Machine Learning Stochastic Design Models

Quentin Falcone

Peyteon High School

Keywords: Conceptual and Preliminary Design; Design Search and Optimisation; Graphical Modelling; Machine Learning; Bayesian Networks.

INTRODUCTION

Due to the fluid nature of the early stages of the design process, it is difficult to obtain deterministic product design evaluations. This is primarily due to the flexibility of the design at this stage, namely that there can be multiple interpretations of a single design concept. However, it is important for designers to understand how these design concepts are likely to fulfil the original specification, thus enabling the designer to select or bias towards solutions with favourable outcomes. One approach is to create a stochastic model of the design domain. This paper tackles the issues of using a product database to induce a Bayesian model that represents the relationships between the design parameters and characteristics. A greedy learning algorithm is presented and illustrated using a simple case study.

BACKGROUND

The design literature is agreed on the need to perform a broad search of the concept space before investing in a more detailed analysis of the final solution concept. Industry's market pressures requires this search to be performed rapidly. To perform such a search methodically requires a conceptual design model that encompasses the breadth of potential solutions for a given product.

The conceptual design space is challenging to model due to the fluid nature of the design at this point. There are two components required for any useful design model: (1) design representation containing the design parameters and characteristics and (2) the relationships between these parameters and characteristics. At the conceptual design stage, the representation needs to cover a vast number of variants with significantly different behaviours. For this reason, the conceptual design stage has resisted formal modelling.

Domain experts are able to navigate through the conceptual design space. It is argued that domain experts have a tacit model of the design space, constructed through their experience in the domain [1]. Some researchers have attempted to extract this model through various methods, however it represents a difficult and expensive task. There have also been investigations into using Machine Learning techniques to analyse product data sets and extract design models. This approach suffers from generating models that are to complex for human designers to understand and verify.

This paper describes research investigating an approach that aims to use product databases for inducing a model, while restricting this model to human cognitive limits. A probabilistic approach is adopted to represent the fluidity that is core to the conceptual design stage. It is worth noting that when inducing a design model from previous products, the resulting model will to a large extent only represent the characteristics and relationships observed from those examples. However, this is not a great problem as a large amount of design work can be considered *redesign*, where the design specification requires a similar product to those already on the market.

Probabilistic Modelling

An early application of probabilistic design modelling was used to evaluate a design's specification [2]. The specifications were described using probability density functions (pdf's). These pdf's were interpreted as acceptability functions, which would gravitate towards the preferred specification. By combining the pdf's of several design parameters, it was possible to compute an overall acceptability score. This overall acceptability score represented the *likelihood* that a designer could successfully carry through the design to completion. Thus, it was feasible to search the *specification space* to fully define a given design subject to a partial specification.

More recently, probabilistic design modelling has been used for forecasting the impact due to the uncertainty in using immature technologies at the conceptual and preliminary design stages [3]. By encoding a product's 'development curve' with the help of domain experts, it is possible to create a set of functions that can be combined to estimate the likelihood of success of a given combination of products or technologies. This information can then be used to help determine the design concepts that are most likely to develop high quality novel products.

Signposting, a method for guiding a designer through the design *process*, uses a probabilistic approach to determine the next design *task* to be undertaken based on the confidence expressed of the current design variables [4]. Signposting's aim is to rapidly obtain a high confidence design definition by suggesting to a designer which of the available tasks would best improve the overall confidence in the design. This provides a framework for collecting domain knowledge from a design state viewpoint. Domain experts provide an approximate (probabilistic) indication of suitable tasks given certain conditions.

These approaches demonstrate that a probabilistic design model can be effectively used to search the design space. However, these approaches all require extensive domain expert input and hence, it would be valuable to be able to induce such models algorithmically.

Cognitive Aspects and Machine Learning

Human cognitive aspects have had little impact on Machine Learning and Data Mining methods. This is largely due to the fact that the results from these methods are most often used by machines rather than humans. However, if Machine Learning approaches are to make an impact on how a design space is searched by humans, the cognitive issues must be addressed. When performing a search, Short Term Memory (STM) is being used to perform rapid comparisons and direct further searching. Unfortunately, STM is limited in size to about seven items [5]. Larger sets must be grouped, or chunked, into sets of related objects which are handled in a more abstract manner.

