ADVANTAGES OF EVOLUTIONARY COMPUTATION USED FOR EXPLORATION IN THE CREATIVE DESIGN PROCESS

Peter von Buelow
College of Architecture and Urban Planning, University of Michigan Ann Arbor, Michigan, USA

In early phases of design a wide exploration of the design space is crucial to the development of creative solutions. In this regard, Evolutionary Computation (EC), and in particular Genetic Algorithms, contain several qualities that can enhance exploration by opening the search process beyond the focus of finding a single "best" solution. Over the years many researchers in the area of creative thinking including Gordon, de Bono, Parnes, Osborn and others, have suggested design strategies that have interesting parallels in EC processes. For instance, a well known inhibitor of creative thinking is design fixation, where the suggestion of a particular solution makes it difficult to imagine other good solutions. Unlike many other computational search algorithms, EC methods work with populations of "fairly good" solutions. Therefore, there is less danger that creativity will be harmed by design fixation on one "best" solution. This paper shows through a specific example of a truss bridge how an EC based design exploration program can aid the designer by providing a selection of "pretty good" solutions rather than a single optimal solution. Other aspects of the EC program are also discussed including drawbacks to the method such as computational intensity as well as directions of future development.

Key Words: *genetic, evolutionary, truss, topology, exploration, optimization, morphogogy*

1. Introduction

Traditionally, the design process has been recognised to proceed in phases. In architectural engineering these phases usually include: Schematic design, Design development and Construction documents (Cryer, 1994). This paper focuses mainly on the early schematic or conceptual phase. In this phase the problem itself is usually not fully defined (Crossley, 1999). Herbert Simon defines the typical "ill structured" design problem as "a problem that admits restructuring through the introduction of such new resources" (Simon, 1973). Designers in most fields readily recognize this as the typical mode of operation in conceptual design. In the early phase, design parameters are established and defined as variable or fixed (Parmee, 2001). As the designer makes continuing decisions about each of these parameters (moving them from variable to fixed), the solution space becomes more restricted, with less freedom for variation in subsequent choices. As more parameters are fixed, the designer has more knowledge about the eventual solution (Hale, 1996). Of course, as a consequence of each decision, the design proceeds in a certain direction. The final solution is determined incrementally by each decision made along the way.

To arrive at a good solution, the designer must explore various possibilities and alternatives at each decision point. This exploration is crucial to finding good solutions (Gero, 1994). Creativity in design can, in one sense, be seen as adept exploration. That is, many aspects of creativity can be described as some type of exploration. Margaret A. Boden, Professor of Philosophy and Psychology at the University of Sussex, has written much about the mechanics creativity. She describes creativity in terms of exploration.

> ... examples show that exploration often leads to novel ideas. Indeed, it often leads to ideas, such as new forms of harmonic modulation, that are normally called creative. In that sense, then, conceptual exploration is a form of creativity. (Boden, 1994)

The path to the better solutions is often missed when the designer makes choices without thorough exploration. One of the greatest hindrances to thorough exploration is design fixation (Gordon, 1961, Purcell and Gero, 1996). Fixation occurs when one "falls in love" with a solution before exploration is complete. The designer sees, or knows of, a solution that is "perfect" for the problem being solved. At that point it is very difficult for the designer to conceive of other possible solutions because of the fixation on this "perfect" solution. As a result, exploration suffers, and many of the steps that are part of that exploration are omitted.

Poor exploration may occur when optimization methods, which are more appropriate for later phases of design, are used too early in the process. The mere selection of parameters may define the problem in a way which obscures many good solutions. If limiting decisions are too quickly assumed, fixation on the part of the designer is hard to avoid when one optimized, "best" solution is revealed. If the tool only offers one solution, there is not much opportunity for creative choice.

Evolutionary Computation (EC), on the other hand, is based on a progressive morphogenic process, and generally operates on populations of solutions. As a result, there is always a pool of solutions made up of parents (selected from the past generation) and children produced by "breeding," which compete with one another to remain in the next generation. Each generation consists of selected individuals from this pool of solutions called a population. One generation follows the next in a cyclic manner20.1913 0 TD(0.000

