainingsphase ermöglicht [Meyer und Eder, 1999].

Modellkonfiguration und -Validierung: Im System MOMBES erfolgt im ersten Schritt der Prozessmodellbildung die Anbindung der Rezepturdaten an das System sowie die Zuweisung der Prozessgrößen als Modelleingangs- und Ausgangsgrößen (siehe Abbildung 8.9).

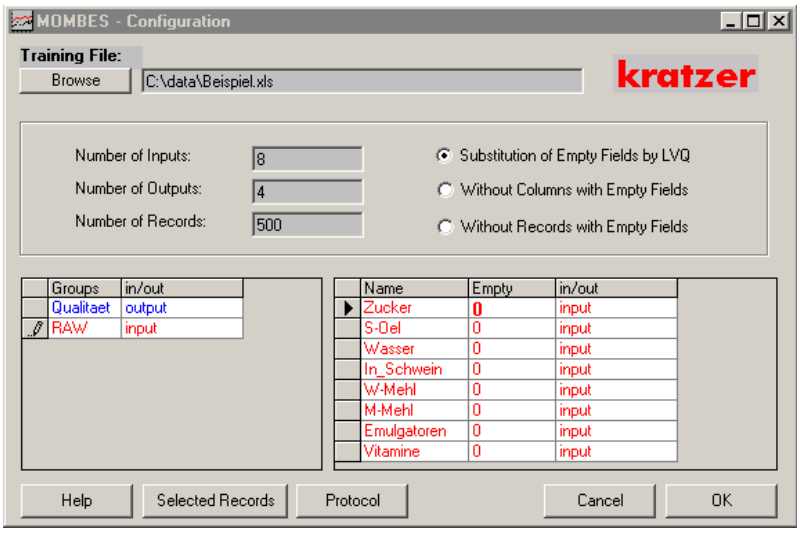


Abbildung 8.9: Dialogfeld zur Definition der Modelleingangs- und Ausgangsgrößen

Im betrachteten Beispiel erfolgt die Berechnung der Zielfunktionswerte f_2 bis f_5 für die Qualitätsgrößen: Festigkeit, Fettbindung, Protein-Vitamin-Gehalt (ProtVit) und Textur durch das Prozessmodell, wohingegen der Preis (d.h. f_1) analytisch als Summe der Preise der im einzelnen verwendeten Rohmaterialienanteile berechnet wird. Welche Eingangsgrößen als Rohmaterialien interpretiert werden, wird durch die Gruppenzugehörigkeit „RAW“ entschieden und der Rezepturtabelle entnommen. Die verwendeten Preise der entsprechenden Rohmaterialien hingegen werden aus einer tages-aktuellen Preisliste entnommen.

Nach der Anbindung der Rezepturdaten wird ein Prozessmodell in Form eines RBF-Netzes (mit 8 Eingabe-, 4 Ausgabe- sowie 300 Zwischenneuronen mit jeweils $\sigma = 0{,}32$) erzeugt[86] und steht für die Validierung und das Training zur Verfügung.

Um die Güte des Prozessmodells zu validieren und so den Anwender in seiner Einschätzung der Prognosesicherheit des Modells zu unterstützen, wurde in MOMBES ein Modul zur Modellvalidierung implementiert, mit dessen Hilfe die vorhandenen Daten genauer untersucht

[86] Die Größe von Sigma wurde entsprechend Gleichung [23] geschätzt, wobei die maximale euklidische Distanz zwischen zwei Input-Vektoren $d = \sqrt{n}$ gesetzt wird. Das ist möglich, da alle Prozessgrößen für das Training auf das Intervall $[0;1]$ normiert werden.

werden können. Mit Hilfe dieses Moduls können sowohl Ausreißer und Meßartefakte erkannt, als auch ein Eindruck von der Qualität der Daten und der zu erwartenden Güte des Vorhersagesystems gewonnen werden.

Bei der Validierung wird das Prozessmodell lediglich mit einer Teilmenge der Rezepturen (Train Set) belehrt. Anschließend wird das Prozessmodell sowohl auf die zum Training verwendeten als auch die nicht verwendeten Rezepturen (Test Set) angewandt. Durch den Vergleich der bekannten und durch das Modell prognostizierten Prozessausgangsgrößen, können Aussagen darüber getroffen werden, inwieweit die zum Training verwendeten Rezepturen einen Schluß auf die Prozessausgangsgrößen unbekannter Rezepturen zulassen. Im Dialogfeld (siehe Abbildung 8.10) der Modellvalidierung werden statistische Informationen über die Trainings- und Testdaten sowie das erzielte Approximationsverhalten des Modells ausgegeben.

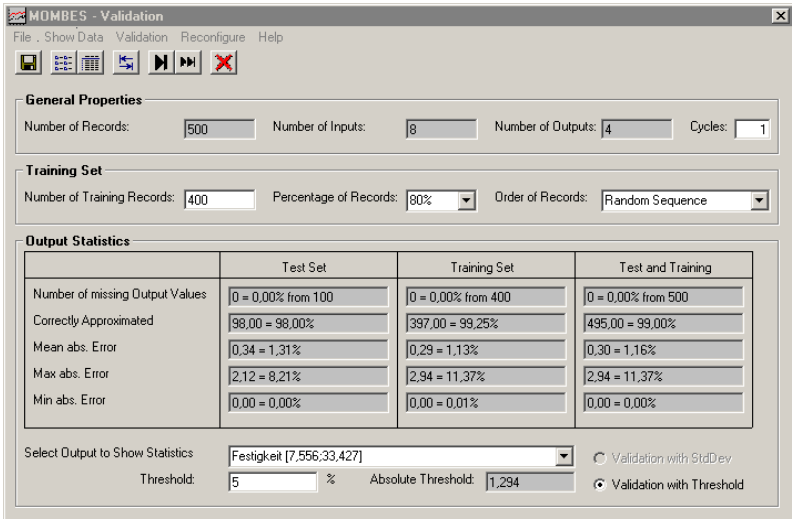


Abbildung 8.10: Dialogfeld zur Konfiguration der Validierung des Prozessmodells

Durch ein Verändern des Verhältnisses der Anzahl der Trainingsdaten zur Anzahl der Testdaten („Number of Trainings Records“ bzw. „Percentage of Records) und dem Analysieren der Vorhersagegüte des Prozessmodells insbesondere bezüglich des Test-Sets können Rückschlüsse auf die Qualität des Modells gezogen werden.

Die Ergebnisse der Validierung werden getrennt nach Test- und Trainingsmenge dargestellt. Zur Einschätzung der Güte des Prozessmodells werden die Kriterien: „Correctly Approximated“ und „Absolute Error“ verwendet, die auf der absoluten Abweichung δ_i^j zwischen der durch das Modell prognostizierten $\hat{y}_i^j$ und dem entsprechenden, durch die Trainingsdaten vorgegebenen Wert y_i^j der Prozessausgangsgröße i der Rezeptur j basieren (Gleichung [86]).

$$\delta_i^j = \left| \hat{y}_i^j - y_i^j \right| \qquad [86]$$

Correctly Approximated: Durch den Wert dieser Größe wird beschrieben, bei wieviel Prozent der untersuchten Datensätze gilt: $\delta_i^j \leq \varepsilon$, wobei ε ein parametrierbarer Schwellwert ist.

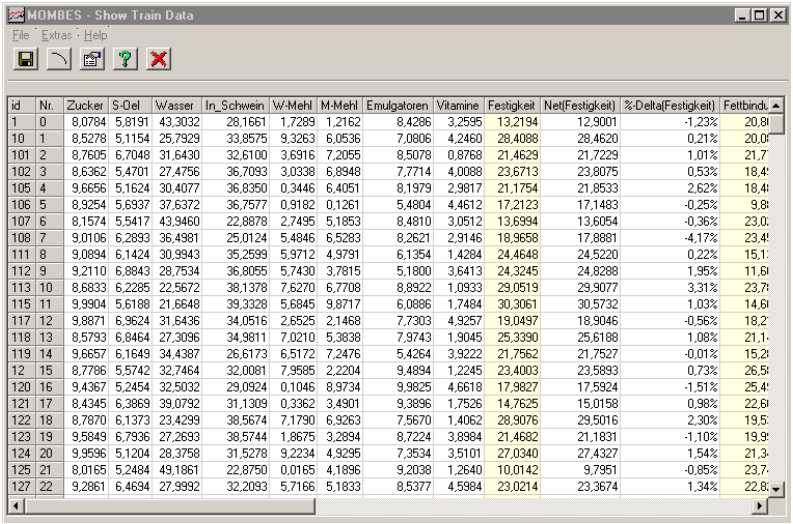
Die prozentualen Werte für die mittlere (*Mean Absolute Error*), maximale (*Max Absolute Error*) und minimale (*Min Absolute Error*) absolute Abweichung δ_i^j beziehen sich immer auf den durch alle Trainingsrezepturen vorgegebenen Wertebereich der entsprechenden Ausgangsgröße i.

Im vorliegenden Beispiel ergaben sich die in Tabelle 8.3 dargestellten Werte für die Prognosegüte des Modells, das mit 80% (d.h. 400) zufällig aus der Rezepturtabelle entnommenen Datensätzen trainiert und sowohl an den für das Training benutzten (Train Set) als auch an den 100 unbekannten (Test Set) Rezepturen getestet wurde.

		Alle	Festigkeit	Fettbindung	ProtVit	Textur
	Wertebereich	-	[7,55;33,42]	[8,47;30,01]	[3,85;58,76]	[21,79;77,31]
	$\varepsilon = 5\%$ des Wertebereichs	-	1,29	1,08	2,74	2,77
Test Set	Correctly Approx.	93,5%	98,0%	97,0%	88,0%	91,0%
	Mean Abs Error in %	1,91%	1,31%	1,99%	2,29%	2,07%
	Max Abs Error in %	11,49%	8,21%	8,61%	11,49%	9,43%
	Min Abs Error in %	0,0%	0,0%	0,02%	0,01%	0,02%
Train Set	Correctly Approx.	98,75%	99,25%	99,0%	98,0%	98,75%
	Mean Abs Error in %	1,41%	1,13%	1,53%	1,52%	1,47%
	Max Abs Error in %	11,37%	11,37%	5,95%	8,14%	6,65%
	Min Abs Error in %	0,0%	0,0%	0,0%	0,01%	0,01%

Tabelle 8.3: Ergebnisse der Modellvalidierung

Die vom Prozessmodell prognostizierten Werte $\hat{y}_i$, die entsprechende Prozessgröße der Trainingsdaten y_i sowie die absoluten und prozentualen Werte für δ_i^j zwischen beiden Größen können für jede Modellausgangsgröße i, für jeden Datensatz j der Test- und Trainingsfraktion tabellarisch und graphisch dargestellt werden.



id	Nr.	Zucker	S-Oel	Wasser	In_Schwein	W-Mehl	M-Mehl	Emulgatoren	Vitamine	Festigkeit	Net(Festigkeit)	%-Delta(Festigkeit)	Fettbindu
1	0	8,0784	5,8191	43,3032	28,1661	1,7289	1,2162	8,4286	3,2595	13,2194	12,9001	-1,23%	20,8
10	1	8,5278	5,1154	25,7929	33,8575	9,3263	6,0536	7,0806	4,2460	28,4088	28,4620	0,21%	20,0
101	2	8,7605	6,7048	31,6430	32,6100	3,6916	7,2055	8,5078	0,8768	21,4629	21,7229	1,01%	21,7
102	3	8,6362	5,4701	27,4756	36,7093	3,0338	6,8948	7,7714	4,0088	23,6713	23,8075	0,53%	18,4
105	4	9,6656	5,1624	30,4077	36,8350	0,3446	6,4051	8,1979	2,9817	21,1754	21,8533	2,62%	18,4
106	5	8,9254	5,6937	37,6372	36,7577	0,9182	0,1261	5,4804	4,4612	17,2123	17,1483	-0,25%	9,8
107	6	8,1574	5,5417	43,9460	22,8878	2,7495	5,1853	8,4810	3,0512	13,6994	13,6054	-0,36%	23,0
108	7	9,0106	6,2893	36,4981	25,0124	5,4846	6,5283	8,2621	2,9146	18,9658	17,8881	-4,17%	23,4
111	8	9,0894	6,1424	30,9943	35,2599	5,9712	4,9791	6,1354	1,4284	24,4648	24,5220	0,22%	15,1
112	9	9,2110	6,8843	28,7534	36,8055	5,7430	3,7815	5,1800	3,6413	24,3245	24,8288	1,95%	11,6
113	10	8,6833	6,2285	22,5672	38,1378	7,6270	6,7708	8,8922	1,0933	29,0519	29,9077	3,31%	23,7
115	11	9,9904	5,6188	21,6648	39,3328	5,6845	9,8717	6,0886	1,7484	30,3061	30,5732	1,03%	14,6
117	12	9,8871	6,9624	31,6436	34,0516	2,6525	2,1468	7,7303	4,9257	19,0497	18,9046	-0,56%	18,2
118	13	8,5793	6,8464	27,3096	34,9811	7,0210	5,3838	7,9743	1,9045	25,3390	25,6188	1,08%	21,1
119	14	9,6657	6,1649	34,4387	26,6173	6,5172	7,2476	5,4264	3,9222	21,7562	21,7527	-0,01%	15,2
12	15	8,7786	5,5742	32,7464	32,0081	7,9585	2,2204	9,4894	1,2245	23,4003	23,5893	0,73%	26,5
120	16	9,4367	5,2454	32,5032	29,0924	0,1046	8,9734	9,9825	4,6618	17,9827	17,5924	-1,51%	25,4
121	17	8,4345	6,3869	39,0792	31,1309	0,3362	3,4901	9,3896	1,7526	14,7625	15,0158	0,98%	22,6
122	18	8,7870	6,1373	23,4299	38,5674	7,1790	6,9263	7,5670	1,4062	28,9076	29,5016	2,30%	19,5
123	19	9,5849	6,7936	27,2693	38,5744	1,8675	3,2894	8,7224	3,8984	21,4682	21,1831	-1,10%	19,9
124	20	9,9596	5,1204	28,3758	31,5278	9,2234	4,9295	7,3534	3,5101	27,0340	27,4327	1,54%	21,3
125	21	8,0165	5,2484	49,1861	22,8750	0,0165	4,1896	9,2038	1,2640	10,0142	9,7951	-0,85%	23,7
127	22	9,2861	6,4694	27,9992	32,2093	5,7166	5,1833	8,5377	4,5984	23,0214	23,3674	1,34%	22,8

Abbildung 8.11: Tabellarische Darstellung der prognostizierten und tatsächlichen Ausgangsgröße „Festigkeit“ des Prozessmodells

Abbildung 8.11 zeigt beispielhaft einen Ausschnitt dieser Darstellung der Ergebnisse der Modellvalidierung, hier für das Test-Set. In der graphischen Darstellung (Abbildung 8.12) sind die Werte der Prozeßgröße „Festigkeit“ und seine durch die Trainingsdaten vorgegebenen Werte abgebildet. Es ist zu erkennen, daß das Prozessmodell auch für unbekannte Rezepturen Ausgangsgrößen prognostiziert, die eine sehr geringe Abweichung (der mittlere absolute Fehler liegt hier bei 1,31%) von den tatsächlichen, durch die Rezeptur vorgegebenen Werten,

aufweisen. Lediglich bei 2% der Testrezepturen liegt diese Abweichung höher und beträgt maximal 8,21% des Wertebereichs der Prozessgröße „Festigkeit“.