These cognitive constraints inform the search heuristics for any design model that is to be used by a human designer. Namely, any particular model that is to be used by a designer to understand how some design variable relates to the rest of the design should not contain more than seven variables. However, when the shape of the design space is so complex that more than seven variables are required, there is the option of using chunking. Rather than use the design variables directly, a latent variable is used. These latent variables represent 'hidden' or lower level design models. Constructing these represents a challenge, as these latent variables need to be meaningful for the chunking to be successful.

GRAPHICAL MODELLING

Graphical modelling provides a means for representing the causal relationships between design variables [6]. It should be noted that design variables include both design parameters, i.e. the aspects of the design that are directly determined by the designer, and design characteristics,

i.e. the aspects of the design that are the result of the design parameters. Graphical modelling differs from parametric modelling in a number of ways. First, there are no exact equations that bind the design variables. The relationships are represented by probability distribution functions, which allow for ambiguity and flexibility that is core to the early design stages. Second, as the relationships are now probabilistic, it is possible to also include relationships that are difficult to model exactly, e.g. aesthetic properties. Third, the approach leads to trivially understanding the dependency path that design variables have. This path allows for a designer to understand which design variables either are affected or need to be changed to obtain a desired result, depending on the causality direction.

Constructing Graphical Models

There are two primary methods for constructing a graphical model: (1) manually identifying the relationships and generating an appropriate pdf, and (2) using machine learning techniques for inducing a graphical model from prior data. It is also possible to use a hybrid method that combines both of these approaches. This work is primarily concerned with the second approach.

There are two components to inducing graphical models: identifying where to place the arcs (or edges) that represent a direct relationship between two variables, and how to compute the pdf that is to be contained by the arc. The focus here will be on identifying where the arcs are needed, as this must be done prior to determining the associated pdf's.

A graph is fully defined by its nodes (, the set of design variables), and the set of edges between them (, the arcs representing direct relationships). The set is determined by the product database description. It now remains to discover where the edges should lie, i.e. to search the space containing all potential sets. For any given edge set , it is possible to compute how likely this would have resulted in the product database. This provides for a utility function on the edge space, measuring the 'goodness' of any given edge set. Using this a number of search algorithms can be applied, e.g. Genetic Algorithms or hill climbing. The approach adopted here is informed from the Bayesian Network Toolbox (BNT), which is publicly available software [7]. Fundamentally, the BNT implements a greedy approach. The algorithm starts with an empty edge set, and constructs a number of new sets with a single edge. These are all evaluated, and the best one is retained as the seed for the next iteration. From this seed, a number of edge sets are created that differ by a single edge, either by adding or removing an edge from the seed set. These are evaluated, retaining the best scoring edge set.

This process is repeated a preset number of iterations. This process does guarantee to find the best edge set. However, as the model is probabilistic, this is sufficient.

Using a Graphical Design Model

A natural case for using a graphical design model is for performing redesign tasks. In this event, a designer starts with a previous example that nearly satisfies the new design constraints. The designer then identifies a variable that fails a constraint. Recall, this could be either a design parameter or characteristic. In the graph, the neighbouring nodes (design variables, again either parameter or characteristic) need to be considered. As the graph is causally directed, the neighbouring nodes are split into the parent and child node sets. The child nodes represent the design variables that will be affected by the change in the chosen variable. The parent nodes represent design variables that will need to be changed to achieve the desired original variable setting. Further, if the arcs have been populated with conditional probability distribution functions, these can be used to provide estimates to what values the neighbouring node variables should take. By propagating this through the network, it becomes possible to estimate the total perturbation to the whole design.

Issues with Graphical Modelling

A common problem with data driven methods is that they tend to require large quantities of training data. This typically presents a problem in the design domain as data is expensive and hence scarce relative to the volume frequently seen in other machine learning applications. To overcome this issue, there are a number of options:

1. Do nothing: train the model with sparse data and measure the statistical significance and proceed with as much caution as this significance requires.

2. Seed the model with expert knowledge: by providing an initial model that is believed to be valid to to its origins, the data volume requirement is reduced.

3. Review the model by domain experts: again, this provides confidence by verifying the model using domain experts.

From the above options, it would appear that using Machine Learning techniques do not offer great advantages over extracting

the models directly from domain experts. However, this does not take into account the different levels of effort required to extract a complete (or near complete) model from a domain expert versus either extracting a seed model containing the most important and obvious relationships or being provided a model for verification, a considerably more passive exercise.