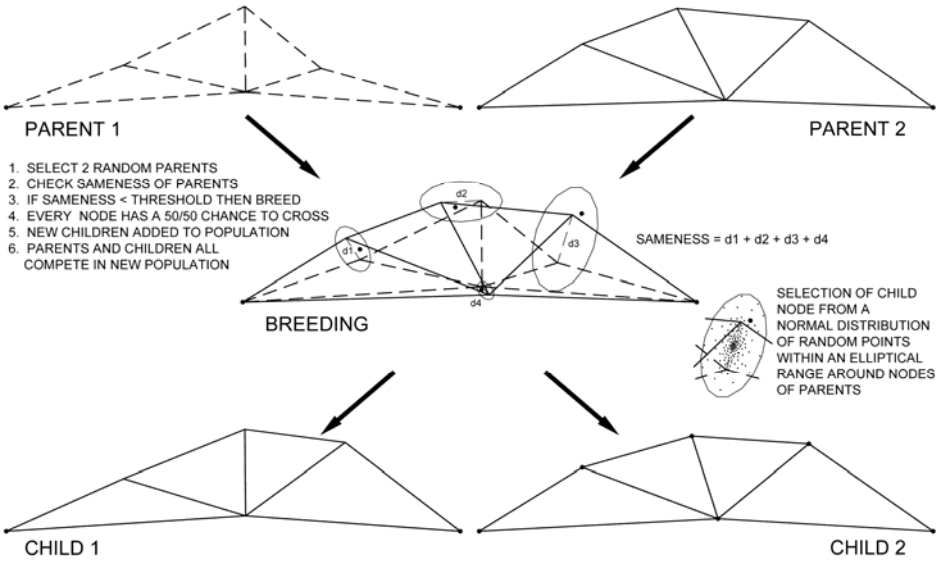


Fig. 1 Breeding of geometry in CHC-GA cycle.

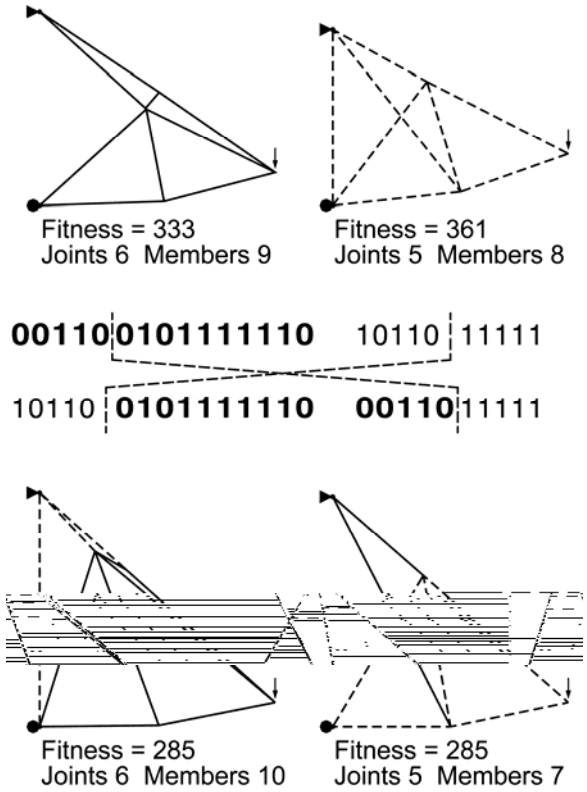


Fig. 2 Breeding of topology in ES-(μ+λ) cycle.

As is shown in Figures 1 and 2, the chromosome coding and breeding procedures are somewhat different in the geometry and topology GA's. This is briefly described in Section 3 below. A more detailed description of the mechanics of the IGDT has recently been published in a book by the author (von Buelow, 2007). The book also describes mutation procedures used by the IGDT. This section presents the basic objectives of the program, and shows why EC is a particularly good method to meet these objectives. In conceptual design as for the IGDT there are basically two types of objectives that guide the exploration: quantitative objectives and qualitative objectives.

2.1. Quantitative Objectives

For the IGDT to be useful as a means of exploring structural form, it must be able to assess and rank, quantitative parameters of any structural form it might evolve. These parameters define the fitness function for the GA. For the examples presented in this article, parameters of weight (material efficiency) and complexity of form (geometric efficiency) have been used. Although it is possible to include singularly or collectively any number of parameters, care is recommended, as the consideration of too many parameters tends to obscure the meaning of the resulting solution.

It is, of course, possible for the IGDT to run using solely predetermined parameters to guide it in exploring the design space. Even in this automatic mode, the IGDT is a more explorative design tool than traditional optimization methods, because it continually presents populations, which represent a range of good solutions, rather than one instance of an "optimal" solution. In addition, the automatic mode, which runs unattended, makes it possible to sometimes reach deeper into the solution space in a shorter amount of time. For example, the program can be set to run unattended through a few hundred topology generations allowing the designer to view a larger variety of solutions. The graphic output from the automatic mode can be filtered to remove any duplicate solutions, and show only solutions that pass a given fitness level (the better solutions). In this way it is possible to control the amount of output, and make it easier to see the dominant patterns. It is then possible to proceed from any point reached with the automatic mode by continuing with the interactive mode.

The goal in the automatic mode is to discover as many good solutions to the predetermined quantitative parameters as possible. In this way the user's creative understanding of the solution space is expanded.

2.2. Qualitative Objectives

EC methods, including the IGDT, have the potential to use qualitative objectives as well as quantitative objectives. This is accomplished through human interaction to supply assessment and selection as the program is running. Qualitative objectives are parameters that are recognizable by the designer, but do not lend themselves readily to quantifiable definition. These include parameters such as aesthetic value, meaning, analogous form, etc. It is characteristic of qualitative values, that they are readily ranked by comparison within a group of solutions, but present difficulties for most people to describe independently. For example, given a set of images, it is not necessarily a difficult task to choose the most aesthetically pleasing one. But it is quite a different task to describe precisely why the choice was made, in a way that defines a consistent rule that can be applied to all future choices. Guided by qualitative objectives the user makes interactive choices, which provide the ranking of the structures found by the IGDT. In this way the IGDT can be guided by the user's own creative curiosity and instinct in exploring the solution space.

