Learning Empirical Knowledge To Assist Preliminary Design

M L MAHER and H LI*

ABSTRACT

The reuse of the experience of design and construction of a major project is ad hoc and depends on the individuals involved in the project being present on a similar project in the future. At the same time, the development of knowledge-based systems to support the design process requires the encoding of previous experience in a form that can be applied to future design projects. Machine learning techniques can be applied to automate the reuse of design experience and to facilitate the development of design knowledge bases. The application of machine learning techniques in a design domain requires the consideration of the representation of the learned design knowledge, that is, a target representation, as well as the content and form of the training data, or design examples. This paper proposes a target representation called a design concept and presents a methodology for learning design concepts from design examples. The method is illustrated by applying it to examples of bridge designs.

Key Words

machine learning; design; conceptual clustering; knowledge-based design

INTRODUCTION

Incorporating learning ability in knowledge based design systems can make direct use of the design data available from design projects and can relieve the burden of maintaining a knowledge base. The direct application of existing machine learning techniques assumes that the target representation, the representation of the knowledge learned, is in an appropriate form for application to a new problem. We propose that the representation of design knowledge is fundamentally important in a knowledge based design system and the learning methodology should be developed based on the target representation rather than on a theoretical approach to machine learning. For example, a methodology reported by Arciszewski (1987) is to learn rules from design data, and a system developed by Reich (1991) is to learn hierarchic representation of design examples. In this paper we consider a target representation of design knowledge, called a design concept, and how such a representation can be learned automatically by adapting and combining existing machine learning techniques and statistical analysis.

Construction Informatics Digital Library http://itc.scix.net/
paper w78-1993-2-301.content

* *Department of Architectural and Design Science, The University of Sydney, NSW 2006, Australia.*

The learning methodology presented has two major components: concept aggregation and concept characterisation. Concept aggregation involves decisions about what entities or attributes are to be selected and grouped into a concept. This component is based on the techniques for learning from examples, except in our case the attributes of the examples may vary from example to example. Concept characterisation involves deriving inherent associations among attributes in the proposed concept from the values of the attributes in the design examples.

The methodology is defined following a specification of a representation of design examples, the input to the learning system, and a specification of a representation of design concepts, the output of the learning system. In the next section we present the representation of design examples and design concepts. The following section presents the methodology for learning design concepts from design examples. Section 4 shows how a design concept can be used in a new design problem. Finally, a summary of the methodology and directions for further research are presented.

Representing Design Examples and Design Concepts

Design Examples Represented as Attribute-value Pairs

We propose a representation of design examples as a set of attribute-value pairs. The use of attributes to describe a design example allows a set of relevant attributes to be identified, the values of the attributes provide quantitative and qualitative descriptions of the attributes. An example of a set of attribute-value pairs describing a continuous bridge design example is shown in Table 1.

This representation for examples is widely used in many machine learning techniques and is suitable for our purposes because the attribute names characterise the design example and distinguish it from other design examples. It is possible for the set of attributes to vary from design example to design example, thereby allowing a design to be charaterised by its attributes as well as described. Also, the use of attribute-value pairs does not require an understanding of the design process that was used to produce the design example. Such process information is hard to articulate and may not be relevant in a new design situation. Given that the purpose of the presented machine learning technique is to aggregate and characterise design concepts, a simple representation for design examples provides the flexibility needed to generate general relationships among attributes, without the bias of the relationships the designer may have used in a particular design project.

BN	Bridge Name	Mr-124, ...
LC	Location	Sutton-interchange...
TT	Traffic Type	vehicle,pedestrian...
CC	Construction Year	1981,1984,...
CS	Cost	low, medium, high
CL	Clearance	0.5 - 70.5 m
PH	Pier Height	1.2 - 39.7 m
TL	Traffic Lanes	2 - 9
CR	Cross a River/Road	river, road
RB	River Bed condition	good, medium, bad
PL	suPerstructure design Load	NAARSA-1976
BL	suBstructure design Load	NAARSA-1976
LN	bridge LeNgth	93.5 - 379.5 m
DW	Deck Width	7.5 - 27.5 m
DC	section Depth at the Center	0.68 - 1.45 m
DS	section Depth at the Support	0.89 - 2.78 m
SN	Span Number	2,3,4,5...
MS	Main Span	50 - 203 m
ST	Superstructure Type	continuous-girder
GH	Girder Height	0.68 - 1.35 m
GW	Girder Width	1.59 - 5.01 m
GM	Girder Material	concrete, steel
GS	Girder Shape	trapezoid
NG	Number of Girders in a section	2, 4,5,6...,18
NTC	Number of Tendons at the Center	88, 124, 56,...
NTS	Number of Tendons at the Support	128, 164,...
FT	Foundation Type	excavative, pile
BT	Bearing Type	Pot-type,...
DJ	Deck Joint	Claw-and-Gland,...
DA	Deck Area	789-4320 cm2
TA	Tendon Area	12.4-268.7 cm2

