Distributed Evolutionary Design: Island-Model Based Optimization of Steel Skeleton Structures in Tall Buildings

Rafal Kicinger, Civil, Environmental and Infrastructure Engineering Department, George Mason University, Fairfax, VA 22032, USA

(rkicinge@gmu.edu)

Tomasz Arciszewski, Civil, Environmental and Infrastructure Engineering Department, George Mason University, Fairfax, VA 22032, USA

(tarcisze@gmu.edu)

Kenneth De Jong, Computer Science Department, George Mason University, Fairfax, VA 22032, USA

(kdejong@gmu.edu)

Summary

This paper presents results of a study on distributed, or parallel, evolutionary computation in the topological design of steel structural systems in tall buildings. It describes results of extensive experimental research on various parallel evolutionary architectures applied to a complex structural design problem. The experiments were conducted using Inventor 2003, a network-based evolutionary design support tool developed at George Mason University. First, a general introduction to evolutionary computation is provided with an emphasis on recent developments in parallel evolutionary architectures. Next, a discussion of conceptual design of steel structural systems in tall buildings is presented. Further, Inventor 2003 is briefly introduced as well as its design representation and evolutionary computation characteristics. Next, the results obtained from systematic design experiments conducted with Inventor 2003 are discussed. The objective of these experiments was to qualitatively and quantitatively investigate evolution of steel structural systems in tall buildings during a distributed evolutionary design process as well as to compare efficiency and effectiveness of various parallel evolutionary architectures with the traditional evolutionary design approaches. Two connectivity topologies (ring topology and fully-connected topology) have been investigated for four populations of structural designs evolving in parallel and using various migration strategies. Also, results of the initial sensitivity studies are reported in which two ways of initializing distributed evolutionary design processes were investigated, using either arbitrarily selected designs as initial parents or randomly generated ones. Finally, initial research conclusions are presented.

1 Introduction

As the field of evolutionary design continues to develop, there are several scaling-up issues that need to be addressed as the design problems increase in size and complexity. The first issue is computation time. When evolving designs, most of the computation time involves evaluating the fitness of a design. As a consequence, in the past when computational costs were high, a variety of techniques were developed to minimize computational design effort, including separating the stages of conceptual design and detailed design, and developing separate tools for each stage.

Today, however, the situation is different. First, the cost of computation continues to decrease. In addition, a variety of low cost parallel computation architectures such as Beowulf clusters are now readily available. Also, evolutionary algorithms have a natural mapping onto parallel architectures. For all these reasons, it is our belief that computational costs should not be the primary factor in developing new EA-based engineering design support systems.

2 Background

2.1 Evolutionary Computation

Computational models of Darwinian evolution and natural selection have been used not only to study artificial life (Holland 1975) but also to solve problems in various disciplines of science and engineering (Parmee 2001). Models of evolutionary mechanisms encoded in *evolutionary algorithms* (EA) have been successfully applied to many structural design problems, especially those related to complex optimization problems with nonlinear, stochastic, or temporal components, where traditional optimization techniques are generally unsatisfactory. Presently, evolutionary computation, or more specifically evolutionary design, is recognized as particularly appropriate for various traditional and novel computational applications in structural engineering.

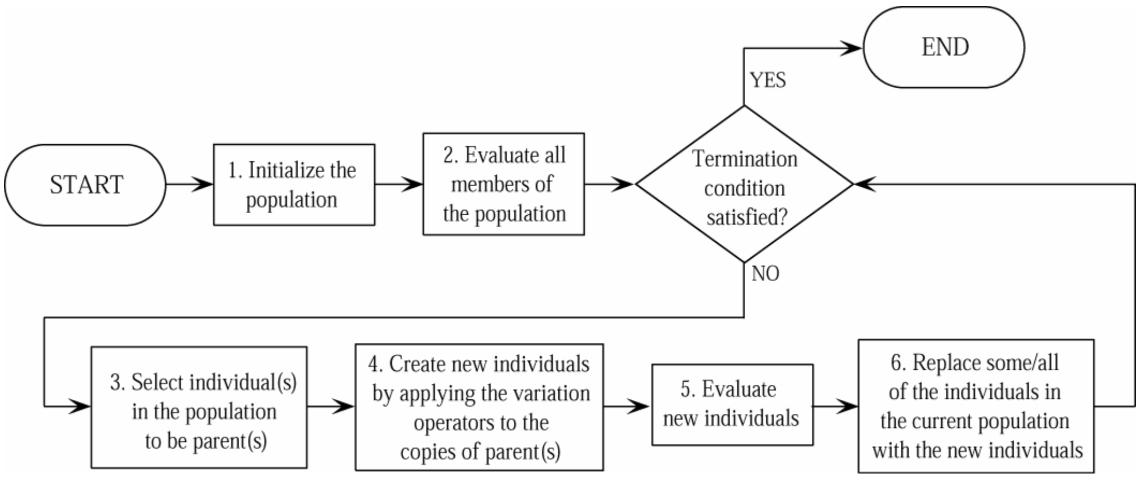
Strong resemblance to biological processes as well as their initial applications for modeling complex adaptive systems (Holland 1975) influenced the terminology used by evolutionary computation (EC) and evolutionary design researchers. It borrows a lot from genetics, evolutionary theory and biology. Thus, a candidate solution (design) to a design problem is called an *individual* while an entire set (or more accurately a superset) of current solutions is called a *population*. For some types of EA a population may be broken into several *subpopulations*. The actual representation (encoding) of an individual is called its *genome* or *chromosome*. Each genome consists of a sequence of *genes*, i.e. attributes that describe an individual. A value of a gene is called an *allele*. When individual solutions are modified to produce new candidate solutions, they are said to be *breeding* and the new candidate solution is called an *offspring* or a *child*. During the evaluation of a candidate solution, it receives a grade called *fitness*, which indicates the quality of the solution in the context of a given problem. When the current population is replaced by offspring, the new population is called a new *generation*. Finally, the entire process of searching for an optimal solution is called *evolution* (Luke 2000).