Considering the qualitative objectives together with the quantitative objectives, the IGDT can be considered a multi-objective search tool. Multi-Objective Evolutionary Algorithms (MOEA) have been used in a variety of applications including engineering optimization (Coello Coello, 2004). In recent years several approaches to MOEA architecture have been put forward (Mehr & Azarm, 2003) which are generally distinguished in the treatment of the fitness function or the population composition and

selection algorithms. The IGDT can be considered an *implicit* multi-objective search tool when guided by the designer's non-coded selection criteria, but it is not explicitly coded to find Pareto non-dominated solutions (a Pareto set).

2.3. Selection without Fixation

Multiple solutions are also important in revealing solutions that respond to objectives that have not been explicitly expressed by the designer. The designer may find a solution that is only marginally less efficient in terms of fitness, but offers either strong visual advantages or makes the designer aware

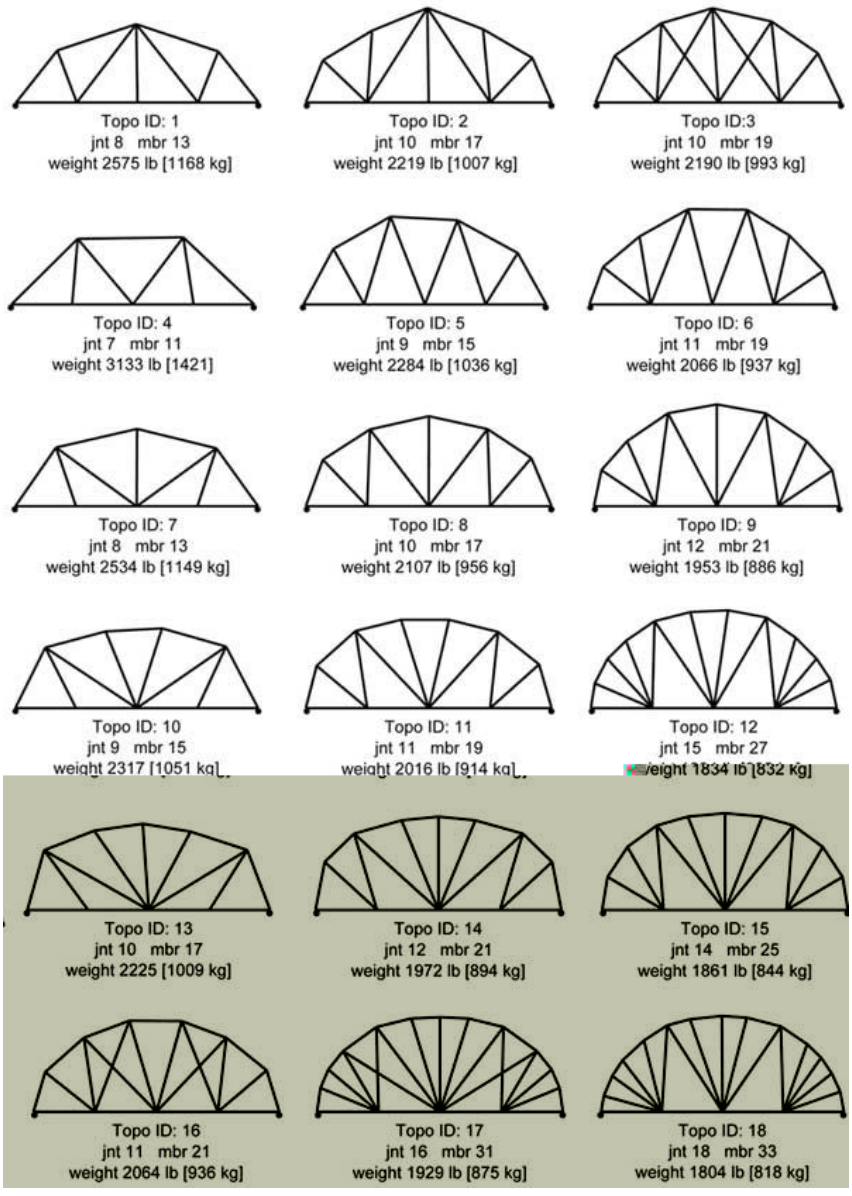


Fig. 3 Selected results from the automatic mode of the IGDT. The results are organized here into topological families (rows) which gain in complexity from left to right.

of missing objectives in the problem formulation. It is interesting that given a selection of "fairly good" solutions, the "best" solution, purely in terms of the stated fitness function, is often not the one that is chosen (Parmee, 2001).