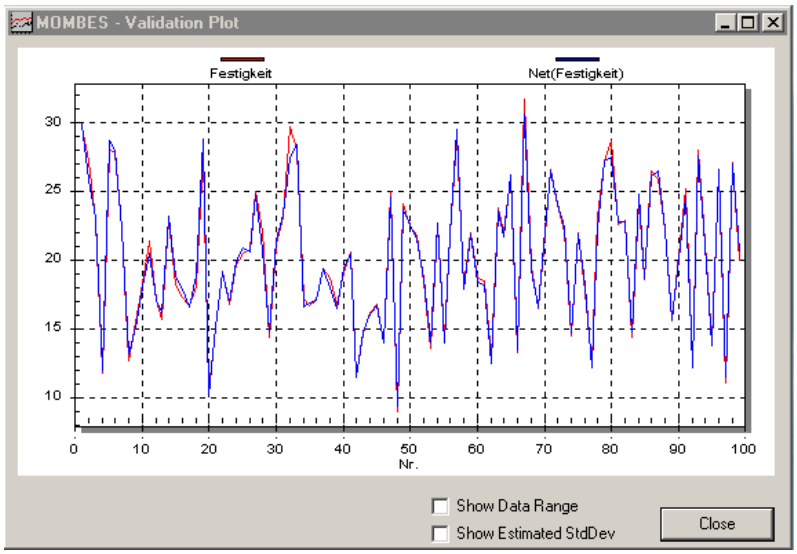


Abbildung 8.12: Graphische Darstellung der prognostizierten und tatsächlichen Ausgangsgröße „Festigkeit“ des Prozessmodells

Die Prognosefähigkeit des Modells bezüglich aller 4 Prozessausgangsgrößen ist etwa vergleichbar gut: die mittlere absolute Abweichung der durch das Modell prognostizierten und tatsächlichen Werte aller Prozessausgangsgrößen des Test-Set beträgt nur ca. 2% der entsprechenden Wertebereiche. Dies läßt den Schluß zu, daß es mit der hier verwendeten Methode und den zur Verfügung stehenden Daten möglich ist, ein Prozessmodell ausreichender Qualität zur Vorhersage der 4 Prozessausgangsgrößen zu erzeugen.

Modelltraining und -Anwendung: Zur Erzeugung des abschließenden Prozessmodells zur Abbildung der Prozesseingangs- auf die Prozessausgangsgrößen wird das RBF-Netz mit allen (hier 500) Rezepturen trainiert. Die Vorhersagegüte des Modells hinsichtlich aller Trainingsdaten, was einer Modellvalidierung mit 100% Trainingsdaten entspricht, wird unmittelbar nach dem Training für den Anwender dargestellt (siehe Abbildung 8.13).

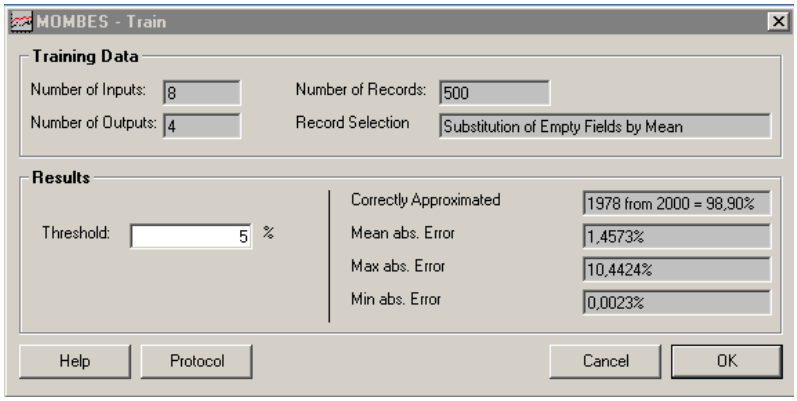
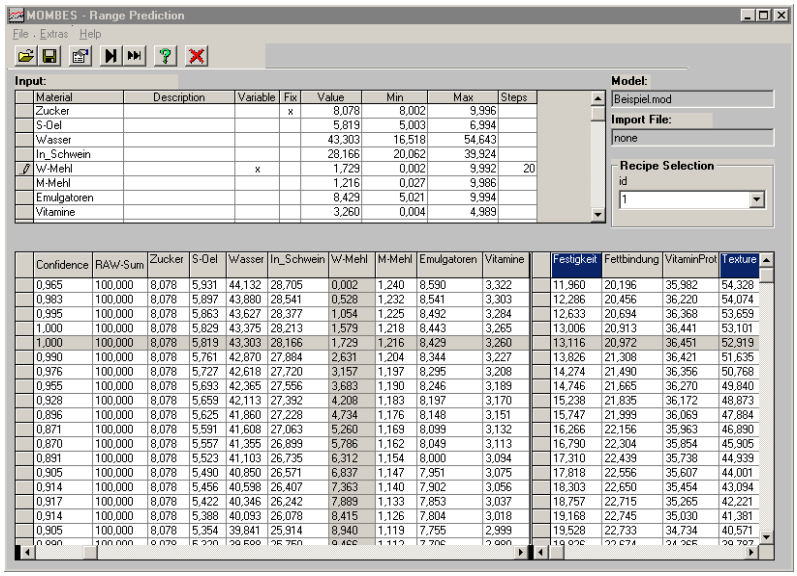


Abbildung 8.13: Dialogfeld zum Training des Prozessmodells

Nachdem das Prozessmodell erzeugt wurde, kann es für die Prognose von Ausgangsgrößen bei manipulierten Eingangsgrößen verwendet werden. Auf diese Weise kann das Verhalten des modellierten Systems bei unbekannten Betriebspunkten (neuen Rezepturen) simuliert und die Wirkung verschiedener Eingangsgrößen auf die Ausgangsgrößen getestet werden.

Beispielhaft für die Anwendung des Prozessmodells soll hier das Modul der Parametervariation erwähnt werden. Bei der Parametervariation wird eine Eingangsgröße einer beliebig modifizierten Rezeptur in einem vorgegebenen Intervall variiert. Auf diese Art entstehen neue Rezepturen, die alphanumerisch und graphisch angezeigt werden und anhand derer der Anwender die Wirkung der Veränderung eines Parameters auf die Prozessausgangsgrößen analysieren kann.

Im dargestellten Beispiel sind alle Eingangsgrößen Rohmaterialien, d.h. eine Veränderung einer Prozesseingangsgröße muß gleichzeitig eine Veränderung anderer Rohmaterialien zur Folge haben, um deren Summe konstant auf 100% zu halten. Wird also ein Rohmaterial variiert, so werden in den neuen Rezepturen automatisch die Werte aller der Rohmaterialien kompensiert, die nicht ausdrücklich durch den Anwender auf einen bestimmten Wert fixiert wurden. Wird also z.B. der Wert eines Rohmaterials verringert, werden die Werte der übrigen, nicht fixierten Rohmaterialien der Rezeptur proportional zu ihren ursprünglichen Werten erhöht.



Material	Description	Variable	Fix	Value	Min	Max	Steps
Zucker			x	8,078	8,002	9,996	
S-Oel				5,819	5,003	6,994	
Wasser				43,303	16,518	54,643	
In_Schwein				28,166	20,062	39,924	
W-Mehl		x		1,729	0,002	9,992	20
M-Mehl				1,216	0,027	9,986	
Emulgatoren				8,429	5,021	9,994	
Vitamine				3,260	0,004	4,989	

Confidence	RAW-Sum	Zucker	S-Oel	Wasser	In_Schwein	W-Mehl	M-Mehl	Emulgatoren	Vitamine	Festigkeit	Fettbindung	VitaminProt	Texture
0,965	100,000	8,078	5,931	44,132	28,705	0,002	1,240	8,590	3,322	11,960	20,196	35,982	54,328
0,983	100,000	8,078	5,897	43,880	28,541	0,528	1,232	8,541	3,303	12,286	20,456	36,220	54,074
0,995	100,000	8,078	5,863	43,627	28,377	1,054	1,225	8,492	3,284	12,633	20,694	36,368	53,659
1,000	100,000	8,078	5,829	43,375	28,213	1,579	1,218	8,443	3,265	13,006	20,913	36,441	53,101
1,000	100,000	8,078	5,819	43,303	28,166	1,729	1,216	8,429	3,260	13,116	20,972	36,451	52,919
0,990	100,000	8,078	5,761	42,870	27,884	2,631	1,204	8,344	3,227	13,826	21,308	36,421	51,635
0,976	100,000	8,078	5,727	42,618	27,720	3,157	1,197	8,295	3,208	14,274	21,490	36,356	50,768
0,955	100,000	8,078	5,693	42,365	27,556	3,683	1,190	8,246	3,189	14,746	21,665	36,270	49,840
0,928	100,000	8,078	5,659	42,113	27,392	4,208	1,183	8,197	3,170	15,238	21,835	36,172	48,873
0,896	100,000	8,078	5,625	41,860	27,228	4,734	1,176	8,148	3,151	15,747	21,999	36,069	47,884
0,871	100,000	8,078	5,591	41,608	27,063	5,260	1,169	8,099	3,132	16,266	22,156	35,963	46,890
0,870	100,000	8,078	5,557	41,355	26,899	5,786	1,162	8,049	3,113	16,790	22,304	35,854	45,905
0,891	100,000	8,078	5,523	41,103	26,735	6,312	1,154	8,000	3,094	17,310	22,439	35,738	44,939
0,905	100,000	8,078	5,490	40,850	26,571	6,837	1,147	7,951	3,075	17,818	22,556	35,607	44,001
0,914	100,000	8,078	5,456	40,598	26,407	7,363	1,140	7,902	3,056	18,303	22,650	35,454	43,094
0,917	100,000	8,078	5,422	40,346	26,242	7,889	1,133	7,853	3,037	18,757	22,715	35,265	42,221
0,914	100,000	8,078	5,388	40,093	26,078	8,415	1,126	7,804	3,018	19,168	22,745	35,030	41,381
0,905	100,000	8,078	5,354	39,841	25,914	8,940	1,119	7,755	2,999	19,528	22,733	34,734	40,571

Abbildung 8.14: Dialogfeld des Modul Parametervariation

Im vorliegenden Beispiel soll die Wirkung von Weizenmehl auf die Festigkeit und die Textur des Produkts untersucht werden. Dazu wird zunächst eine Ausgangsrezeptur gewählt. Dies kann entweder durch direktes Laden des entsprechenden Datensatzes aus der Rezepturtabelle oder durch Modifikation der Eingangsgrößen erfolgen. Anschließend bestimmt der Anwender, welche Modelleingangsgröße, in welchem Intervall, in wie vielen äquidistanten Schritten variiert werden soll und welche Modelleingangsgrößen von der Kompensation ausgeschlossen werden sollen. Hier wird die Größe „W-Mehl“ im Intervall [0,002;9,992] in 20 Schritten variiert, wobei die Größe „Zucker“ nicht kompensiert, sondern auf dem Wert von 8,078 konstant gehalten wird. MOMBES generiert nun eine Menge von Rezepturen, die sich durch die Variation von „W-Mehl“ ergeben. Diese Rezepturen werden inklusive ihrer Konfidenz tabellarisch (siehe Abbildung 8.14) und graphisch (siehe Abbildung 8.15) dargestellt.

Es ist zu erkennen, daß eine Erhöhung des Anteils von Weizenmehl eine Erhöhung der Festigkeit und eine Verringerung der Textur des Produkts zur Folge hat, wobei die Abhängigkeiten nahezu linear sind.

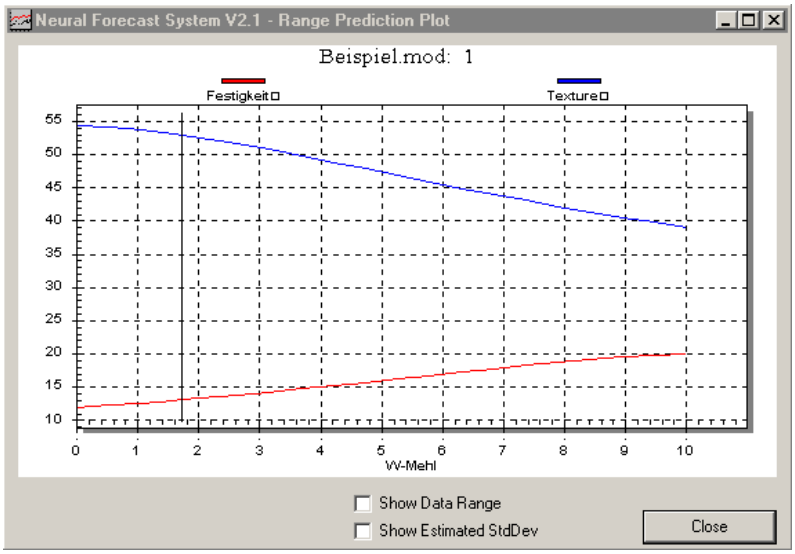


Abbildung 8.15: Graphische Darstellung der Wirkung der Variation des Anteils von Weizenmehl auf die Festigkeit und Textur des Produkts

Dies ist mit der Bindungsfähigkeit von Mehl zu erklären: indem sein Anteil in der Rezeptur erhöht wird, erhöht sich auch die Bindefähikeit der Mischung, was sich durch eine größere Festigkeit und einen geringeren Anteil einzelner, kleiner „Bröckchen“ – also einer Verringerung der Textur - im Produkt äußert.

8.2.4 Modellgestützte Versuchsplanung

Die modellgestützte Versuchsplanung ist ein Modul von MOMBES, das im Rahmen der Anpassungen (NFCS) auf Probleme der Rezepturoptimierung entwickelt wurde. Hier werden durch das System Rezepturen vorgeschlagen, die zusätzlich zu den Trainingsrezepturen getestet werden sollten, um die Prognosesicherheit des Prozessmodells zu erhöhen. Setzt man voraus, daß die Prognosesicherheit und Qualität eines Modells verbessert werden kann, indem die Verteilung der ihm zugrunde liegenden Trainingsvektoren im Raum der Trainingsvektoren homogenisiert wird, muß nach Datensätzen gesucht werden, die möglichst weit von bekannten Stützstellen (Trainingsvektoren) entfernt sind. Dabei ist der Suchraum durch die Wertebereiche der ursprünglichen Trainingsvektoren begrenzt.