From the algorithmic perspective, evolutionary algorithms form a family of population-based search algorithms that simulate the evolution of individual structures by interrelated processes of selection, reproduction, and variation. There is a variety of EA that have been proposed and studied. They all share a common set of underlying assumptions but differ in the breeding strategy and representation of solutions on which EA operate (De Jong to appear).

Historically, three major EA have been developed: evolution strategies (ES) (Rechenberg 1965; Schwefel 1965), evolutionary programming (EP) (Fogel et al. 1966), and genetic algorithms (GA) (Holland 1975). Traditionally, these algorithms have been mostly used to evolve solutions to problems which can be parameterized, i.e. encoded in terms of attributes, symbolic or numerical, and their values. On the other hand, the fourth major EA developed more recently, genetic programming (GP) (Koza 1992), has been used to evolve actual computer programs. There are also many hybrid models incorporating various features of the above canonical EA, e.g., CHC algorithm (Eshelman 1991) .

From the engineering point of view, EC can be understood as a search and optimization process in which a population of solutions (designs) undergoes a process of gradual changes. This process is guided by the fitness understood here as a formal measure of the perceived performance of the individual designs.

Figure 1 shows the architecture of a canonical evolutionary algorithm. Before an actual evolutionary design process begins, an initial population of individuals (designs) is created. Traditionally in the EC literature, the initial population is generated randomly but several other initialization techniques have also been used (e.g. starting from a set of previously known or arbitrarily assumed solutions). Next, each design in the initial population is evaluated and assigned a fitness value. The fitness scores are subsequently used by the selection mechanism to choose a subset of the current population (usually one or two designs, called *parents*). Selected parent(s) are then copied and subsequently changed by the variation operators and thus form new individuals, called *offspring*, which are modified versions of the parents. When the selection mechanism uses bias toward designs with better fitness, the created offspring will, on average, have higher fitness. The newly created designs are evaluated and assigned fitness values. Then, depending on the selection strategy, either all or only a subset of the current population is replaced by the new designs.



START
1. Initialize the population
2. Evaluate all members of the population
Termination condition satisfied?
YES
END
NO
3. Select individual(s) in the population to be parent(s)
4. Create new individuals by applying the variation operators to the copies of parent(s)
5. Evaluate new individuals
6. Replace some/all of the individuals in the current population with the new individuals

Figure 1. Architecture of a canonical evolutionary algorithm

The two most popular variation operators are *mutation* and *recombination*. Mutation acts on a single design and works by applying some variation to one or more genes in the design's chromosome (similar to variation operator used in other search mechanisms like hill climbing or simulated annealing). Recombination, on the other hand, operates on multiple designs (usually two) and combines parts of these designs to create new ones.

The three main issues in applying EA to an engineering design problem are:

- Selecting an appropriate representation for engineering designs.
- Defining efficient variation operators.
- Providing an adequate evaluation function for estimating the fitness of generated solutions (points in the search space).

Appropriate representation of an engineering system is one of the most crucial elements of evolutionary design. This issue is particularly important when creativity/novelty of designs produced by evolutionary processes is one of the major goals. The process of creating an efficient and adequate representation of an engineering system for evolutionary design is complicated and involves elements of both science and art. One has to take into account not only important aspects of understanding traditional modeling of an engineering system, but also relevant computational issues that include search efficiency, scalability, and mapping between a search space (genotypic space) and a space of actual designs (phenotypic space).

Appropriate choice and implementation of variation operators, i.e. mutation and recombination, and careful tuning of their rates is an important issue as it can have a big impact on the final results. Variation operators are primary sources of exploration of the design spaces by an EA. On the other hand, a selection mechanism provides an EA with exploitative power. Thus, by properly defining and controlling variation mechanisms, one can achieve "an effective balance between further exploration of unexplored regions of the search space and exploiting the regions already explored" (De Jong to appear).

Another important issue in successful application of EA is the choice of an adequate fitness evaluation function for a given problem. Evaluation functions provide EA with feedback about the fitness of each design in the population. EA use this feedback to bias the search process in order to improve the population's average fitness. Naturally, the details of a particular fitness function are problem specific.