Figure 3 shows an example of actual results from an IGDT run. The problem in this case is the truss bridge described later in Section 4. Here we can see that the overall lightest solution (the dominant fitness used was weight) is number 18, but its visual complexity, slender diagonals, and many connections might make it a less desirable choice. Having the other 17 solutions to choose from is definitely an aid to the designer. Had the program only revealed the "best" solution, the other "fairly good" (perhaps better overall) solutions, albeit a bit heavier, would have remained unseen and unknown. Of course, this is the situation that traditional multi-objective methods attempt to address. However, in traditional multi-objective optimization, all of the objective functions need to be explicitly determined and defined in advance, which is not the case with the IGDT. As was pointed out above, in the early phase of design, exploring a number of solutions actually helps to define the objectives of the problem. Even without multi-objectives being defined, the IGDT returns multi-solutions which can inform the designer of further objectives.

Fig. 4 A team of students using the IGDT in interactive mode to design a bridge.

2.4. Exploration

Exploration is essential if a creative solution is to be found. The first level of exploration is satisfied by providing for multi-solutions described above. But for better exploration, the program needs to be interactive and responsive to the direction of the designer. In recent years Interactive Evolutionary Computation (IEC) has become an area of research in itself (Takagi, 2001). In an IEC, selection from the EC population is made either solely by the user or by the user in conjunction with some predefined parameters. Rather than elitist or other coded selection algorithms, the choice of parents for the subsequent generation is made by the user directly. This allows the user to explore areas of the solution space based on non-coded criteria such as aesthetics. The IGDT can operate in this mode, or it can be set to run automatically and exclusively with a pre-set fitness function.

The drawback with IEC is usually related to time spent in making the selections or waiting time for analysis of the solutions bred from the selected parents – that is the delay time between opportunities

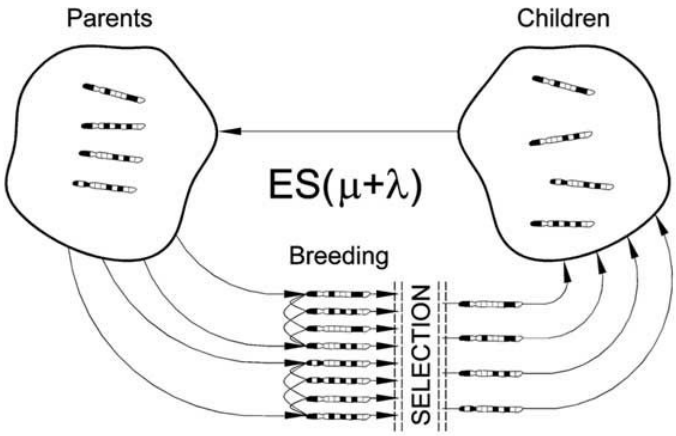


Fig. 5 Flow diagram of the ES-(μ+λ) used in the topology cycle.

for selections. In the automatic mode the IGDT, as typical with most EC implementations, runs through hundreds of generations in a cycle. It would be totally impractical to attempt to choose parents from so many generations by hand. A more productive use of IEC is attained when it is combined with some number of automatically selected generations. For instance, the designer might select by hand one generation, and then let the program run 10 or 20 generations without the designer's interaction. Then the designer could select another generation, and again let the program run several generations without interaction. The IGDT has in fact been setup to run in this way.

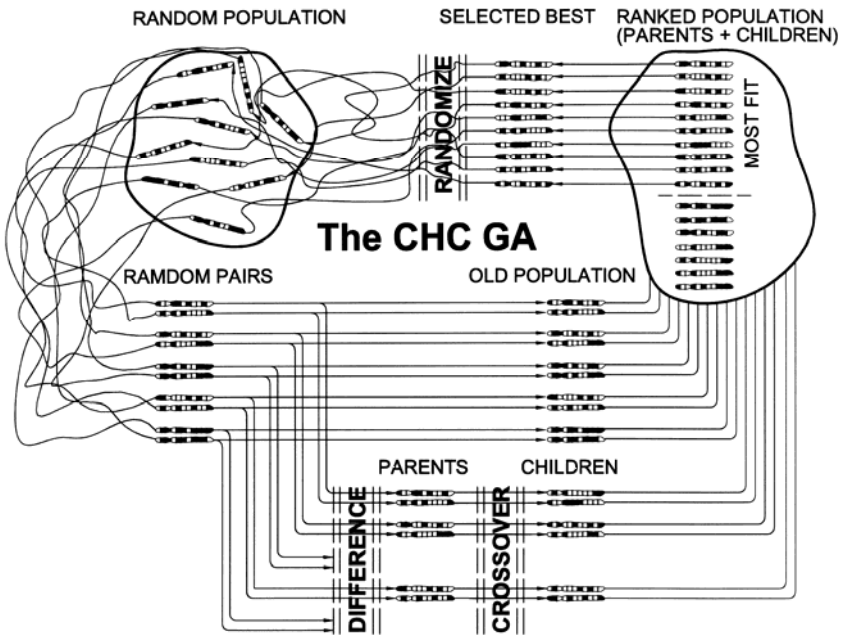


Fig. 6 Flow chart of the CHC-GA used in the geometry cycle.

