Learning Intelligent Modification Strategies in Design Synthesis

Christopher A. W. Vale and Kristina Shea

Engineering Design Centre, University of Cambridge,
Trumpington Street, Cambridge, CB2 1PZ, UK

Abstract

A method that exploits machine learning to aid modification-based computational design synthesis is presented. The algorithm makes use of statistical inference to identify appropriate modification strategies that guide the selection of modifications to develop a design solution. The procedure is especially suited to computational design tasks that cannot be easily formulated for standard algorithms. In line with the needs of most practical design tasks, the technique is oriented towards multi-objective search, yielding an archive of pareto-optimally directed design solutions. Results of application to example design tasks, truss synthesis and bitmap design, are presented.

Introduction

Many design tasks can be investigated using a succession of modifications applied to a starting design. Each modification either improves the existing design or facilitates improvement after subsequent modifications. Computational design synthesis is oriented toward controlling the selection and extent of design modification in order to generate a beneficial solution. In the simplest of cases this might involve the modification of design variables with the aid of a numerical optimizer. However, in many practical applications a class of more diverse modifications, including adding and removing design components, are required to realize high-level functionality. The challenge is then to develop automated techniques for design synthesis capable of pursuing intelligent modification strategies that result in suitable design outcomes in an acceptable number of iterations.

Within the framework of a stochastic generate-and-test approach to computational design synthesis, this paper presents a machine learning-based technique to learning and utilizing such intelligent modification strategies in scenarios where the possible design modifications can be classified as a finite set of *schemas* or types. Each schema represents a class of modification types, related to the synthesis domain, that have a repeatable effect on the design but typically an unpredictable effect on the quality of the resulting modified design, requiring re-evaluation of the modified design. In a simple example of a child's building block game, the modification schemas might be defined as *select block, lift block, drop block, move block,* etc. Each modification schema has a number of potential implementations, e.g. which block to select, or where to move the block. Evaluation of the design might involve one or more design objectives, such as calculation of the height or the base footprint of the tower being built or the number of blocks being used. The repeatable aspect of the schemas are evident in their names, e.g. selecting, lifting, etc. The unpredictable aspects of their application are the resultant effects on the design objectives, e.g. dropping a block may or may not result in increased tower height.

The general schema-based interpretation of design modification allows for the development of automated procedures for identifying suitable and unsuitable modification strategies. This is done through observation of the effects of past modifications on the current design being generated or previous knowledge from past similar design problems. The use of observation to aid decision making has been widely applied in such applications as robotic control (Bresina, 1990) and design (Wang, 1994). Though in the latter application, knowledge is acquired through experimentation, not through observation of a third party expert. As the knowledge base increases, modification strategies that prove to be reliable are favored and previously observed modifications that are unproductive are avoided. Assuming that the schemas are selected randomly, in the building block example, repeated lifting and dropping of a block is unproductive, as is repeated selection of blocks without any other action, whereas lifting, then moving, then dropping of a block is more reliably productive. Further specification of the modification, such as how high the block was lifted or which size of block was selected, can serve to refine the identified strategies. For example, it may be observed that lifting a smaller block higher results in better outcomes.

As suggested by the example above, modification strategies can be embodied by partial sequences of modification schemas. The length of the partial sequences set the degree of intelligence captured by the procedure. Longer sequences can capture more involved productions but are observed with far lower frequency. The effects of shorter sequences, however, are supported with more past observation, though they may embody more simplistic productions and therefore may be less useful. The algorithm presented here takes a statistical observation-based approach to machine learning using an architecture comprised of two elements. The first, termed the *modification sequence modeler*, analyses the effects of sequences of modification schemas on the design objectives, also taking into account implementation specifics that detail how the modification schemas are applied. These observations form a statistical database that allows inference of the possible future effects of sequences of modifications. The second element, termed the *modification advisor*, uses the sequence models to rate potential imminent modifications based on the history of past modifications and design objectives. Highly rated modifications typically facilitate sequences that are favorable to the future development given its current state.

To illustrate the robustness and wide applicability of the approach, it is applied to two diverse problem scenarios. The first involves the multiobjective design of planar truss structures with the aid of shape grammar modification rules. In this capacity, the procedure exhibits a higher degree of search efficiency when applied in conjunction with a stochastic multiobjective optimization scheme. A second synthesis task is the design of bitmap patterns using a sliding tile modification formalism where increases in search efficiency result in improved final designs in a limited number of modification-analysis iterations.

This paper first outlines the 'modification-based design' framework before presenting an overview of the design algorithm. The next two sections deal with the two components of the algorithm in greater detail. The results of application of the algorithm to sample design tasks are then presented before conclusions and an outlook for further work are discussed.