Die Grundidee der modellgestützten Versuchsplanung bei MOMBES besteht daher darin, ausgehend von einer Startrezeptur und definierten Randbedingungen der Prozessein- und Prozessausgangsgrößen (Intervallgrenzen) Rezepturen mit möglichst niedriger Prognosekonfidenz zu finden. Dabei wird die Konfidenz, die nicht überschritten werden soll, durch den Anwender parametriert. Da die Konfidenz einer Rezeptur ein Maß dafür ist, welche minimale euklidische Distanz bezüglich der Modelleingangsgrößen sie zu den Trainingsrezepturen aufweist, ist sie als Suchkriterium geeignet (siehe S. 98).

Die Suche nach Rezepturen, die den durch den Anwender angegebenen Randbedingungen entsprechen und eine möglichst geringe Konfidenz besitzen, wird mit dem in Abschnitt 6.4 beschriebenen Optimierungskern von MOMBES realisiert. Die Berechnung der Prozessausgangsgrößen erfolgt dabei durch das Prozessmodell. Jedoch wird die Berechnung

der Fitness Φ der Individuen hier nicht mehr entsprechend Gleichung [44] (S. 74) sondern nach Gleichung [87] durchgeführt, da das Auffinden von Rezepturen minimaler Konfidenz kein MOP, sondern nunmehr ein Problem zur Minimierung einer einzelnen Zielfunktion darstellt.

$$\Phi(\vec{a}_i) = FIT(\vec{a}_i, P) = \begin{cases} \dfrac{1}{Conf(\vec{a}_i)+1}; & \text{für } \vec{x} \in X_{feasible} \\ \dfrac{1}{Conf(\vec{a}_i)+p(\vec{a}_i)+2}; & \text{für } \vec{x} \notin X_{feasible} \end{cases} \quad \text{mit: } \vec{x} = \Gamma(\vec{a}_i) \qquad [87]$$

Hierbei ist $Conf(\vec{a}_i)$ die Konfidenz, die sich bei der Verarbeitung des Vektors der Modelleingangsgrößen $\vec{x} = \Gamma(\vec{a}_i)$ des Individuums $\vec{a}_i$ der Population P durch das Prozessmodell ergibt (Definition der Konfidenz: S. 98). Der Grad der Verletzung der Randbedingungen des Individuums (Überschreitung der durch den Anwender parametrierten Wertebereiche der Modelleingangs- und Ausgangsgrößen bezogen auf den durch die Trainingsrezepturen definierten Wertebereich der jeweiligen Prozessgröße) wird durch $p(\vec{a}_i)$ (siehe Gleichung [45] S.74) bestimmt.

Die bei der Anwendung von MOMBES zur Rezepturoptimierung geforderte Einhaltung der (harten) Randbedingung, daß die Summe aller Rohmaterialien stets 100% ergeben muß, ist durch eine Anpassung der Operatoren für die Initialisierung, Mutation und Rekombination der Individuen gewährleistet: Hier werden nicht, wie in Algorithmus 4 (S.75) und Algorithmus 5 (S.76) definiert, jeweils *alle* Entscheidungsvariablen verändert, sondern immer nur eine, zufällig ausgewählte Komponente i aus $\vec{x}$ rekombiniert bzw. mutiert. Alle anderen, nicht durch den Anwender fixierten Komponenten $x_j \mid \forall j \neq i$ werden jedoch im Anschluß daran proportional zu ihrem ursprünglichen Werten angepaßt, so daß die Summe aller Komponenten von $\vec{x}$ wieder 100% beträgt. Die Berechnung der Fitness nach Gleichung [87] gewährleistet, daß Individuen, die eine Verletzung der Randbedingung bestimmen (d.h. für die gilt: $p(\vec{a}_i) > 0$), in jedem Fall eine geringere Fitness zugeordnet wird, als Individuen, die den Randbedingungen genügen. Dagegen wächst bei gültigen Individuen (siehe Definition 15, S.44) die Fitness mit der Verringerung ihrer Konfidenz. Bei zwei Individuen $\vec{a}_i$ und $\vec{a}_j$ mit $p(\vec{a}_i) = p(\vec{a}_j)$, ist die Fitness desjenigen Individuums größer, dessen Konfidenz geringer ist.

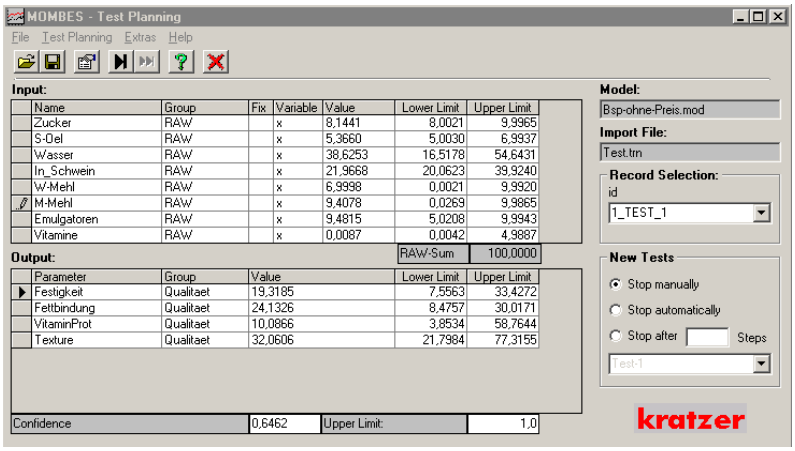


Abbildung 8.16: Dialogfeld des Moduls der modellgestützten Versuchsplanung

In Abbildung 8.16 ist das Dialogfeld der modellgestützten Versuchsplanung dargestellt. Im vorliegenden Beispiel sollten durch das System Rezepturen vorgeschlagen werden, deren Konfidenz geringer als 1 ist. Dabei konnten durch MOMBES alle Prozessgrößen innerhalb des Wertebereichs variiert werden, der durch die entsprechenden Werte in den Spalten „Upper Limit“ und „Lower Limit“ der Tabellen „input“ und „output“ definiert ist.

Nach ca. 300 Generationen wurde mit Hilfe des Optimierungskerns von MOMBES (mit $\mu = 15$, $\lambda = 100$, $\rho = 2$, $p_a = 0$ und $\overline{N} = 1$[87]) eine Rezeptur mit einer Konfidenz von $Conf = 0{,}64$ generiert. Diese hypothetische Rezeptur wurde von MOMBES als diejenige identifiziert, die bezüglich der Prozesseingangsgrößen am wenigsten von den Trainingsrezepturen repräsentiert wird und die vorgegebenen Randbedingungen erfüllt.

Indem die Eingangsgrößen der durch diese Versuchsplanung vorgeschlagenen Rezepturen am realen Prozess getestet und mit den entsprechend gemessenen Werten der Prozessausgangsgrößen in die Trainingsdatenbank aufgenommen werden, kann eine sukzessive Verfeinerung der Trainingsdatenbank (Rezepturtabelle) erreicht werden. Wird das Prozessmodell mit dieser angepaßten Rezepturtabelle trainiert, kann eine Erhöhung seiner Modellqualität erreicht werden.

8.2.5 Optimierung: Generierung näherungsweise pareto-optimaler Lösungen

Ziel der Optimierung ist es, ausgehend von einer Startrezeptur und definierten Randbedingungen der Eingangs- und Ausgangsgrößen (Wertebereichsgrenzen), unter Minimierung der Kosten für die Gesamtrezeptur die angegebenen Qualitätsgrößen zu maximieren.

Im vorliegenden Beispiel bedeutet dies die Optimierung sich widersprechender Ziele: nämlich qualitativ hochwertigste Rezepturen zu finden, deren verwendete Rohmaterialien minimale Kosten verursachen. Dabei sollen zunächst nur die in Spalte „Preis-1“ von Tabelle 8.2 definierten Preise berücksichtigt werden.

Das sich die Optimierungsziele widersprechen, macht folgendes Gedankenexperiment deutlich: Rezepturen mit einem hohen Anteil des (teuren) Vitamin-Mix, einem hohen Anteil des proteinhaltigen Fleisches sowie einem hohen Anteil von Mehl und einem geringen Anteil von Wasser weisen eine hohe Festigkeit, einen hohen Anteil von Vitaminen und Proteinen sowie eine starke Färbung und Textur auf. Diese Rezepturen sind jedoch aufgrund der hohen Anteile von Rohmaterialien mit hohen Preisen auch teuer. Dagegen sind die billigsten Rezepturen diejenigen mit einem hohen Anteil von Wasser, die jedoch auch eine geringe Festigkeit, einen geringen Anteil von Vitaminen und sowie eine geringe Färbung und Textur aufweisen. Preis und Qualität widersprechen sich also hier. Andererseits bestimmen diejenigen Rezepturen mit dem größten Anteil an Vitaminen und Proteinen zwar den größten Preis, jedoch nicht die maximale Festigkeit. Das ist damit zu erklären, daß der teure Vitamin-Mix prozentual einen höheren Vitamin-Anteil im Endprodukt bestimmt, als die natürlichen Rohstoffe: Fleisch, Mehl und Soja-Öl. Um den Anteil des Vitamin-Mix bei minimalen Anteil von Wasser zu maximieren, müssen andere, die Festigkeit stark bestimmende Rohmaterialien (aufgrund der Einhaltung der 100%-Bedingung) verringert werden. Das führt dazu, daß Rezepturen mit maximaler Festigkeit zwar einen hohen, jedoch nicht den maximalen Preis verursachen. Diese einfachen Überlegungen zeigen, wie komplex die Zusammenhänge zwischen den Rohmaterialien und Qualitätsgrößen sowie ihrem Preis bereits bei dieser einfachen synthetischen Rezepturtabelle sind und eine multikriterielle Optimierung unter Berücksichtigung einer großen Anzahl von Randbedingungen

[87] Da hier nur eine Zielfunktion minimiert wird, existiert auch immer nur ein dominantes Individuum, d.h. die Pareto-Front enthält nur ein Element.

notwendig machen. Erschwerend kommt hinzu, daß die Prozesseingangsgrößen aufgrund der zur Einhaltung der 100%-Bedingung notwendigen Kompensation eine Epistasis (siehe S. 32) aufweisen. Damit ist es nicht möglich, die Wirkung der Veränderung einer einzelnen Prozesseingangsgröße zu analysieren, sondern von mindestens 2 Prozesseingangsgrößen, die sich in ihrer Wirkung jedoch u.U. widersprechen.

Die multikriterielle Optimierung zur Generierung näherungsweise pareto-optimaler Lösungen wird mit dem Optimierungskern von MOMBES (siehe Abschnitt 6.4) durchgeführt, wobei die Berechnung der Qualitätsgrößen, d.h. der Werte $\hat{y}_i$ für $i = 2,..,5$ der Zielfunktionen f_i für $i = 2,..,5$, mit Hilfe des oben (siehe Abschnitt 8.2.3) beschriebenen Prozessmodells erfolgt.

Da MOMBES die simultane *Minimierung* einer Menge von Zielfunktionen realisiert, sollen die in Tabelle 8.2 bestimmten Zielfunktionen f_i für $i = 2,..,5$ derart in f_i' konvertiert werden, daß eine Minimierung von f_i' eine Maximierung von f_i bewirkt. Da die Prozessausgangsgrößen f_i für $i = 2,..,5$ prozentuale Werte sind und nicht negativ sein können, ist eine solche Konvertierung nach [88] möglich.

$$f_i' = 100 - f_i \qquad \text{mit: } i = 2,..,5 \qquad [88]$$

Der Preis der aktuellen Rezeptur wird mittels der Zielfunktion f_1 entsprechend Gleichung [89] bestimmt, wobei $cost_i$ der in der aktuellen Kostentabelle (Spalte „Preis-1“ oder „Preis-2“ von Tabelle 8.2) festgelegte Preis für 1Kg der Rohmaterialie x_i ist. Hier erfolgt die Kostenberechnung zunächst unter Berücksichtigung von Preis-1.

$$f_1 = \sum_{i=1}^{8} x_i \cdot cost_i \qquad [89]$$

Ebenso wie bei der modellgestützten Versuchsplanung wird auch hier die zu berücksichtigende harte Randbedingung, daß die Summe aller Rohmaterialien stets 100% betragen muß, in Form einer Anpassung der Evolutionsoperatoren für die Initialisierung, Mutation und Rekombination von Individuen berücksichtigt (siehe S. 154).

Neben der Berücksichtigung des Einhaltens des durch den Anwender parametrierbaren Wertebereiches jeder Prozesseingangs- und Prozessausgangsgröße wird hier eine weitere Randbedingung berücksichtigt, nämlich die bei der Berechnung der Modellausgangsgrößen durch das Prozessmodell mindestens einzuhaltende Prognosesicherheit. Dazu muß der Anwender eine bei der Berechnung der Prozessausgangsgrößen während der Optimierung durch das Modell mindestens einzuhaltende Konfidenz ($MinConf$) festlegen.

Der Optimierungskern von MOMBES wurde mit folgender Parametrierung für die Erzeugung der Menge P^* von näherungsweise pareto-optimalen Lösungen und der durch sie bestimmten Approximation PF^* der Pareto-Front entsprechend des oben beschriebenen MOP´s zur simultanen Minimierung der 5 Zielfunktionen: f_1 sowie f_2',.., f_5' verwendet:

Anzahl der Generationen:	$t_{max} = 500$
Anzahl der Eltern:	$\mu = 15$
Anzahl der Nachkommen:	$\lambda = 100$
Anzahl der Wettkämpfer:	$\rho = 2$

Größe der Elitepopulation: $\left|\overline{P}_t\right| = 250$

Parameter, der die Wahrscheinlichkeit für das Einfügen approximierter Individuen bestimmt: $p_a = 0{,}2$

Mindestens einzuhaltende Prognosesicherheit des Prozessmodells: $MinConf = 0{,}8$

Da in diesem Beispiel 5 Zielfunktionen minimiert werden sollen, wurde die Größe der Elitepopulation mit maximal 250 Individuen relativ[88] groß gewählt.

