Struct Multidisc Optim 23, 241–251 © Springer-Verlag 2002
Digital Object Identifier (DOI) 10.1007/s00158-002-0182-6

A parallel nodal-based evolutionary structural optimization algorithm

Y.-M. Chen, A. Bhaskar and A. Keane

Abstract This paper is concerned with the minimum weight design of structures using Finite Element Analysis (FEA). A new evolutionary structural optimization (ESO) algorithm is presented. This method departs from previous studies of ESO in that it exploits the movements of the nodes in an unstructured finite element mesh in an appropriate way. An attractive feature of the scheme presented is that it carries out topology optimization in the interior of the domain concurrently with shape optimization of the exterior of the domain. Circular cavities are inserted into the interior of the domain from which the internal topology is then revealed by migration of the cavity edge nodes. Due to the complexity of the resulting cavity geometry the FE mesh tends to be refined internally. A scheme for maintaining a roughly uniform density unstructured finite element mesh throughout the optimization in a two-dimensional domain is presented. The designs produced posses smooth internal and external boundaries. The method uses iterative finite element analysis and re-meshing to correct for any element distortion. The benchmark "Michell Arch" problem is used to demonstrate the approach.

Key words evolutionary, node, optimization, topology, shape

1 Nomenclature

$A_{\text{domain}}^{\text{initial}}$ = area of initial design domain.

A_{domain} area of design domain in each cycle of optimization.

$ED_{\text{min}}^{\text{initial}}$ prescribed initial minimum edge distance between two adjacent boundary design variables.

ED_{min} variable minimum edge distance between two adjacent boundary design variables.

$\text{CV}_{\text{min}}^{\text{initial}}$ prescribed initial minimum edge distance between two adjacent cavity design variables.

CV_{min} variable minimum edge distance between two adjacent *cavity* design variables.

R initial radius of circular cavities.

C a user defined constant.

X_i X co-ordinate of node i.

Y_i Y co-ordinate of node i.

X_d unit movement in X direction.

Y_d unit movement in Y direction.

$xstep$ stress search step size in the X direction.

$ystep$ stress search step size in the Y direction.

σ_i nodal stress of design variable i.

$\sigma_{\text{initial\_max}}^{N}$ initial maximum nodal von Mises stress.

σ_{X_ref} the magnitude of the highest stress in the X direction relative to that at the design variable position.

σ_{Y_ref} the magnitude of the highest stress in the Y direction relative to that at the design variable position.

OR optimum ratio.

δOR increment in the optimum ratio.

P_{X_ref} the relative distance between the highest stress in the X direction relative to that at the design variable position. This term is expressed as a multiple of $xstep$.

P_{Y_ref} the relative distance between the highest stress in the Y direction relative to that at the design variable position. This term is expressed as a multiple of $ystep$.

Received February 26, 2001
Revised manuscript received May 20, 2001

Y.-M. Chen, A. Bhaskar and A. Keane

Computational Engineering and Design Centre (CEDC), School of Engineering Sciences, University of Southampton, SO17 1BJ, UK
e-mail: andy.Keane@soton.ac.uk

2 Introduction

Shape and topology optimization of structures is a rapidly developing field (e.g., see Rozvany *et al.* 1995) that has significant industrial importance. Within this field most real-world analysis is carried out using Finite Elem-

ent Analysis (FEA). In finite element formulations, the structural geometry is defined using nodes, elements and material properties such as element thickness, elastic modulus, etc. The objective of structural optimization is usually to produce low mass designs while at the same time ensuring smoothness of the structure boundaries. In early work on finite element based shape optimization, the coordinates of the boundary nodes were treated as design variables (see Hsu 1994) because it was believed that boundary smoothness could be achieved by slowly migrating the edge nodes. However, this idea was discarded because experience showed that it was very difficult to maintain a smooth boundary shape using the approach (see Hsu 1994). The more recent approach has been to treat the element or material properties as design variables.

This more intuitive evolutionary method pioneered by Xie and Steven (1997) is based on removing inefficient (low stress level) material from an initially oversized domain. The removal process can be carried out by either varying the elastic modulus as a function of the stresses or by deleting regions occupied by elements with low stresses. By repeating this step and "removing" small amounts of material at each stage, the topology for the structure gradually evolves. There have been a number of modifications and refinements proposed for this basic approach, (e.g., see Querin *et al.* 1998, 2000; Kim *et al.* 2000; Yang *et al.* 1999). However, the weakness of classical ESO methods (adding and/or removing elements) in generating optimum topologies was recently pointed out by Zhou and Rozvany (2001). They demonstrated through a simple test example that the classical ESO rejection criteria for compliance or stress design can produce an extremely non-optimal topology.

In contrast the soft kill method developed by Walther and Matteck (1993) is based on varying the elastic modulus by using a simple *linear* relationship with some scalar measure of the stresses evaluated within each element. The hard kill method developed by Hinton and Sienz (1995) used a *step* function to adjust the elastic modulus.

The homogenization method originally developed by Bendsøe and Kikuchi (1988) is based on defining the initial design domain with an infinite number of microscale cells with voids. The porosity of this medium is then optimized. The optimization problem is defined in such a way that the solid/void ratio in the base cells are the design variables. If a portion of the medium consists only of voids, it is assumed that no material lies in that area. On the other hand, if there is no porosity in some portion, solid structure must be located there. Bendsøe (1989) also proposed various ways of predicting the optimum topology of a mechanical element by introducing an artificial density. He concluded that the most satisfactory method is to employ a porous material approach, using simple square voids at the microscale.

A similarity in all the above methods is that the initial design domain is filled with elements of equal size and shape. Analysis of this mesh then provides a measure of the stress distribution within the domain. Based on the relative stress levels within the structure, the shape and topology of the structure are then modified. However, by using identical elements in the optimization, the final shape of the converged geometry is limited by the shape of the elements used. As a result, a "stair-case" effect may emerge. Therefore, post-processing is often required to smooth the final geometry.

It has also been demonstrated that a solid isotropic microstructure with penalty (SIMP) method for intermediate densities combined with new optimality criteria methods (see Rozvany and Zhou 1991) results in very satisfactory solid-empty type topologies in generalized shape optimization. In addition, the solution obtained with a SIMP model is much closer to theoretical solutions than those obtained using square cells (see Rozvany *et al.* 1992).