Modification-Based Design

Modification-based design describes a process by which one or more designs are gradually changed through successive applications of modification operators. Typically, the design algorithm seeks to implement modifications that lead to improved designs after a number of modifications have been made. Implementation of the modifications and evaluation of the resulting new designs demand computational resources and, as a result, it is desirable to minimize the number of modification-evaluation iterations in the design process. The framework described above not only accounts for simple optimization tasks but also more complex design formalisms where the modifications employed not only modify but also introduce and remove design variables. An example of such a formalism used in the design of pin joint planar truss structures is summarized by figure 1 (Shea, 1997). A set of modification schemas are shown that can be applied iteratively to generate an infinite number of planar truss topologies.

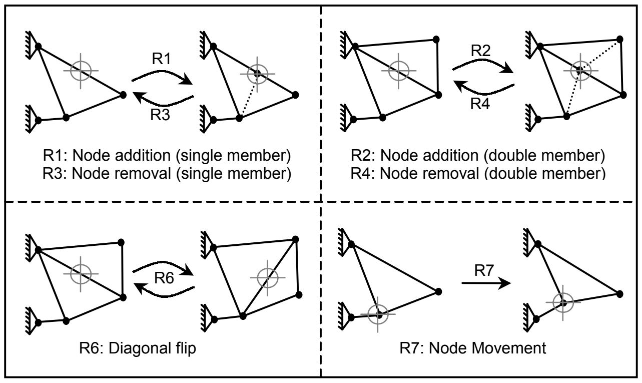
R1
R3
R1: Node addition (single member)
R3: Node removal (single member)
R2
R4
R2: Node addition (double member)
R4: Node removal (double member)
R6
R6: Diagonal flip
R7
R7: Node Movement

Figure 1. The truss modification rules. (Rule application locations are defined by crosshair indicators)

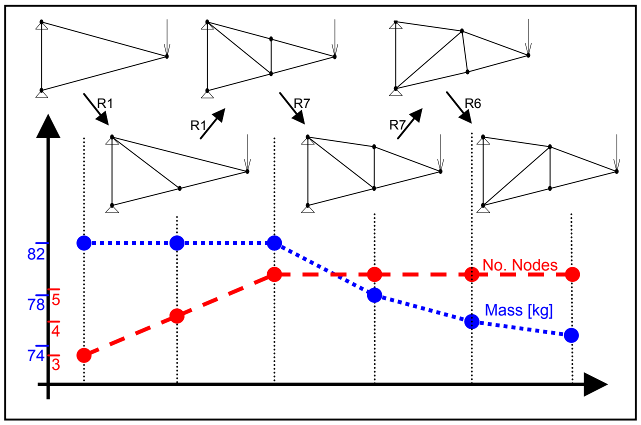

Figure 2. Development of a truss through application of grammar rules

Figure 2 shows the development of a typical loaded planar truss through repeated application of the rules. Note that with each application of the modification rules, design objectives, such as truss mass and number of nodes can change. Truss design will be revisited later as an example. It should be noted that the generate-and-test synthesis approach in conjunction with a modification-based formalism supports a bottom-up synthesis model, which does not bias synthesis by prescribing a design process, thereby allowing the generation of novel design solutions.

In addition to a basic modification rule type, such as shown in Fig 1, modifications may also be further qualified by application variables, providing a more complete description of the applied modification. In the example above, such a parameter may be the point of application of the modification rule in Cartesian co-ordinates. Each modification type is assigned an integer label from 1 to N_R, with N_R the total number of modification types. In the truss example above, N_R = 7. In combination, the modification type label and the application variables provide a detailed description of each rule application. However, it will be shown later that is it preferable to keep the number of rule types and application variables low as a higher level of specification requires more data samples to support statistical inference through observation.

Algorithm Overview

The algorithm is incorporated as two modules into a standard generate-and-test computational synthesis framework as shown in figure 3. The modification sequence modeler is responsible for analyzing the past effects of sequences of modifications and inferring future behavior. The modification advisor uses the accumulated knowledge of the modification sequence modeler together with analysis of the current design state to suggest the most suitable modifications to make on the next iteration. There are a number of implementation possibilities for each module.

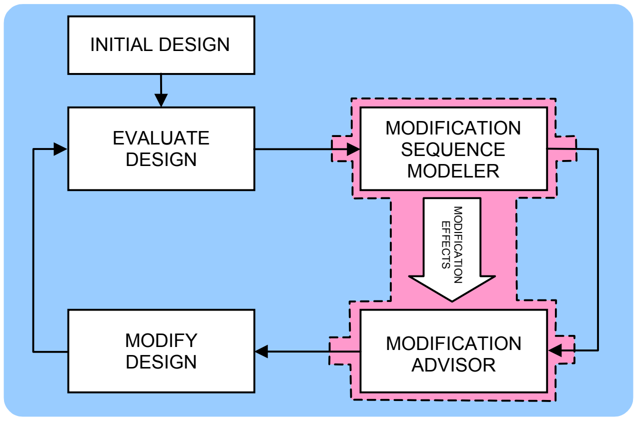

Figure 3. An iterative generate-and-test search procedure (e.g. simulated annealing) augmented by learning.