Micro-Models

The concept of micro-models is inspired from the cognitive limits imposed by human Short Term Memory (see Section 2.2). Micro-models are designed to 'fit' into STM, allowing a designer to get a good understanding of the behaviour of some aspect of the design. It is important to note here that these micro-models represent only partial models of the design space, and therefore a number of micro-models are needed to cover the whole design model.

Definition

Micro-models are defined principally by their size: the model should not contain more than seven variables. Where this is not possible, *meaningful* latent variables should be used in place of design variables. The latent variables are in their own right micro-models, and this layering represents the chunking function that can be used by STM.

More important than the construction of any single micro-model is the construction of the set of micro-models. These must fulfil two requirements:

1. The micro-models must cover the full design model; and
2. The micro-models must be meaningful.

The first of these requirements can be algorithmically achieved by verifying that the whole design model has been covered. The second requirement is more difficult to achieve algorithmically. It is possible to flag certain design variables as being particularly meaningful or important, and then devise search heuristics that search for micro-models that each contain one of these variables. Another option is to manually label each micro-model with a descriptive name. This is especially useful where micro-models are used as latent variables, as this meaningful name provides the mechanism for STM to relate to the latent variable in the same manner that a designer can relate to a direct design variable.

Micro-model intersection

The method used to bridge the gap between micro-models and a complete design model is by ensuring appropriate overlap between micro-models. Thus, the union of the micro-models should be a good approximation to the total design model.

The intersection of two models provides a communication channel between these models. However, the aim of using micro models is to remove the need to consider the whole de sign model. Therefore, a designer should only have to consider two models when absolutely necessary. To illustrate how this message passing between models should work, the following scenario will be considered. Two micro-models, and contain a total of five design variables. These two intersect at variable , as illustrated in Figure 1.

The following three heuristics can be applied to using micro-models:

1. Variable : propagate only in ;
2. Variable : propagate in , highlight potentially affected;
3. Variable : propagate in both and .

This allows a designer to proceed with exploring the design space with minimal concern to the rest of the model. The designer need only propagate information through to another micromodel when modifying variables that exist in the intersection of two micro-models. A warning is provided when there is a potential need to propagate information.

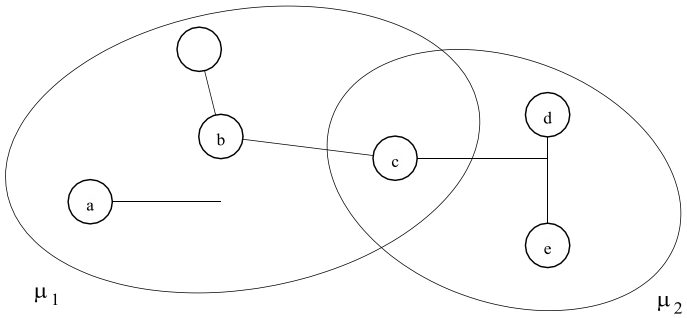

Figure 1. Illustration of two overlapping micro-models: design variable lies in the intersection and is used to propagate design information between the two micro-models.

LEARNING GRAPHICAL MODELS FOR STOCHASTIC DESIGN

There are two aspects of learning for graphical models: structure learning and distribution learning. This research is primarily concerned with the former, and generating the latter 'on the fly' using the supplied parametric product databases. There are a huge number of graphical models for a given set of design variables, and the challenge is to develop heuristics to search for 'good' models. 'Good' models are determined by their performance against a set of metrics, such as *validity*, *understandability*, and *interestingness* [8]. Validity measures what proportion of the data can be covered by the model. Understandability provides a complexity measure that can represent how easy it is for a designer to understand a model. Finally, interestingness measures the novelty of representation of a model in a design domain. These metrics have been listed in order of difficulty of measuring. Validity can be measured directly against the database supplied. Understandability requires a measure of human ability to understand a given model. Interestingness must be measured against the current state of domain knowledge and combined with a subjective element supplied by the domain expert.

The model search algorithm this research proposes is an evolutionary one. The results of one iteration provide the starting point for the next iteration and this is repeated until some termination criteria have been met. The algorithm requires seeding before the first iteration can begin. These initial seed models are created using the pairwise information content as determined by the data set and Equation 2. These provide a reasonable starting point from which to evolve more accurate graphical models.