Nowadays commercial packages such as ANSYS are commonly used for geometric and finite element modelling. Most commercial modelling packages have one or more auto-meshing tools. The meshes generated by these tools usually consist of irregular sized elements with different orientations. Increasingly, layout and topology optimization methods are being incorporated into these commercial packages. The idea of using optimization strategies with irregularly meshed domains and evolutionary methods is, therefore, a natural next step in the use of FE methods in design. Methods applicable to this way of working are the subject of this paper.

In this paper a parallel nodal-based evolutionary structural optimization (PNESO) method that is capable of optimizing unstructured meshes by treating boundary node positions as the design variables is proposed. In this method, boundary smoothness is achieved by migrating boundary nodes from their initial lightly loaded positions to higher stress locations. Both the magnitude and direction of each nodal movement are calculated by the proposed "NESO equation". This algorithm carries out three optimization operations in an iterative fashion. The first of these moves the external edges of the structure and is similar in logic to the "nibbling" ESO, of Xie and Steven (1997). The second operation of the algorithm is an internal cavity formation stage, where small circular holes are made in the structure. The final operation is the movement of the boundaries of these holes. The overall structure of the method is illustrated in Fig. 1.

An outline of the basic NESO method is presented in Sect. 2. Details of the PNESO algorithm are then presented in Sect. 3. Mesh control considerations are discussed in Sect. 4. Cavity formation considerations are discussed in Sect. 5. Section 6 discusses the computational costs involved in the implementation of the PNESO algorithm. Section 7 presents an application in which the PNESO algorithm is validated using the benchmark "Michell Arch" problem.

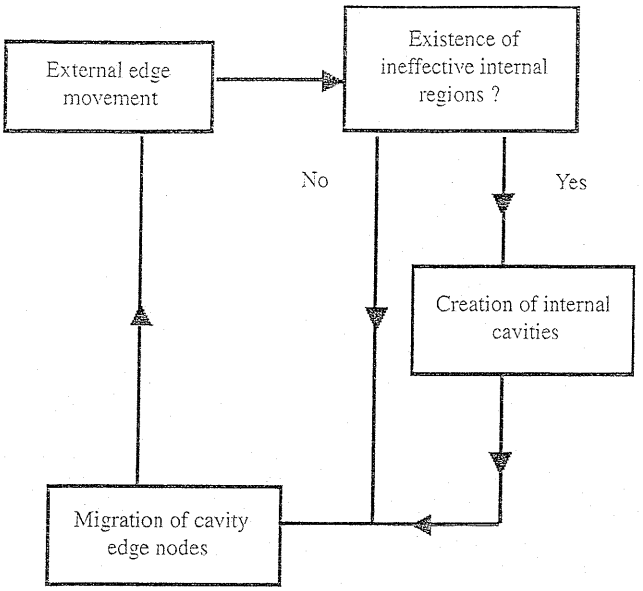

Fig. 1 PNESO Process

3 The Nodal-Based Evolutionary Structural Optimization (NESO) method

The NESO method is based on relocating structural boundary nodes (design variables) repeatedly from initially lightly loaded (low stress) locations towards higher stress locations. The procedure outlined has been implemented on a two dimensional domain. The extension to three dimensions is, in principle, straightforward. The details, however, may be significantly different and will not be discussed here. The basic procedure starts with a 'blank' which is usually taken as a rectangular domain. The domain is meshed using unstructured triangular elements by the Matlab mesh generator and the loads and geometric boundary conditions are applied. Finite element analysis is invoked at this stage and the stress components found. Here, the scalar measure of the average of the stress components has been taken as the von Mises stress at a point.

The strategy of node migration is based on the expectation that moving nodes at the boundary towards regions of high stress should lead to improvement of the design in the sense of reducing weight from the relatively unstressed regions. By shifting the boundary nodes towards higher values of stress, we are likely to retain material in the regions of high stress, whereas material from the low stress regions is likely to be "trimmed off" by shifting nodes away from these areas. Note that the conventional ESO approach (see Xie and Steven 1997) trims off material from the regions of relatively low stress too: the difference is that the present method does not use elimination of an element from the domain and, therefore, affords the possibility of controlling the process of shedding material from the design more smoothly. We require two pieces of information to move a node in a plane: we need to decide the *magnitude* of movement and the *direction* of movement.

The present strategy finds a reference point within the domain in both the horizontal and the vertical directions relative to each boundary node. The reference coordinates are taken as the locations of the highest stress within the domain with respect to each boundary node, i.e. with respect to the same horizontal and vertical coordinate. Each boundary node is then migrated towards these reference points in the X or Y-direction.

To accurately determine these reference points, a small step size is required when searching through the design domain. For each search point, the corresponding stress is calculated by interpolation from the FE results. This is equivalent to exploring the domain stress distribution using a fine rectangular grid but with the advantage of not needing to use this fine grid when deriving the FE solution. This saves significant computational cost in the cycles of finite element analysis. Consider a two-dimensional continuous domain as shown in Fig. 2. The steps required for finding the reference points are listed below.

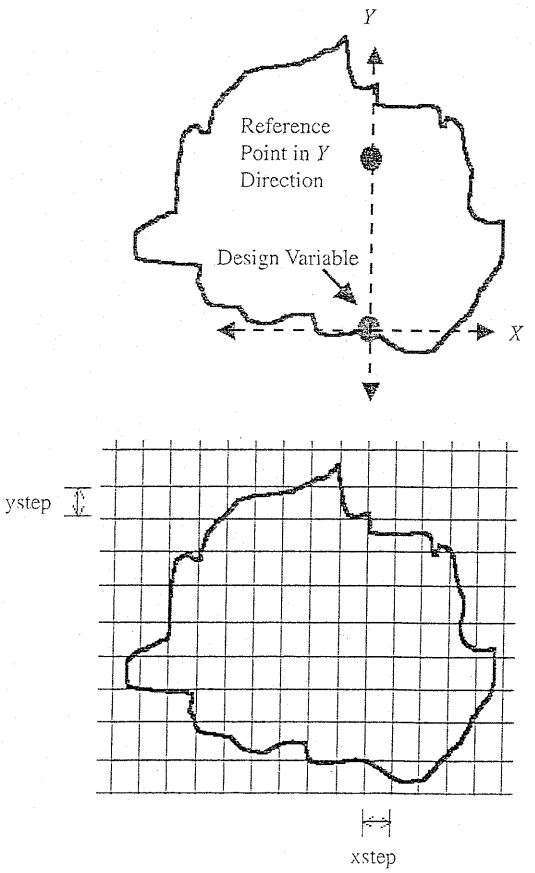

Fig. 2 Reference node search strategy

Starting from the position of each design variable (i.e. a boundary node), search the domain with a given step size in the two orthogonal directions, relative to the pos-