Two basic approaches have been investigated in this research. The first and more simpler uses the data captured by the modification sequence modeler to assign a figure of merit to each sequence or segment, thereof. The modification advisor then rates the possible imminent modifications based on the scores of the sequences they would contribute to. Parallels can be drawn with reinforcement learning techniques (Mitchell, 1997) where the figure of merit is associated with the standard penalty/reward system.

The second approach, which is detailed here, uses the recorded data to predict the probability distribution of future objective values due to imminent modifications being made. Based on the predicted outcomes the modification advisor rates imminent modifications according to their potential to produce results.

The Modification Sequence Modeler

The modification sequence modeler encompasses recording and analyzing the effects of sequences of modifications on the design objectives. In this implementation of the technique, it provides a model that allows prediction of the effects of future modifications based on known recent modifications and the corresponding objective values.

Observing the effects of modification sequences

A sample of the progress of a typical search in a multi-objective design problem is captured in Table 1. Referring to the table, if the current iteration is 193, then the search has just applied modification type 1 to the outcome of iteration 192. In so doing, it has completed a single modification sequence, <1>, a double modification sequence, <3-1>, and a triple modification sequence, <1-3-1>. Each sequence is completed subject to the application variables listed in the last column.

Table 1. Sample output of a search procedure

Iterate	Obj. Values Before	Obj. Values After	**Rule Applied** $\in[1,N_R]$	**Application Variables** $[v_1; v_2; v..]$
…	…	…	…	…
191	[1.0; 3.0]	[0.2; 6.0]	1	[0.2; 1.0]
192	[0.2; 6.0]	[0.1; 8.0]	3	[0.1; 0.1]
193	[0.1; 8.0]	[0.4; 1.0]	1	[1.0; 3.0]
…	…	…	…	…

By looking even further back into the history of modification applications, it is possible to observe the effects of arbitrarily long sequences of modifications. However, as the length of the modeled sequences increases, so do the number of possible sequences, implying that within a finite observation time, there will be fewer observed effects per sequence to support inference of the effects of each sequence. Currently, the algorithm models at fixed pre-defined sequence lengths, though further research will develop modelers capable of dynamically varying the modeled sequence length as the number of observations increase.

Assuming, for the purposes of explanation, that the algorithm is modeling at the level of two modifcations per sequence, it will then concern itself only with the just completed <3-1> sequence. The algorithm calculates the changes in objective values:

$$\Delta\mathbf{O}_{<3-1>|\mathbf{v}} = \mathbf{O}_{193}^{After} - \mathbf{O}_{192}^{Before} \quad (1)$$

$$\Delta\mathbf{O}_{<1>|<3>,\mathbf{v}} = \mathbf{O}_{193}^{After} - \mathbf{O}_{193}^{Before} \quad (2)$$

The change in objectives in Eqn. (1) is due to the complete double rule sequence, while that in Eqn. (2) is the change in objective from when modification type 3 has been applied to the end of the sequence. For an N_S length sequence, there will be N_S changes in objective values sampled in this fashion at each iteration. In this way, all sequences of length equal to, or shorter than the current level of sequence length modeling are recorded. Note that the observations can be made subject to the application variables, $\mathbf{v}$, which would more completely describe the sequence.

Subdividing sequences

An additional task of the modification sequence modeler is to group observed sequence effects according to the application variables in so that, for any particular sequence, each subdivision exhibits differing effects on the design objectives when applied within a particular range of application variables. Figure 4 plots the observed change in truss mass for a truss design problem over the sequence of modifications <1-3> as a function of a single application variable defined here as the geometric distance separating the application of the first and second modifications in the sequence.

It can be seen that the observed changes in the mass are generally much wider spread out in the middle ranges of geometric separation. When the separation distance was zero, all the observed changes in mass were also zero. This is to be expected for the truss grammar, as rules 1 and 3 are a rule/anti-rule pair. <1-3> applied with zero distance separating them amounts to addition and immediate removal of a node, leaving the truss unchanged. With the general effect of <1-3> clearly changing as a function of the application variables, each general sequence can be resolved into discrete segments, $S_{1..N}$, chosen so that the difference in the statistical properties (mean and variance) of the observed changes in objective values of neighboring segments is maximized. In Figure 4, the segments that result from subdivision are indicated by vertical lines.

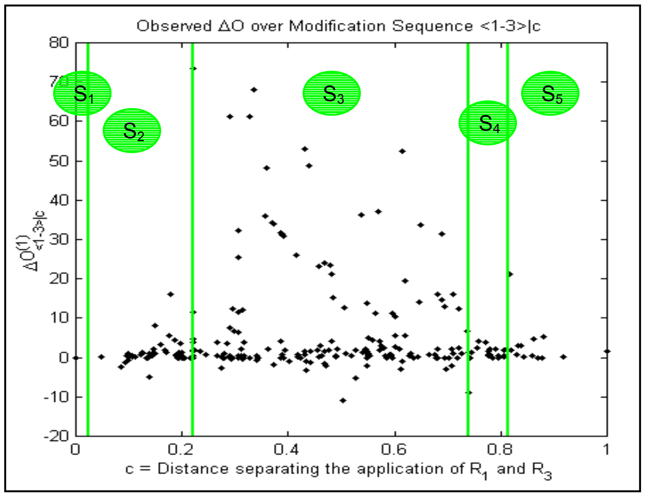

Figure 4. Changes in objective due to <1-3> as a function of the segmentation variable.