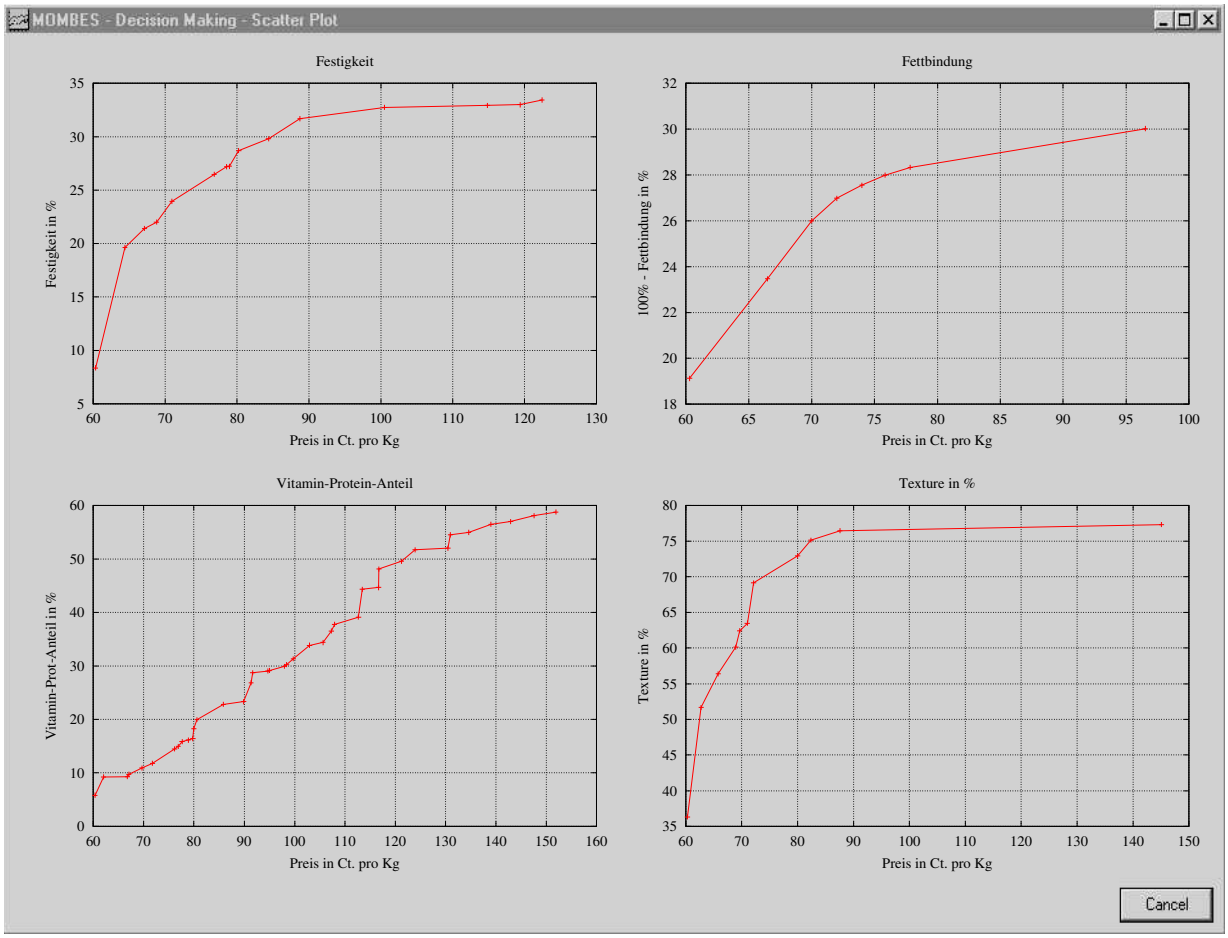


Abbildung 8.17: scatter-plots der durch MOMBES erzeugten Approximation der Pareto-Front

In Abbildung 8.17 ist beispielhaft die durch MOMBES erzeugte PF^* als scatter-plot-Matrix der nicht dominierten Menge folgender Projektionen dargestellt:

- $f_2 = (100 - f_2')$ als Funktion von f_1: Festigkeit als Funktion des Preises (Abbildung 8.17, links oben)

[88] Bei der Anwendung von MOMBES zur Lösung der in Kapitel 7 dargestellten MOP, wurde die Größe der Elitepopulation auf 100 beschränkt.

- $f_3 = (100 - f_3')$ als Funktion von f_1: Fettbindung als Funktion des Preises (Abbildung 8.17, rechts oben)
- $f_4 = (100 - f_4')$ als Funktion von f_1: Protein-Vitamin-Anteil als Funktion des Preises (Abbildung 8.17, links unten)
- $f_5 = (100 - f_5')$ als Funktion von f_1: Textur als Funktion des Preises (Abbildung 8.17, rechts unten)

Es ist gut erkennbar, daß die Qualitätsgrößen mit zunehmenden Preis maximiert werden, wobei lediglich die Abhängigkeit zwischen dem Vitamin-Protein-Anteil und dem Preis linear ist. Alle anderen Qualitätsgrößen verfügen im nicht-dominierten Bereich über eine nichtlineare Abhängigkeit vom Preis.

Auf die generierten Mengen P^* und PF^* aufbauend, können nun in der Phase des Decision-Makings diejenigen Rezepturen ausgewählt werden, die den Kompromiß zwischen Preis und Qualität für den Anwender am besten beschreiben.

8.2.6 Decision Making

Abbildung 8.18 zeigt das Dialogfeld des Decision-Making-Moduls von MOMBES für den Zugriff auf die nicht-dominierten Lösungen.

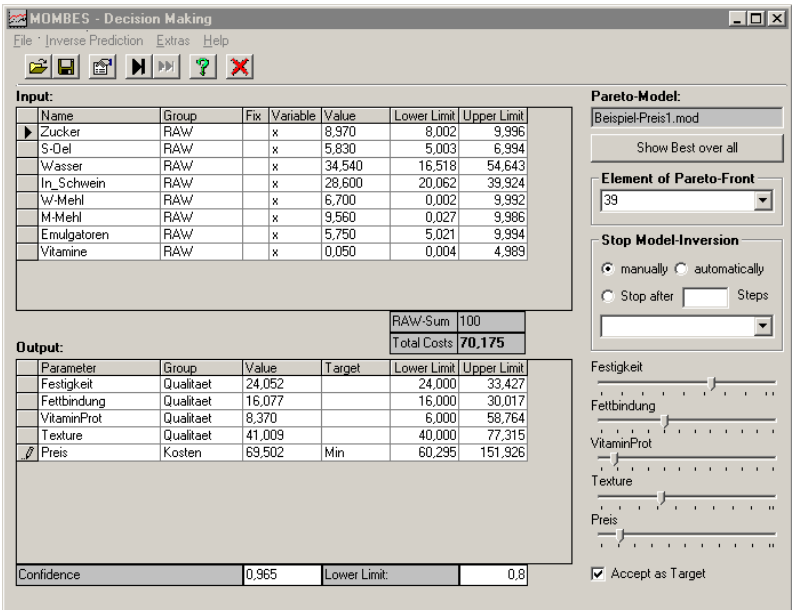


Abbildung 8.18: Dialogfeld des Decision-Making-Moduls von MOMBES

Es werden die Werte der Prozesseingangsgrößen der aktuellen Rezeptur in der Tabelle „Input“, die durch das Prozessmodell berechneten Prozessausgangsgrößen in der Tabelle „Output“ sowie die dabei ermittelte Konfidenz im Feld „Confidence“ dargestellt. Darüber hinaus wird der Wert jeder Prozessausgangsgröße als Zeigerwert im entsprechenden Schiebereglerdiagramm (Abbildung 8.18; rechts unten) angezeigt. Die aktuelle Rezeptur kann durch Auswahl eines Elements aus PF^*, aus der dem Training zugrunde liegenden Rezepturtabelle und/oder durch Editieren der Werte der Prozesseingangsgrößen gewonnen werden.

Eine Besonderheit von MOMBES besteht darin, die Abhängigkeit zwischen den Zielfunktionen innerhalb von PF^* durch die Teilmodelle MOD_i abzubilden (siehe Abschnitt 6.5.2.1, S.91). Indem der Anwender den Zeiger auf den Schiebereglern einer Prozessausgangsgröße verändert, werden durch Anwenden der Modelle MOD_i die Zeiger der Schieberegler der anderen Zielfunktionen synchron angepaßt. Die Kombination der durch die Zeigerwerte der Schiebereglerdiagramme dargestellten Prozessausgangsgrößen entspricht daher immer einem Punkt auf PF^*. Durch das Beobachten der Reaktion auf das Verschieben des Zeigers einer Prozessgröße auf die Zeiger der anderen, erhält der Anwender einen ersten Eindruck über das Verhalten der Prozessausgangsgrößen innerhalb von PF^*.

Um den Zugriff auf die näherungsweise pareto-optimalen Lösungen zu spezifizieren, sind die konkreten Optimierungsziele, d.h. der Target-Vektor $\vec{y}^{target}$, durch den Anwender anzugeben. Dazu können die Werte für y_i^{target} direkt in die Spalte „Target“ der Tabelle „Output“ eingetragen werden. Alternativ können die Target-Werte, bei Aktivierung des Kontrollkästchens „Accept as Target“ aus den Schiebereglerdiagrammen in die Output-Tabelle übernommen werden.

Wird kein konkreter Wert der Prozessausgangsgröße als „Target“ vorgegeben, kann die Richtung der Optimierung durch Angabe von „Max“ oder „Min“ bestimmt werden. Für jede Prozesseingangs- und -ausgangsgröße kann der bei der Optimierung einzuhaltende Wertebereich durch Editieren der Spalten „Lower Limit“ und „Upper Limit“ angegeben werden.

Sind die Startrezeptur, die einzuhaltenden Wertebereiche der Prozessgrößen sowie die Optimierungsziele bestimmt, wird entsprechend der Darstellung im Abschnitt 6.5.2[89] durch Modellinvertierung eine Rezeptur generiert, die diesen Anforderungen genügt.

Im vorliegenden Beispiel seien durch MOMBES Rezepturen vorzuschlagen, die näherungsweise pareto-optimal bezüglich der Minimierung der Zielfunktionen: $f_1, f_2', .., f_5'$ seien und den in Tabelle 8.4 dargestellten Bedingungen genügen.

Prozessgröße:	Preis	Festigkeit	Fettbindung	VitaminProt	Textur
Zielfunktion:	f_1	f_2	f_3	f_4	f_5
Optimierungsziel:	Minimieren	$f_2 \geq 24{,}0$	$f_3 \geq 16{,}0$	$f_4 \geq 6{,}0$	$f_5 \geq 40{,}0$

Tabelle 8.4: Optimierungsziele

Die Startrezeptur wurde hier durch Auswahl des 39-ten Elements aus der durch MOMBES generierten Approximation der Pareto-Front gewonnen. Um den Einfluß der Schwankungen der Rohmaterialienpreise auf die Gesamtkosten der Rezeptur zu illustrieren, wurden die Berechnungen auf Basis der Einzelpreise sowohl der Spalte „Preis-1“ als auch der Spalte „Preis-2“ aus Tabelle 8.2 durchgeführt.

Es sind also Rezepturen zu finden, die möglichst geringe Kosten verursachen und über folgende Einschränkungen bezüglich der Qualitätsgrößen verfügen: die Festigkeit der Rezeptur soll größer gleich 24% sein, ihre Fettbindung größer gleich 16%, ihr Vitamin-Protein-Anteil größer gleich 6% und ihre Textur größer gleich 40%.

In Tabelle 8.5 sind die Ergebnisse der Optimierung sowohl Basis von Preis-1 als auch auf Basis von Preis-2 dargestellt. Im folgenden sollen beispielhaft nur einige Schlußfolgerungen und

[89] An dieser Stelle soll aus Gründen der Vereinfachung zur Berechnung der Zielfunktionswerte nicht das finale Pareto-Modell, sondern das in Abschnitt 8.2.3 dargestellte Prozessmodell verwendet werden.

Kommentare zur den optimierten Rezepturen gegeben werden, die im Rahmen dieser Arbeit natürlich nicht vollständig sein können.

Es ist zu erkennen, daß die Startrezeptur noch nicht die Qualitätskriterien einhält: die Fettbindung beträgt hier nur 12,15% und liegt damit bezüglich des Wertebereichs dieser Prozessgröße um fast 20% unter dem geforderten Wert von 16,0%. Dagegen sind die Qualitätskriterien Festigkeit, Protein-Vitamin-Gehalt und Textur deutlich über das geforderte Maß erfüllt. Die Kosten der Startrezeptur betragen bezüglich der Rohmaterialienpreise 79,09 Ct bei Preis-1 und 76,84 Ct entsprechend Preis-2.

		Start-rezeptur	Rezeptur1: Optimierung auf Basis von Preis-1		Rezeptur2: Optimierung auf Basis von Preis-2	
Prozessgröße	**Werte-bereich**	**Wert**	**Wert**	**Abweichung zum Startwert**	**Wert**	**Abweichung zum Startwert**
Zucker	[8,0 ; 10,0]	9,54	8,97	-28,50%	8,97	-28,50%
S-Oel	[5,0 ; 7,0]	5,83	5,83	0,00%	5,83	0,00%
Wasser	[16,5 ; 54,5]	31,65	33,8	5,66%	36,32	12,29%
In_Schwein	[20,0 ; 40,0]	38,12	28,9	-46,10%	28,6	-47,60%
W-Mehl	[0,0 ; 10,0]	5,75	6,89	11,40%	9,28	35,30%
M-Mehl	[0,0 ; 10,0]	3,29	9,9	66,10%	4,9	16,10%
Emulgatoren	[5,0 ; 10,0]	5,45	5,75	6,00%	5,75	6,00%
Vitamine	[0,0 ; 5,0]	0,32	0,05	-5,40%	0,05	-5,40%
Festigkeit	[7,5 ; 33,5]	25,23	24,2	-3,96%	24,14	-4,19%
Fettbindung	[8,5 ; 30,0]	12,15	16,07	18,23%	16,8	21,63%
ProtVit	[3,8 ; 58,8]	9,11	11,18	3,76%	7,77	-2,44%
Textur	[21,8 ; 77,4]	67,31	40,02	-49,08%	69,17	3,35%
Konfidenz	[0,0 ; 1,0]	0,998	0,844	-15,40%	0,977	-2,10%
Preis1 in Ct	[61,0 ; 152,0]	79,09	70,671	-6,19%		
Preis2 in Ct	[61,0 ; 152,0]	76,84			66,63	-11,22%

Tabelle 8.5: Ergenisse der Optimierung entsprechend der Ziele aus Tabelle 8.4

Bei der Optimierung auf Basis von Preis-1 ist durch MOMBES eine Rezeptur (Rezeptur1) generiert worden, bei der das Verhältnis der Rohmaterialien gegenüber der Startrezeptur so angepaßt wurde, daß der Anteil von Wasser um 5,66% erhöht, der Anteil an Schweine-Innereien drastisch um 46,1% gesenkt und der Anteil von Weizen- und Maismehl ebenfalls erheblich um ca. 11,4% und 66,1% erhöht wurde. Der Anteil des Vitamin-Mixes wurde um ca. 5,4% gesenkt, wobei der Anteil der Emulgatoren wiederum um 6% erhöht wurde. Auf diese Weise wurde eine deutlich billigere Rezeptur (die Preisreduzierung liegt bei fast 6,2%) gefunden, die dennoch allen geforderten Qualitätskriterien entspricht. Die drastische Reduktion des Anteils von Schweine-Innereien ist darauf zurückzuführen, daß diese Rohmaterialie mit einem Preis von 0,83€ pro Kg relativ teuer ist. Offensichtlich kann das billigere Mehl ebenso zu einer Erhöhung der Festigkeit und der Fettbindung führen. Sein Anteil wurde daher dramatisch erhöht. Die veränderte Mischung führt zwar gegenüber der Startrezeptur zu Qualitätseinbußen bezüglich der Festigkeit (-3,96%) und der Textur (-49,08%), konnte jedoch bezüglich des Protein-Vitamin-Anteils (+3,76%) und der Fettbindung (+18,23%) zu einer Qualitätssteigerung führen. Die Werte aller Prozessausgangsgrößen liegen innerhalb der geforderten Toleranzen und (erwartungsgemäß) eher an den unteren Grenzwerten der zulässigen Wertebereiche.