Table 1 Illustration of the X directional stress and position counter (NaN (not a number) indicates the search has crossed the domain boundary)

Stress value (MPa)	NaN	96	100*	73	66	55	NaN
X direction position counter (Px)	−2 [xstep]	−1 [xstep]	0 [xstep]	1 [xstep]	2 [xstep]	3 [xstep]	4 [xstep]

*Current design variable ($\sigma_i = 100$)

ition of the current boundary node. Searches in both the positive and negative senses in the two directions are carried out (i.e. left as well as right; and up as well as down) until a domain boundary is crossed.

During search, the nodal stresses are interpolated from the values given by the unstructured grid FE analysis. The nodal stresses are recorded in two vectors for the X-direction and the Y-direction searches, respectively. The highest values and their corresponding locations are stored.

The number of '*xstep*'s and '*ystep*'s in the X and Y-direction vectors, respectively, required to reach the reference point from the current design variable position are noted. For example, based on Fig. 2 , considering both the X and Y- direction vectors, respectively, these are zero and ten, see Tables 1 and 2.

From Table 1 it is clear that the location of the current design variable itself has the highest stress value among all the locations in the horizontal direction of the domain. In this situation, the current design variable itself becomes the reference point in the horizontal direction ($\sigma_{X\_\mathrm{ref}} = 100$ and $P_{X\_\mathrm{ref}} = 0$). As will be seen later, this node will not move in the horizontal direction ($xstep = 0$).

From Table 2, the vertical direction vector reveals that the reference point has a magnitude of $\sigma_{Y\_\mathrm{ref}} = 225$ MPa and is situated 10 steps in the Y-direction from the position of the boundary node in question, i.e. $P_{Y\_\mathrm{ref}} = 10$. Therefore, the current design variable needs to be moved in the vertical direction. Moreover, the positive sign of the term P_{Y_ref} indicates that the movement should be in the positive Y-direction, i.e. upwards. So far, the method of finding the directions for each nodal movement has been illustrated. The magnitude of nodal movement is calculated by the use of the "NESO" equation as explained below. Consider the X and Y co-ordinates of each boundary node; nodes are shifted according to the following equation:

$$X_i = X_i + \left\{ sgn(P_{X\_\mathrm{ref}}) \left(1 - \frac{1}{|P_{X\_\mathrm{ref}}|}\right) \left[1 - \frac{\sigma_i}{\sigma_{X\_\mathrm{ref}}}\right] \right\} X_d \,,$$

$$Y_i = Y_i + \left\{ sgn(P_{Y\_\mathrm{ref}}) \left(1 - \frac{1}{|P_{Y\_\mathrm{ref}}|}\right) \left[1 - \frac{\sigma_i}{\sigma_{Y\_\mathrm{ref}}}\right] \right\} Y_d \,. \qquad (1)$$

First term: the previous value of the co-ordinate.

Second term: the signum function is used to decide the direction of each nodal movement with respect to the local nodal position.

Third term: provides a relative magnitude for each nodal movement. The unit distance counted as P_{X_ref} or P_{Y_ref} is an integer value that indicates the relative distance between the boundary node and the reference point location. In addition, if the position of the reference point is located far away from the current design variable position ($|P_{X\_\mathrm{ref}}| \gg 1$ and/or $|P_{Y\_\mathrm{ref}}| \gg 1$), then this node should move a relatively large distance towards the reference position, i.e. movement increases as distance to the reference point increases.

Fourth term: this term smooths out the nodal movement. This term together with term B provides a more realistic estimation on the magnitude of the nodal movement. In other words, if the reference nodal stress is high ($\sigma_{X\_\mathrm{ref}} \gg \sigma_i$ and/or $\sigma_{Y\_\mathrm{ref}} \gg \sigma_i$) compared with the nodal stress of the boundary point in question, then the boundary node needs to make a larger step towards the

Table 2 Illustration of the Y-directional stress and position counter (NaN (not a number) indicates that the search has crossed the domain boundary)

Stress value (MPa)	Y-direction position counter (Py)
NaN	16 [ystep]
194	15 [ystep]
202	14 [ystep]
204	13 [ystep]
212	12 [ystep]
222	11 [ystep]
225*	10 [ystep]
202	9 [ystep]
196	8 [ystep]
188	7 [ystep]
182	6 [ystep]
180	5 [ystep]
168	3 [ystep]
157	2 [ystep]
148	1 [ystep]
100**	0 [ystep]
NaN	−1 [ystep]

*Reference node in the Y-direction ($\sigma_{Y\_\mathrm{ref}} = 225$ MPa)
**Current design variable ($\sigma_i = 100$ MPa)

reference point, i.e. movement increases as stress difference increases.

Final term: prescribed unit nodal movement distance. This term is directly related to the computational cost. A smaller value results in a smoother boundary but the number of steps required to achieve it is larger and hence the process becomes more expensive.

At first sight, the behaviour of the second term in the NESO equation for the movements may appear to be contrary to intuition. It appears that nodal movements for a boundary node are affected most when the reference point is far off whereas this is smallest when the reference point is close to the boundary node in question. Note that the minimum value of this term is zero when $P_{X\_\text{ref}} = 1$ (the minimum possible value; recall that this variable takes only integer values) and it is unity when P_{X_ref} tends to infinity. In addition, when $P_{X\_\text{ref}} = 0$ (the design variable position itself is a reference point), the second term in the NESO equation tends to infinity but the third term becomes zero. In other words, when P_{X_ref} is zero, the total contribution to the nodal displacement is zero and hence the node does not move at all. A careful examination of the node-movement process reveals that the desired nodal movements should indeed have the trend indicated above (with respect to distance from the reference point), i.e. the algorithm presented here is based on the idea that points with low stress values should be pushed towards regions of high stress.

