REPRESENTATION AND ITS IMPACT ON TOPOLOGICAL SEARCH IN EVOLUTIONARY COMPUTATION

Yu Zhang [1], Kai Wang[1], David Shaw[1], John Miles[1], Ian Parmee[2] & Alan Kwan[1]

ABSTRACT

Evolutionary methods such as genetic algorithms are proving to be powerful tools for solving Engineering problems. The generally accepted wisdom is that the only connection between the algorithm and the real problem is via the fitness function. This results from the common practice of using a parametric representation. For some problems this works well but when there is a need to use the algorithm to reason about shape or topology, a parametric representation can be limiting. Topological reasoning problems are important in many areas of Architecture, Engineering and Construction (AEC) industry and so it is vital that appropriately formed problem representations are used. In this paper, there is an initial discussion on the use of parameters and ground structures and it is shown how these can restrict the search for good solutions. The paper then looks at recent work by the authors using voxels, graphs, computational geometry and generative shape theory as representation strategies. It is shown how for various classes of problems, these can be useful techniques.

KEY WORDS

Evolutionary Computation, Representation, Shape, Topology, Design

INTRODUCTION

Design is a major step during the process of creating a new product. For many disciplines, the resulting product is a three dimensional object or objects and so reasoning about the geometrical form of the product and the relationships between its components is a major part of the design process. To date, the availability of computational tools to help designers with topological / spatial reasoning has been limited. There are many forms of topological reasoning that occur in, for example, the design of a building. There is the proximity relationship between the different rooms and spaces within the building, there is the relationship between these and the structure, the influence of the shape of the building on its heating / ventilation and lighting characteristics, the need to provide space for the building services and the relationship between the building and its surroundings. Each of these is a complex design problem in its own right and if building design is to get somewhere close to

---

[1] Cardiff School of Engineering, Cardiff University, The Parade, Cardiff CF24 3AA UK, Phone 442920875694, FAX 442920874597, email milesjc@cf.ac.uk

[2] ACDDM Lab, School of Computer Science, University West of England, Coldharbour Lane, Bristol, BS16 1QY, UK, Phone 441173283137, FAX 441173282587, email ian.parmee@uwe.ac.uk

an "optimum" solution, then they all have to be considered in a single, multi-criteria search process. At the moment the available reasoning techniques used in computational search are no where near powerful enough for this to be possible.

Designers have traditionally used heuristics to search for good design solutions. Over the past two decades, it has become apparent that design search spaces are so huge, that searches which just use heuristics have an extremely small chance of coming up with solutions which are any better than satisficing. For example, when considering the conceptual design of a typical office building, Khajehpour & Grierson (2003) estimate there are 167 million design options just for the major structural systems although not all of these are feasible. With such huge search spaces, a major problem is conveying information about the search space to the designer(s). Obviously it isn't possible to visualise the whole search space but techniques for looking at areas of high performance have been developed by Parmee (2004)

For at least 4 decades, methods of searching for good shape and topological design solutions have been investigated. Many initial attempts were linked to finite element analysis and used techniques which started from an initial design and then gradually removed material in areas of low stress (e.g. Rozvany, 1992). Obviously with such a procedure, the shape of the initial design is a crucial factor in determining how good the resulting solution is and given the limited search, the process cannot be considered as being optimisation. The later development of the homogenisation method by Bendsoe and Kikuchi (1988) is a much more successful way of using finite element analysis which can be used to search on any criterion which can be expressed using finite elements. The method is sensitive to the choice of discretisation scheme and does rely on the assumption that anisotropic materials of infinitely varying density are available. This obviously isn't feasible and so there is a need for interpretation which results in a final structure which is someway from that produced by the method. Also when considering topology in design there are many aspects which cannot be represented by finite element analysis such as proximity relationships and aesthetics.

Analysis based techniques have produce some useful results, for example, the work of Pritchard (2004) coupling linear programming to structural analysis techniques to find good solutions for three dimensional truss structures. The problem with this type of technique is the limited ability of the algorithms used, to search the sort of highly complex search spaces that typically occur in design.