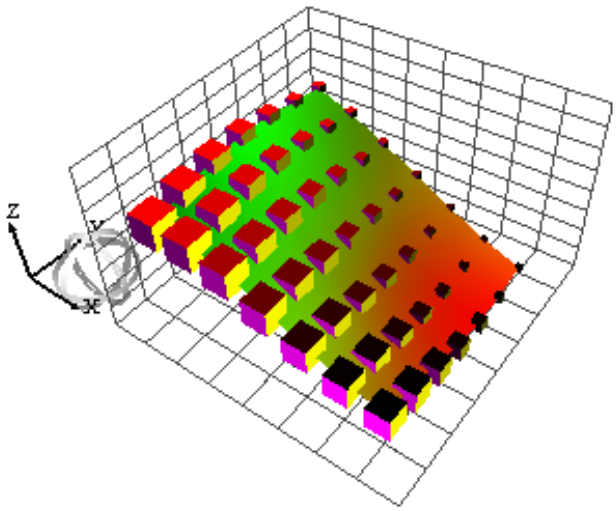
Die Optimierung auf Basis der Kostenberechnung mit Preis2 führt prinzipiell zu ähnlichen Ergebnissen (Rezeptur2) wie die, die durch die Optimierung auf Basis von Preis1 erreicht wurden. Jedoch ist hier zu beobachten, daß der Wasser-Anteil gegenüber Rezeptur1 nochmals erhöht wurde und hier bei ca. 36,22% liegt. Die Verringerung der Festigkeit von Rezeptur2 auf

Grund des hohen Wasseranteils wurde auch hier durch eine Erhöhung des Mehl-Anteils kompensiert, wobei der etwas geringer ist als bei Rezeptur1. Das ist damit zu erklären, daß Weizen-Mehl bessere Bindungseigenschaften besitzt als Mais-Mehl und damit zu Erreichung der gleichen Festigkeit (ca. 24,14%) insgesamt weniger Mehl eingesetzt werden muß, wenn der Anteil von Weizen-Mehl hoch genug ist. Da der Einzelpreis für Weizen-Mehl bei Preis1 höher als der bei Preis2 ist, kann also bei gleichem Preis bei Rezeptur2 mehr Weizen-Mehl eingebracht werden. Dies führt insgesamt nochmals zu einer Preisreduktion der gesamten Rezeptur. Dagegen ist der Vitamin-Protein-Anteil bei Rezeptur2 gegenüber Rezeptur1 leicht gefallen (von 9,11% auf 7,77%), was damit begründet ist, daß auch in Mehl viele Vitamine enthalten sind und durch seinen höheren Anteil in Rezeptur1 mehr Vitamine in die Mischung gelangen.

Die Ergebnisse der Optimierung zeigen, daß es zum einen möglich ist, bei Einhaltung von Grenzwerten der Qualtitätskriterien eine Kostenreduktion der Rezepturen zu erreichen. Darüber hinaus wird deutlich, daß es mit diesem System gelingt, Schwankungen innerhalb der Rohmaterialienpreise auszugleichen, indem teure Rohmaterialien durch die kostengünstigeren Materialien mit ähnlicher Wirkung auf die Produktqualität ersetzt werden. Illustriert wird dies am Beispiel der Optimierung auf Basis von Preis-1 und Preis-2 durch die Veränderung des Anteils von Weizen- und Mais-Mehl bei unterschiedlichen Preisen dieser Rohmaterialien.

8.2.7 Qualitative Analyse der Approximation der Pareto-Menge und Pareto-Front

An dieser Stelle werden die qualitativen Zusammenhänge zwischen den Prozesseingangs- und Prozessausgangsgrößen im nicht-dominierten Bereich, nach der in Abschnitt 6.5.1 dargestellten Methode der qualitativen Clusteranalyse mit Hilfe einer Kohonenkarte untersucht.



Festigkeit: Höhe der Oberfläche

Preis: Größe der Würfel

W-Mehl: grüne Oberfläche

Wasser: rote Oberfläche

M-Mehl: mangenta-farbige Würfelfläche

Vitamine: gelbe Würfelfläche

In_Schwein: rote Würfelfläche

Abbildung 8.19: Mit P^* und PF^* trainierte Kohonenkarte

Abbildung 8.7 zeigt die Visualisierung der verwendeten Kohonenkarte ($7 \times 7 = 49$ Neurone mit gaussförmiger Erregungsfunktion in einer Nachbarschaft von 5 Neuronen), die in 500 Schritten mit einer Lernrate von $\alpha_{START} = 0{,}05$ mit den Vektoren aus P^* und PF^* belehrt wurde.

Folgende Information kann auf Basis der Interpretation der Kohonenkarte mit den in Abbildung 8.7 (rechts) dargestellten Zuordnungen der Kartenattribute zu den Prozessgrößen gewonnen werden:

- Die Höhe der Kartenoberfläche ist dort stark ausgeprägt, wo eine grüne Färbung der Kartenoberfläche und eine rote Färbung der Würfelflächen auftritt sowie die Kartenoberfläche nicht rot gefärbt ist. Daraus folgt: *große Festigkeit korreliert mit einem hohen Anteil an Weizen-Mehl und In_Schwein sowie einem geringen Anteil an Wasser.*
- Die Höhe der Kartenoberfläche ist dort eher gering ausgeprägt, wo die Würfelflächen mangenta-farbig sind. Dabei treten die mangenta-farbigen Würfelflächen alternativ zu den rot gefärbten Würfelflächen auf. Daraus folgt: *ein hoher Anteil an Mais-Mehl bewirkt eher eine geringe Festigkeit. Wird Weizen-Mehl durch Mais-Mehl ersetzt, verringert sich die Festigkeit des Produkts, d.h. die Wirkung von Mais-Mehl auf die Festigkeit ist geringer als die von Weizen-Mehl.*
- Die Würfel sind überall dort am größten, wo die Würfelflächen gelb gefärbt sind, und dort am kleinsten, wo die Kartenoberfläche niedrig und stark rot gefärbt ist sowie die Würfelflächen nicht rot und nicht mangenta gefärbt sind. Daraus folgt: *ein hoher Anteil des Vitamin-Mixes verteuert die Rezeptur. Billige Rezepturen weisen einen hohen Wasseranteil, einen geringen Anteil von Mais- und Weizen-Mehl sowie Vitaminen auf und besitzen eine geringe Festigkeit.*

An dieser Stelle wurden beispielhaft nur einige Aspekte der Wirkung der Prozesseingangs- auf die Prozessausgangsgrößen sowie die Zusammenhänge der Prozessgrößen untereinander im näherungsweise pareto-optimalen Bereich dargestellt. Es wurden dabei die Wechselwirkungen von 7 Prozessgrößen gleichzeitig visualisiert. Die Interpretationen der Kohonenkarte konnten dabei die Überlegungen des Gedankenexperiments (siehe S. 155) sowie die Ergebnisse der Parametervariation des Prozessmodells (siehe S. 153) bestätigen. An dieser Stelle kann die qualitative Analyse natürlich nicht umfassend sein und lediglich die Arbeitsweise von MOMBES zur qualitativen Analyse der durch MOMBES generierten Approximation der Pareto-Menge und der durch sie bestimmten Approximation der Pareto-Front illustrieren.

8.2.8 Zusammenfassung

Das beschriebene Gesamtsystem zur modellgestützten Mehrzieloptimierung konnte erfolgreich zur Modellbildung und Rezepturoptimierung in der Tierfutterherstellung eingesetzt werden. Die Trainings- und Optimierungszeiten lagen durch den Einsatz des beschriebenen Verfahrens im Bereich weniger Minuten.

Die Modellierung der Abhängigkeiten zwischen 150 Prozesseingangs- und 30 Prozessausgangsgrößen erfolgte auf Basis von 500 gemessenen Rezepturen, wobei ca. 20% der Komponenten der Rezepturen nicht vorhanden waren und die gemessenen Prozessgrößen einen Rauschanteil von ca. 4% aufwiesen. Bei der Validierung (siehe 8.2.3) des Modells (Training mit 80% zufällig entnommenen Rezepturen und Test mit den verbliebenen 20%) wurde beim Test-Set ein mittlerer, absoluter Fehler ca. 8% erreicht. Bei einer zulässigen Abweichung von 10% des Wertebereichs der jeweiligen Prozessgröße wurden 87% der Prozessausgangsgrößen der Rezepturen des Test-Set durch das Prozessmodell korrekt prognostiziert. Bei der Optimierung auf Basis einer linearen Kostenfunktion und Einhaltung von ca. 50 Randbedingungen (hier wurde der Wertebereich der Prozessgrößen gegenüber dem durch die Trainingsdaten bestimmten Intervall stark eingeschränkt) konnte eine Kostensenkung um ca. 5% erreicht werden.

9 Schlußbetrachtungen

In diesem Kapitel werden die im Rahmen der vorliegenden Arbeit erzielten Ergebnisse unter dem Aspekt der in Kapitel 1 formulierten Ziele zusammengefaßt und ein Ausblick auf mögliche weitere Forschungsziele gegeben.

9.1 Zusammenfassung

In dieser Arbeit wurde ein neues System (MOMBES) zur modellbasierten Mehrzieloptimierung kontinuierlicher multikriterieller Optimierungsprobleme auf der Basis Evolutionärer Algorithmen und Neuronaler Netze entworfen. MOMBES setzt sich dabei aus drei, aufeinander aufbauenden Modulen zusammen:

- einem Modul zur Prozessmodellbildung, das eine modellbasierte Berechnung der Zielfunktionen und Randbedingungen des zu optimierenden und durch eine Menge von Beispielsituationen beschriebenen Prozesses realisiert,
- einem Optimierungskern zur Generierung einer Menge von näherungsweise pareto-optimalen Kompromißlösungen zur simultanen Minimierung dieser Zielfunktionen unter Berücksichtigung der Einhaltung einer Menge von Randbedingungen sowie
- einem interaktiven Decision-Making-Modul zum kontinuierlichen Zugriff auf die durch den Optimierungskern generierten Kompromißlösungen, das auch eine Extraktion von qualitativen Zusammenhängen zwischen den Zielfunktionen und Entscheidungsvariablen im näherungsweise pareto-optimalen Bereich gestattet.

Dabei lag der Schwerpunkt der hier dargestellten Arbeiten auf der Entwicklung eines effizienten, selbstadaptierenden, multikriteriellen Optimierungsalgorithmus sowie der Erarbeitung eines modellbasierten Decision-Making-Moduls, das sowohl einen kontinuierliche Lösungszugriff als auch das interaktive Extrahieren von Wissen über diejenigen Zusammenhänge zwischen den Prozessgrößen gestattet, die zur Herausbildung pareto-optimaler Lösungen führen. Darüber hinaus wurde die praktische Anwendbarkeit des entwickelten Verfahrens am Beispiel einer Applikationen aus dem industriellen Umfeld gezeigt.

Das entwickelte System wurde anhand künstlicher, multikriterieller Testprobleme validiert und mit bisher bekannten Evolutionären Algorithmen zur multikriteriellen Optimierung verglichen. Bei diesen Vergleichen konnte die große Leistungsfähigkeit von MOMBES dargestellt werden. Dabei zeigte sich, daß es durch die implementierten Mechanismen der Selbstanpassung innerhalb des Optimierungskerns gelungen ist, die Optimierung gezielt in Richtung homogen und breit verteilter Pareto-Front zu beeinflussen und auf diese Weise ein Lernen der Evolutionsrichtung zu realisieren. Weiter konnte dargestellt werden, daß es mit Hilfe der hier entwickelten Modelle innerhalb des Decison-Making-Moduls möglich war, die Zusammenhänge innerhalb des pareto-optimalen Bereichs des Lösungs- und Zielraums zu modellieren und dieses Modellwissen für den Anwender zur Verfügung zu stellen.

Im Gegensatz zu den bisher entwickelten multikriteriellen Evolutionären Algorithmen gestattet der in dieser Arbeit entwickelte Ansatz zum einen eine Extraktion von Wissen aus der generierten Approximation der Pareto-Menge und Pareto-Front und nutzt zum anderen die Eigenschaft lernfähiger Neuronaler Netze zur Beeinflussung der Suchrichtung des Optimierungsalgorithmus. Die zusätzliche Kombination mit einem gradientenbasierten Optimierungsalgorithmus machen den Ansatz zu einem Multi-Hybrid-System, das sich durch sehr gute Konvergenzeigenschaften und eine hohe Qualität der generierten Pareto-Front auszeichnet. Als Besonderheit dieses Ansatzes sei die Hybridisierung von ES und NN als Kombination von adaptiven, lernfähigen Methoden der CI genannt. Dabei wurde die

Modellierung der Verteilung der Elemente der Pareto-Front in Form der Belehrung einer Selbstorganisierenden Merkmalskarte während der Optimierung und die Modellierung der Zusammenhänge, die zur Herausbildung pareto-optimaler Lösungen führen, nach Abschluß der Optimierung als Unterstützung für das Decision-Making neu entwickelt. Der beschriebene Ansatz weist folgende Vorteile auf: Der Benutzer

- wird nicht gezwungen, vor dem Optimierungslauf eine tatsächlich in der Praxis schwer zu realisierende numerische Gewichtung der einzelnen Zielfunktionen vorzunehmen, sondern
- kann die Zielvorgaben auch nach der Optimierung variieren, d.h. mehrere Szenarien simulieren. Die interaktive Rolle des Benutzers wird hervorgehoben und die Zielvorgaben werden unscharf formulierbar.
- Weiterhin ist der Anwender in der Lage, Aussagen über die Zielrichtung der einzelnen Optimierungsziele zu formulieren und so konkurrierende und nicht konkurrierende Zielfunktionen zu identifizieren.

Zudem wurden neue Ansätze für die Visualisierung der Zusammenhänge zwischen den einzelnen Zielfunktionen bei multikriteriellen Optimierungsproblemen mit höherdimensionalen Zielfunktionsvektor vorgeschlagen. Die im Rahmen des Decision-Making-Moduls von MOMBES erzeugten Modelle bilden die Basis einer einfachen und für den Praktiker nützlichen Visualisierung der Zusammenhänge zwischen den einzelnen Zielfunktionen innerhalb der generierten Approximation der Pareto-Front. Darüber hinaus wird durch die Anwendung selbstorganisierender Merkmalskarten eine interaktive, qualitative Analyse der Zusammenhänge zwischen denjenigen Prozessgrößen ermöglicht, die die Herausbildung pareto-optimaler Lösungen bestimmen.