Metric implementation

The three named metrics above are conceptual and require detailed implementation. The most straightforward of these is measuring the model's validity. This can be done by using information content statistics. In effect, 'interesting' graphical models are those whose arcs contain useful information about the causal relation between the nodes. This represents a hypothesis about the structure of the relationships between the various design variables. The information content measures the variance of the conditional probability distribution induced from the product data base. The greater the information content, the better the model.

Measuring understandability requires a combination of measuring the complexity of a model

and how well this model can be mentally absorbed by a designer. Using the simple interpretation that the 'size' of Short Term Memory (STM) is large enough to hold seven objects [5], the metric is designed to identify good structures having less that seven variables and edges. The most understandable model has only two variables and one edge, and thus the metric increases the penalty slowly until the total size is greater than seven at which point the penalty rapidly increases.

Without a reasonable interpretation for interestingness, this metric cannot be implemented at this stage. Future work will compare models to the current state of domain knowledge. Where the models are good and there is a significant difference in the model to the current domain knowledge, this will incur a high score.

These metrics are specified for evaluating a *single* model. The aim is to identify a collection of diverse micro-models that together provide a good explanation of the design domain. However, the above metrics, with small modifications, can also be applied to sets of models. The problem is now a matter of identifying a good set portfolio of models to provide a covering representation of the design domain.

Graph Search Heuristic

The graph search algorithm implements a greedy search heuristic based on a measure of the information content of the conditional probability distribution. Recall the definition of conditional probability:

(1)

Where the events and are independent, Hence, when and are independent . By considering the difference between the *observed* conditional and prior probability distributions, it is possible to measure the mean variation in this difference:

(2)

The variation, represents how much more information is contained in the conditional probability distribution above the information contained in the prior probability distribution. A large value for indicates that the conditional probability distribution contributes greatly to the knowledge of the domain while a small value indicates that the two variables are likely to be relatively independent of each other.

The graphical model search algorithm begins by measuring the pairwise information content between each variable pair. This is computed for both directions as in general

. For each design variable, the system is seeded with a micro-model containing the given variable and the variable that has the greatest information content of its conditional probability distribution. Where a micro-model would be repeated, the variable with the next highest information content is selected.

These micro-models are ordered in increasing information content order. The next step is to merge micro-models with low information content, creating a new micro-model whose information content is given by the sum of its parts. The first two 'smallest' models with a shared depth width

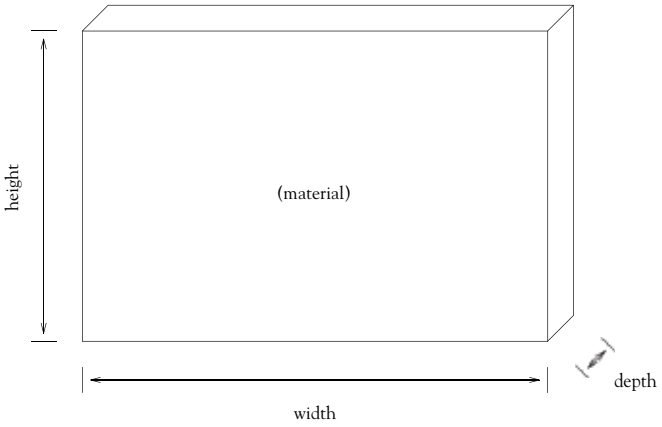

Figure 2. Design parameters of the flat screen display. The design evaluation criteria are: weight, cost, expected life, and expected sales volume.

variable are merged, resulting in a new smaller set of micro-models. Where there are more than two candidate models for combining, the tie breaker is determined by (1) resulting model complexity followed by (2) lower information score. This is repeated until all micro-models are exhausted.

The above greedy algorithm results in a single graphical model. This is not as useful as a set of micro-models that describe various roughly independent aspects of a design domain. The point at which the algorithm should be stopped therefore needs to take into account the average complexity and the micro-models' total representation of the design domain.