Note that this strategy of edge movement is capable of modifying the external shape only, it cannot alter the topology in any way. To allow for changes in topology, cavities are introduced based on local values of stress within the domain.

The NESO equation is a mathematical representation of the fact that the boundary nodes move relative to each other. When the nodes are far from the reference points, the nodal movement is relatively large compared with the nodes that are close to the reference points. Conversely, the nodal movement becomes smaller when the nodes are near the reference points. This has the advantage of ensuring a slow geometrical convergence into the optimum shape in high stress regions.

4 The Parallel Nodal-Based Evolutionary Structural Optimization (PNESO) algorithm

Here, we propose a parallel version of the nodal-based evolutionary structural optimization algorithm (PNESO), in which cavities can be inserted into the domain before the external edge movement is completed. Additionally, the cavity nodes are then treated as design variables and participate in the node-shifting process. In other words, both the external (shape) and internal (topology) design domains are optimized at the same time.

In the normal evolutionary optimization method, the process of optimization continues until some prescribed objective function goal is satisfied, e.g. optimization continues until the lowest stress is, say, 80 % of the original peak stress. However, it often requires experience to set this goal successfully. Here, the optimization is allowed to continue indefinitely and is terminated by the user when no further substantial changes in the design are observed.

Consequently, the product of the initial maximum nodal von Mises stress ($\sigma^N_{\text{initial\_max}}$) and a large integer constant C is taken as the goal [see (2)]. The objective function for each stage of the evolutionary optimization is then set as the product of a control parameter termed the optimum ratio OR (which is gradually adjusted) and $C\sigma^N_{\text{initial\_max}}$ to encourage a gradual convergence from the initial geometry, i.e.

$$\text{objective function} = OR\ C\sigma^N_{\text{initial\_max}}. \tag{2}$$

During the evolutionary process, a low value of OR (say 0.01) is used to begin with. When the minimum value of the von Mises stress at all the boundary nodes reaches this value, the optimum ratio is incremented by a fixed amount,

$$OR = OR + \delta OR. \tag{3}$$

With this increased optimum ratio, the node-shifting and cavity formation operations take place again until the minimum over all the boundary nodes reaches this new value.

The constant used here plays two roles in the optimization: (a) it allows the user to terminate the optimization process when appropriate; (b) it speeds up the optimization process. The explanation for why the introduction of C reduces the optimization time is as follows: in each stage of optimization the objective function is defined as the product of the optimum ratio OR and $C\sigma^N_{\text{initial\_max}}$. Only those design variables with nodal stress *less* than this product are shifted, however, after each edge movement the stresses throughout the whole design domain tend to increase. Without the introduction of the constant, edge movement tends to spend most of the time moving a few low stress boundary nodes before moving to the next step of the optimization. This increases the computational cost because FE analysis is required after each movement. By introducing the constant, more boundary nodes are shifted per iteration and this means that the design domain shrinks faster. Thus, the evolutionary optimization process requires less iterations to produce a given shape and topology for the design domain.

For each cycle of edge movement, only those boundary nodes that have stresses lower than $OR\ C\sigma^N_{\text{initial\_max}}$ are relocated to new positions. All boundary nodes with nodal stresses greater than this are not moved. Similarly for cavities, only those cavity design variables with nodal stress lower than the minimum nodal stress of the boundary design variables are shifted, i.e. the minimum bound-

ary nodal stress is also the driving force for cavity shape design. This cycle of optimization continues until the upper limit of OR is reached. A flow chart that illustrates the inter-connectivity of the various processes involved in PNESO is shown in Fig. 3.

Note that when cavities are introduced into the design domain, the mesh tends to refine around the cavities because of the complexity of shape. It is, however, important to ensure an approximately uniform mesh density throughout optimization to minimize the computational cost. The mesh density of the design domain can be controlled through appropriate selection of both external boundary design variables and cavity design variables.

5
Mesh control: selection of external boundary design variables

The Matlab^{TM} partial differential equation (PDE) toolbox is used to mesh the geometries studied here; each geometry being defined by a set of edge nodes. When remeshing is invoked, the number of edge nodes tends to increase. As a result, the edge nodes tend to become more closely packed (leading to a higher mesh density) and edge movement becomes difficult and expensive. One way of controlling mesh density is by an appropriate selection of edge nodes for deletion. This requires prescribing