Table 1: Attributes and values for a continuous bridge design example

Design Concepts Represented as Empirical Networks

We present a representation called an empirical network to structure empirical design concepts which captures both quantitative and qualitative attributes as nodes which are linked either by empirical formulas (for quantitative attributes) or design patterns (for qualitative attributes). The set of design attributes in an empirical network can be viewed as the identity of a design concept. Each attribute further defines a design concept by including a default value and value range. An empirical network has two

representational components: nodes and interrelationships, as shown in Figure 1.

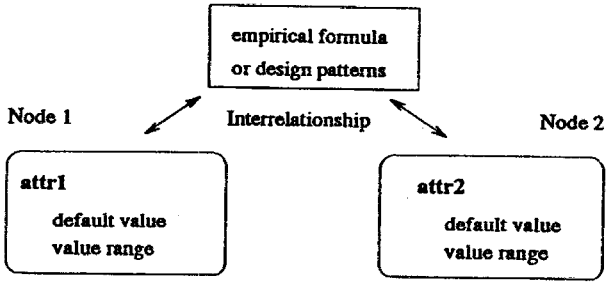

Figure 1. Components in an Empirical Network

A node is the representation of a design attribute, its default value and value range. The nodes in an empirical network comprise both the requirement and description attributes of a learned design concept, although the identification of a single node as one or the other is not made. A node can be either a quantitative or qualitative type. The default value represents the most frequently assigned value of the attribute in the given set of design examples. The value range for the quantitative type is the gap between its maximum and minimum values, and for the qualitative type is the accumulation of all possible discrete values. A node is represented as a corner-smoothed rectangle in Figure 1.

The interrelationships reflect empirical associations among design attributes. Two kinds of interrelationships are represented: an empirical formula captures associations among quantitative attributes, and design patterns capture associations among qualitative attributes. Interrelationships are shown in a rectangle in Figure 1.

An example of an empirical network for a design concept is illustrated in Figure 2. The nodes are design attributes for a cable-stayed bridge. Relationships among the attributes are empirical formulas or design patterns relevant for the preliminary design of a cable-stayed bridge.

(a) Empirical formulas

It is largely accepted that designers are able to generalise numerical relations through design experience. Implicitly we develop a proportional profile for familiar objects. For example, we know the width of a door is approximately half of its height. Proportions are important in preliminary design, and empirical formulas are explicit forms of such proportional relations. The complexity of the mathematical formula in a machine learning

system can initially be limited to those that people already easily generate without machine learning, for example linear equations. As the techniques for automatically learning empirical formulas is more widely used and accepted, the formulas may be more complex, for example exponential equations and inequalities.