2.2 Distributed Evolutionary Algorithms

The fact that EA manipulate a population of independent solutions makes them well suited for parallel computation architectures. The motivation for using parallel EA is twofold. First, one wants to improve the speed of evolutionary processes by conducting concurrent evaluations of individuals in a population. Second, one hopes to improve the problem solving process by overcoming difficulties such as premature convergence that face traditional EA.

Distributed EA can be categorized into 3 major groups:

- Coarsely-grained "island" models
- Finely-grained "diffusion" models
- Agent-oriented models

In coarsely-grained island models (Cohoon et al. 1987), evolution occurs in multiple parallel subpopulations, each running a local EA, evolving independently with occasional "migrations" of selected individuals among subpopulations. The following experimental design decisions have to be made to define island-model EA:

• *Number of the subpopulations*:	2, 3, 4, more
• *Size of the subpopulations*:	uniform, or non-uniform
• *Type(s) local EA*:	uniform, or non-uniform
• *Connectivity topology*:	ring, star, fully-connected, random

- *Migration mechanisms*:
    - *how often migrations occur*
    - *which individuals migrate*

Finely-grained diffusion models (Collins and Jefferson 1991) assign one individual per processor. The local neighborhood topology is assumed and individuals are allowed to mate only within the neighborhood, called a deme. The demes overlap by an amount that depends on their shape and size and in this way create an implicit migration mechanism. Each processor runs identical EA which select parents from the local neighborhood, produces an offspring, and decides whether to replace the current individual with an offspring.

In agent-oriented models (Holland 1994), the agents interact with other agents in a simulated environment. The fitness is determined as a function of agent interactions.

2.3 Evolutionary Design in Structural Engineering

Evolutionary computation in structural engineering has relatively long history. Initial applications of EA considered sizing and shape optimization of relatively simple structural systems, including trusses (Goldberg and Samtani 1986; Hajela 1990) and frames (Grierson and Pak 1993). Topology optimization of discrete-member trusses were conducted by Shankar and Hajela (1991), and Hajela and Lee (1995). Bohnenberger et al. (1995) applied GA to optimize topologies of truss structures in pylons. Rajan (1995) applied GA to optimize topology, shape and member sizing of truss structures. Murawski et al. (2001) and Kicinger et al. (2004b) applied ES to optimize topology of steel structural systems in tall buildings. Soh and Yang (2001) introduced GP based approach to topological optimum design (TOD) of truss structures.

Initial applications of parallel EA in structural engineering are due to Adeli and Cheng (1994). They applied canonical GA with evaluation conducted concurrently by 8 processors to sizing optimization of complex spatial truss systems. Topping and de Barros Leite (1998) applied parallel GA to sizing optimization of cable-stayed bridge. Sarma and Adeli (2001) used island model with local fuzzy GA and migrations to sizing optimization of spatial multistory frame structures. Recently, Dimou and Koumousis (2003) used a parallel GA with subpopulations competing for limited resources (computation time) to solve a sizing optimization problem of a simple planar truss. More information is provided in (Kicinger et al. 2004a).

2.4 Steel Structural Systems in Tall Buildings

Steel skeleton structures in tall buildings are considered most complicated structures designed and built. Their conceptual and physical complexity can only be compared to such complex structural systems as, for example, large span bridges or large span space structures. Usually, steel structural systems in tall buildings are designed as a system of vertical members called columns, horizontal members called beams, and various diagonal members called wind bracings, since they are added to columns and beams to increase the flexural rigidity of the entire system and that is driven mostly by stiffness requirements related to wind forces.

Skeleton structures are designed to provide a structural support for tall buildings. They have to satisfy numerous requirements regarding the building's stability, transfer of loads, including gravity, wind and earthquake loads, deformations, vibrations, etc. For this reason, the design of structural systems in tall buildings requires the analysis of their behavior under various combinations of loading and the determination of an optimal configuration of structural members. It is difficult, complex, and still not fully understood domain of structural engineering, particularly as the generation/development of novel structural concepts is concerned.

2.5 Inventor 2003

In the mid-eighties, an extensive research program on evolutionary design was initiated in the Information Technology and Engineering School at George Mason University with support from the NASA Langley Center. Its focus has been on the fundamental issues of evolutionary design, including both the design methodology and computational issues, and on the development of various evolutionary design support tools. Several generations of such tools have been developed, starting with Inventor 2001 (Murawski et al. 2001). The most recent tool is called Inventor 2003 and it uses multi-population evolutionary computation as a search mechanism. It has been written in Java using Java spaces technology and is capable of distributing time-expensive computations through a network.

3 General Design Characteristics

In this paper, topological optimum design of steel structural systems in tall buildings is considered. It is conducted as a two-stage process. In the first stage, distributed EA produces a design concept. By this term, an abstract description of a future structural system is understood, and it is provided in terms of symbolic attributes (see section 3.1). It identifies the configuration of the following members of a structural system: wind bracings, beams, and supports. The configuration of columns is assumed constant (the location and nature of columns do not change) and is not evolved. In the second stage, sizing optimization of all structural members, including wind bracings, beams, and columns, is conducted for the design configuration determined in the previous stage.