There is one class of search algorithm that has a proven ability to cope with complex search spaces and that is Evolutionary Computation (EC). An early example of using EC for structural design was the work of Jenkins (1993) who used a Genetic Algorithm (GA). Since then there have been numerous other examples including many on shape and topological search and optimisation. Most of the latter have used either a parameter based or a ground structure representation. Within this paper, it is argued that both of these limit the search procedure and potentially prevent it from finding good solutions. A number of different forms of representation are examined, these being based on work undertaken by the authors. The strengths and weaknesses of these alternative methods are discussed as is their appropriateness for various classes of problems. The methods that are covered in the paper are only those of which the authors have extensive experience so there are, for example, some techniques, such as shape grammars, which are not included.

REPRESENTATION

Within software engineering and computer science the word representation has many meanings and even to those with expertise in EC, there are differences of opinion as to what exactly the term representation covers within EC. For the purposes of this paper, the word representation will be defined as the technique used to describe the problem domain within the algorithm. This, however does not include the fitness / objective function but is restricted to the encoding procedure and how this relates to the actual problem.

Parameter Based Representation

For most applications of EC in design, parameters have been the main form of representation. For example Azid & Kwan (1999) use the X and Y locations of joints within a truss to search for the optimum topology. Miles et al (2001) use X and Y coordinates to represent the location of columns within buildings. For some classes of problems, where the topology of the final solution is reasonably fixed, then the use of parameters is a suitable technique but for all other types of problems, it should be avoided. The reason for this can be shown by the simple example given in fig.1

Fig.1 Parameter Representation Example

In a domain where the topology of the solution is unknown, then a representation which allows the algorithm to describe a rectangle would, using parameters, require an X and Y dimension. However to also allow for a solution where the answer is a circle would require a radius. One can of course go on adding further possible shapes and for each one there would have to be a separate parameter set and the algorithm would have to be able to recognise each one. It becomes even worse if one considers a population of possible solutions with different topologies and then considers how one would cope with typical operators such as crossover. So for search problems where the topology is a part of the search, the use of parameters is very restrictive and almost certainly will lead to a poor solution.

Ground Structures

Ground structures are mostly used when a truss is the preferred form. A ground structure then consists of a truss with nodes in fixed locations. Each node is initially connected to all the other nodes by a structural member. The “optimisation” process consists of selectively removing structural members and searching for that combination which best satisfies the search criteria (fig.2). The method has the attraction of being simple but by fixing the number of nodes and their location, the search is severely constrained and so the resulting structure is unlikely to be the optimum for the given criteria. Azid & Kwan (1999) present a development of the ground structure approach where the nodes are allowed to move and this does remove some of the restrictions but introduces a need for extensive repair which disrupts the search.

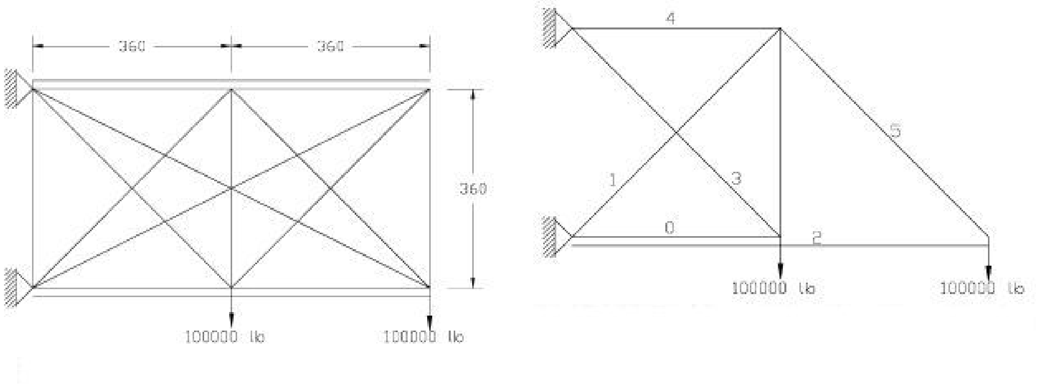


Figure 2 An example of "optimisation" using a ground structure. On the left is the ground structure and on the right is the "optimised" structure (Deb & Gulati, 2001)