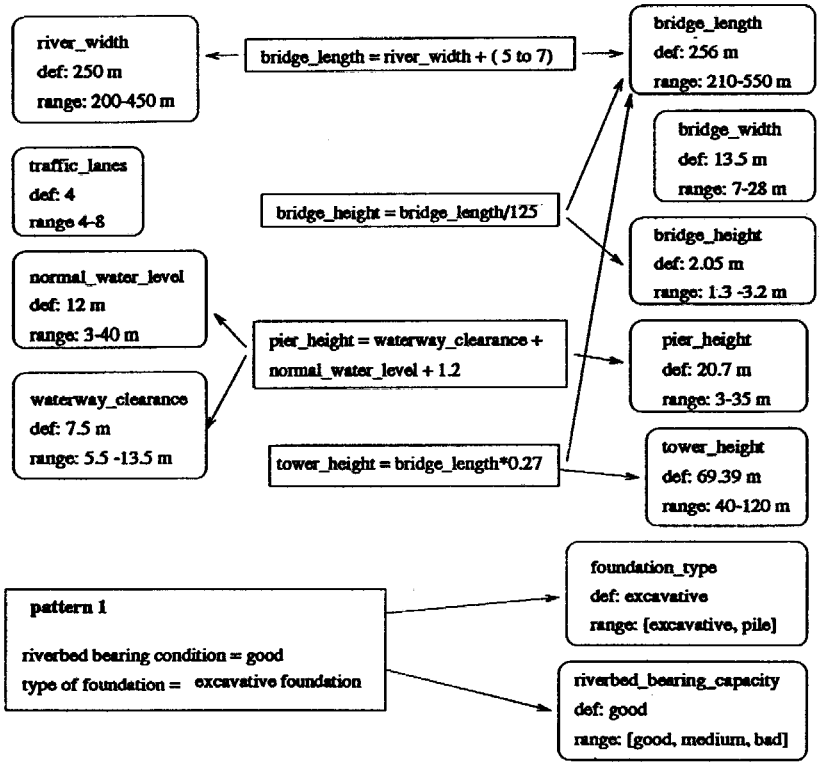

Figure 2. An Empirical Network for Cable-stayed Bridge Design

Our use of the empirical formula is not to model causal relationships among design attributes but rather to model an empirical relationship that can generate a reasonable new design in the range of attributes considered. In other words, the adequacy of the empirical formula is dependent on the collective inference from all included attributes, thus a procedure for assigning a value to the individual attribute is de-emphasized.

(b) Design patterns

Qualitative attributes interact with each other as do quantitative attributes, although the interaction cannot be expressed as mathematical formulas.

Usually what value a qualitative attribute should take is dominated by how that value will be compatible with other qualitative values. As a result, qualitative attributes and their values in a design solution define a feasible state, a style or a pattern, of how those attributes appear together. The frequently occurring states are interesting because they reflect a combination of values those attributes typically take. We refer to those feasible states as design patterns.

The representation of a design pattern can be a logical expression, a rule, or a collection of attribute-value pairs. We choose here to use a collection of attribute-value pairs because of the non-directional nature of the representation and the anticipated use of the pattern to derive a new design solution. As design is an ill-structured problem, the attributes of a design concept are not predefined as requirements or artifact description. An attribute that serves as a requirement in one design example may be considered part of the description of the design solution in another. Therefore, it is advantageous to have a representation of design concepts in which the design patterns are non-directional, and do not imply which attributes are known first.

Automated Learning of Design Concepts from Design Examples

A methodology for learning design concepts is considered in two major parts: concept aggregation and concept characterisation. In concept aggregation, the design examples provide a basis for learning descriptive attributes for a design concept. Each design concept is a set of attributes, these attributes will become the nodes in the network introduced in Section 2. Which collection of attributes belong in a design concept and how those attributes are described is the focus of concept aggregation. The result of concept aggregation is collection of sets of attributes, where each set forms a concept, and each attribute in a concept is further defined by its default value and value range. During concept characterisation, the relationships among the attributes in a concept are defined. These relationships are the links between the nodes in the network introduced in Section 2. The design examples are used to provide the data needed to derive the relationships.

Concept Aggregation

Many of the concept learning techniques have a goal of providing a representation that can be used to classify or recognise a new example (Winston, 1975; Fisher, 1987, Reich 1991). In design, the goal of using concept learning techniques is to generate a representation that can be used to generate new design examples, not to classify examples that have not been seen before. Also, many techniques that learn from examples require that each example is classified as positive or negative. In design it is unusual to

categorise design examples in this way. A design example may have certain positive characteristics and certain negative ones, but it is unusual to classify an example as entirely positive or negative. For our purposes, we assume that all design examples are positive. Finally, limiting the description of design examples to a predefined set of attributes implies that the learning method can learn about one concept, not about many concepts. In design, different concepts are characterised by the use of a different set of attributes to describe the example. For these reasons, we have developed an approach to learning multiple concepts that accepts varied attributes across the set of design examples, and uses the values of the attributes to further characterise a design concept.