In this implementation, subdivision is not only applied to the highest level of observed change in objectives, e.g. $\mathbf{\Delta O}_{<1\text{-}3>}$ as above, but also applied to multi-level observed data, e.g. $\mathbf{\Delta O}_{<3>|<1>}$ in the example above.

As with some other components of the system to be presented later, implementation of the subdividing algorithm is beyond the scope of this paper. More detail of their implementation can be found in (Vale 2002). Effectively this algorithm finds a 'happy medium' by considering all the levels of observation at once and subdividing the sequence so that the biggest grouping patterns over all the levels of observation decide the segments of the sequences.

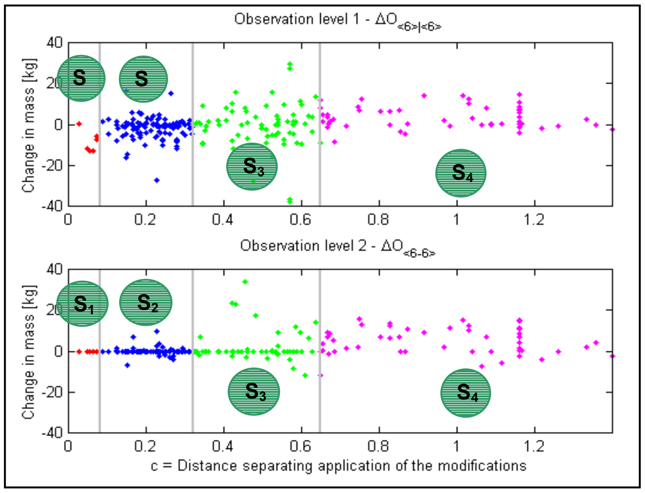

Figure 5. Multi-level subdivision of <6-6> in a truss problem

Although not done in this implementation, it would be possible and more appropriate to consider each level of observation in turn, generating $N_S \times N_O$ sets of segments for each sequence (with N_O equal to the number of objectives). Further research and development of the approach will address this issue. Figure 5 plots the two levels of observed changes in mass for a sample truss design problem using the truss modification rules for the <6-6> sequence showing the resulting segments in different greys for the set of observations.

In multiobjective problems, subdivision is performed independently for each objective, resulting in differently segmented sequences for each objective.

Modelling the Observed Data as Mixture Gaussian Distributions (MGDs)

In this implementation, the distributions of changes in objective associated with each segment are used to aid prediction of future objective values. It serves our purpose to approximate the observed frequency distribution plots as mixture gaussian distributions (Vaseghi 1996) to aid the prediction mechanisms to be used later. MGDs are formed by a weighted sum of gaussian (normal) distributions and possess certain favourable statistical qualities including a simple and fast convolution process. The MGDs are generated for each level of observed changes in objective, in each segment and for each objective independently. The operation is performed immediately after subdivision of the sequences. Figure 6 shows some of the mixture gaussian distributions resulting from the segments of the <6-6> sequence shown in Figure 5. Note the impulse functions that result from strong inclinations for zero change in objective.