The sizing optimization is conducted by SODA. It is a commercial computer program for the analysis of internal forces, dimensioning and numerical optimization of steel structural systems. In the project, a modified SODA program developed by the Waterloo Systems in Waterloo, Ontario, Canada, has been used. The optimization method used in SODA is described in (Grierson 1989). In the structural analysis conducted by SODA, dead, live, and wind loads as well as their combinations are considered. The structural elements are designed using several groups of sections for beams, columns, and bracings. In the performed experiments the first order analysis was used.

3.1 Representations of Steel Structural Systems

In Inventor 2003, similarly to Inventor 2001 (Murawski et al. 2001), a structural system of a tall building is considered as a system of identical parallel planar transverse structures, which are the subject of design. Representations of steel structural systems in tall buildings considered here encode the following types of structural members: bracings, beams, and supports.

Figure 2 shows the values of attributes representing bracing elements in a steel structural system at the phenotypic, symbolic, and genotypic level. Each such attribute has seven symbolic values (see Figure 2b)) encoding various types of bracings (no bracing, diagonal bracing \, diagonal bracing /, K bracing, V bracing, simple X bracing, and X bracing). Their phenotypic, or design, representation is presented in Figure 2a). Figure 2c) shows genotypic values of the attributes representing bracing elements where alleles take on subsequent integer values from 0 to 6.

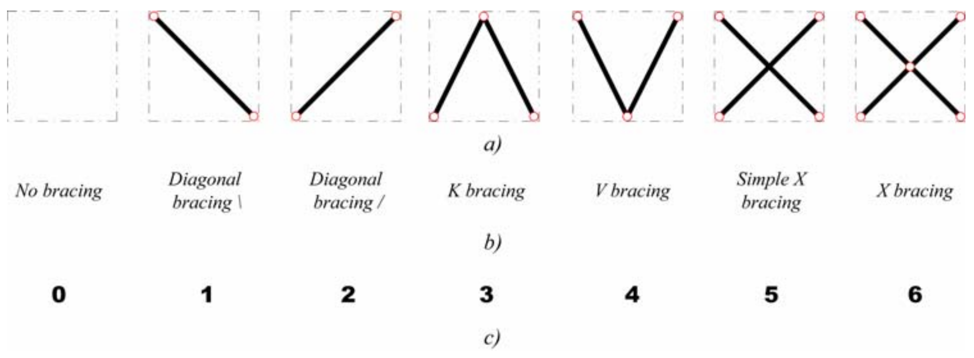


Figure 2. a) Phenotypic, b) symbolic, c) genotypic values of attributes representing wind bracing.

In Figure 3 values of attributes representing beams and supports are presented. Each attribute representing a beam in a steel structural system has two symbolic values (binary attributes) (see Figure 3c)) encoding two types of beams (pinned beam, and fixed beam) (see Figures 3a) and 3b)). Similarly, each attribute representing a support in a steel structural system is binary (see Figures 3f)) and encodes two types of supports (pinned support, and fixed support) (see Figures 3d) and 3e)).

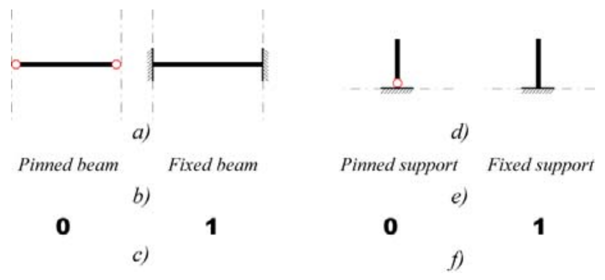


Figure 3. a) Phenotypic, b) symbolic, c) genotypic values of attributes representing beams.

The representation space in Inventor 2003 has been developed using the concept of division of the structural grid of the building (the system of vertical and horizontal lines of columns and beams, respectively) into cells. A cell can be described as a part of the vertical structural grid contained within the adjacent vertical and horizontal grid lines (Murawski et al. 2001). Each such cell is defined by an attribute determining the type of a wind bracing (7 possible values) and an attribute describing the type of a beam (2 possible values). Additionally, two types of supports can be used. Using this representation space, a given steel structural system can be encoded as the sum of representations of its individual cells, each described by attributes identifying the existence and the types of the considered structural elements.

The actual genotypic representation, or genome, that is manipulated by an evolutionary algorithm, is linearized and encoded as a string of integer values. In this paper, fixed-length genomes are used as representations of steel structural systems. The length of a genome used in a given case depends on the number of cells in the structural system being considered, and that is obviously related to the number of stories.

3.2 Distributed Evolutionary Design

In this paper, coarsely-grained island models are investigated with four subpopulations. The number of subpopulations was chosen to be equal to four in order to compare the results obtained using distributed EA with the results reported in (Kicinger et al. 2004b) in which four subpopulations of steel structural design of tall buildings were evolved independently. Each subpopulation was of equal size and evolved by the same EA. Two connectivity topologies were investigated: ring topology and fully-connected topology. They are shown in Figure 4. Arrows in Figure 4 indicate migrations to and from each of the subpopulations. Additionally, initial sensitivity studies were conducted in which the influence of initialization strategy was tested. Two methods of initialization of a distributed EA were investigated: arbitrarily selected designs as initial parents or randomly generated ones. When the former initialization strategy was used, the selected designs were chosen to be exactly the same as the ones used in the experiments reported in (Kicinger et al. 2004b). They are described in section 4.1.