3. Encoding Techniques

The IGDT presented in this paper is used in determining good structural forms for truss systems. In order to do this, it explores topologies and geometries in a way which exposes a set of "fairly good" solutions to the designer (shown in Figure 3). The search and exploration engine used in the IGDT has been patterned after the CHC Genetic Algorithm, developed by Larry J. Eshelman and J. D. Schaffer of Philips Laboratories (Eshelman, 1991). The CHC-GA combines a highly disruptive recombination operator, which allows for thorough exploration, with an elitist selection operator for good convergence velocity. CHC is successful with small populations (ca. 50). In addition, it makes use of a breeding filter that allows only the fraction of the parent population which promises more productive pairings to produce children. This makes the process of breeding more efficient. These operators combine to give the CHC good exploration qualities, while minimizing computation. In the IGDT, good exploration qualities are desirable to provide the designer with a broad view of the solution space. Also, for the process to function interactively, speed is needed in finding good solutions. Since the CHC offers advantages in thorough exploration and rapid convergence, it was chosen as the basis for the IGDT. Figure 6 shows the CHC-GA cycle.

3.1. Geometry and Topology

In the design of discrete structures like trusses, it is necessary to distinguish between different geometries of the same topology as well as different topologies. The topology of a truss is, in this case, the ordering of joints and members – how many of each and their relative connectivity. The geometry of the truss is described by the actual nodal coordinates of each joint. Figure 7 shows the distinction between geometry and topology.

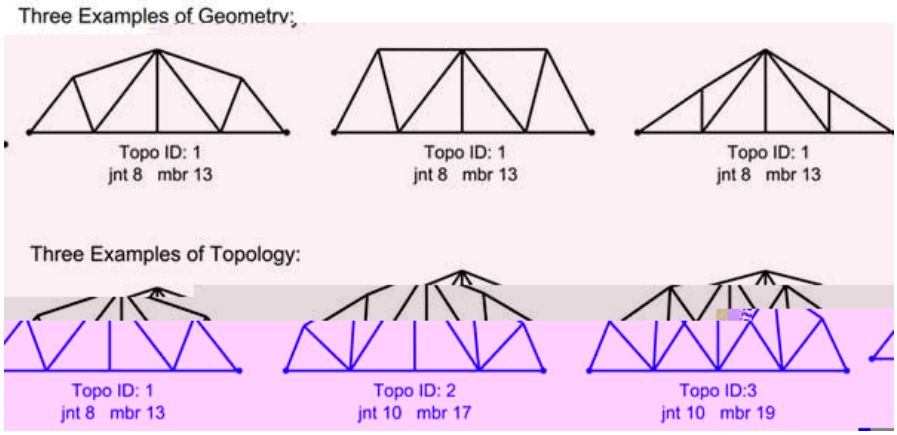


Fig. 7 Distinction of forms showing in the top row three different geometries with the same topology, and in the bottom row three different topologies.

Both different geometries with the same topology as well as the range of topologies need to be explored. This is accomplished in the IGDT by a nested pair of EC cycles. The outer cycle explores topology. Each topology cycle contains many inner (nested) geometry cycles. The geometry EC is based on the CHC-GA discussed above and shown in Figure 6. The topology optimization is actually a little simpler, and is based on an Evolutionary Strategy developed by Schwefel (Bäck et al., 1992, Bäck, 1996) and called an ES-(μ+λ). Figure 5 shows the flow diagram for the topology cycle. For problems of the scale shown in this article, a geometry population of 50 parents and a topology

population of 20 parents work well. More complex structures with a greater number of joints and members require larger populations. The size of the population is based on the amount of "genetic" material that is evolving. To facilitate recombination (breeding) and mutation operations, parent individuals in both topology and geometry cycles need to be described as "chromosomes" or coded strings. In order to reach any point in space, real (base 10) numbers are used to code the geometry nodes. Because topology members are either present or not present, a binary coding is convenient and efficient. Figures 1 and 2 show how geometry and topology members are coded and bred.

The binary strings shown in Figure 2 which describe the topology, are derived from the incidence matrix used in the finite element analysis of the structural systems. This has a great advantage over the use of a ground structure which is commonly used for other optimization methods. The advantage is that the incidence matrix can describe any topology without reference to geometry in the most efficient way (least number of digits). This efficiency is an important consideration for EC methods that otherwise have the drawback of being computationally intensive.

The separation of topology and geometry exploration into nested cycles has the additional advantage that it lends itself well to coding for parallel processing. This allows for substantial processing speed increases which can allow relatively large systems to be interactively explored. The current version of the IGDT uses message passing libraries of the Parallel Virtual Machine software (Geist et al., 1994), and runs on a cluster of Linux workstations, (currently 100, p3 and p4 machines, although the example below only used 30). The interactive processing used a topology generation of 10 parents plus 10 children. This works well for interactive viewing on a computer monitor (see Figure 4). Each cycle was run in under 10 minutes. In the automatic mode 5 topology cycles with populations of 20 parents, each run to convergence and restart in about 1.5 hours.