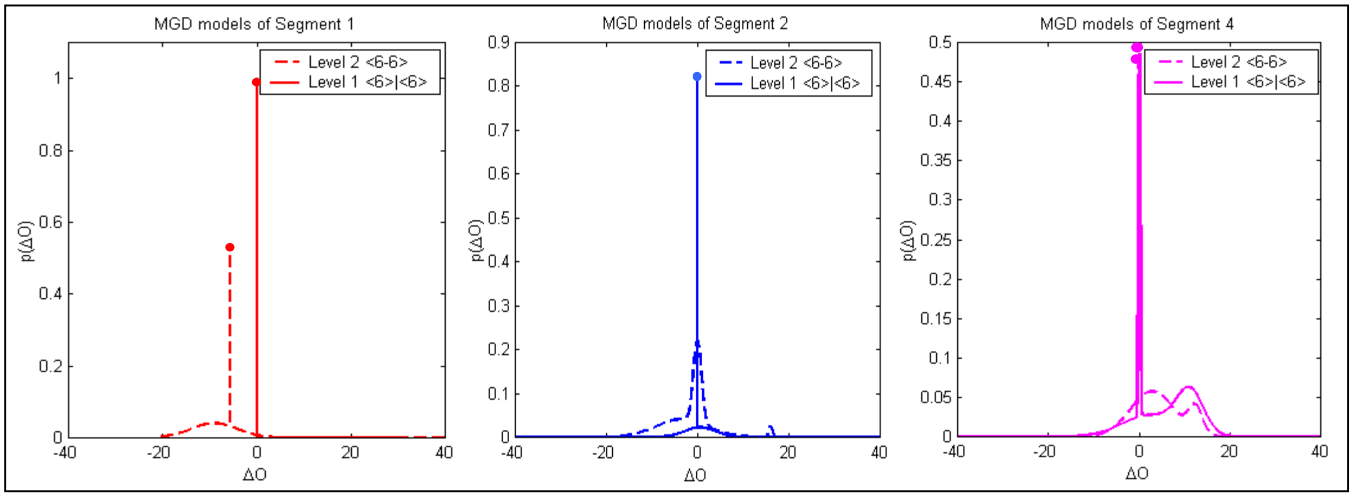

Figure 6. MGD models of some of the distributions form Figure 5.

Objective Prediction and Hypothesis Evaluation

Given the MDG models of the past effects of sequences of modifications, it is possible to infer the probability distribution functions of future objective values given knowledge of the modifications and objective values for the past $N_S - 1$ iterations.

Consider the task of predicting the objective value distribution after a single modification is applied to the current design. Application of a modification type, *t*, will complete a sequence, <…-r-s-t>, Given that all the levels of observed changes in objectives have been modelled and that the objective values for the last $N_S - 1$ iterations are known, there are now N_S possible hypotheses for the probability distribution of the objective value resulting from application of modification type *t* corresponding to each of the observed levels within the sequence. Which hypothesis yields a more accurate prediction depends on the nature of the interaction between subsequent modifications in the sequence and is specific to the particular sequence (and/or segment thereof). In this implementation, the hypotheses are assessed on an on-going basis throughout the search procedure, by the modification sequence modeller. This is done by comparing the predictions of the various hypotheses with what is observed to occur during the search. Each hypothesis within a sequence is then assigned a dynamic quality metric through these observations. The details of the hypothesis evaluation and scoring procedure are beyond the scope of this paper but can be found in (Vale 2002).

The procedure above for predicting a single iteration in the future can be extended in a similar fashion for predicting up to N_S iterations away, though it should be noted that as the predictions become more long-term, the ability of the inferred future objective value probability distributions to aid short term decision making decreases.

Also worthy of note is the fact that objective distribution prediction may result in a future probability distribution function close to an impulse function. In truss design, for example, the <1-3> sequence applied with rules 1 and 3 in close proximity *always* leads to zero change in the truss mass over the sequence. In such cases, where the prediction mechanism is certain of the exact value of the future objective value, it may actually be used in lieu of the true objective evaluation module. The effects of exploiting this side-effect of the algorithm are shown later.

The Modification Advisor

The purpose of the modification advisor is to provide an ordered list of appropriate modifications to make on the next iteration. It is comprised of three primary sub-modules, each invoked in turn at each iteration.

The Performance Observer

When the complexity of escaping local optima exceeds the scope of the sequence models, the procedure can trap itself by strictly following its preferential treatment of 'good' design strategies. The role of the performance observer is to analyze the convergence of the archive of solutions and instruct completely random modifications if local convergence is observed. As soon as an advance is made, the Modification Advisor switches back to its standard practice. Under this regime the search exhibits rapid archive development and high efficiency until it reaches a point where a higher level of creativity is required. The performance observer uses an exponential distribution of observed advances made throughout the search to detect when the probability that an advance in the search should have occurred. If this probability exceeds a threshold value, local convergence is assumed and random modifications then provide a burst of creativity before the algorithm switches back to using the more informed search.

The Pareto Navigator

Relevant only in multiobjective problems, the aim of the pareto navigator is to provide an indication of the most suitable change in objective values to best explore the Pareto frontier given the past search history and the current location in objective space. Typically, it tries to steer the search away from areas that have been well searched with little success, as well as objective values excessively far away from the developing archive of Pareto-optimal Solutions. The *Pareto Navigator* outputs a metric. M_w, for each objective, w, in the interval [-1,1], with -1 or 1 indicating a strong need to decrease or increase the corresponding objective respectively. The magnitude of the metric stresses the importance of the change. Zero indicates that changes of the objective are irrelevant at the current iteration. Implementation details can be found in (Vale, 2002).

The Modification Ranker

The primary difficulty in ranking imminent modifications is that of associating a utility or score to each possible imminent modification. This is the responsibility of the modification ranker, which forms the heart of the algorithm. In this implementation the algorithm compares the ultimate outcomes of its imminent decisions, much like a chess algorithm, and selects the imminent modification that promises the best long-term future.