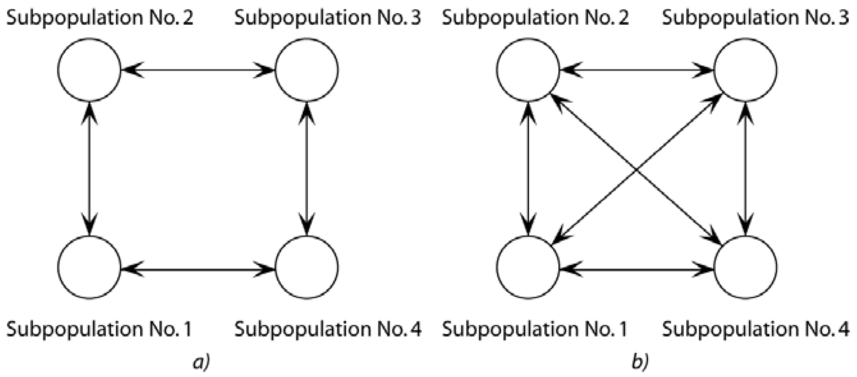


Figure 4. Investigated connectivity topologies a) ring, b) fully connected.

4 Design of Experiments

4.1 Domain Parameters

The objective of the experiments reported in the paper was to determine the feasibility of distributed EA for topological optimum design problems in structural engineering. It has been accomplished through the analysis of the results of a number of experiments. Figure 5 shows the domain parameters and their values that were used in the conducted experiments.

Domain Parameter	*Value(s)*
Number of bays	3
Number of stories	36
Bay width	20 feet (6.01 m)
Story height	14 feet (4.27 m)
Distance between transverse systems	20 feet (6.01 m)
Structural analysis method	First order
Serviceability conditions (constraints)	Max deflection < 1/600 · height of the building
Beams	Pinned, fixed (see Figure 3)
Column	Not evolved
Supports	Pinned, fixed (see Figure 3)
Wind bracings	No bracing, diagonal bracing (/), diagonal bracing (\), K bracing, V bracing, simple X bracing, and X bracing (see Figure 2)

Figure 5. Domain parameters and their values used in the reported experiments.

As discussed earlier, two initialization strategies were studied: random initialization and initialization using arbitrarily selected parents. When the latter approach was used, the island-based EA was started using a group of 12 feasible designs which have been pre-selected to form a pool of initial parents for the distributed evolutionary processes. The initial group of 12 parents was divided into four subpopulations of size three in the same way as in the authors' previous experimental study reported in (Kicinger et al. 2004b). The group of 12 designs consisted of designs that were considered as appropriate (called here "sub-optimal") for a given design situation, as well as designs that could be characterized as rather inappropriate. Examples of several designs discussed above are presented in Figure 6.

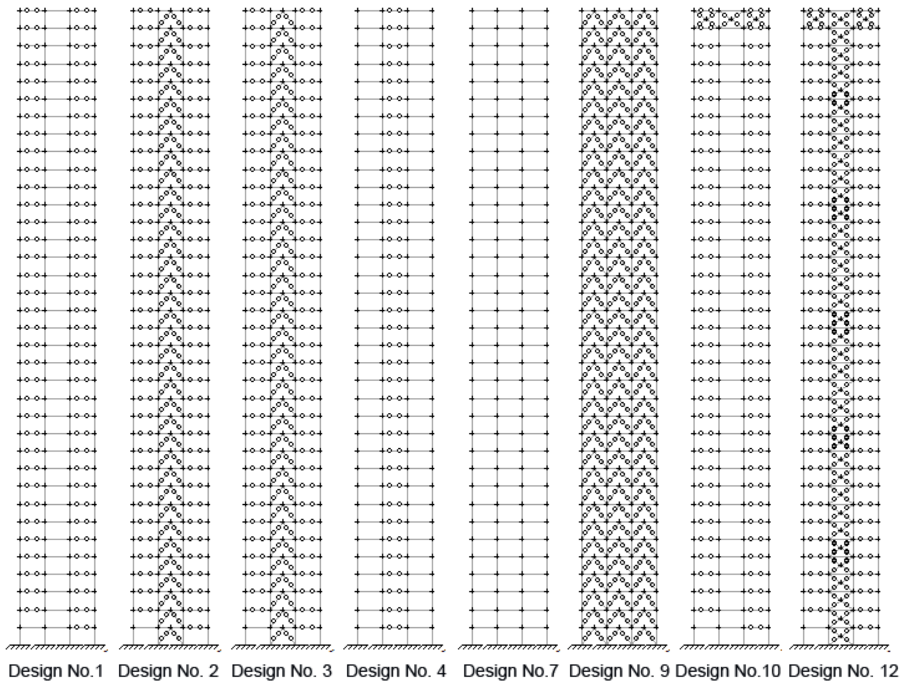


Figure 6. Examples of several designs used as initial parents in the reported experiments.

4.2 Evolutionary Computation Parameters

Each design of a steel structural system of a tall building defined in the previous section was represented by a fixed-length genome consisting of 220 non-homogenous integer-valued genes.