Graphs

Various forms of graph based representation have been used with EC. As with ground structures, their application has mostly been to trusses although they can be used for other layout problems such as building floor plans. Graphs allow one to model the connectivity between nodes while varying the location of the nodes (Borkowski & Grabska, 1995). This gives them a distinct advantage over the ground structure method and generally for problems which involve linear and connected elements, a graph representation is very good. Yang (2000) presents a simple method of using a tree based structure to describe trusses (fig.3).

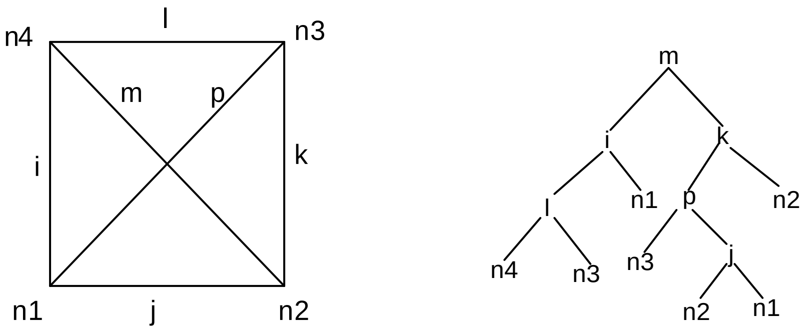
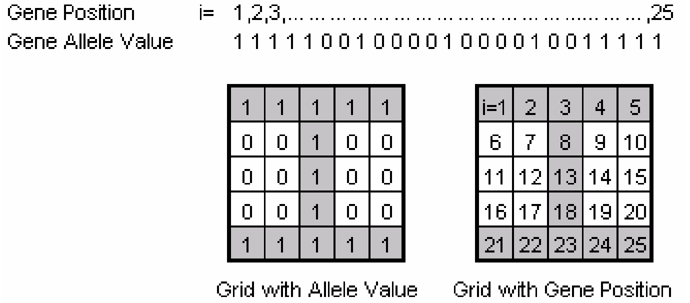


Figure 3: Truss Structure and Graph Representation

Voxels

Voxels are typically used to represent artefacts that, unlike trusses, have a relatively large solids to voids ratio. For example Griffiths & Miles (2003) used them in the search for optimal beam cross sections. A typical representation of a simple beam shape in a Genetic Algorithm (GA) is given in fig.4. An alternative representation for the same shape can be achieved by using a 2 dimensional array. In concept, voxels are easy to handle. If a GA is used then crossover is conceptually simple to implement as is mutation. However, as Zhang and Miles (2004) show, the choice of crossover mechanism has a significant impact on the ability of the algorithm to find good solutions. They show that the physical shape represented by the crossover mechanism affects the search ability of the algorithm and can influence the final shape. It is therefore vital in situations where the topology of the solution is unknown in advance to use crossover mechanisms with more than one "shape". So for example one might combine in the same search mechanisms which select a square shaped block of voxels with mechanisms which select vertical and horizontal strips. The choice of mechanism for any given operation would then be random. This impact of the operators on the solution is an important finding because as well as the fitness function and the representation, it shows that the operators can also have a link with the problem domain.



Gene Position i= 1,2,3,………………………………………………..……,25
Gene Allele Value 1 1 1 1 1 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 1 1 1 1 1

1	1	1	1	1
0	0	1	0	0
0	0	1	0	0
0	0	1	0	0
1	1	1	1	1

Grid with Allele Value

i=1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Grid with Gene Position

Figure 4: Example of voxel representation, GA string at the top (1 = material present, 0 – void). (Griffiths & Miles, 2003)

There are some significant problems when using voxels to search for good topological solutions. In such a search, it is usually desirable to have a solution with a high degree of connectivity between the voxels. This is particularly the case if load bearing properties are required. However, voxel-based representations tend to result in isolated voxels, jagged edges and structures with no clear load path. Griffiths & Miles (2003) show that with a well designed fitness function, plus suitable operators (e.g. crossover & mutation) that despite these problems, sensible solutions can be obtained. Fig. 5 shows how, starting from a random distribution of voxels, sensible structural forms can be generated.