ILLUSTRATIVE CASE STUDY

For the purposes of illustration, an 'artificial' design domain has been developed. From this hand-crafted domain model, a random sample of designs were created. This data set serves two purposes. First, it forms the basis of the illustration of the machine learning algorithm. Second, it is used to compare the nature of the probabilistic design model with the original (source) model. The models that produced by this algorithm represent the causal structure between the design variables and include the conditional probability density function between the two variables. The probabilistic model generated in this case study is critically compared to the original model in Section 6. Once a model exists, it can then be used as a design exploration tool. A proposal for achieving this is introduced in Section 7.

To illustrate the graphical models that are generated, a small design case study was created. A flat screen display domain was constructed (Figure 2) by defining a small set of relationships between the design parameters (aspects of the design determined by the designer) and objectives (aspects of the design determined by the parameters). These relationships were designed to be sufficiently complex that all the design objectives could not only be expressed in terms of the design parameters. As the relationships were known before the analysis, it was possible to measure how well the analysis method performs.

Table 1. Information measures for seed micro-models and causal direction.

Micro-model		Information
	life	0.1322
	cost	0.1171
	weight	2.2337
	life	1.4385
weight	units	0.1388
cost	units	2.8349
life		0.1066
units		0.1243

Design space definition

The design space of the disply panel was represented by four design parameters and four objective criteria, forming an eight-dimensional space. The design parameters were: width ( ), height ( ), depth ( ), and material ( ), all of which were randomly sampled from a uniform distribution. The objective criteria were: weight, cost, life expectancy, and sales volume. These were related as follows:

(3)

(4)

(5)

(6)

where and represent noise and are randomly sampled values from a normal and uniform distribution, respectively. Note that in this model the number of units sold was modelled only by a random value. This represents the subjective nature of the customer population. A key aim of this case study was to find out an explicit relationship for this objective based on the remaining parameters. In addition, all objectives had a small amount of gaussian noise added. This again was to represent the noise occurring in real world domains due to other factors that this simple model did not include.

Finally, a database of 200 examples was sampled from this model. This represented the 'past designs' that would form the basis of the analysis. The size of this database was set similar to other product databases that would be analysed.

Generating Graphical Models

Using the algorithm described in Section 4.2, a set of seed micro-models were generated. These models are listed in Table 1. From this table it can be seen that the lowest scoring micro-model is given in line 7 (life, = 0.1066) and the second lowest is in line 2 (cost, = 0.1171). Both these micro-models share , and hence are merged to form the micro-model in Figure 3.

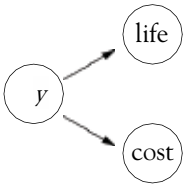

Figure 3. Micro-model resulting from the first merge step in the greedy algorithm, with information content 0.1066 + 0.1171 = 0.2237.

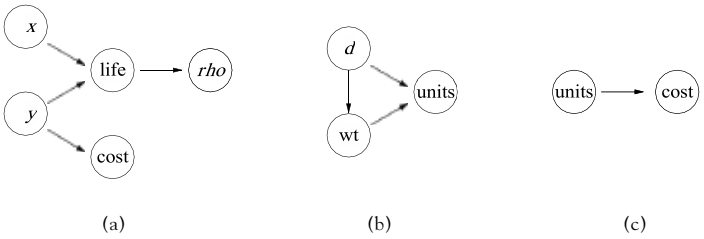

Figure 4. Micro-models resulting from five iterations.

Repeating this process for another 4 iterations results in the set of micro-models depicted in Figure 4. This model provides a good balance between simplicity and completeness of model. The process ends with the graphical model shown in Figure 5. This represents the 'total' graphical model of the domain, as determined by the greedy search algorithm. Although it appears to be manageable in this design case, this grand-unifying model is not desirable for more complex design domains involving significantly larger numbers of design variables.

DISCUSSION

There are two aspects that need to be considered with respect to the creation and use of design micro-models. Firstly: do the micro-models provide a *reasonable* description of the behaviour of the design domain? Secondly: how does the micro-model representation compare to other probabilistic design modelling approaches?

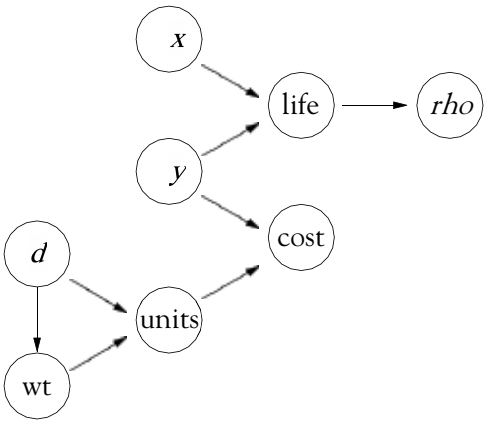

Figure 5. The final graphical model.