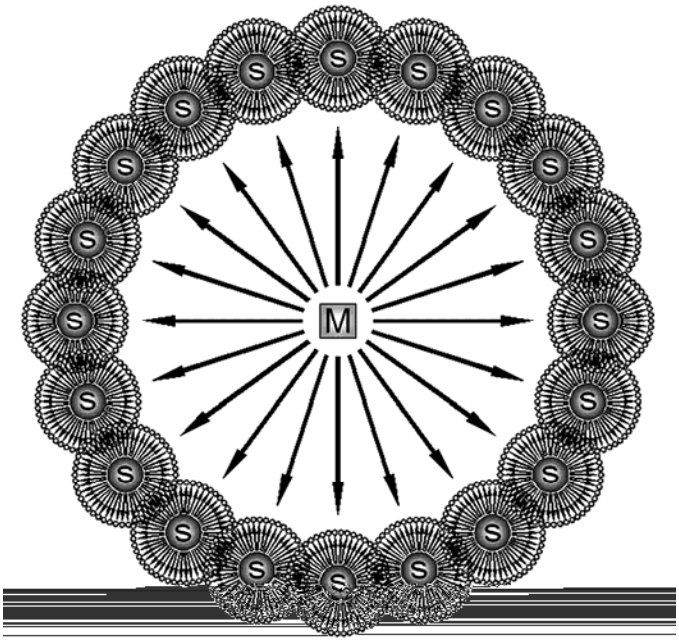
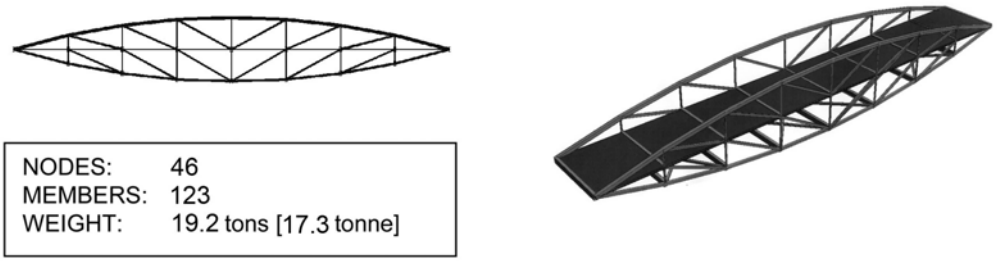


Fig. 8 A diagram of the parallel processing configuration. The "M" is the master process that collects and sorts the best solutions. Each "S" is a slave processes - one spawned for each topology. The small circles surrounding each "S" are the geometry individuals.

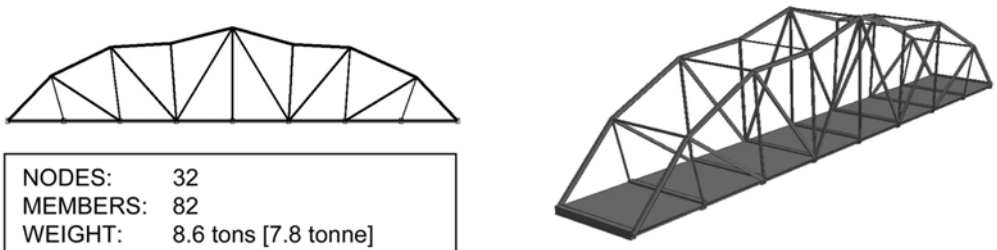
4. An Example Design Problem – Truss Bridges

To initially demonstrate the IGDT, an example is used which can be compared with other published solutions. The solutions in Figure 3 have the same geometric constraints and loadings as a similar example which uses the homogenous method by Ma et al. (1995). The total span is 60 feet (18.3 m), and the truss is loaded at the three panel nodes with 23 ton (200 kN) point loads. Although the current coding allows for multiple load cases (e.g. moving truck loads) a static load case was used in order to allow for a better comparison with the other published results. In addition, the actual weight of the structure was added into the analysis. Members were sized as steel pipe sections to meet the criteria of the AISC-ASD steel code, including buckling.



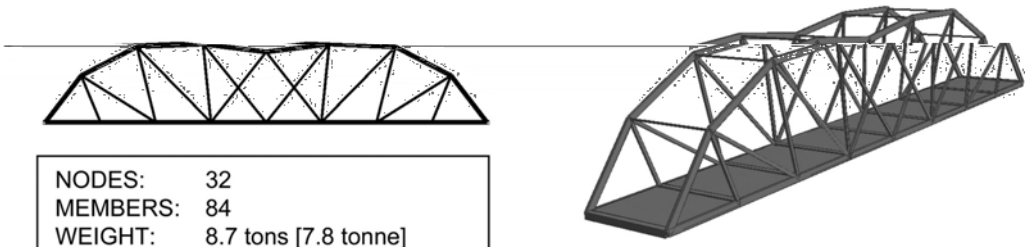
NODES:	46
MEMBERS:	123
WEIGHT:	19.2 tons [17.3 tonne]

Design by Jong Yun, Hyuntak Oh, Sang Ahn, Sung Kim, Nayara Islam.