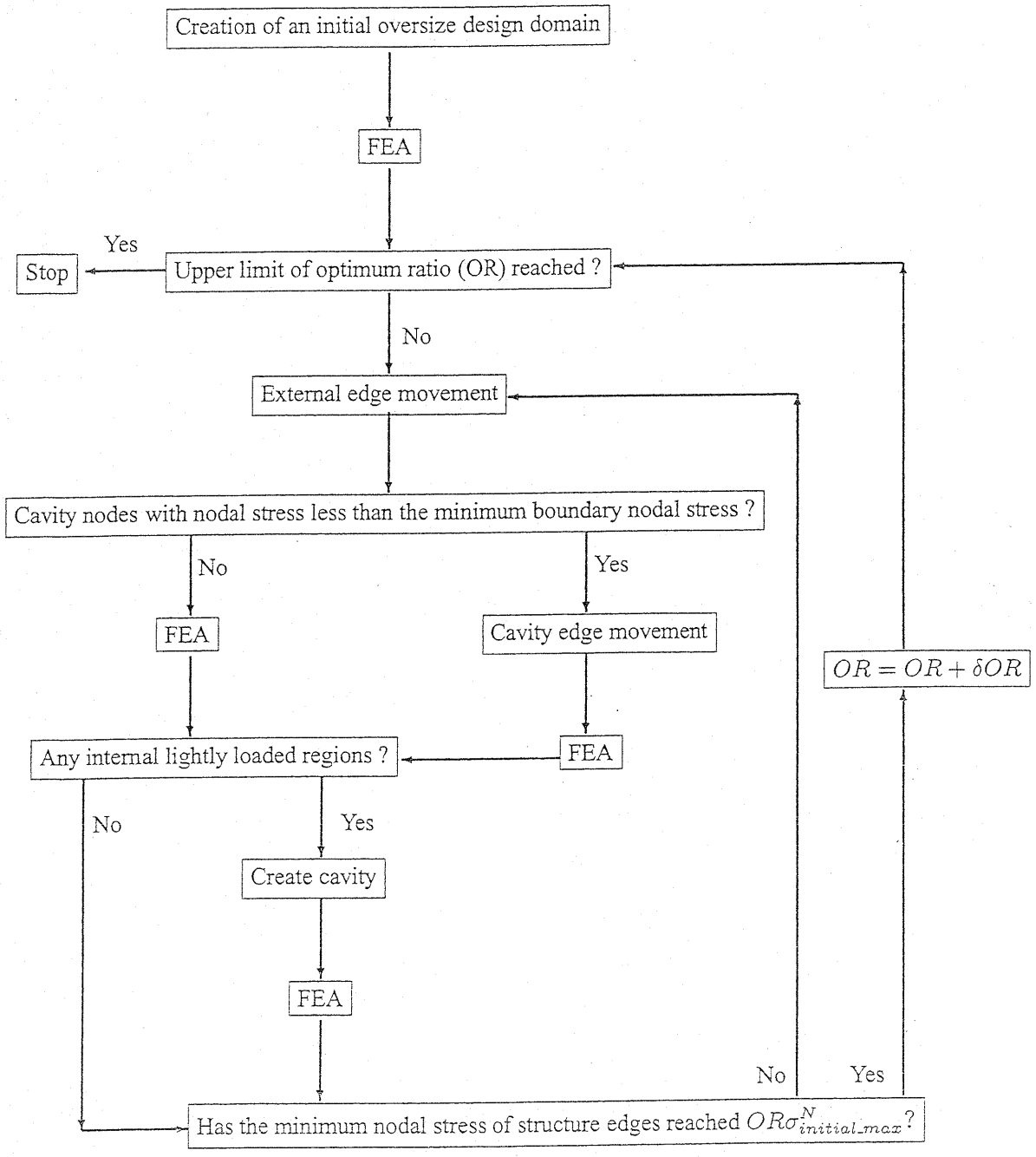

Fig. 3 Schematic diagram of PNESO algorithm

a minimum edge distance $ED_{\min}$ (i.e. the minimum spacing between nodes), which is used to control the minimum distance between two adjacent edge nodes before a node is deleted. This leads to the design variable criterion

$$ED \geq ED_{\min}. \tag{4}$$

A reference edge node is selected, and the edge distance ED is calculated between the reference node and next node in the sequence defining the geometry. If the edge distance is greater than the prescribed minimum edge distance $ED_{\min}$ then such a node qualifies to be treated as a design variable. Additionally, this qualifying node becomes the reference node for the next adjacent node in the sequence. Conversely, if the edge distance between nodes fails to satisfy the design variable criterion, then this node is deleted. The design variable criterion is applied to the whole set of edge nodes each time after re-meshing. As a result, the number of design variables varies at each re-meshing. However, the mesh density is kept nearly uniform because the spacing between the edge nodes (design variables) is properly controlled and it is from these nodes that the mesh is created.

6
Cavity formation considerations

In the basic NESO algorithm, boundary node migration is completed before cavity insertion and the basic shape of the structure remains fixed during cavity development. With the PNESO algorithm, cavities are introduced into the domain according to the changing stress distributions during edge movement. In other words, the location of the most lightly loaded regions within the domain constantly varies. An interesting question then arises as to the correct location of the cavities within the domain and their possible migration.

In the basic NESO algorithm, cavities are created when the nodal stresses of the internal domain are less than the minimum nodal stress of the boundary design variables. The lowest such stresses are chosen for cavity centers. In the PNESO algorithm, however, the same strategy cannot be used, due to the constant variation in stress distribution. Often, as the external boundaries shift inwards, there are a few adjacent low stress points. Moreover, low stress points also appear near to existing cavities (see Fig. 4). The areas covered by these low stress points are generally very small and they usually disappear after the next few iterations of structure edge and/or cavity edge movement. Hence, we can state that cavities should be introduced within the domain only as long as their locations are neither near existing cavities or adjacent to the structure boundary.

Ideally, as many cavities as possible should be placed inside the design domain because forming cavities reduces the structural weight. In addition, after cavities are created within the domain, the edge nodes of cavities qualify for the node-shifting process and this allows freedom to these nodes to explore the surrounding low stress areas, i.e. a further saving of structural weight is achieved.

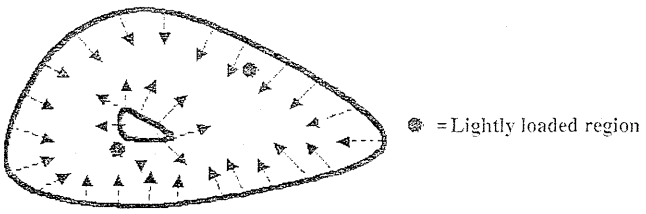

Fig. 4 Low stress locations near an existing cavity and adjacent to the external boundary

For the external structure boundary, the set of boundary design variables is limited using the *design variable criterion*. The selected design variables are adequate for representing the *shrinking behavior* of the design domain. In contrast to the external structural boundary, cavities exhibits a *growing behavior*. Therefore, additional cavity nodes are needed to ensure a smooth cavity boundary. Re-meshing the domain generates these additional nodes. Additionally, the number of cavity design variables is controlled using a similar design variable criterion for external nodes. The only difference is that the prescribed minimum spacing between two adjacent cavity nodes $CV_{\min}$ is set less than the prescribed minimum distance between two adjacent boundary nodes. It should be pointed out here that selecting more boundary and edge nodes (decreasing the prescribed minimum boundary $ED_{\min}$ and cavity edge $CV_{\min}$ distances) tends to produce a better shape, but the required computational cost increases. Control of computational costs is discussed next.