Figure 7 summarizes how the modification ranker assigns a utility score, u, to each potential imminent modification for the example situation where $N_S = 3$, the total number of modification types is two and there is only one objective. In the figure, a sequence, <1-2-1>, has just been completed. Since it is not necessarily certain that a modification will be able to be applied to the design at an arbitrary iteration, there is an associated probability of application for each modification type. In this implementation, this figure is inferred through observation of past attempts at applying modifications. More accurate probabilities of application can be determined with more complex models (also potentially sequence-based) or through user intuition. Note that in some cases, modifications may be applicable with 100% certainty.

Using the prediction model, the modification ranker can predict the objective value probability distributions arbitrarily far into the future. In this case, this is done to a range of N_S iterations. The objective value distributions that result are shown, as well as those of earlier iterations.

Distribution Scoring. Each final distribution outcome at the prediction horizon is assigned a score, Z, based on its properties. The scoring mechanism favours distributions that promise a high certainty of large improvement in the objective value (whether that may be higher or lower than the current value). Essentially the score corresponds to the maximum possible value of *improvement* × *probability of improvement* for a particular distribution. *Improvement* is defined as the magnitude of the difference between the current objective value and the future objective value in the currently desired direction of change. The *probability of improvement* is found from the probability distribution function and the corresponding value of *improvement*. The Z score found in this way corresponds to the utility at the prediction horizon.

Propagation of the utilities. Once assigned, the utilities at the prediction horizon must 'propagate back' to the imminent decision point. This is done using weighted summation of the utilities based on the rule application probabilities, P_R, according to the principles of sequential decision making (Bernardo, 1998). The calculations performed by the algorithm to determine the imminent utility for rule 1 are shown in figure 7.

Combining utilities and ranking modifications. Having determined the utilities for each objective in turn using the approach outlined above, the algorithm then calculates a combined score for each imminent modification using a weighted sum of the utilities, where the weights correspond to the magnitudes of the metrics, M_w from the pareto navigator for all W design objectives.

A weighted roulette wheel ranks imminent modifications with weights equal to the combined utility scores.

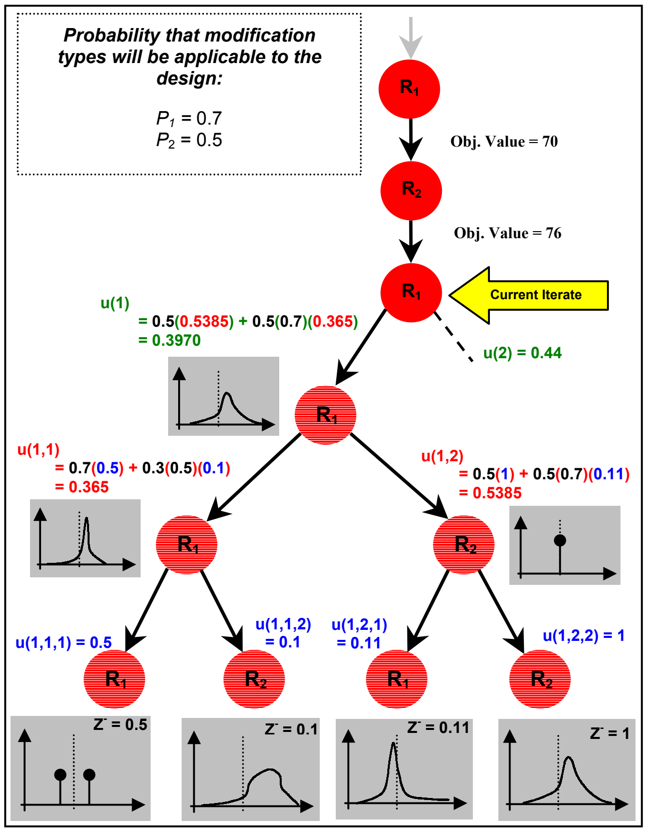

Figure 7. The calculation of utilites for imminent modifications.

Application to Design Tasks

The algorithm presented was applied to two sample design tasks. The first involves truss design using the shape rules described earlier. The second deals with bitmap design.

Truss Design – Cantilever Example

The first example is the design of a cantilever truss structure. The starting structure and loadings are given in Figure 8. Figure 9 shows the typical output of a basic search algorithm, which simply applies rules at random, archiving any new non-dominated solutions discovered in the process. In the context of Pareto optimality, a non-dominated solution implies one that is unbeaten in at least one of its objectives, by all other solutions found. Also shown in the figure are some of the corresponding design topologies. Note that the solutions approach the known optimal Prager topology for the cantilever design (Prager 1977). The trade-off between the number of nodes and the truss mass is visible in the plot of archived objective values.

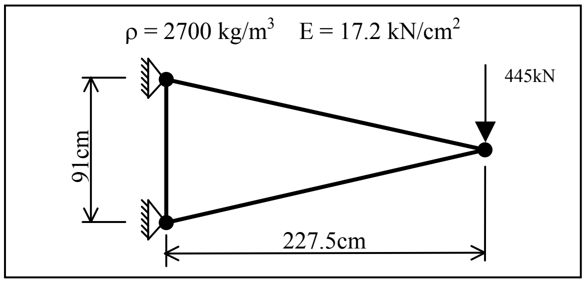

Figure 8. Cantilever truss design problem for minimum mass/minimum number of nodes.