108 genes represented wind bracings and had seven values, 108 binary genes represented beams and 4 binary genes encoded supports.

Figure 7 shows EA parameters and their values that were used in the experiments reported in section 5. First, island-model EA parameters and their values were set to define the overall "design ecosystem" consisting of 4 populations. Subsequently, the parameters defining local EA operating on individual subpopulations were specified.

EA Parameter	*Value(s)*
Island-model EA parameters:	
Number of subpopulations	4
Size of the populations	Uniform
Type of local EA	Uniform
Connectivity topology	Ring, fully connected
Migration mechanism	
- How often migrations occur	Every 10 generations
- Which individuals migrate	Best individual replaces a random one in a subpopulation
Local EA parameters:	
EA	ES
Pop. sizes (parent, offspring)	(3,15)
Generational model	Overlapping
Selection (parent, survival)	(uniform stochastic, truncation)
Mutation rate	0.05
Crossover (type, rate)	(uniform, 0.2)
Fitness	Weight of the steel structure (minimization problem)
Initialization method	Random, arbitrarily selected initial parents (see section 4.1)
Constraint handling method	Death penalty (infeasible designs assigned 0 fitness)
Simulation parameters:	
Termination criterion	100 generations (for each subpopulation)
Number of runs	5 (in each experiment)

Figure 7. Domain parameters and their values used in the reported experiments.

5 Experimental Results

In the case of island-model EA, the conducted experiments revealed high sensitivity of results, in terms of fitness, to the quality of initial designs. Figure 8a) shows that significant improvements in fitness of generated designs as well as progress of evolutionary design process were achieved for population 1 consisting of initial designs, which had relatively low fitness. In the case of the fully connected topology and the ring topology, designs of comparable fitness were produced. The first used topology (fully connected) produced good solutions significantly faster than the second one. On the other hand, when island-model EA was initialized with designs of high fitness (as it is the case in population 3), almost no improvement was achieved when compared to independent evolution (when there is no migration) (see Figure 8b)). Figure 8 shows the average best-so-far fitness (total weight of the steel structure) obtained in the reported experiments and vertical bars denote 95% confidence intervals calculated using Johnson's modified t test. Connectivity topology and migrations within island-model EA provide opportunities for transferring good genetic material from a subpopulation containing designs of high fitness to subpopulations consisting of designs of relatively low fitness individuals.

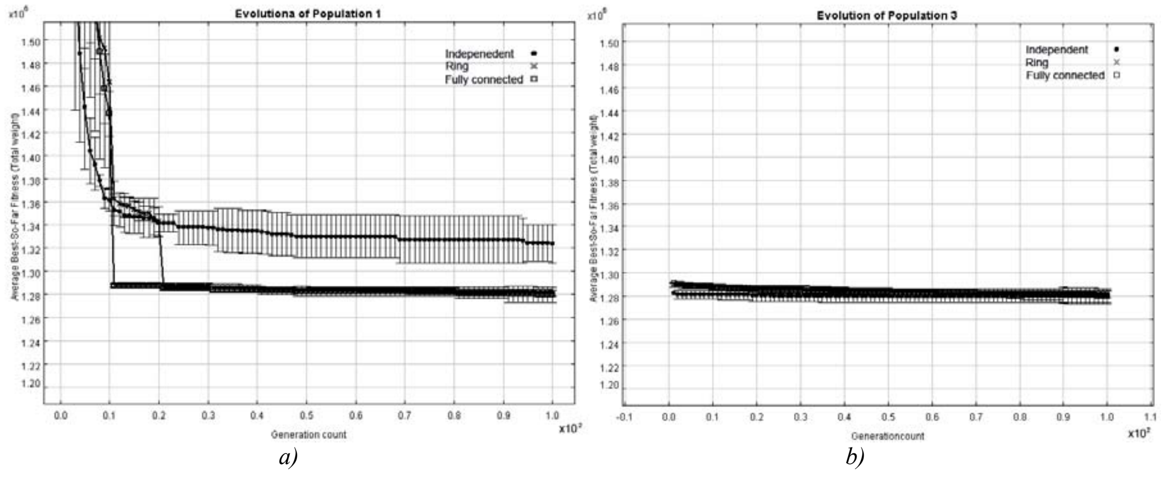


Figure 8. Comparison of evolutionary progress for population 1 a) and population 3 b) for ring and fully connected topologies and independent evolution

Intuitively, fully connected topology should transfer good genetic material faster than ring topology. This is confirmed in Figure 9, which shows the evolutionary progress of all four subpopulations using the ring topology (see Figure 9a)) and the fully connected topology (see Figure 9b)). It is clear that all four populations of structural designs converge in about 10 generations when fully connected topology is used (see Figure 9b)) while it takes twice as many generations when the ring topology is applied (see Figure 9a)).