7
Computational cost considerations

The prescribed minimum edge distance parameter is directly linked to the computational cost involved in the evolutionary optimization. In an evolutionary optimization process, iterative finite element (FE) analysis is required, and within each FE analysis, inverting the stiffness matrix is an expensive step. Therefore, a fine mesh should only be used when needed. Here, we control the numbers of elements produced after each re-meshing via constantly varying the minimum edge distance parameters $ED_{\min}$ and $CV_{\min}$. We mentioned earlier that, due to boundary edge movement, the design domain exhibits a shrinking behavior. In other words, a portion of material has been "trimmed off" after each edge movement. The boundary smoothness of this smaller design domain is maintained by increasing the numbers of edge nodes (i.e. decreasing the value of the minimum edge distance $ED_{\min}$). Conversely, we can say that the initially over-

sized design domain does not need a very fine mesh (i.e. the minimum edge distance parameter may be assigned a larger value). Hence we vary the minimum edge distance parameter as a function of the domain area as follows:

$$ED_{\min} = ED_{\min}^{\text{initial}} \frac{A_{\text{domain}}}{A_{\text{domain}}^{\text{initial}}} \tag{5}$$

Equation (5) simply takes advantage of the shrinking behavior of the design domain $A_{\text{domain}} \leq A_{\text{domain}}^{\text{initial}}$ and uses the area ratio to calculate the minimum edge distance value per iteration. As a result, a finer mesh is automatically used when the area of the domain becomes smaller (the number of elements approximately doubles every time the enclosed area halves).

In contrast, the cavities tend to exhibit a growing behavior when the cavity design variables explore the low stress areas inside the domain. Increasing the minimum cavity edge distance, $CV_{\min}$, then results in a poor cavity shape because fewer cavity design variables are used to shape the cavity. Conversely, decreasing $CV_{\min}$ results in an increase in computational cost. Therefore, a fixed $CV_{\min}$ is recommended for use throughout the evolutionary optimization.

An alternative arrangement is to link both $CV_{\min}$ and $Ed_{\min}$ to the square root of the domain area ratio. This has the attraction that topologically similarly shaped domains have similar numbers of elements whatever their size. Experiments with these control parameters are continuing.

8 Example: Michell arch

In this section the benchmark "Michell arch" problem is used to illustrate the working of the PNESO method introduced in this paper. A two-dimensional plane stress problem is considered. The initial domain is a rectangle with length 5 m and width 2.5 m. The Matlab™ PDE toolbox is used to mesh the design domain in an unstructured fashion using triangular elements. A point force of magnitude of 50×10^6 N is applied at the bottom center; and the lower left and right corners are rigidly fixed in both the X- and Y-directions (see Fig. 5).

The goal for this optimization was set to $240\,\sigma^N_{\text{initial\_max}}$. When internal lightly loaded points were identified during edge movement, circular cavities with fixed radius R of 0.1 m were inserted at these locations.

Each element possesses a thickness of 100 mm, elastic modulus of $21 \times 10^{10}\,\text{N/m}^2$ and Poisson's ratio = 0.3. Both the X- and Y-direction unit nodal movements were set to 0.1 m. Re-meshing is carried out over the domain every cycle. The initial minimum edge distance $ED_{\min}^{\text{initial}}$ for design variables located on the external boundaries was set to 0.5 m. The cavity design variable selection criterion used a fixed value of $CV_{\min}^{\text{initial}}$ of

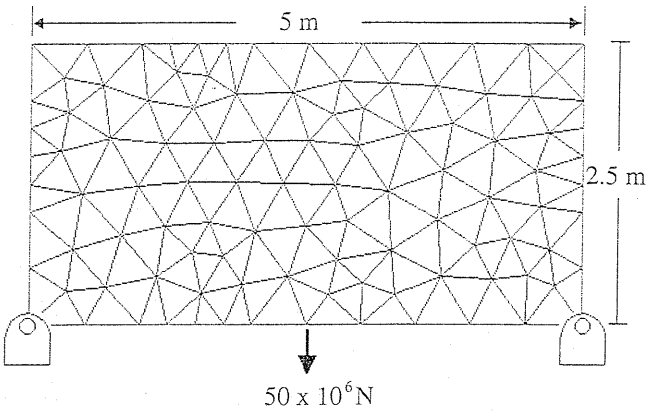

Fig. 5 Initial oversize domain

0.0628 m ($2\pi R/10$). The increment of the optimum ratio $OR = 0.01$.

The evolutionary structural optimization process was terminated after 112 iterations. The final shape is shown in Fig. 11, while Fig. 12 plots the initial domain and the result from Fig. 11 together. The first circular cavity was created after 31 iterations of edge movement (see Fig. 8)