Domain behaviour representation

The 'true' behaviour of the display design domain were defined in Section 5.1. This definition will be used to compare the results of the learning process after five iterations, as displayed in Figure 4.

Micro-model (a) does not provide a very good explanation of the behaviour of the design domain. At best, the relationship between Life and is accurate, although the causality arrow should ideally be reversed. The other relations bear no similarity to the source model.

Micro-model (b) is a closer representation of the source model. Although 'units sold' was modelled as an unknown externally affected design variable, it was related to cost and life. Life is explicitly determined by material, , which in turn is strongly related to both and weight. So while this model does not provide an explicit description of the underlying model, it does provide meaningful suggestions as to how the design variable interact.

Micro-model (c) is very simple and relates the design variables 'units sold' and cost. This relationship does have a very high information content (hence why it has not been merged with another micro-model), and is very close to the version in the source model.

While the micro-models have not provided a close reconstruction of the design domain model, certain high information aspects have emerged through the data. In the current implementation of this algorithm, no data preprocessing was performed. Thus it is entirely possible that the nature of the raw data biased the results. Further, this design model had some very na¨ıve assumptions placed upon it, namely that the design parameters were to be sampled uniformly. In reality this would not be the case, and there would be some tacit knowledge embedded within the sample of design parameters.

Comparison to alternative methodologies

The use of probabilistic and stochastic modelling techniques in design is not new. Similar to most modelling approaches, the bulk of probabilistic methodologies strongly rely on human expert input. For probabilistic models there are two aspects involved: (1) the 'structural' modelling of identifying which variables are related and (2) the distribution modelling, explicitly determining the shape of the distribution.

Prior domain knowledge about the nature of the variable distributions and relationships are used to achieve flexibility in the design process [9]. This approach uses a 'Design Preference Index' to indicate 'goodness' of flexible designs. The design performance is determined probabilistically, and coupled with the designer's preferences, concepts are selected for further detailing. In a similar manner, uncertainty can be probabilistically represented in the early stages of variant design and can be used to estimate the performance and other objectives of the final design [10]. Uncertainty in design has also been used for process modelling [11] and for systems integrity [12].

These methods provide excellent examples of the use for probabilistic modelling methods, but do not address the challenge of creating the model in the first instance. There has been some work addressing the representation of product databases suitable for learning probabilistic models [13]. The work reported in this paper addresses how to use that to generate both the structural representation and the distribution functions, and thus provides an important contribution to the probabilistic modelling efforts.

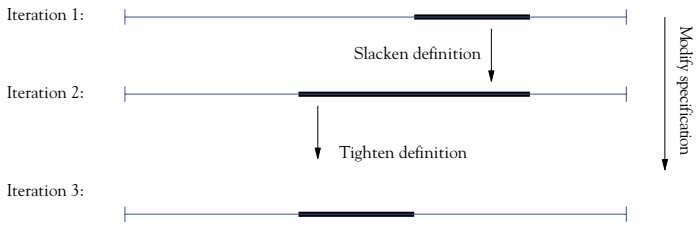

Figure 6. Iterative process of slackening and then tightening of a design variable to shift its value. For each iteration, the thin horizontal line represents the whole possible design range. The thicker line represents the range of values currently specified.

INTERACTIVE DESIGN SEARCH TOOL

Given a set of micro-models, it is necessary to provide an intuitive interface for the designer. This interface provides the means for the designer to 'query' the design model, and obtain suggestions for further specifying the design. It is important that this interface provides sufficient transparency to the design micro-models, otherwise it runs the risk of being viewed as a black box. Designers lack trust in such black boxes, which results in poor uptake of new technologies.

As this approach is to be used during the early stages of design, its main use will be to complete an initial design specification. A designer would supply a partial design specification to start the search process. By using the probabilistic design model, it is possible to compute the probability density function for the remaining unspecified design variables. The nature of the graphical model directs the designer to first consider the design variables neighbouring those that have already been specified, as these will have the most accurate pdf's. By considering the graphical design model, it is clear to the designer which variables should be considered in the next iteration. Further, it becomes possible to get a preview of the impact of a change to the design.