Automatically generating design concepts involves assigning an example to a specified concept. If the example does not fit into an existing concept, a new concept is formed. This process is roughly similar to conceptual clustering (Fisher, 1987), but the attributes across design examples can vary and the result is a set of unconnected spaces rather than a hierarchy. A computer program called CONCEPTOR is used to implement the process of separating design examples into several spaces. In CONCEPTOR, descriptive attributes of a design concept are stored in a specific space in which examples possessing those attributes stay. Each space represents a particular design concept. After the examples are grouped into spaces, the second learning stage of CONCEPTOR is to derive default values and value ranges. Formation of default values and value ranges involves processes of using probabilistic generalisation through a survey of all examples in a concept space, whereby all possible values for an attribute are extracted.

A concept space is initialised as the set of attributes that describe the first example. When a new example is introduced, two steps involved are (1) measuring the fit of a new example to existing concepts, and (2) if the best fit is above a threshold, the example is assigned to the concept with the best fit, otherwise a new concept is formed based on the new example. Currently, the fit is calculated as the number of attributes that match. The mechanism used to measure the similarity of attributes is based on syntactic match, as this learning method presumes that design attributes are described using an accepted design vocabulary in a specific domain. The result, therefore, is dependent on how a design example is described symbolically.

Two processes are described to form defaults for numeric and nominal values. The default values for *numeric attributes* are determined by the weighted average shown in equation 1.

Equation 1 is a formula widely used in statistics. This equation has been used in engineering to produce the most possible value for a large dataset, or the default. The weight W_i of value A_i is the frequency in which value A_i occurs in all design instances. It represents the frequency of value A_i occurring

$$default = \frac{\sum_{i=1}^{n} W_i \cdot A_i}{\sum_{i=1}^{n} W_i} \qquad (1)$$

W_i weight of A_i, it is calculated as the frequency of A_i
A_i one value of a design variable

and its influence on the default value.

The default values for *nominal attributes* are determined using the frequencies in which certain nominal values occur in the design examples, the most frequent ones become the defaults. For example, assume in bridge design, the shape of the cross section of a concrete beam has three values in a group of design examples: rectangular, I and circular. The values and their frequencies are shown below.

Shape of cross section = rectangular	(0.83)
Shape of cross section = trapezoid	(0.10)
Shape of cross section = circular	(0.07)

The Shape of cross section = rectangular has the largest frequency (0.83), which denotes rectangular is the most frequent value for the nominal attribute Shape of cross section. Consequently, rectangular becomes the default value.

As shown in equation (2), the frequency of a specific nominal attribute-value pair, such as Shape of cross section = rectangular, is determined as the number of examples possessing this attribute-value pair divided by the number of examples possessing a specific attribute, which is Shape of cross section in the above case.

$$F(attr,value) = N(attr,value)/N(attr,-) \qquad (2)$$

where, F(attr,value) is the frequency for an attribute-value pair, N(attr,value) is the number of examples possessing the attribute-value pair, and N(attr,-) is the number of examples possessing the attribute.

The range of a numeric attribute comprises the subgaps within its maximum and minimum values. Subgaps are determined by considering the distribution of values of each numeric attribute and gaps with relative dense population of examples are taken as value ranges, whereas the range for a nominal attribute is the accumulation of all values that appear in the given

examples. The value range of each numeric attribute is dissected into three grades: low, medium, high (or small, medium, large).

As shown in Figure 3, examples are distributed according to their values of numeric attribute i, distance-1 and distance-2 are the largest and second largest distance between two adjacent examples. Therefore, the length between Minimum and Maximum is cut along distance-1 and distance-2, three subgaps are produced as gap1, gap2 and gap3 representing low, medium and high (or small, medium, large) grades.

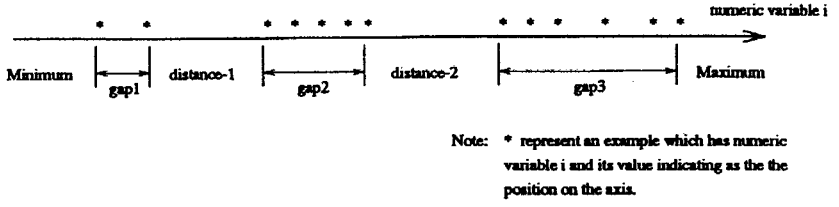

Figure 3. Three Gaps for Numeric Value Ranges

Concept Characterisation

In concept characterisation, the design concepts formed above are further defined by the empirical associations among the design attributes. The examples in a design concept are used to derive two types of associations: empirical formulas among numerically-valued attributes and design patterns among nominally-valued attributes.