Im Rahmen der durchgeführten Benchmarks wurde ein neues Gütemaß eingeführt, das die Qualität von Approximationen einer Pareto-Front hinsichtlich der Pareto-Dominanz ihrer Elemente und deren Verteilung im Zielraum berücksichtigt. Dieses Gütemaß, das als Kombiniertes-Dominanz-Ausdehnungskriterium bezeichnet wird, erlaubt im Gegensatz zu den bisher bekannten Gütekriterien die gleichzeitige Charakterisierung der Elemente einer nicht-dominierten Front sowohl hinsichtlich der Annäherung ihrer Elemente zur globalen Pareto-Front als auch ihrer Anordnung im Zielraum.

Zusammenfassend wird festgestellt, daß es mit dem hier vorgestellten System MOMBES gelungen ist, ein Gesamtkonzept zur modellbasierten Mehrzieloptimierung vorzustellen, das seine Stärken sowohl bei der Lösung künstlicher Testprobleme, als auch bei der Anwendung zur Lösung konkreter Probleme aus der industriellen Praxis zeigt.

9.2 Ausblick

An dieser Stelle werden einige Anregungen für weiterführende Forschungen gegeben, die ausgehend von den hier erzielten Resultaten besonders naheliegend erscheinen.

Multikriterielle Optimierung diskreter Zielfunktionen: Die hier dargestellten Arbeiten beziehen sich ausschließlich auf die multikriterielle Optimierung kontinuierlicher Zielfunktionen. Ein interessantes Thema stellt die Erweiterung der hier dargestellten Ansätzen auf diskrete Probleme z.B. der Ablaufplanung dar. Erste Simulationen bei der Lösung von multikriteriellen Scheduling-Problemen[90] haben gezeigt, daß mit dem Optimierungskern von MOMBES auch ohne die Modellierung der aktuellen, nicht-dominierten Zielfunktionswertevektoren[91] Resultate

[90] Eine Übersicht der erzielten Ergebnisse findet sich in [Meyer, 2000].

[91] d.h. ohne die Funktionalität des Einfügens approximierter Individuen

erzielt wurden, die nur geringfügig unter denen moderner MOEA wie z.B. SPEA2 lagen und hochwertiger als die mit Standardverfahren des Operation Research erzielten Ergebnisse waren. Es wäre daher zu prüfen, inwieweit eine Modellierung diskreter Verteilungen von Zielfunktionswerten möglich und damit die Lösungskonzeption von MOMBES auf diskrete multikriterielle Probleme übertragbar ist, bzw. welche Anpassungen innerhalb von MOMBES notwendig sind, um effizient diskrete multikriterielle Optimierungsprobleme zu lösen.

Erweiterungen des Prozessmodells: Das hier vorgeschlagene Modell zur datengetriebenen Abbildung der Prozesseingangs- auf die Prozessausgangsgrößen zur Berechnung der Zielfunktionen und Randbedingungen stellt ein relativ einfaches Modell dar. Es wäre sinnvoll, dieses Modell derart zu erweitern, daß ergänzend u.U. vorhandenes, regelbasiertes Wissen über die Zusammenhänge zwischen den Prozessgrößen berücksichtigt werden könnte. Dies könnte realisiert werden, indem das Gesamtmodell aus Teilmodellen aufgebaut ist, für die jeweils die am besten geeignete Modellstruktur ausgewählt wird[92]. Dabei könnten mathematische Zusammenhänge ebenso wie in Form von Fuzzy-Regeln dargestelltes regelbasiertes Wissen berücksichtigt werden.

Auffinden unterrepräsentierter Bereiche im hochdimensionalen Datenraum: Es ist zu prüfen, ob das hier im Rahmen der Funktionalität des Einfügens approximierter Individuen vorgeschlagene Verfahren zum Auffinden von Bereichen geringer Dichte innerhalb einer Menge höherdimensionaler Vektoren[93] mittels selbstorganisierender Merkmalskarten genutzt werden kann, um generell Bereiche geringer Datendichte im hochdimensionalen Datenraum zu identifizieren. Damit könnte ein neuer Weg in Richtung modellbasierter Versuchplanung eingeschlagen werden.

Visualisierung höherdimensionaler Pareto-Fronten: Mit der hier vorgeschlagenen Anwendung von selbstorganisierenden Merkmalskarten und Teilmodellen zur Abbildung der Zusammenhänge zwischen den Zielfunktionen und Entscheidungsvariablen im pareto-optimalen Bereich ist ein weiterer Schritt in Richtung Visualisierung höherdimensionaler Zusammenhänge innerhalb der Approximationen der Pareto-Menge und Pareto-Fronte gemacht worden. Wie dargestellt, ist die Invertierung dieser Teilmodelle für multikriterielle Optimierungsproblem mit mehr als zwei Zielfunktionen nicht mehr eindeutig. Es ist zu prüfen, welche zusätzlichen Möglichkeiten entwickelt werden könnten, um den Anwender bei der Auswahl eines Punktes auf der aktuellen Approximation der Pareto-Front zu unterstützen und damit das Decision-Making anwenderfreundlicher zu gestalten.

[92] Einen solchen Ansatz zur lokalen Modellbildung entwickelte M.Sturm mit seinem LEMON-Algorithmus in [Sturm, 2000].

[93] Hier stellt diese Menge die aktuelle Approximation der Pareto-Front dar.

Anhang A

A.1 Ausgewählte Abkürzungen

DM	**D**ecision-**M**aker
FPM	**F**inales **P**areto-**M**odell
HLGA	**H**ajela and **L**ins **G**enetic **A**lgorithm
LVQ	**L**earning **V**ector **Q**uantization
MLP	**M**ulti **L**ayer **P**erceptron
MOEA	**M**ulti**o**bjective **E**volutionary **A**lgorithm
MOES	**M**ulti**o**bjective **E**volutionary **S**trategy
MOGA	**M**ulti**o**bjective **G**enetic **A**lgorithm
MOMBES	**M**ulti **O**bjective **M**odel **B**ased **E**volution **S**trategy
MOP	**M**ultiobjective **O**ptimization **P**roblem
NN	**N**euronales **N**etz
NPGA	**N**iched **P**areto **G**enetic **A**lgorithm
NSGA	**N**ondominated **S**orting **G**enetic **A**lgorithm
PESA	**P**areto **E**nvelope-based **S**election **A**lgorithm
RBF	**R**adial **B**asis **F**unction
SOM	**S**elf **O**rganizing **M**aps (selbst organisierende Merkmalskarte)
SOP	**S**ingle **O**ptimization **P**roblem
SPEA	**S**trength **P**areto **E**volutionary **A**lgorithm
VEGA	**V**ector **E**valuated **G**enetic **A**lgorithm

A.2 Ausgewählte Symbole

$X_{feasible}$	Menge der zulässigen Lösungen eines MOP´s
$Y_{feasible}$	Menge der zulässigen Zielfunktionswerte eines MOP´s
$\vec{x} = [x_1,.., x_n]$	Vektor der Entscheidungsvariablen
$\vec{f}(\vec{x}) = [f_1(\vec{x}),.., f_k(\vec{x})]$	Vektor der Zielfunktionen
$\vec{y} = [y_1,.., y_k]$	Vektor der Zielfunktionswerte mit $y_i = f_i(\vec{x})$
P_{true}	Pareto-Menge, Menge aller Lösungen eines MOP´s
PF_{true}	durch P_{true} bestimmte Pareto-Front
P^*	Approximation der Pareto-Menge
PF^*	durch P^* bestimmte Approximation der Pareto-Front
$\vec{y}^p = [y_1^p,..., y_k^p]$	Zielfunktionswertevektor $\vec{y}^p \in PF^*$
$\vec{y}^{ideal} = [f_1^*, f_2^*,.., f_k^*]$	Idealvektor, Vektorkomponenten sind die Minimalwerte der einzelnen Zielfunktionen
$\vec{x}^{ideal}$	Ideallösung, $\vec{f}(\vec{x}^{ideal}) = \vec{y}^{ideal}$

$\vec{x}^B$	best-over-all-Lösung
$\vec{a}$	Individuum
Γ	Dekodierungsfunktion, $\vec{x} = \Gamma(\vec{a})$
$\Phi(\vec{a})$	Fitness des Individuums $\vec{a}$
$p(\vec{a})$	Straffunktion des Individuums $\vec{a}$
ρ	Anzahl der an der Rekombination beteiligten Eltern
λ	Anzahl der Nachkommen pro Generation
μ	Anzahl der Eltern pro Generation
q	Anzahl der Kandidaten bei Wettkampfselektion
$\vec{\sigma}$	Mutationsschrittweiten bei ES
t	Nummer der aktuellen Generation
$t_{\max}$	maximale Anzahl der Generationen
$P(t)$	Population nach t Generationen
$\overline{P}(t)$	externe Elitepopulation nach t Generationen
$N = \lvert P(t) \rvert$	Populationsgröße: Anzahl der überlebenden Individuen pro Generation
$\overline{N} = \lvert \overline{P}(t) \rvert$	Größe der externen Elitepopulation
$n_{function\_call}$	Anzahl der Aufrufe zur Berechnung des Zielfunktionswertevektors
σ_{share}	Nischenradius beim sharing
t_{dom}	Größe der Vergleichsmenge beim NPGA; Angaben in Prozent
p_c	Wahrscheinlichkeit für das Anwenden des Rekombinationsoperators
p_m	Wahrscheinlichkeit für das Anwenden des Mutationsoperators
p_{approx}	Parameter, der die Wahrscheinlichkeit für das Einfügen approximierter Individuen bestimmt
$\vec{w}^p$	Pseudo-Gewichtsvektor für ein Element von PF^*
$\vec{w}^{Ziel}$	Zielgewicht für die Optimierung in Richtung Ecke oder Bereich geringster Dichte von PF^*
$\vec{\hat{y}}$	Modelloutput
$\vec{y}_{Ni} = [y_1,..,y_{i-1},y_{i+1},..,y_k]$	Zielfunktionswertevektor mit den komplementären Komponenten zu y_i
MOD_i	Modell zur Berechnung von y_i in Abhängigkeit von $\vec{y}_{Ni}$
δ^{mod_i}	Vorhersagefehler von MOD_i
δ_i^{FPM}	Vorhersagefehler des FPM für die i-te Komponente von $\vec{\hat{y}}$
$\vec{y}^{target}$	Target-Vektor
$\vec{x}^{target}$	Target-Lösung
δ^{mod_i}	Vorhersagefehler der MOD_i
δ_i^{FPM}	Vorhersagefehler der FPM für die i-te Komponente von $\vec{\hat{y}}$

$\delta_i^{INV\_FPM}$ Invertierungsfehler des FPM für die i-te Komponente von $\vec{y}^{target}$

$\delta^{INV\_mod\_i}$ Invertierungsfehler von MOD_i

A.3 Internet-Informationsquellen

- Für Einsteiger in die Problematik von EA besonders geeignet: EvoWeb (European Network of Excellence in Evolutionary Computing): http://evonet.dcs.napier.ac.uk/
- Liste von Veröffentlichungen zur Kombination von EA und NN, Zugang über: gann-request@cs.iastate.edu
- Handbook of Evolution Computation: http://www.iop.org/Books/CIL/HEC/
- Parallel Problems Solving from Nature: http://ls11-www.informatik.uni-dortmund.de/PPSN/
- Bibliographische Zusammenstellung vieler wichtiger Aufsätze zm Thema MOEA (auch zum download geeignet: http://www.lania.mx/~ccoello/EMOO/EMOObib.html
- frei erhältlicher Simulator für NN: Stuttgart Neural Network Simulator (SNNS): www.informatik.uni-stuttgart.de/ipvr/bv/projekte/snns/snns.html
- Software und Dokumentation für LVQ und SOM: http://www.cis.hut.fi/research/som_pak/ und http://www.cis.hut.fi/research/lvq_pak/
- Umfassende und ständig aktualisierte Übersicht der wichtigsten mathematischen Methoden und Begriffe: http://mathworld.wolfram.com/

Konferenzen zu MOEA:

- First International Conference on Evolutionary Multi-Criterion Optimization (EMO‘01) 2001 in Zürich: http://www.tik.ee.ethz.ch/emo/
- Second International Conference on Evolutionary Multi-Criterion Optimization (EMO'03) 2003 in Portugal: http://conferences.ptrede.com/emo03

Testdaten zu MOEA:

- Künstliche kontinuierlich-reellwertige Optimierungsprobleme: http://www.tik.ee.ethz.ch/~zitzler/
- Scheduling- und andere OR-Probleme: http://mscmga.ms.ic.ac.uk/info.html

10Literatur

[Angeline, 1993]: Angeline, P.J.: Evolutionary Algorithms and Emergent Intelligence, Dissertation, Ohio State University, Columbus 1993

[Bäck et al, 1997]: Bäck, T.; Fogel, D.; Michalewicz, Z. (Hrsg.): Handbook of Evolutionary Computation, Oxford University Press, Oxford 1997

[Bäck und Schwefel, 1993]: Bäck, T.; Schwefel, H.P.: An Overview of Evolutionary Algorithms for Parameter Optimization, in: Evolutionary Computation 1(1) 1993

[Bäck, 1994]: Bäck, T.: Evolutionary algorithms in theory and practice, Dr.rer.nat. Dissertation, Universität Dortmund, Fakultät für Computer Science, Februar 1994

[Bäck, 1996]: Bäck, T.: Evolutionary Algorithms in Theory and Practice, Oxford University Press, New York, 1996

[Battiti und Tecchilio, 1995]: Battiti, R.; Tecchilio, G.: Local Search with memory: Benchmarking RTS, in: OR Spektrum 17 (1995) 2/3, S.67-86

[Ben-Tal, 1980]: Ben-Tal, A.: Characterization of Pareto and Lexicographic Optimal Solutions, in: Fandel, G.; Gal. T. (Hrsg.): Multiple Criteria Decision Making - Theory and Applikation 177, Lecture Notes in Economics and Mathematical Systems, S.1-11, Springer Verlag, 1980

[Biethahn und Nissen, 1995]: Biethahn, J.; Nissen, V. (Hrsg.): Evolutionary Algorithms in Management Applications, Springer Verlag, Berlin1995

[Blickle, 1996]: Blickle, T.: Theory of Evolutionary Algorithms and Application to System Synthesis, Ph.D.Thesis, Swiss Federal Institute of Technology, Zürich 1996

[Box, 1957]: Box, G.E.P.: Evolutionary Operation: A Method of Increasing Industrial Productivity, in: Applied Statistics, A Journal of the Royal Statistical Society 6(1957), S. 81-101, 1957

[Box, 1965]: Box, M.J.: A new method of constrained optimization an a comparison with other methods, Computer Journal 8, 1965