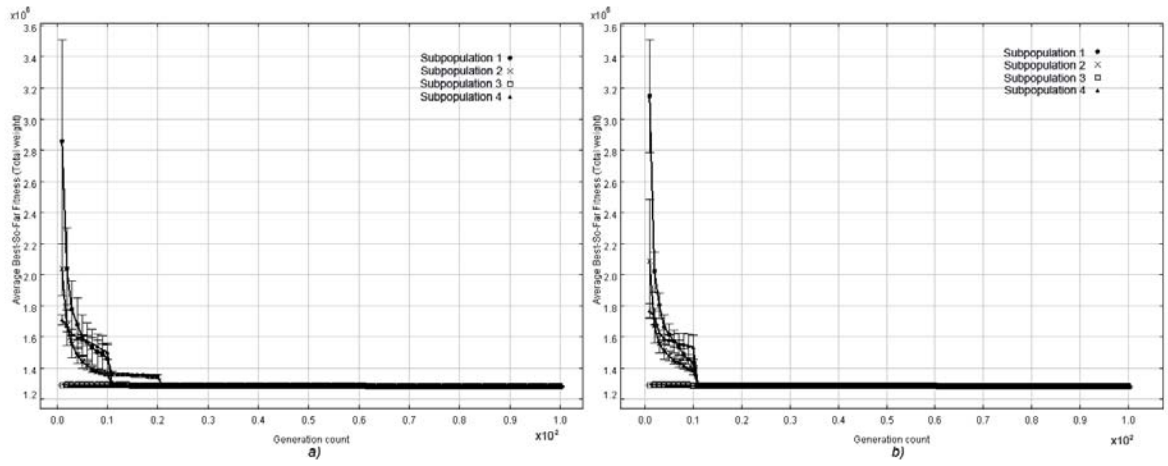


Figure 9. Evolution of four subpopulations in the island-model EA using ring topology a) and fully connected topology b).

In the experiments which used pre-selected initial parents, all subpopulations fairly quickly converged to a sub-optimal design (Design No. 9 in Figure 6) contained in population 3, or its slight mutation. Figure 9 shows that after 20 generations all other subpopulations in the design ecosystem became homogenous and almost no further exploration of the design space was conducted. However, when distributed evolutionary design processes were initialized randomly, the obtained results were dramatically different. Figure 10 shows the progress of evolutionary design processes of four subpopulations initialized randomly and compared to the progress of subpopulation 3 which was initialized using pre-selected initial designs of high fitness. It is clear that in the case of random initialization of the island-model EA constant evolutionary design progress is sustained and that eventually better designs are produced.

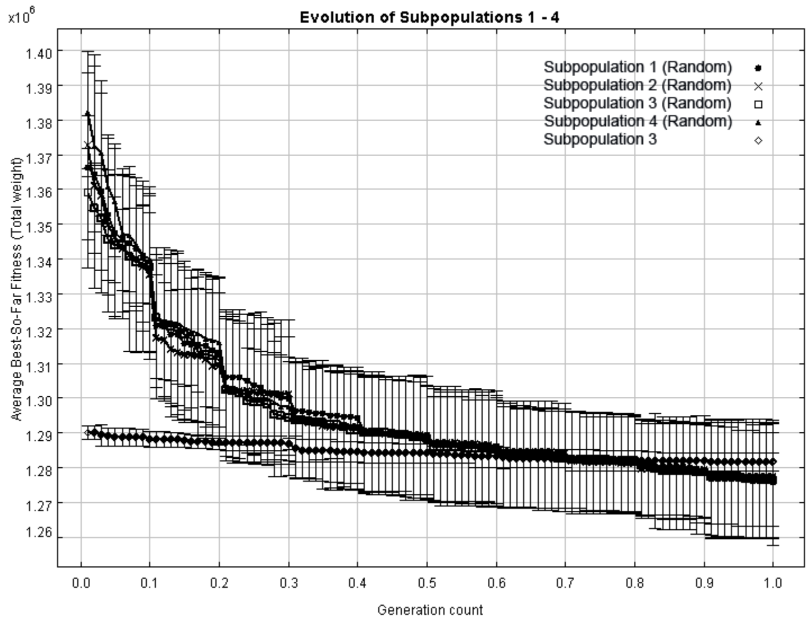


Figure 10. Evolution of four subpopulations 1-4 initialized randomly and subpopulation 3 intiialized with pre-selected high quality designs (in all cases the ring topology was used).

6 Conclusions

The conducted experiments revealed the feasibility of using distributed EA in structural design. This form of EA is obviously more conceptually and computationally complicated than a single-population EA, but it provides several essential advantages.

Specifically, the research results can be summarized as follows:

1. The multi-island model allows the exchange of good genetic material and thus improves the quality of designs. Also, in this case, the process of evolutionary design is significantly faster than in the case of single population.
2. There is potential "danger" that distributed evolutionary design process can be attracted to a local minimum (especially when small sizes of subpopulations are used) when good designs are used as initial parents. However, this danger can be avoided by using random initialization.

The reported results are preliminary. The area of distributed evolutionary design is practically unexplored and much more work has to be done, including the extension of the conducted experiments.

7 References

Adeli, H., and Cheng, N. T. (1994). "Concurrent genetic algorithms for optimization of large structures." *Journal of Aerospace Engineering*, 7(3), 276-296.

Bohnenberger, O., Hesser, J., and Männer, R. (1995). "Automatic design of truss structures using evolutionary algorithms." *Proceedings of the Second IEEE International Conference on Evolutionary Computation (ICEC'95)*, Perth, Australia, 143-149.