NODES:	32
MEMBERS:	82
WEIGHT:	8.6 tons [7.8 tonne]

Design by Xuezhen Chen and Jin Jeon.



NODES:	32
MEMBERS:	84
WEIGHT:	8.7 tons [7.8 tonne]

Design by Lauren Bostic, William Marquez, Jennifer Siegel, and Faye Whittemore.

Fig. 9 Three student designs found using the interactive mode of the IGDT.

4.1. Automatic Mode

A threshold fitness was chosen for the "fairly good" set which delivered about 50 solutions. These represented 50 different peak values gleaned from the search space. This is a size which is manageable for visual inspection and still captures a large degree of variation in the "fairly good" range. From this set, the 18 were selected which are shown in Figure 3. They represent a good set of topologies that the designer could use for further exploration. These results were all attained using only the automatic mode in order to better illustrate what the IGDT is capable of on its own. Of course, in the interactive mode, any individual or number of these solutions could be chosen for a more detailed exploration in that area of the solution space.

In viewing the results, familiar patterns are recognized, and names could be given to several. Topology 7 is a Waddell truss; topology 8 is a Parker; topology 5 is the Warren; topology 3 is a Bowstring Arch. As pointed out by Boden (1994), Koza et al. (1999), and others, the fact that these topologies are successful enough to have names, indicates the IGDT is finding good solutions.

Fig. 10 Site rendering of the third design from Figure 8 (by Bostic, Marquez, Siegel, and Whittemore).

4.2. Interactive Mode

The example used to demonstrate the interactive mode of the IGDT was executed in the context of a graduate structures course at the University of Michigan, School of Architecture. The problem is another bridge, although a bit larger and more complex than the previous example. There was in fact an actual site with a span of 118.7 feet (36.2 m). An AASHTO HS 20 moving truck load was applied at the six panel locations plus a lane load.

The students used the IGDT to explore possible truss forms. One can assume that because they were architecture students they had more than the average interest in the appearance of the final design, and they explored topologies that met their own aesthetic criteria. Due to time constraints the students were only able to use the IGDT in one session to generate forms. The sessions ran about 30 minutes which allowed for between 4 and 6 interactive cycles. A topology population size of 10 parents plus 10 children was used. Each group made the run independently and without knowledge of the work of the other groups.

There were seven groups altogether and none of them chose the same design topology. Most designs selected tended to be in the same weight range as the original design, with some heavier and some lighter. Finding a lighter solution was an obvious objective, but some students treated weight more as a secondary objective compared to some form related requirement. For example the first bridge shown in Figure 9 was a deliberate attempt to find a solution that included truss work above and below the deck.

Figure 9 shows three of the final designs. Because only four cycles were run, the scope of exploration was rather limited, but this was the level of time investment expected for a simple class assignment. In actual practice, it is likely that more time could be invested in the initial design phase, and a more detailed exploration be made. Nonetheless, the students were able to use the tool, and arrived at reasonably efficient forms that met the criteria. Although a complete analysis with transverse loads and modal frequencies was not made, the students did develop a complete bridge frame and sized the steel members for the AASHTO loadings.

5. Conclusion

Evolutionary Computation has been shown to be especially well suited in developing tools for use in early phases of form exploration and design. In taking advantage of the inherent use of populations in Evolutionary Computation methods, the design space can be better explored without risking design fixation. An example application, the IGDT, has been shown to be effective in producing a variety of solutions to structural engineering problems. This allows the designer to make selections based on coded objectives of least weight, while enabling an expanded exploration of the solution space based on non-coded criteria, such as visual appearance.

Currently the IGDT is limited to truss structures; however, the same method could be applied to a wide range of structural forms by expanding the analysis module. There are plans to do this in the near future. One of the reasons the program was initially limited to trusses is that the computational cost of truss analysis is less than the cost of other frames or shells. With the advent of faster machines and parallel processing, this is less of a problem. In our case, the recently enlarged parallel processing cluster, now at 100 nodes, also make complex problems more practical. As the number of members in the structure increases, it is generally necessary to increase the population size of the GA to ensure adequate exploration of the solution space. Larger clusters are better able to deal with the larger population sizes. In the end, speed of computation is a limiting factor. When the interactive mode of the program is being used, it is better if the solution can be generated within about 10 minutes. In the automatic mode, runs can be left overnight, but run times which take weeks are just not very practical.

As is the case with most optimization procedures, in office practice, the time investment usually makes their application uneconomical for routine use. The time schedule constraints placed on typical

architectural projects, simply cannot allow a detailed exploration of multiple solutions. This is why architectural researchers commonly pursue first their research topic, and then find an opportunity to apply the results. Nonetheless, some projects make better candidates for detailed structural consideration than others. One example would be a project in which some structural element plays a particularly significant roll. Another example would be a project in which the repetition of some element makes it worthwhile to give that element particularly detailed consideration.