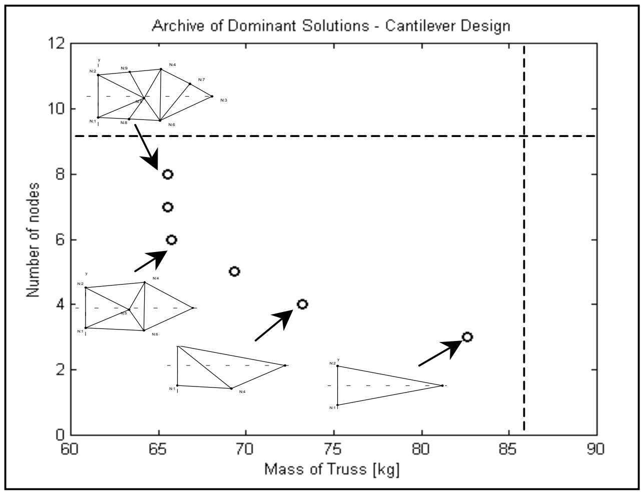

Figure 9. The cantilever truss solution archive

Figure 10 plots the average convergence for the developing archive when augmenting simple random generate-and-test synthesis with the technique presented here. Pareto 'convergence' is measured by calculating the hyper-volume of objective space that is dominated by the archive of non-dominated solutions. The effective gain in efficiency that results from use of objective value prediction is clear from the figure. Here, the search converged to the same archive in about half the number of objective function evaluations used by the un-augmented method. It should be stressed that the augmentation used here only used a sequence modeling length of $N_S = 2$. It is expected that with longer sequence lengths the speed-up effect will be greater.

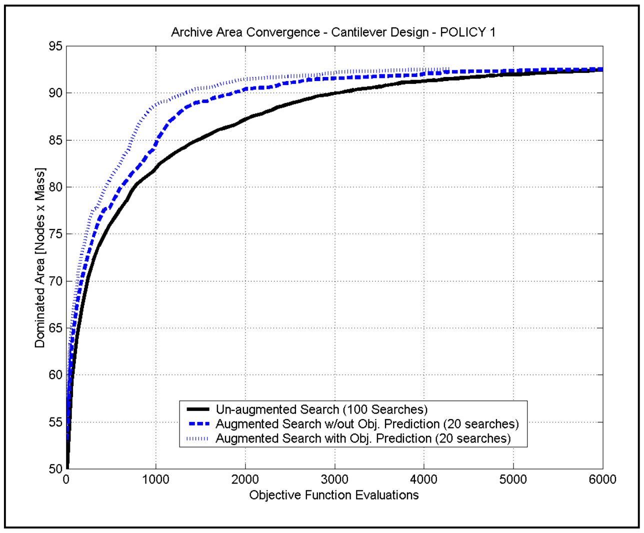

Figure 10. Convergence of the cantilever example.

Sliding Tile Bitmap Synthesis

The second synthesis task revolves around the well-known puzzle of sliding tiles. The classic version of this puzzle consists of a number of tiles arranged in a frame in a grid-like arrangement with one tile missing, as shown in white in Figure 11. When arranged correctly, the tiles typically form an image or sequence of numbers. The challenge of the puzzle is to reconstruct the proper arrangement of tiles from an initial jumbled up arrangement by successively sliding tiles into the free space.

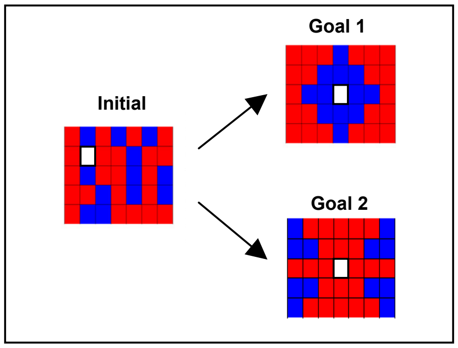

Figure 11. Sliding tile bitmap design task

This is a particularly interesting puzzle as it requires a considerable amount of planning to solve successfully, but is based on a very simple set of allowable actions. In this investigation, the modification formalism is applied to multiobjective design of dual-color bitmap images. The initial bitmap image is a jumbled combination of the two sets of grey tiles needed for the final images and one black 'free space'. Each objective to be minimized is the difference between the current bitmap and a unique 'goal' bitmap. Unless all the goal bitmaps match, it is impossible to reduce all the objectives simultaneously to zero. Expressed as a multiobjective problem, however, this promises to reflect the trade-offs between conflicting goals that might lead to the development of 'morphed' bitmaps of varying degree, with the goal bitmaps obtained at the extremes of the Pareto-front.