Cohoon, J. P., Hegde, S. U., Martin, W. N., and Richards, D. S. (1987). "Punctuated equilibria: a parallel genetic algorithm." *Proceedings of the Second International Conference on Genetic Algorithms (ICGA'87)*, J. J. Grefenstette, ed., Cambridge, MA, USA, 148–154.

Collins, R. J., and Jefferson, D. R. (1991). "Selection in massively parallel genetic algorithms." *Proceedings of the Fourth International Conference on Genetic Algorithms (ICGA'91)*, R. K. Belew and L. B. Booker, eds., San Diego, CA, USA, 249-256.

De Jong, K. A. (to appear). *Evolutionary computation: a unified approach*, MIT Press, Cambridge, MA.

Dimou, C. K., and Koumousis, V. K. (2003). "Genetic algorithms in competitive environments." *Journal of Computing in Civil Engineering*, 17(3), 142-149.

Eshelman, L. J. (1991). "The CHC adaptive search algorithm: how to have safe search when engaging in nontraditional genetic recombination." *Proceedings of the Second Workshop on Foundations of Genetic Algorithms*, G. J. E. Rawlins, ed., Vail, CO, USA, 265--283.

Fogel, L. J., Owens, A. J., and Walsh, M. J. (1966). *Artificial Intelligence through simulated evolution*, John Wiley, Chichester, UK.

Goldberg, D. E., and Samtani, M. (1986). "Engineering optimization via genetic algorithm." *Proceedings of the Ninth Conference on Electronic Computation*, University of Alabama, Birmingham, 471-482.

Grierson, D. E. (1989). "Computer-automated optimal design for structural steel frameworks." *Proceedings of the NATO ASI Conference on Optimization and Decision Support Systems in Civil Engineering*, B. H. V. Topping, ed., Edinburgh, UK, 327-354.

Grierson, D. E., and Pak, W. (1993). "Discrete optimal design using a genetic algorithm." Topology Design of Structures, M. P. Bendsoe and C. A. Mota Soares, eds., Kluwer Academic Publishers, Dordrecht, The Netherlands, 89-102.

Hajela, P. (1990). "Genetic search - an approach to the nonconvex optimization problem." *AIAA Journal*, 26, 1205-1212.

Hajela, P., and Lee, E. (1995). "Genetic algorithms in truss topological optimization." *Journal of Solids and Structures*, 32(22), 3341-3357.

Holland, J. H. (1975). *Adaptation in natural and artificial systems*, University of Michigan Press, Ann Arbor, Michigan.

Holland, J. H. (1994). "Echoing emergence: objectives, rough definitions, and speculations for Echo-class models." Complexity: Metaphors, Models and Reality, G. Cowan, D. Pines, and D. Meltzer, eds., Addison-Wesley, Reading, MA.

Kicinger, R., Arciszewski, T., and De Jong, K. A. (2004a). "Evolutionary computation and structural design: a survey of the state of the art." *Computers & Structures*, submitted.

Kicinger, R., Arciszewski, T., and De Jong, K. A. (2004b). "Evolutionary designing of steel structures in tall buildings." *Journal of Computing in Civil Engineering*(tentatively approved).

Koza, J. R. (1992). *Genetic programming : on the programming of computers by means of natural selection*, MIT Press, Cambridge, Mass.

Luke, S. (2000). "Issues in scaling genetic programming: breeding strategies, tree generation, and code bloat," Ph.D. Dissertation, Department of Computer Science, University of Maryland, College Park, Maryland.

Murawski, K., Arciszewski, T., and De Jong, K. A. (2001). "Evolutionary computation in structural design." *Journal of Engineering with Computers*, 16, 275-286.

Parmee, I. C. (2001). *Evolutionary and adaptive computing in engineering design*, Springer, London, New York.

Rajan, S. D. (1995). "Sizing, shape, and topology design optimization of trusses using genetic algorithm." *Journal of Structural Engineering*, 121, 1480-1487.

Rechenberg, I. (1965). *Cybernetic solution path of an experimental problem*, Royal Aircraft Establishment, Farnborough, UK.

Sarma, K. C., and Adeli, H. (2001). "Bilevel parallel genetic algorithms for optimization of large steel structures." *Computer-Aided Civil and Infrastructure Engineering*, 16, 295-304.

Schwefel, H.-P. (1965). "Kybernetische Evolution als Strategie der experimentelen Forschung in der Stromungstechnik," Master's thesis, Hermann Föttinger Institute for Hydrodynamics, Technical University of Berlin.

Shankar, N., and Hajela, P. (1991). "Heuristics driven strategies for near-optimal structural topology development." Artificial Intelligence and Structural Engineering, B. H. V. Topping, ed., Civil-Comp Press, Oxford, UK, 219-226.

Soh, C. K., and Yang, Y. (2001). "Genetic programming-based approach for structural optimization." *Journal of Computing in Civil Engineering*, 31, 31-37.

Topping, B. H. V., and de Barros Leite, J. P. (1998). "Parallel genetic models for structural optimization." *Engineering Optimization*, 31(1), 65-99.