(a) Learning empirical formulas

The purpose of establishing relationships between quantitative attributes is to provide a basis for assigning values to attributes, given an initial set of values for a subset of the attributes in a design concept. Learning empirical formulas includes two steps: 1) formulating potential equations; 2) computing the coefficients of each equation. The methodology for learning empirical formulas can be regarded as an adaptation of KEDS (Rao et al, 1991). This methodology is implemented as a computer program called EFD, Empirical Formula Derivation.

In EFD, a two-attribute linear template, $ax + by + c = 0$, is used to represent the proportional relationship between two continuous attributes. In this template, x and y are two continuous attributes in a design concept, and a, b, c are constants. Although we did not pursue it in this implementation, an extension to nonlinear models is possible when the linear models are insufficient to characterise the numerical associations in the space.

To comply with the dimensional homogeneity, EFD classifies continuous attributes by their units. Only attributes with the same unit are allowed to pair

to form an equation. The x and y in the template are single attributes of the same units. Although there may be certain combinations of attributes that can be added or multiplied together to give the same dimension, we assume these combinatorial attributes with second or higher order (the degree of order is the maximum number of attributes in the combinatorial attribute) have comparatively less influence on the accuracy of the equations, thus they are excluded.

Each candidate equation is considered separately, its response and predictor attributes depicts a two-dimensional space and the corresponding values in the dataset provide the points in the two-dimensional space. The coefficients of the equation determine the best estimate of the relationship between the attributes. EFD uses regression analysis to determine the coefficients in the two-attribute linear model. The standard procedure to estimate the best-fit regression line is called the method of least squares.

Not all formulas are valid approximations to the relationship among the attributes. EFD uses two criteria to determine the validity of a formula: MR and e. The majority ratio, MR, denotes the minimum fraction of instances in the dataset that must be covered by an equation within a $1 \pm e$ band; where e is the error bound. In the current implementation, MR = 0.75 and e = 0.5. All equations that cover at least MR percent of the design examples are valid empirical formulas. Other equations found using the regression analysis are assumed not to be representative enough to generalise and characterise the design space. However, those attributes that do not participate in empirical formulas using the linear model may be better characterized in formulas using other models, such as a polynomial model, logarithmic model, etc.

(b) Learning design patterns

In contrast to having continuous distributions and unit-related values of quantitative attributes, qualitative attributes appear to be discrete and incommensurable. The discrete nature of the values indicates that a qualitative attribute has a finite number of allowable options. For example, a qualitative attribute "the shape of cross section" in beam design normally has to be one of the following: rectangular, circular, I-shape, trapezoidal. A qualitative decision can therefore be formally identified as the selection of a plausible option for the qualitative attribute from its range. Usually a design problem involves several qualitative attributes, such as "the kind of material", "the type of foundation", etc. In the representation of a design solution, all qualitative attributes take one value, and we define a combination of qualitative attributes and their values as a feasible state indicating their compatibility of concurrence in design. Particularly, those frequently used feasible states are interesting, because they demonstrate a strong association, or trend, among the qualitative values in those feasible states. We name the strong associations in the feasible states *design patterns*, to emphasize that they

represent a kind of prevalent expertise on relevant qualitative decisions-making processes.

We propose a probabilistic representation for design patterns as shown in Figure 4, which is diagrammatically similar to other dependency graphs, such as semantic networks (Woods, 1975), and constraint networks (Montanari, 1974). The nodes are qualitative attributes. Each node has a number of possible values as its value range. A link between values of two attributes denotes the compatibility of concurrence in design, and the attached frequency indicates the degree to which the linked qualitative attribute-value pairs occur together in individual designs. Associations between two attributes are built up through the links of their values. The thickness of the line in the figure indicates the magnitude of concurrence frequency. Values without links indicate the inhibitory effect, which can be interpreted as the infeasibility of concurrence.