Figure 12 plots the average archive convergence for the sliding tile bitmap synthesis task. Improvements in search capability are afforded through use of this technique, with the objective value prediction once again increasing efficiency by more than 15%. Figure 13 plots some examples of the archive members and the pareto front discovered using this search method.

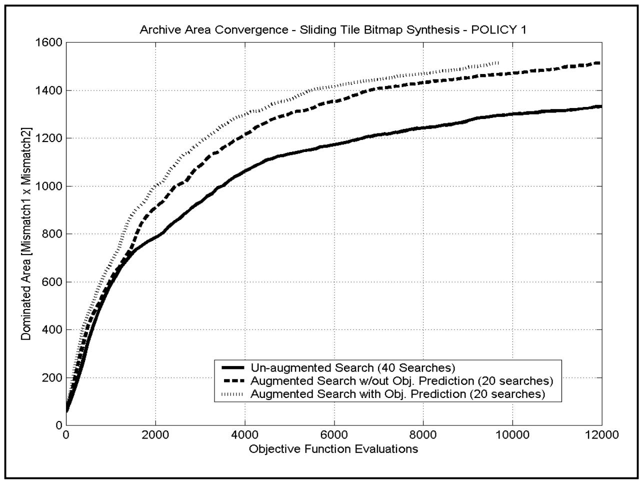

Figure 12. Convergence of the sliding tile bitmap synthesis task.

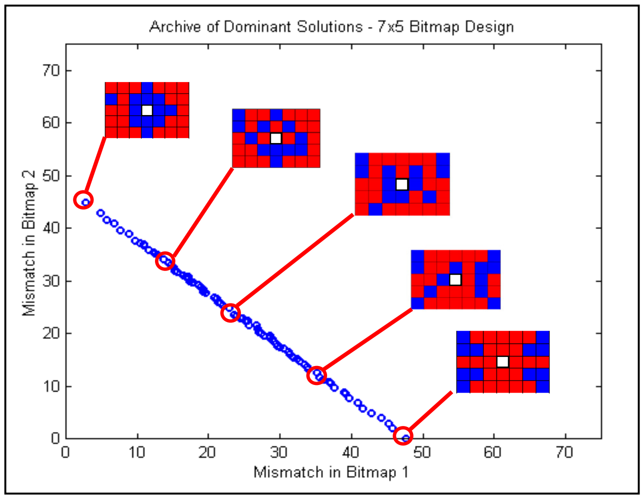

Figure 13. The pareto-front and sample archive members in the sliding tile problem.

Conclusions

The learning technique presented represents a fully automated means of accumulating and re-using effective design modification knowledge for synthesis tasks, resulting in increased synthesis efficiency and consequent improvements in solution quality when using a finite number of search iterations. The user benefits from conclusions that can be drawn from the knowledge base developed by the modification sequence modeler, which can lead to improvements in the modification set and insight into the nature of the problem. In both synthesis tasks presented, the algorithm is shown to result in improvements in exploration efficiency. It should however be noted that in its current form, it is best applied to problems where the modifications are generally applicable, and there is uncertainty in the effect of the modifications on the design objectives. Other classes of problems, such as the constrained geometric knapsack problem (Cagan, 1993), which exhibit a high degree of uncertainty in the applicability of modifications, and a co-incidentally high degree of certainty in the effects of those modifications are not well dealt with by this technique. Future research will focus on overcoming these limitations.

Acknowledgements

Current research support is provided by the EPSRC (UK) and a Philip Leverhulme Prize through the Leverhulme Trust (UK).

References

Bernardo, J.M., Smith, A.F.M., "Sequential Decision Problems," in *Bayesian Theory,* John Wiley & Sons, 1998, pp 13-104.

Bresina, J., Drummond, M., "Integrating planning and reaction: A preliminary report," in J. Hendler, editor, *SRC TR 90-45*. Systems Research Center, University of Maryland, 1990.

Cagan, J., and W.J. Mitchell (1993), "Optimally Directed Shape Generation by Shape Annealing," Environment and Planning B, 20:5-12.

Mitchell, T.M., *Machine Learning,* McGraw-Hill, Singapore, 1997

Prager, W. 'Optimal layout of Cantilever Trusses,' *Journal of Optimization Theory and Applications,* Vol. 23, no. 1, Sept. 1977, pp. 111-117

Shea, K., Cagan J., Fenves, S.J. "A Shape Annealing Approach to Optimal Truss Design With Dynamic Grouping of Members," *Transactions of the ASME,* Vol. 119, Sept. 1997, pp. 388-394.

Vale, C.A.W., *Multiobjective Dynamic Synthesis via Machine Learning,* M.Phil Thesis, University of Cambridge, UK, 2002.

Vaseghi, A.V. *Advanced Digital Signal Processing and Noise Reduction,* John Wiley & Sons, ltd. New York, 1996, pp. 117-125.

Wang, X. "Learning planning operators by observation and practice," Proc. *2nd Intl Conf. AI Planning Systems*, 1994.