In addition to tightening the specification, it will also be necessary under certain circumstances to be able to slacken a specification. This will occur in the event that a design becomes infeasible due to conflicting requirements. By considering each design variable's pdf just outside the current specified range, searching for likely areas, a set of slackening suggestions can be made. This then allows for a design variable's requirement to be shifted along by tightening the specification in the next iteration (see Figure 6).

CONCLUSIONS

This paper has presented a machine learning algorithm for learning a Bayesian network from a design database. The algorithm used the information content of the conditional probability distribution, and implemented a greedy approach to construct the Bayesian network. It was argued that this construction should be terminated before all the design variables form a single monolithic network, but rather form a set of smaller networks. These smaller networks are more easily interpreted by human designers who are able to extract domain knowledge. Finally, the paper proposed how these networks could be used as part of an interactive design search and optimisation tool.

The results generated by the display case study were mixed. It is not clear if this is due to the artificial nature of the underlying data generating model. This model in effect used a design of experiments approach, and therefore the design parameters did not contain any implicit knowledge as they would have had the data arisen from an industrial case study. In addition, no preprocessing had been performed on the data.

Further work is also required in evaluating the work psychology aspects of these small stochastic networks, specifically how well designers do understand them in practise. There is also a need to further explore the interface between the designer and the networks when searching for good design concepts. Suitable data preprocessing methods must also be considered to ensure the data is well conditioned for this type of learning algorithm by removing any biases that might exist within the data. Finally, there is a need to enable the encoding of prior knowledge into the system where it exists.

REFERENCES

1. S Ahmed. *Understanding the Use and Reuse of Experience in Engineering Design*. PhD thesis, Cambridge University Engineering Department, 2001.

2. D R Wallace, M J Jakiela, and W C Flowers. Design search under probabilistic specification using genetic algorithms. *Computer Aided Design*, 28(5):405–421, June 1996.

3. M R Kirby and D N Mavris. Forecasting technology uncertainty in preliminary aircraft design. In *4th World Aviation Congress and Exposition*, San Francisco, CA, 1999. SAE Paper No 1999-01-5631.

4. P J Clarkson and J R Hamilton. 'Signposting': a parameter-driven task-based model of the design process. *Research in Engineering Design*, 12(1):18–38, 2000.

5. G A Miller. The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological Review*, 63:81–97, 1956.

6. D Heckerman. *Learning in Graphical Models*, chapter A Tutorial on Learning with Bayesian Networks, pages 301–354. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 1999.

7. K P Murphy. The Bayes Net Toolbox for Matlab. In Edward J. Wegman, Amy Braverman, Arnold Goodman, and Padhraic Smyth, editors, *Computing Science and Statistics*, volume 33, pages 331–351. Interface Foundation of North America, 2001.

8. S Mitra and S K Pal. Data mining in soft computing framework: A survey. *IEEE Transactions on Neural Networks*, 13(1):3–14, 2002.

9. W Chen and C Yuan. A probabilistic-based design model for achieving flexibility in design. *ASME Journal of Mechanical Design*, 121(1):77–83, 1999.

10. R Crossland, J H Sims Williams, and C A McMahon. An object-oriented modeling framework for representing uncertainty in early variant design. *Research in Engineering Design*, 14:173–183, 2003.

11. Y M Goh, J D Booker, and McMahon C A. Evaluation of process modelling approaches to support probabilistic design analysis. In A Folkeson, K Grale´n, M Norell, and U Sellgren, editors, *Proceedings of the 14th International Conference on Engineering Design*, Stockholm, 2003. Design Society.

12. D J Pons and J K Raine. Design with uncertain qualitative variables under imperfect knowledge. *Proceedings of the Institution of Mechanical Engineers Part B: Journal of Engineering Manufacture*, 218:977–986, 2004.

13. B Johansson, D A DeLaurentis, and D N Mavris. Managing design data for probabilistic evaluation of aircraft concepts. In A Folkeson, K Grale´n, M Norell, and U Sellgren, editors, *Proceedings of the 14th International Conference on Engineering Design*, Stockholm, 2003. Design Society.