Current work by the authors is looking at the use of voxels to generate three dimensional forms; this of course requires 3D voxels.

Computational Geometry

Computational Geometry is the collective name given to a host of techniques that can be used to generate and handle shapes (Shamos 1978, O'Rourke, 1998). Within Engineering the use of these techniques has largely been confined to mesh generation for numerical analysis software but it also has huge potential for topological search. Shaw et al (2005a, 2005b) have used computational geometry to design two structural forms, building floor plans and domes. Only the floor plans are discussed here.

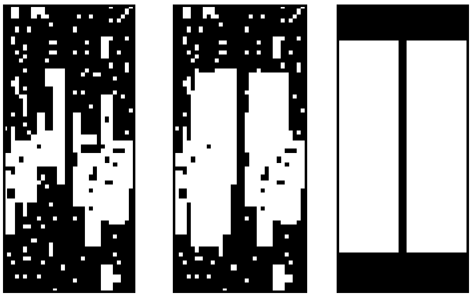


Figure 5: The evolution of an I beam from on the left, the best solution after 10 generations to, on the far right, the final answer. (Griffiths & Miles, 2003)

The search is directed towards finding a good arrangement of columns in a typical beam, column, slab type building such as a hotel or office. Various constraints come into play such as the size of work spaces within the building, keeping the column spaces as regular as possible, to minimise the number of section sizes that are required, and minimising beam depths thus keeping the building height and cladding costs within bounds. There is the possibility of the conflicting constraint of maximising building flexibility by having as much column free space as possible, thus increasing beam spans and depths. The result is a complex and highly constrained search space.

Early work on the generation of building floor plans used parametric representations (e.g. Miles et al,2001) and this worked well with buildings with rectangular floor plans but very few buildings are truly rectangular. Shaw et al (2005) explore the use of a sweep line algorithm for use with buildings whose floor plan is more complex in shape (Fig.3)

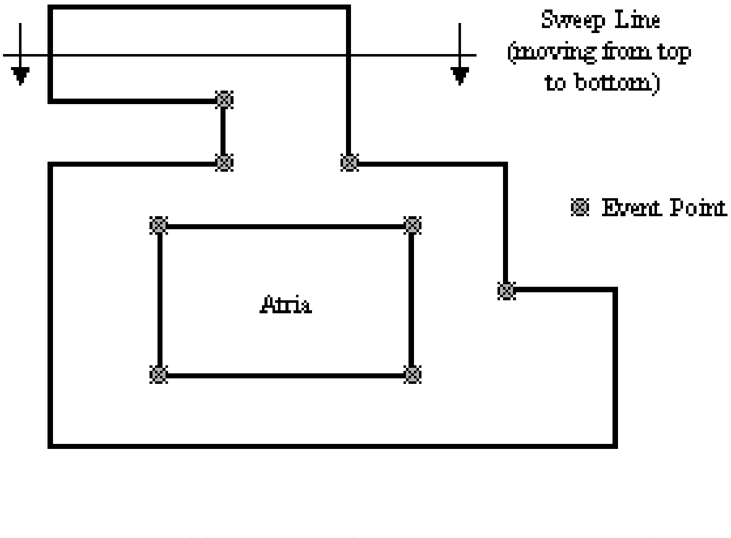


Fig.6. Building Floor Plan; Sweep Line Example

A major area within computational geometry is the partitioning of polygons. Shaw et al (2005) use the sweep algorithm to partition a floor plan into rectangles that are then assigned column spacings. The algorithm sweeps an imaginary line over the floor plan from either left to right or top to bottom. During the sweep, the line stops at "event points". These can be any reflex vertex on the boundary or an internal form such as an atrium (fig.6). Next the algorithm does another sweep from right to left. This further partitions the rectangles (fig. 7).