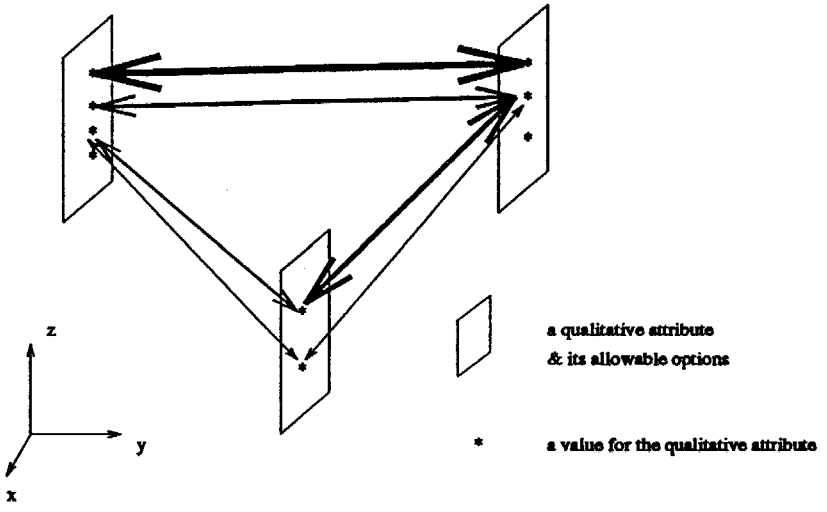

Figure 4. A Three Dimensional Illustration of a Pattern Structure

DPF, the Design Pattern Formulator, is an implementation of a methodology for generating design patterns using the qualitative attributes and their values from the examples in a given design concept. The learning task separates into two subtasks: learning the qualitative attribute for structuring a pattern topology, and identifying the topology itself - specifically the missing links. Although the two subtasks are not independent, it is convenient for us to present the learning process in two separate phases: structure learning to set up the basis for representing design patterns, and link generation in order

to finally assemble design patterns.

A group of qualitative attributes is a by-product of learning descriptive attributes from design concepts. In a design concept, we are able to distinguish qualitative attributes because they do not have units. We postulate m out of n attributes are qualitative. Then, examples are used again to gain the distribution of the discrete values of m attributes. In every example, the collection of qualitative attributes, along with their discrete values, provide a feasible state within which each value pair has a link with one unit weight.

Thereafter, DPF allocates each feasible state over the nodes of the proposed design patterns. If a value of a qualitative attribute exists in its allocated node, its surrounding links are used to augment the weight distribution by adding one unit to the existing links; otherwise, DPF treats the value as new and inserts it to the value range of this node. The links around the new value are established in same ways as they are in the feasible state. As such, all feasible states are processed to determine the weight distribution of links, and the weight distribution is converted into frequency distribution by averaging weights with the total number of feasible states.

We compute the frequency distribution $P(i=1, n)$, where n is the number of feasible states. The distribution is nonnegative and attains the value 0 if there is no link at all among the qualitative attributes being considered. A design pattern is determined in the following manner. The links with frequencies exceeding a predetermined threshold are considered as strong links. Values of qualitative attributes associated with strong links are extracted as design patterns.

Overall Learning Model

The overall model for learning design concepts is illustrated in Figure 5. In this model, design examples are first separated into concept spaces which are described by a set of attributes, their default values, and their value ranges. This is followed by the application of two methods for learning associations among the attributes in each space: one method for learning numerical associations and a second for learning associations among nominal values of attributes.

The model was tested using 72 bridge designs collected from a database at the Roads and Traffic Authority (RTA) in Sydney. The 72 bridge design examples generate 4 spaces as shown in Figure 6. Each space stands for a specific design concept, where the design concept is, at this point in the process, described by a set of attributes as well as their default values and value ranges. The examples that fit in each space are recorded as part of the space for the application of the methods that learn associations among the attributes. Although the learning program does not assign indicative names to the spaces, it is not difficult to recognise the types of bridge designs the

concepts represent. For example, designs in space 1 are all simply supported girder bridges, and the attributes, their defaults and value ranges for space 1 are drawn from these bridge examples. Similarly, it is recognised that concept 2 represents plank bridge design; concept 3 for continuous bridge design and concept 4 cable-stayed bridge design.