[Bremer, 1998]: Bremer, J.H.: Beschleunigte Evolutionsstrategie zur Optimierung von Fertigungsprozessen, Dr.Ing. Dissertation, Technische Universität Berlin, Fachbereich Maschinenbau und Produktionstechnik, Berlin 1998

[Brewka et al, 1999]: Brewka, G; Der, R.; Gottwald, S.; Schierwagen, A. (Hrsg.): Fuzzy-Neuro-Systems 1999, Leipziger Universitätsverlag, 1999

[Broekmeulen, 1995]: Broekmeulen, R.A.C.M.: Facility Management of Distribution Centres for Vegetables and Fruits, in [Biethahn und Nissen, 1995], S. 199-210

[Bronstein und Semendjajew, 1978]: Bronstein, I.N.; Semendjajew. K.A.: Taschenbuch der Mathematik, VEB Bibliographisches Institut Leipzig, 1978

[Büche et al, 2002]: Büche, D.; Milano. M.; Koumoutsakos. P.: Self-Organizing Maps for Multi-Objective Optimization, in: Barry, A.M. (Hrsg.): GECCO 2002: Proceedings Genetic and Evolutionary Computation Conference, S.152-155, AAAI, New York, 2002

[Butz, 1997]: Butz, D.: Neuronale Funktionsapproximation mit RBF-Schwerpunktnetzen, Shaker Verlag Aachen, 1997

[Chang, 1995]: Chang, C.S.: Genetic Algorithm Based Bicriterion Optimization for Traction Substations in DC Raylway Systems, in: [Fogel, 1995], S. 11-16, 1995

[Coello, 1996]: Coello, C.A.C.: An Empirical Study of Evolutionary Techniques for Multiobjective Optimization in Engineering Design, Ph.D. Thesis, Department of Computer Science, Tulane University, New Orleans, April 1996

[Coello, 1999]: Coello, C.A.C.: List of references on Evolutionary Multiobjective Optimization: http://www.lania.mx/~ccoello/EMOO/EMOObib.html

[Corne et al, 2000]: Corne, D. W.; Knowles, J. D.; Oates, M.J.: The Pareto Envelope-based Selection Algorithm for Multiobjective Optimization, in: Schoenauer, M.; Deb, K.; Rudolph, G.; Yao, X.; Lutton, E.; Merelo, J.J.; Schwefel, H.-P. (Hrsg.): Proceedings of the Parallel Problem Solving from Nature VI Conference, S. 839-848, Springer, 2000

[Cunha et al, 1997]: Cunha, A.G.; Oliviera, P.; Covas, J.: Use of Genetic Algorithms in multicriteria optimization to solve industrial problems, in: Proceedings of the 7. International Conference on Genetic Algorithms, S. 682-688, 1997

[Davidor et al, 1994]: Davidor, Y.; Schwefel, H.-P.; Männer, R. (Hrsg.): Parallel problem solving from nature 3, Proceedings of the 3rd PPSN Conference, vol. 866 of Lecture Notes in Computer Science, Springer, Berlin 1994

[Davis, 1991a]: Davis, L. (Hrsg.): Handbook of genetic algorithms, Van Nostrand Reinhold, New York, 1991

[Davis, 1991b]: Davis, L.: Genetic Algorithms Tutorial, in: [Davis, 1991a], S.1-101

[De Jong und Spears, 1993]: De Jong, K.A.; Spears, W.: On the state of evolutionary computation, in: [Forrest_1993], S. 618-623, 1993

[De Jong, 1975]: De Jong, K.A.: An Analysis of the Behavior of a Class of Genetic Adaptive Systems, Ph.D. Thesis, University of Michigan, 1975

[Deb und Goel, 2001]: Deb, K; Goel, T: A Hybrid Multi-Objective Evolutionary Approach to Engineering Shape Design, in: [EMO, 2001], S. 385-399, 2001

[Deb und Goldberg, 1989]: Deb, K.; Goldberg, D.E.: An Investigation of Niche and Species Formation in Genetic Function Optimization, in: [Schaffer, 1989], S.42-50

[Deb, 1999]: Deb, K.: Multi-objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems, Evolutionary Computation 7(3), S. 205-230, 1999

[Deb, 2000]: Deb. K.; Agrawal, S.; Pratab, A.; Meyarivan, T.: A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II, KanGAL report 200001, Indian Institute of Technology, Kanpur, India, 2000.

[Deb, 2002]: Deb, K.: Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons, Chichester 2002

[Duran und Odell, 1974]: Duran, B.; Odell, P.: Cluster Analysis, Springer Verlag, Berlin 1974

[Eiben und Kemenade, 1995]: Eiben, A.E.; van Kemenade, C.H.M.: Performance of Multi-Parent Crossover Operators on Numerical Function Optimization Problems, Technical Report 95-33, Rijksuniversiteit te Leiden, 1995

[EMO, 2001]: Zitzler, E; Deb, K; Thiele, L.; Coello, CAC and Corne, D. (Hrsg.): First International Conference on Evolutionary Multi-Criterion Optimization, Springer-Verlag, Lecture Notes in Computer Science No. 1993, 2001

[Eshelmann et al, 1989]: Eshelmann, L.J.; Caruana, R.A.; Schaffer, J.D.: Biases in the Crossover Landscape, in: [Schaffer, 1989], S. 10-19

[Eshelmann und Schaffer, 1993]: Eshelmann. L.J.; Schaffer, J.D.: Real Coded Genetic Algorithms and Interval-Schemata, in: [Whitley, 1993], S. 187-202

[Fletcher und Powell, 1963]: Fletcher, R; Powell, M.J.D.: A rapidly convergent descent method for minimizing, Computer Journal 6, 1963

[Fletcher_1987]: Fletcher, R.: Practical Methods of Optimization, John Wiley, Chichester, 1987

[FlorMond_1996]: Floreano, D.; Mondala, F.: Evolution of Homing Navigation in a Real Mobile Robot, IEEE Transactions on Systems, Man and Cybernetics – Part B, 26(3), S.396-407, 1996

[Fogel et al, 1966]: Fogel, L.J.; Owens, A.J.; Walsh, M.J.: Artifical Intelligence through Simulated Evolution, John Wiley, 1966

[Fogel, 1992]: Fogel, D.B.: Evolving Artifical Intelligence, Ph.D.Thesis, University San Diego, 1992

[Fogel, 1995]: Fogel, D. (Hrsg.) Proceedings of the 2nd IEEE Conference on Evolutionary Computation, IEEE Service Center, 1995

[Fogel_1962]: Fogel, L.J.: Autonomous Automata, in: Industrial Research 4, S.14-19, 1962

[Fonseca und Fleming, 1993]: Fonseca, C.M.; Flemming, P.J.: Genetic Algorithm for multiobjective optimization: Formulation, Discussion and Generalization, in: [Forrest_1993], S. 416-423, 1993

[Fonseca und Fleming, 1995]: Fonseca, C.M.; Fleming, P.J.: An Overview of Evolutionary Algorithms in Multiobjective Optimization, in: Evolutionary Computation 3(1), 1995 MIT

[Fonseca und Fleming, 1998]: Fonseca, C.M.; Fleming, P.J.: Muliobjective Optimization and multible Constraint Handling with Evolutionary Algorithms – Part I: A unified Formulation, IEEE Transactions on Systems, Man and Cybernetics – Part A: Systems and Humans, 28(1), S. 26-37

[Forrest_1993]: Forrest, S. (Hrsg.): Proceedings of the 5th International Conference on Genetic Algorithms, University of Illinois, Urbana-Champaign IL, Morgan Kaufmann, San Mateo CA, 1993

[Fourman, 1985]: Fourman, M.: Compaction of Symbolic Layout using Genetic Algorithm, in: [Grefenstette, 1985], S.141-153

[Geoffrion et al, 1972]: Geoffrion, A.M.; Dyer, J.S.; Feinberg, A.: An interactive aproach for multicriterion optimization with an application to the operation of an academic department, Management Science 19(4), S.357-368, 1972

[Goldberg und Richardson, 1987]: Goldberg, D.E.; Richardson, J.: Genetic Algorithm with sharing for multimodal function optimization, in: Genetic Algorithms and their Applications: Proceedings of the 2. International Conference on Genetic Algorithm, S. 41-49, Hillsdale, 1987

[Goldberg, 1989]: Goldberg, D.E.: Genetic algorithms in search, Optimization and machine learning, Addison-Weshley, Reading MA, 1989

[Gomez et al, 1998]: Gomez-Skarmeta, A.F.; Jimenez, F.; Ibanez, J.: Pareto-optimality in fuzzy modeling, in: 6th Euopean Congress on Intelligent Techniques and Soft Computing, S. 694-700, 1998

[Goos, 1998]: Goos, G.: Vorlesungen über Informatik, Band 4: Paralleles Rechnen und nicht-analytische Lösungsverfahren, Springer Verlag, 1998

[Greenwood et al, 1996]: Greenwood, G.W.; Hu, X.; D'Ambrosio, J.G.: Fitness functions for multible objective optimization problems: Combining preferences with pareto rankings, in: Foundations of Genetic Algorithms 4 (FOGA-1996), S. 437-455, 1996

[Grefenstette, 1984]: Grefenstette, J.J.: GENESIS: A System for using Genetic Search Procedures, in: Proceedings of the Conference on Intelligent Systems and Maschines, S. 161-164, 1984

[Grefenstette, 1985]: Grefenstette, J.J. (Hrsg): Genetic Algorithms and Their Applications: Proceedings of the 1. International Conference on Genetic Algorithms, Lawrence Erlbaum, 1985

[Grefenstette, 1986]: Grefenstette. J.J.: Optimization of Control Parameters for Genetic Algorithms, in: IEEE Transactions on Systems. Man and Cybernetics SMC-16, S. 122-128, 1986

[Gueugnuiaud et al, 1997]: Gueugnuiaud, P.Y.; Bertin-Maghit, M.; Hirschauer, C.; Bouchard, C.; Vilasco, B.; Petit, P.; Gen, M.; Ida, K.; Lee, J.; Kim, J.: Fuzzy nonlinear goal programming using genetic algorithm, in: Computers and industrial engineering 33(1), S.39-42, 1997

[Hajela und Lin, 1992]: Hajela, P.; Lin, C.-Y.: Genetic search strategies in multicriterion optimal design, in: Structural Optimization 4, S. 99-107, 1992

[Hansen, 1997]: Hansen, M.P.: Tabu search in multiobjective optimization: MOTS, in: 13th International Conference on Multi-Criterion Decision Making (MCDM'97), University of Cape Town, 1997

[Herdy, 1992]: Herdy, M.: Reproductive isolation as strategy parameter in hierachical organized evolution strategies, in: [Mäner und Manderick, 1992], S. 207-217, 1992

[Hoffmeister und Bäck, 1992]: Hoffmeister, F.; Bäck, T.: Genetic Algorithms and Evolution Strategies: Similarities and Differences, Technical Report SYS-1/92, Technische Universität Dortmund, 1992

[Hoffmeister und Sprave, 1996]: Hoffmeister, F.; Sprave, J.: Problem-independent handling of constraints by use of metric penalty functions, in: Fogel, L. J.; Angeline, P. J.; Bäck, T. (Hrsg.): Evolutionary Programming V - Proceedings of the Fifth Annual Conference on Evolutionary Programming (EP'96*)*, S. 289-294, San Diego CA, MIT Press, Cambridge MA, 1996

[Holland, 1975]: Holland, J.: Adaptation in Natural and Artifical Systems, University of Michigan Press, 1975

[Horn und Nafpliotis, 1993]: Horn. J.; Nafpliotis. N.: Multiobjective optimization using the niched pareto genetic algorithm, IlliGAL Report 93005, University of Illinois USA, 1993

[Horn, 1997]: Horn, J.: Multicriterion Decision Making; in: [Bäck et al, 1997] S. F1.9:1-F1.9:15

[Hornik et al, 1989]: Hornik, K.; Stinchcombe, M; White, H.: Multilayer feedforward networks are universal approximators, Neural Networks, Bd. 2, S.359-366, 1989

[Hwang und Masud, 1979]: Hwang, C.L.; Masud, A.S.M.: Multible Objective Decision Making – Methods and Applikations: Springer Verlag Berlin, 1979

[Isermann, 1974]: Isermann, R.: Prozessidentifikation – Identifikation und Parameterschätzung dynamischer Prozesse mit diskreten Signalen, Springer Verlag, 1974

[Ishbuchi und Murata, 2001]: Ishibuchi, H.; Murata, T.: Minimizing the fuzzy rule and maximizing its performance by a multi-objective genetic algorithm, in: [EMO, 2001], S. 259-264, 2001

[Jordan und Rumelhart, 1992]: Jordan, M.I.; Rumelhart, D.E.: Forward Models: Supervised Learning with a Distal Teacher, Cognitive Science 16, S.307-354, 1992

[Kaas et al, 1983]: Kaas, J.H.; Merzenich, M.M.; Killackey, H.P.: The Reorganization of Somatosensory Cortex Following Peripheral Nerve Damage in Adult and Developing Mammals, Annual Rev Neurosci (6), S. 325-356, 1983

[Kleppmann, 1998]: Kleppmann, W.: Taschenbuch der Versuchsplanung: Produkte und Prozesse optimieren, Hanser Verlag München Wien, 1998

[Knowles und Corne, 2000]: Knowles, J. D.; Corne, D. W.: Approximating the nondominated front using the Pareto archived evolution strategy, in: Evolutionary Computation 8(2), S.149-172, 2000

[Knowles, 2002]: Knowles, J.D.: Local-Search and Hybrid Evolutionary Algorithms for Pareto Optimization, Ph.D.Thesis, University of Reading, Department of Computer Science, Reading UK 2002

[Koeppen und Rudlof, 1998]: Köppen, M; Rudlof, S.: Multiobjective Optimization by Nessy Agorithm, in: Roy, R.; Furuhashi, T.; Chawdhry, P.K. (Hrsg.): Advances in Soft Computing, S. 357-368, Springer, 1998

[Kohonen, 1982]: Kohonen. T.: Self Organized Formation of Topologically Correct Feature Maps, in: Biol. Cybernetics (43), S. 59-69, 1982

[Kohonen, 1995]: Kohonen, T.: Self-organizing maps, Springer series in information sciences, Vol 30, 1995

[Kohonen, 2001]: Kohonen. T.: Self-organizing maps, in: Springer series in information sciences, 3rd ed.; 2001

[Korhonen et al, 1992]: Korhonen, P.; Moskowitz, H; Wallenius, J.: Multible Criteria Decision Support – A Review, in: European Journal of Operational Research (63), S. 361-375, 1992

[Koza, 1992]: Koza, J.R.: Genetic Programming, Cambridge/MA: MIT Press, 1992

[Koza, 1994]: Koza, J.R.: Genetic Programming II, Cambridge/MA: MIT Press, 1994

[Kursawe, 1991]: Kursawe, F.: A Variant of Evolutionary Strategies for Vector Optimization, in: [Schwefel und Männer, 1991], 1991