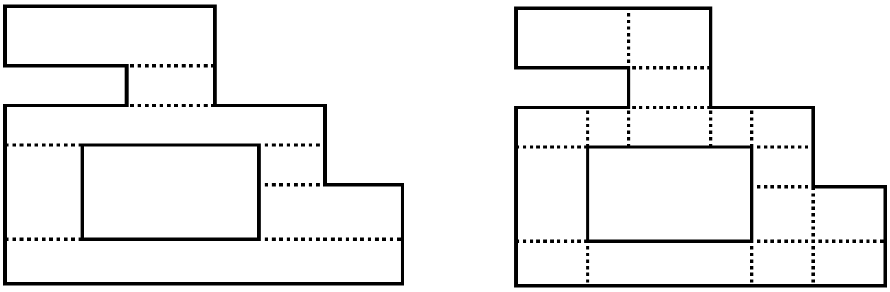


Fig.7: First partioning on the left and the second on the right

At the end of this process, each rectangle shares at least one edge with another rectangle. This is modeled within the algorithm using an 'adjacency graph' which is created by associating a node with each partitioned section and linking it to the adjacent sections.

Next a genome is generated for each rectangle. The initialisation process starts by randomly selecting a rectangle and starting at the top left, a random distance is generated in the X direction (the distance is constrained so it cannot exceed the size of the rectangle). Each distance represents the X column spacing for that part of the column grid. The process is repeated until it reaches the right hand side of the rectangle. The Y column spacings are then generated using a similar process. Having initialised the first rectangle, the algorithm selects an adjacent section and generates a new genome for it. However as the next partition must share a common edge, the algorithm firstly copies the column spacings for this edge before generating a new set of spacings for the remaining parts of the rectangle.

The algorithm uses the adjacency graph to locate the relevant rectangles and to ensure that the whole building is covered and the result is a coherent solution in that the column spacings match up with adjacent areas. The genome is then a series of X and Y coordinates for each rectangle, plus other information such as the floor to ceiling height and the structural system that is to be used. Shaw et al (2005) use a GA as the search algorithm and the method has been found to be robust and to give good results.

Generative Theory of Shape

Usually when defining the geometry of an object, it is expressed as a series of fixed points with connecting lines. Over a period of years, Leyton has developed a new approach to geometry, the main points of which are given in Leyton (1998) and amplified in Leyton (2005). Leyton's theory, which he calls the "Generative Theory of Shape" has two properties which he defines as being fundamental to intelligence and these are:-

The maximisation of transfer and

Maximising the recoverability of generative operations.

In Leyton's theory, complex shapes are seen as arising from a series of simple operations which generate features such as circles, cubes, etc. These are defined from fundamental building elements such as points, lines and planes. The shapes are then built up from these elements and are specified using group theory. The recoverability therefore describes the actions used to create the shape. A simple explanation of how this applies to curved forms is given in Leyton (1988). The transfer can be thought of as the operations to create the object. By reducing these operations to fundamental primitives one is maximising transfer and then by ensuring recoverability, the complexity of the final object can be understood. This is a very simple description of what is a complex theory. Thus rather than describing shapes by their geometrical properties in terms of coordinates etc, Leyton argues that they should be described in terms of the actions used to create them. This potentially has the ability to enable topological search algorithms to handle complex and difficult topologies, like, for instance, suspension bridges. Even more attractive would be the ability to search through a variety of topologies using the same representation and algorithm and thus for example, in one search, to look at a range of bridge options from arches to suspension bridges. Conceptually Leyton's approach allows this. In practice, its application is challenging and much remains to be done before its full potential can be defined. The following describes the early stages of work to translate Leyton's theory into a tool for topological search.

Work by the authors is examining the use of Leyton's theory to look at the aesthetics beam / slab and arch bridges. This is intended to provide an alternative form of representation to the construction and repair agent approach of Machwe et al (2005). In the following, the discussion will focus on arch bridges as this is the area in which the work is most developed.