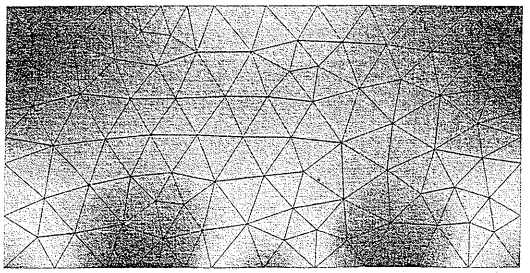

Fig. 6 Iteration 1

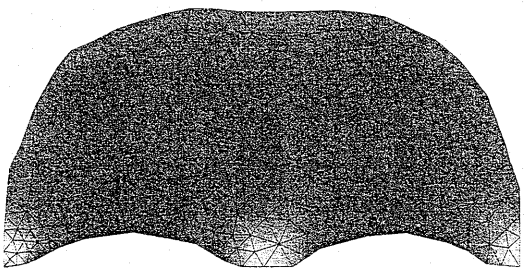

Fig. 7 Iteration 7

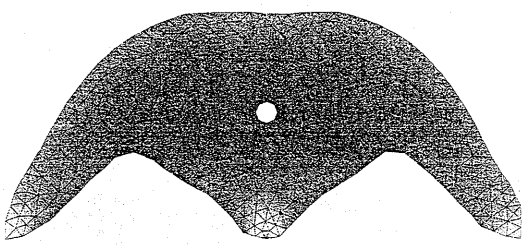

Fig. 8 Iteration 31

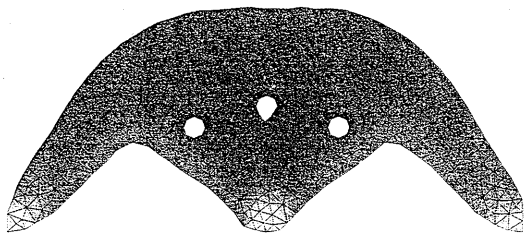

Fig. 9 Iteration 35

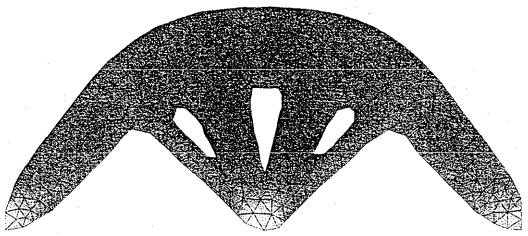

Fig. 10 Iteration 60

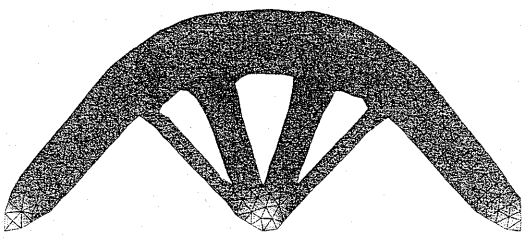

Fig. 11 Iteration 112

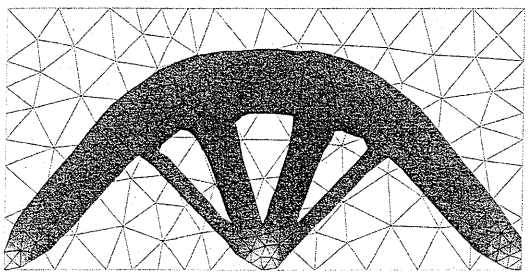

Fig. 12 Before and after

and two additional cavities followed four iterations later (see Fig. 9). Once the cavities are introduced into the domain, the cavity nodes start to grow inside the interior of the domain (see Fig. 10) during ongoing external edge movement. After 100 iterations of optimization the edge movement is seen to be less effective and no obvious changes are made to the geometry. The final design produced by this process is in accord with theoretical predictions, (see Rozvany *et al.* 1995) save only that the finite thickness and widths of the members used limit the number of cavities that are present in the design.

The changes in the domain area are shown in Fig. 13. The final design uses only 36.8 % of the material of the

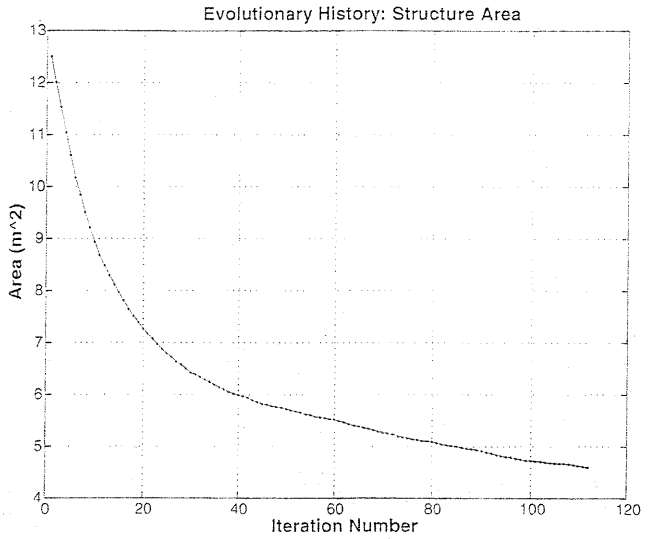

Fig. 13 Area of the domain during the evolutionary optimization process

original blank from which the process was started. The mesh density is automatically adjusted according to the domain area in this optimization following the design variable spacing criterion set out earlier (see Fig. 14).

The history of the design variable nodal stresses is shown in Fig. 15. The maximum stress in the final domain is 3760 MN/m^2 and the minimum is 434.13 MN/m^2. These may be compared to equivalent values of 894.17 MN/m^2 and 17.65 MN/m^2 for the original blank. The minimum nodal stress gradually rises at each cycle of node migration. The maximum nodal stress increases sharply in the early stages of optimization and is rather less affected by the edge movement in the later stages because a steady state has been reached. In the early stages of optimization, the applied load is shared by an initially large domain area. After each edge movement where por-