It is to the advantage of the IGDT concept that the problem input is no more complex than a typical FEA program (e.g., STAAD-Pro) routinely used in offices. There are numerous examples on the market where FEA programs have been coupled with CAD programs and a rather intuitive input interface, allowing even the beginning student ready access. As it now stands, the major limitation to the IGDT concept is the computational intensity inherent to GAs. Although this translates into long run times with today's technology, the time may soon arrive when computing speeds increase sufficiently to make GA applications more practical.

In principal, the approach would certainly have application in other areas where some objective criteria need to be balanced against a visual assessment. But in any case, there would need to be a clear underlying and relatively simple algorithm which defines the geometry of the design. This is why examples involving clearly defined structural systems work well. In those areas, methods like the IGDT will be better at finding solutions which go beyond traditional optimization to enhance the designer's own creativity.

5.1. References

Bäck, T., Hoffmeister, F., and Schwefel, H.P., 1992, "A survey of evolution strategies," in Proceedings of the fourth international conference on genetic algorithms, Morgan Kaufmann. San Mateo, California, pp. 2-9.

Bäck, T., 1996, Evolutionary algorithms in theory and practice, Oxford University Press, Oxford, England.

Boden, M.A., 1994, "What is creativity?" in Boden (ed.), Dimensions of creativity, The MIT Press, Cambridge, Massachusetts, USA.

von Buelow, P, 2007, Genetically Engineered Architecture - Design Exploration with Evolutionary Computation, VDM Verlag Dr. Mueller, Saarbrücken, Germany.

Coello Coello, C.A., 2004, "An Introduction to MOEAs and Their Applications", in: Coello Coello & Lamont (eds) Applications of Multi-Objective Evolutionary Algorithms, Advances in Natural Computation, Vol.1., World Scientific. pp. 1-28.

Crossley, W., 1999, "Using genetic algorithms to encourage engineering design creativity," in Roy, Furuhashi and Chawdhry (eds.) Advances in soft computing, Springer, London, pp. 69-84.

Cryer, John N., 1994, "Design Team Agreements 3.43", in: Haviland (ed.) The Architect's Handbook of Professional Practice, Vol. 2 The Project. AIA Press, Washington, USA.

Eshelman, L., 1991, "The CHC adaptive search algorithm: how to have safe search when engaging in nontraditional genetic recombination," in Rawlins, G. (ed) Foundations of genetic algorithms, Morgan Kaufmann, San Mateo, California, USA.

Geist, A., Beguelin, A., Dongarra, J., Jiang, W., Manchek, R., and Sunderam, V., 1994, PVM: parallel virtual machine. a users' guide and tutorial for networked parallel computing, The MIT Press, Cambridge, Massachusetts, USA.

Gero, J.S., 1994, "Towards a model of exploration in computer-aided design," in Gero and Tyugu (eds.), Formal design methods for CAD, North-Holland, Amsterdam, pp. 315-336.

Gordon, W.J.J., 1961, Synectics: the development of creative capacity, Harper & Row, New York.

Hale, M.A., 1996, An open computing infrastructure that facilitates integrated product and process development from a decision-based perspective, Ph.D. Dissertation, Georgia Institute of Technology, School of Aerospace Engineering. Atlanta, USA.

Ma Z.-D., Kikuchi N., and Cheng H.-C., 1995, “Topological design for vibrating structures,” Computer Methods in Applied Mechanics and Engineering, Vol. 121, No. 1, pp.259-280.

Mehr, A.F.; Azarm, S., 2003, "An Information-Theoretic Performance Metric for Quality Assessment of Multi-Objective Optimization Solution Sets". in: ASME Journal of Mechanical Design, Vol. 125, No. 4, pp. 655-663.

Koza, J.R., Bennett III, F.H, Andre, D., and Keane, M.A. , 1999, Genetic programming III: Darwinian invention and problem solving, Morgan Kaufmann, San Francisco.

Parmee, I.C, 2001, “Poor-definition, uncertainty, and human factors – satisfying multiple objectives in real-world decision-making environments,” in Zitsler et.al. (eds.) EMO 2001 Springer-Verlag, Berlin, pp. 52-66.

Purcell, T.A. and Gero, J.S., 1996, “Design and other types of fixation,” Design studies, Vol 17, No. 4, pp. 363-383.

Simon, H.A., 1973, “The structure of ill structured problems,” Artificial intelligence, Vol 4, No 3-4, pp. 181-201.

Takagi, H, 2001, “Interactive evolutionary computation: fusion of the capabilities of EC optimization and human evaluation,” in Proceedings of the IEEE, Vol. 89, No .9, pp. 1275-1296.

6. Author’s Biography

Dr. Peter von Buelow is on the TCAUP faculty at the University of Michigan. He holds degrees in both architecture and engineering as well as professional certification in both fields. His research centers on the use of Evolutionary Computation for structural optimization and form finding in the design of structural systems for architecture. Professionally, he has worked for RFR-Stuttgart, Greiner Engineering, SL – Rash, and architectural firms in Bonn and Hamburg. He was also a Fulbright Scholar at IL at the University of Stuttgart (Frei Otto director) where he also received his doctorate (Werner Sobek director). For more information visit: www.umich.edu/~pvbuelow.