The search process for the arch bridge involves both aesthetic concepts and transfers (i.e. movements). The discussion will focus on the latter. The arch bridge in phenotypic form is shown in fig.8 where the position of the arch and the main variables are defined.

The genome is hierarchical with currently two levels. At the upper level the arrangement is [L,H,LL,LR,NL,NR], where L is the total bridge length, H is the height, LL is the nodal information for the nodes to the left of the centre, LR is the nodal information for the nodes to the right of the centre, NL is the number of nodes to the left of the centre and NR is the number of nodes to the right of the centre (fig.8). Currently the extreme left hand and right hand nodes are fixed as is the centre node. This is a very cautious approach and future research will look at relaxing these constraints.

At the next level in the hierarchy are the control groups, of which there is one for each node which is represented by 4 items as follows :-

[*X coordinate, Y coordinate, Action command, Action distance*]

Rather than use a typical binary representation for the arch node actions, 3 characters are currently used, these being -1, 0 and 1. These are then defined as: -1 is down, 1 is up and 0 is no change. For example: [12,4,0,0] would represent a node at position 12,4 which is not to move (*action command* = 0).

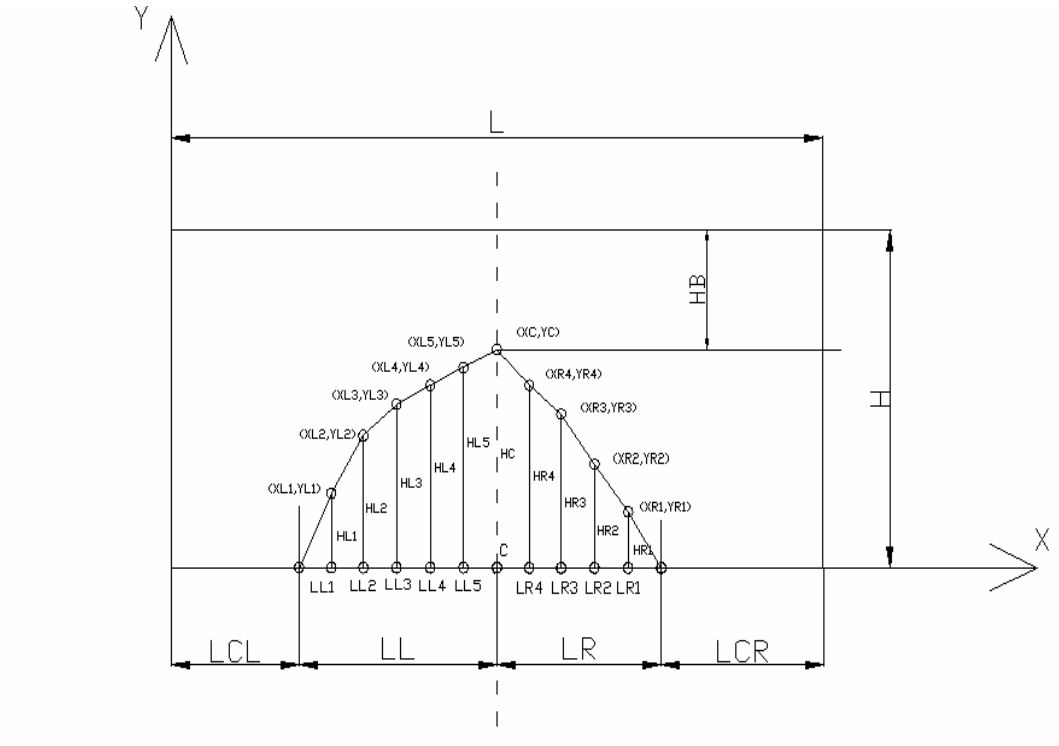


Figure 8: Notation for one span arch bridge

If this was [12,4,1,1] then this would be a node at 12,4 which was to move up (*action command* =1) by amount 1 so it would move to position 12,5. As yet, movement in the X direction has not been allowed although this is planned. Also note that currently there is no memory in the process. Leyton's theory states that this should be present but in terms of topological search, the benefits of doing this are not immediately apparent.