[Lichtfuss, 1965]: Lichtfuss, H.J.: Evolution eines Rohrkrümmers, Diplomarbeit, Technische Universtät Berlin, 1965

[Lis und Eiben, 1997]: Lis, J.; Eiben, A.E.: A multi-sexual genetic algorithm for multiobjective optimization, in: Proceedings of 1997-IEEE International Conference on Evolutionary Computation (ICEC'97), S. 59-64, 1997

[Luenberger, 1984]: Luenberger, D.G.: Linear and Nonlinear Programming, Addison-Wesley, 1984

[Manas, 1982]: Manas, M.S.: Graphical methodes of multi-criterion optimization, in: Zeitschrift für angewandte Mathematik und Mechanik 62(5), S. 375-377, 1982

[Mäner und Manderick, 1992]: Männer. R.. B. Manderick (Hrsg.): Parallel problem solving from nature 2, Proceedings of the 2nd PPSN Conference, North-Holland, Amsterdam 1992

[McCulloch und Pitts, 1943]: McCulloch, W; Pitts, W.: A Logical calculus of Ideas Immanent in Nervous Activity, Bulletin of Mathematical Biophysics (5), S.113-133, 1943

[Meyer und Eder, 1999]: Meyer. D.; Eder. K.: Data Driven Process Modeling and Batch Optimization with Neural Networks and Modified Evolutionary Strategies, in [Brewka et al, 1999] 1999, S.125-132

[Meyer, 2000]: Meyer, D.: Modellbasierte Mehrzieloptimierung verfahrenstechnischer Produktionsprozesse mit Neuronalen Netzen und Evolutionsstrategien, in: VDI Berichte 1526, Computational Intelligence im industriellen Einsatz, Baden-Baden, 2000

[Meyer, 2002]: Meyer, D.: MOMBES – Multiobjective Modellbased Evolution Strategy, Forschungsberichte Künstliche Intelligenz FKI-246-02, Institut für Informatik, Technische Universität München, 2002

[Michalewicz, 1996]: Michalewicz, Z.: Genetic Algorithms + Data Structure = Evolution Programs, 3. Auflage, Springer Verlag, Berlin 1996

[Milano et al, 2001]: Milano, M.; Schmidhuber, J.; Koumotsakos, P.: Active Learning with Adaptive Grids, International Conference on Artifical Neural Networks, Wien 2001

[Minsky und Papert, 1969]: Minsky, M.; Papert, S (Hrsg.): Perceptrons, Cambridge/MA: MIT Press, 1969

[Mitchell, 1996]: Mitchell, M.: An Introduction to Genetic Algorithms, Cambridge/MA: MIT Press, 1996

[Moody und Darken, 1989]: Moody, J.; Darken, C.J.: Fast learning in networks of locally-tuned processing units, Neural Computation (1), S.281-294, 1989

[Mühlenbein et al, 1993]: Mühlenbein, H.; Schlierkamp-Voosen, D.: The Science of Breeding and ist Application to the Breeder Genetic Algorithm, Evolutionary Computation 1(4), S. 335-360, 1993

[Mühlenbein, 1992]: Mühlenbein. H.: How Genetic Algorithms Really Work I. Mutation and Hillclimbing, in: [Mäner und Manderick, 1992], S. 15-25, 1992

[Nauck et al, 1996]: Nauck, D.; Klawonn, F.; Kruse, R.: Neuronale Netze und Fuzzy-Systeme, Vieweg & Sohn, 1996

[Nelder und Mead, 1965]: Nelder, J.A.; Mead, R.: A simplex method for function minimizing, Computer Journal 7, 1965

[Nissen, 1994]: Nissen, V.: Evolutionäre Algorithmen – Darstellung, Beispiele, betriebswirtschaftliche Anendungsmöglichkeiten, Deutscher Universitäts Verlag Wiesbaden, 1994

[Nissen, 1997]: Nissen, V.: Einführung in Evolutionäre Algorithmen: Optimierung nach dem Vorbild der Natur, Verlag Vieweg, 1997

[Numerical Recipies, 1994]: Press, W.H.; Teukolsky, S.A.; Vetterling, W.T.; Flannery, B.P.: Numerical Recipies in C, 2. Auflage, Cambridge University Press, 1994

[Ocyczka, 1984]: Ocyczka. A: Multicriterion Optimization in Engineering, John Wiley, New York 1984

[Ocyczka, 1985]: Ocyczka. A: Multicriteria Optimization for Engineering Design, in: Design Optimization, S. 193-227, Academic Press, 1985

[Oei et al_1991]: Oei, Ch. K.; Goldberg, D.; Chang, Sh.: Tournament Selection, niching and the preservation of diversity, IlliGAL Report 91011, University of Illinois, 1991

[Parker, 1985]: Parker, D.: Learning Logic; Techn. Ber. TR-47, Center for Computational Research in Economics and Management Science, MIT Press, Cambridge, MA, 1985

[Poggio und Girosi, 1989]: Poggio, T; Girosi, F.: A Theory of Networks for Approximation and Learning, A.I. Memo, No. 1140, MIT Artificial Intelligence Laboratory, 1989

[Powell, 1962]: Powell, M.J.D.: An iterative method for finding stationary values of a function of severals variables, Computer Journal 5, 1962

[Powell, 1985]: Powell, M.J.D.: Radial basis functions for multivariable interpolation: A review, in: IMA conference on Algorithms for the Approximation of Functions and Data, RMCS Shrivenham, 1985

[Quagliarella und Vicini, 1997]: Quagliarella, D.; Vicini, A.: Coupling genetic algorithms and gradient based optimization techniques, in: Genetic Algorithms and Evolutionary Strategy in Engineering and Computer Science, S. 289-309, Wiley, Chichester 1997

[Rechenberg, 1965]: Rechenberg, I.: Cybernetic solution path of an experimental problem, Royal Aircraft Establishment, Library Translation Number 1122, Farnborough, UK 1965

[Rechenberg, 1973]: Rechenberg, I.: Evolutionsstrategie, Optimierung technischer Systeme nach Prinzipien der biologischen Evolution, Frommann-Holzboog Verlag, Stuttgart 1973

[Rechenberg, 1994]: Rechenberg, I.: Evolutionsstrategie ’94, Frommann-Holzboog Verlag, Stuttgart 1994

[Ritter et al, 1990]: Ritter, H.; Martinetz, T.; Schulten, K.: Neuronale Netze, Eine Einführung in die Neuroinformatik selbstorganisierender Netzwerke, Addision-Wesley, 1990

[Rojas, 1993]: Rojas, R.: Theorie der Neuronalen Netze, Eine systematische Einführung, Springer Verlag, Heidelberg 1993

[Rosenbrock, 1960]: Rosenbrock, H.H.: An Automatic Method for finding the Greatest or Least Value of a Function, in: Computer Journal 3, 1960

[Rosenthal, 1985]: Rosenthal, R. E.: Concepts, Theory and Techniques, Principles of Multiobjective Optimization, in: Decision Sciences 16, S.133-152, 1985

[Rumelhart und McClelland, 1986]: Rumelhart, D.; McClelland, J. et al: Parallel Distributed Processing: Exploration in the Microstructure of Cognition, Bd. 1: Foundations, MIT Press, Cambridge 1986

[Sachs, 1982]: Sachs, L.: Statistische Methoden; Springer Verlag, 1982

[Salomon, 1996]: Salomon, R.: Some Comments on Evolutionary Algorithm Theory, in: Evolutionary Computation 4(4), S. 405-415, 1996

[Salomon, 1997]: Salomon, R.: Optimierung und Evolutionäre Ansätze, 14. ITG/GI-Fachtagung von Rechnersystemen ARCS’97, Workshop on optimization in physiks and computer science – concepts and realizations, Rostock 1997

[Salomon, 1998]: Salomon, R.: Evolutionary Algorithm and Gradient Search: Similarities and Differences, IEEE Transactions on Evolutionary Computation 2(2), S. 45-55, 1998

[Sammon, 1969]: Sammon, J.W. Jr.: A nonlinear mapping for data structure analysis, IEEE Transactions on Computers, C-18 (5), S. 401-409, 1969

[Schaffer, 1984]: Schaffer, J.D.: Some experiments in Maschine Learning using Vector Evaluated Genetic Algorithms, Ph. Thesis, Vanderbilt University, Nashville 1984

[Schaffer, 1985]: Schaffer, J.D.: Multible objective optimization with vector evaluated genetic algorithms, in: [Grefenstette, 1985], S. 93-100

[Schaffer, 1989]: Schaffer, J.D.; (Hrsg.) Proceedings of the Third International Conference on Genetic Algorithms, San Matoe/CA, 1989

[Schwefel und Bäck, 1992]: Schwefel, H.P.; Bäck, T.: Künstliche Evolution eine intelligente Problemlösungsstrategie? KI Künstliche Intelligenz, Vol. 6. (2), S. 20-27, 1992

[Schwefel und Männer, 1991]: Schwefel, H.P.; Männer, R. (Hrsg.): Parallel Problem Solving from Nature, LNCS 496, Springer Verlag, Berlin 1991

[Schwefel, 1968]: Schwefel, H.-P.: Projekt MHD-Staustrahlrohr: Experimentelle Optimierung einer Zweiphasendüse, Teil 1, Technischer Report 11.034/68. 35., AEG Forschungsinstitut, Berlin 1968

[Schwefel, 1977]: Schwefel, H.P.; Numerische Optimierung von Computer-Modellen mittels Evolutionsstrategie, Birkhäuser Verlag, Basel 1977

[Schwefel, 1981]: Schwefel, H.P.: Numerical Optimization of Computer Modells, John Whiley & Sons, Chichester 1981

[Schwefel, 1995]: Schwefel, H.P.: Evolution and Optimum Seeking; John Whiley & Sons, Chichester 1995

[Shewchuk, 1994]: Shewchuk, J.R.: An introduction to the conjugate gradient method without the agonizing pain, Technical Report: CS-94-125, Carnegie Mellon University, 1994

[Shonkwiler und Miller, 1992]: Shonkwiler, R; Miller, K.R.: Genetic Algorithm/Neural Network Synergy for Nonlinearly Constraint Optimization Problems, in: [Whitley und Schaffer, 1992] S. 248-257

[Skiena, 1990]: Skiena, S. Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica, Reading, MA: Addison-Wesley, 1990

[Srinivas und Deb, 1994]: Srinivas, N.; Deb, K.: Multiobjective Optimization using Nondominated Sorting in Genetic Algorithms, in: Evolutionary Computation Vol. 2, S. 221-248, 1994

[Steinhausen und Langer, 1977]: Steinhausen, D.; Langer, K.: Clusteranalyse, De Gruyter, Berlin 1977

[Steuer, 1986]: Steuer, R.E.: Multible Criteria Optimization: Theory, Computation and Application, Wiley, New York 1986

[Sturm, 2000]: Sturm, M.: Neuronale Netze zur Modellbildung in der Regelungstechnik, Dissertation am Institut für Informatik der TU München, 2000

[Tamaki et al, 1996]: Tamaki, H.; Kita, H.; Kobayashi, S.: Multi-objective Optimization by Genetic Algorithms: A review, in: Proceedings of 1996 IEEE Interational Conference on Evolutionary Computation (ICEC'96), S.517-522

[Thiersen und Goldberg, 1994]: Thiersen, D.; Goldberg; D.E.: Convergence Models of Genetic Algorithm Selection Schemes, in: Y. Davidor. H.P. Schwefel. R. Männer. (Hrsg.): Proceedings of Parallel Problem Solving from Nature 3, Springer Verlag, S.119-129, 1994

[Thomas, 1998]: Thomas, M.W.: A pareto frontier for full stern submarines via genetic algorithms, Ph.D. Thesis, Ocean Engeneering Department, MIT Cambridge, 1998

[Veldhuizen und Lamont, 1999]: Veldhuizen, D.A.; Lamont, G.B.: Multiobjective Evolutionary Algorithm Test Suits, in: Carroll, J.; Haddad, H.; Oppenheim, D.; Bryant, B. and Lamont, G.B. (Hrsg.): Proceedings of the 1999 ACM Symposium on Applied Computing, S. 351-357, San Antonio (Texas) 1999

[Veldhuizen und Lamont, 2000]: Veldhuizen, D.A.; Lamont, G.B.: Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-Art, in: Evolutionary Computation 8(2), S.125-147, 2000

[Veldhuizen, 1999]: Veldhuizen, D.V.: Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations, Ph.D. Thesis, Dayton, OH: Air Force Institute of Technology, Technical Report No.: AFIT/DS/ENG/99-01, 1999

[Walz, 1967]: Walz. F.M.; An Engineering Approach: Hierarchical Optimization Criteria, IEEE Trans. Automatic Control, AC-12, 1967

[Wernstedt, 1989]: Wernstedt, J.: Experimentelle Prozessanalyse, Oldenburg Verlag München Wien, 1989

[Whitley und Schaffer, 1992]: Whitley, L.D.; Schaffer, J.D.: COGANN-92 International Workshop on Combinations of Genetic Algorithms and Neural Networks, Los Alamitos/CA: IEEE Computer Society Press, 1992

[Whitley, 1993]: Whitley, L.D.: Foundations of Genetic Algoritms 2, San Mateo, Morgan Kaufmann, 1993

[Wolpert und Macready, 1995]: Wolpert, D.H.; Macready, W.G.: No Free Lunch Theorems for Search, Bericht: SFI-TR-95-02-010, Santa Fe Institut, 1995

[Zell, 1994]: Zell, A.: Simulation Neuronaler Netze, Addision-Wesley, 1994

[Zitzler et al, 2001]: Zitzler, E.; Laumanns, M.; Thiele, L.: SPEA2: Improving the Strength Pareto Evolutionary Algorithm, TIK-Report Nr. 103, Swiss Federal Institute of Technology (ETH), 2001

[Zitzler et al, 2002]: Zitzler, E.; Laumanns, M.; Thiele, L.: SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization, in: Giannakoglou, K.; Tsahalis, D.; Periaux, J.; Papailliou, K. (Hrsg.): Evolutionary Methods for Design, Optimisation and Control with Applications to Industrial Problems, CIMNE Barcelona 2002

[Zitzler und Thiele, 1998]: Zitzler, E.; Thiele, L.: An Evolutionary Algorithm for Multiobjective Optimization: The Strength Pareto Approach, TIK-Report Nr. 43, Swiss Federal Institut of Technology (ETH), 1998

[Zitzler und Thiele, 1999]: Zitzler, E.; Thiele, L.: Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach, IEEE Transactions on Evolutionary Computation, 3(4), S.257-271, 1999

[Zitzler, 1999]: Zitzler, E.: Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications, Dissertation an der Eidgenössischen Technischen Hochschule Zürich (ETH), TIK-Report Nr. 30, Swiss Federal Institut of Technology (ETH), 1999