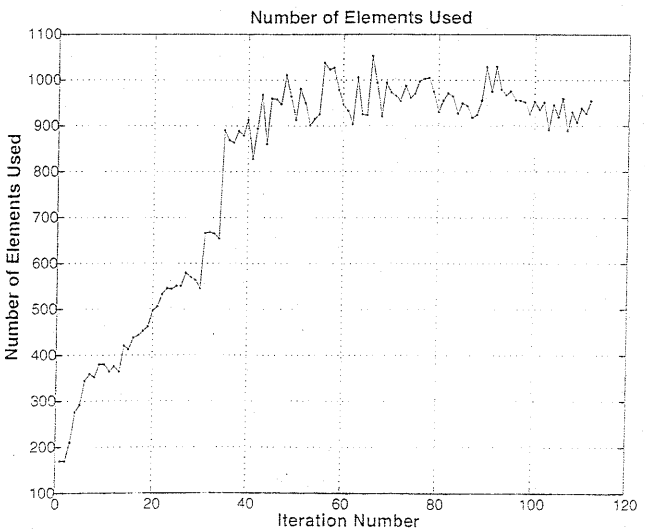

Fig. 14 Number of elements used during the evolutionary optimization process

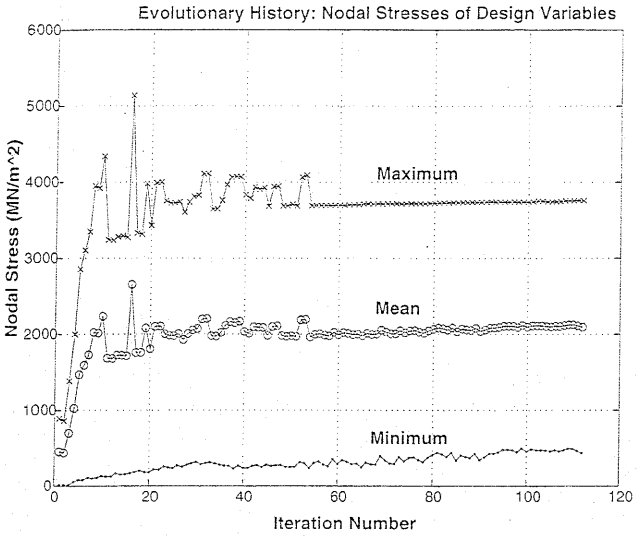

Fig. 15 History of min, mean and max nodal design variables stresses

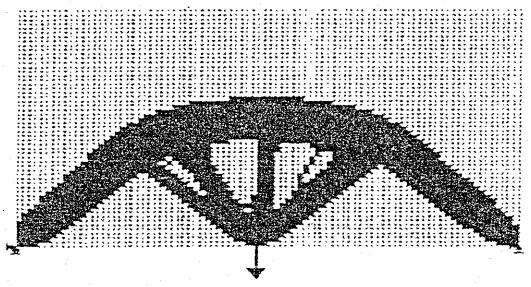

Fig. 16 An optimum shape produced by Evolve 97 version 2.0

tions of material have been trimmed off the same load is shared by a smaller design domain area. In other words, each portion of material carries more load after each edge movement and this explains why there is a sharp increase in the maximum design variable nodal stress in the early stages of optimization. However, in the final stages of optimization the structure is more efficient i.e. the local design variables are closer to the reference points and the stress gradients between the local design variables and the reference points are smaller. As a result, the edge movements become smaller and this means that the domain area is only slightly reduced at each edge movement. This explains why the maximum nodal stress in Fig. 15 shows a fairly constant behaviour in the later stages of optimization.

It is noteworthy that the evolved geometry (Fig. 11) possesses a highly symmetric shape in spite of the completely random initial mesh. Also, the internal and the external boundaries are exceptionally smooth when compared with the typical conventional ESO results for this problem (see Fig. 16). The result in Fig. 16 was obtained by using the "Evolve 97 version 2.0" software written by Xie and Steven (1997), using the same loading and boundary conditions given earlier. In addition, the maximum removed volume was set to 63.2 %, i.e. the same as achieved here. The maximum rejection ratio was set to 15 % and evolutionary rate (ER) was set to 0.1 %. A fine mesh was used in the initial rectangular domain. Note the poorer symmetry exhibited by the result in Fig. 16, c.f. that of Fig. 11.

9 Conclusions

A new evolutionary optimization algorithm has been presented in this paper. This algorithm performs layout optimization on the exterior of the domain and topology optimization in the interior of the domain at the same time. Mesh density control of the unstructured mesh over the design domain is implemented. The validity and quality of the designs produced using this algorithm have been demonstrated using the benchmark "Michell Arch" problem as an illustrative example. The optimization process is automated once it is started and the resulting designs possesses good quality with respect to stress levels and smooth boundaries.

Future work will focus on extending the two-dimensional PNESO method described here to deal with 3D problems.

Acknowledgements This work was funded by the University Technology Partnership for Design, which is a collaboration between Rolls-Royce, BAE Systems and the Universities of Cambridge, Sheffield and Southampton.

References

Bendsøe, M.P. 1989: Optimum shape design as a material distribution problem. *Struct. Optim.* **1**, 193–202

Bendsøe, M.P.; Kikuchi, N. 1988: Generating optimal topologies in structural design using homogenisation method. *Comp. Meth. Appl. Mech. Eng.* **71**, 197–224

Hinton, E., and Sienz, J. 1995: Fully stressed topological Design of structures using an evolutionary approach. *Eng. Computations* **12**, 229–244

Hsu, Y.L. 1994: A review of structural shape optimisation. *Computers in Industry* **26**, 3–13

Kim, H.; Garcia, M.J.; Querin, O.M.; Steven, G.P. 2000: Introduction of fixed grid in evolutionary structural optimisation. *Eng. Computations* **17**, 427–439

Querin, O.M.; Steven, G.P.; Xie. Y.M. 1998: Evolutionary structural optimisation (ESO) using a bidirectional algorithm. *Engrg. Computations* **15**, 1031–1048

Querin, O.M.; Steven, G.P.; Xie, Y.M. 2000: Evolutionary structural optimisation using an additive algorithm. *Finite Element Anal. Des.* **34**, 291–308

Rozvany, G.I.N.; Bendsøe, M.P.; Kirsch, U. 1995: Layout optimization of structures. *Appl. Mech.Rev.* **48**, 41–119

Rozvany, G.I.N.; Zhou, M. 1991: Applications of the COC algorithm in layout optimization. In: Eschenauer, H.A.; Mattheck, C.; Olhoff, N. (eds.) *Engineering optimization in design processes* (Proc. Int. Conf., held in Karlsruhe, 1990), pp. 59–70. Berlin, Heidelberg, New York: Springer

Rozvany, G.I.N.; Zhou, M.; Birker, T. 1992: Generalized shape optimization without homogenization. *Struct. Optim.* **4**, 250–252

Walther, F.; Mattheck, C. 1993: Local stiffening and sustaining of shell structures by SKO adn CAO. *Proc. Int. Conf. on Structural Optimization* (held in Southampton, UK). Computational Mechanics

Xie, Y.M.; Steven, G.P. 1997: *Evolutionary structural optimisation*. Berlin, Heidelberg, New York: Springer

Yang, V.; Querin, O.M.; Steven, G.P. 2000: 3D and multiple-load case bi-directional evolutionary structural optimization (BESO). *Struct. Optim.* **18**, 183–192

Zhou, M.; Rozvany, G.I.N. 1991: The COC algorithm. Part II: topological, geometrical and generalized shape optimization. *Comp. Meth. Appl. Eng.* **89**, 309–336

Zhou, M.; Rozvany, G.I.N. 2001: On the validity of ESO type methods in topology optimization. *Struct. Multidisc. Optim* **21**, 80–83