In the above representation, crossover and mutation currently only operate on the *action commands* and *action distances*. This is in line with the cautious approach adopted for the other aspects of the representation. One of the problems with topological search is ensuring that the process is stable and experience shows that, when developing new methods, it is far better to start with a highly constrained initial approach from which results can be obtained and to then gradually relax the constraints, rather than to use a more ambitious approach.

CONCLUSIONS

The ability to find good solutions to topological problems is something which would be of great assistance to designers. Evolutionary computation offers a variety of powerful search techniques but if the topological search is based on parametric or ground structure types of representation, then the ability of the algorithm to search is constrained. A variety of other techniques have been discussed. Some of them, such as graphs and voxels are of use for certain types of problems. The use of computational geometry has been described for a particular domain. There are other methods within computational geometry that can be applied to other domains but each technique is domain specific. The generative shape theory possibly has the potential to provide a generic representation for topological search but at present this is yet to be proven.

REFERENCES

Azid, I.A. & Kwan, A.S.K., 1999. A layout optimisation technique with displacement constraint, in Topping BHV & Kumar B (eds) Optimization & Control in Civil & Structural Engineering, Civil-Comp Press, Edinburgh UK, 71-77.

Bendsoe, M & Kikuchi N, 1988. Generating optimal topologies in structural design using a homogenisation method, Comput. Methods in Applied Mech Eng., 71, 197-224.

Borkowski A & Grabska E, 1995. Representing designs by composition graphs, in Smith I (ed) Knowledge Support Systems in Civil Engineering, IABSE report 72, Zurich, 27-36.

Deb, K & Gulati S, 2001. Design of truss structures for minimum weight using genetic algorithms, Finite Element Analysis & Design, 37, 447-465.

Griffiths, D.R. & Miles, J.C., 2003. Determining the optimal cross-section of beams, AEI, 17, 59-76.

Jenkins, W, 1993. An enhanced genetic algorithm for structural design optimisation, in Topping, B & Khan A (eds) Neural Networks and Combinatorial optimization in Civ & Struct Eng,, Civil-Comp Press, Edinburgh UK, 109-126.

Khajehpour, S. & Grierson, D. 2003. Profitability versus safety of high-rise office buildingsJournal of Structural & Multidisciplinary Optimization, 25, pp. 1-15.

Leyton, 1988, A process grammar for shape, AI, 34, 213-247.

Leyton, M, 1998. A generative theory of shape, Springer-Verlag, Berlin, 554pp.

Leyton, M, 2005. Shape as memory storage, AI, 3345, 81-103.

Machwe A, Parmee I & Miles J, 2005. Overcoming representation issues when including aesthetic criteria in evolutionary design, Soibelman L & Pena-Mora F (eds) Comp in Civ Eng, ASCE, Reston VA, USA, 12pp.

Miles, J.C., Sisk, G., & Moore C.J., 2001. The conceptual design of commercial buildings using a genetic algorithm, Computers & Structures, 79, 1583-1592.

O'Rourke, J, 1998. Comp Geometry in C, 2nd Ed., Cambridge Univ Press, Cambridge, UK.

Parmee I & Abraham J, 2004, Supporting implicit learning via the visualization of COGA multi-objective data, Proc EVOTEC, IEEE, 395-402.

Pritchard T, 2004. Novel techniques in structural layout optimisation, PhD thesis, Department of Civil Engineering, University of Sheffield, UK.

Rozvany, G.I.N., 1992. Shape and layout optimization of structural systems and optimality criteria methods, Springer-Verlag Wien, 496pp.

Shamos, M,1978, Comp Geometry, PhD Thesis, Yale Univ, New Haven, UMI #7819047.

Yang, Y., 2000. Genetic programming for structural optimisation, PhD thesis, School of Civil and Structural Eng, NTU, Singapore

Zhang, Y & Miles J, 2004. Representing the problem domain in stochastic search algorithms, in Schnellenbach-Held, M & Hartmann, M. (eds) Next Generation Intelligent Systems in Engineering, EG-ICE, Essen, 156-168.