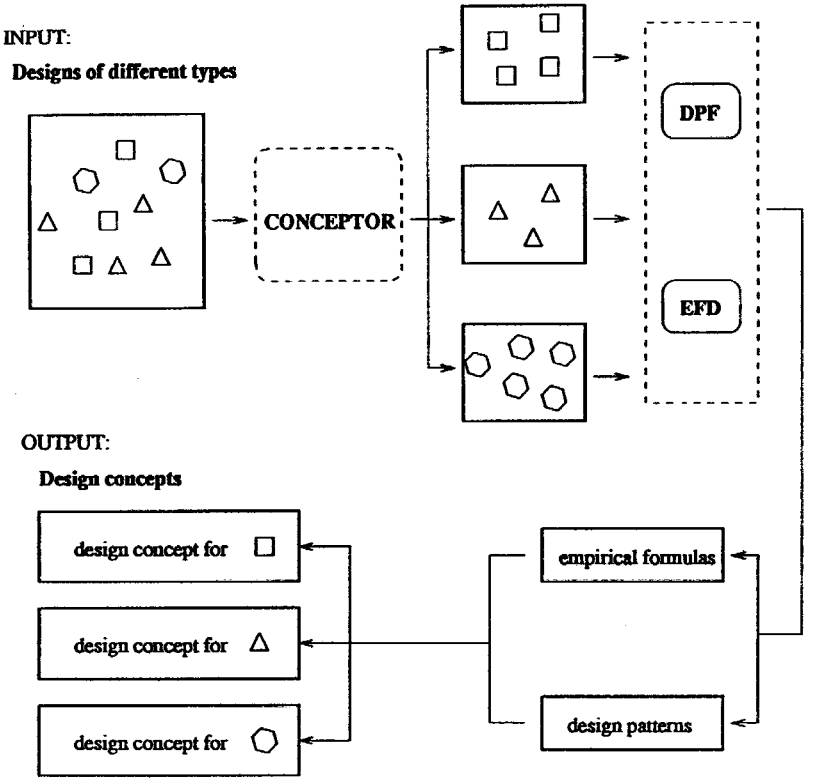

Figure 5. An Overall Model of the Learning System

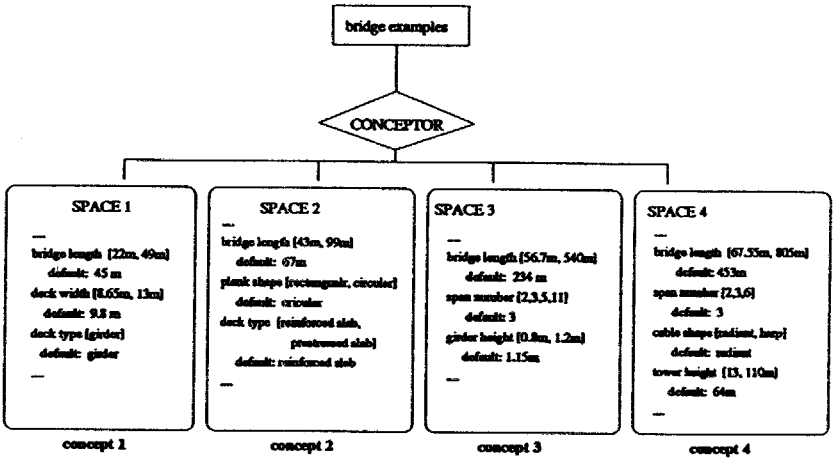

Figure 6. 4 Concepts from CONCEPTOR

The examples in each concept space provide the data to determine the relationships between the attributes. Figure 7 shows the empirical network produced from the examples in concept 3.

Using Design Concepts in a New Design Project

The empirical networks produced by the learning system represent a generalisation of the examples from previous design projects. The generalised knowledge provides assistance in preliminary design by associating a set of attributes with a concept and by providing initial values for the attributes once a concept has been chosen. In the preliminary stages of design, concept selection is assisted by making a set of concepts explicit and assigning initial values for attributes is assisted by the representation of default values and empirical relationships among the attributes and their values.

For illustration, assume the following design attributes are given with values as part of the design requirements.

Design requirements:

width of girder (GW)	3.0 m
waterway clearance (CL)	7.8 m
condition of riverbed	good

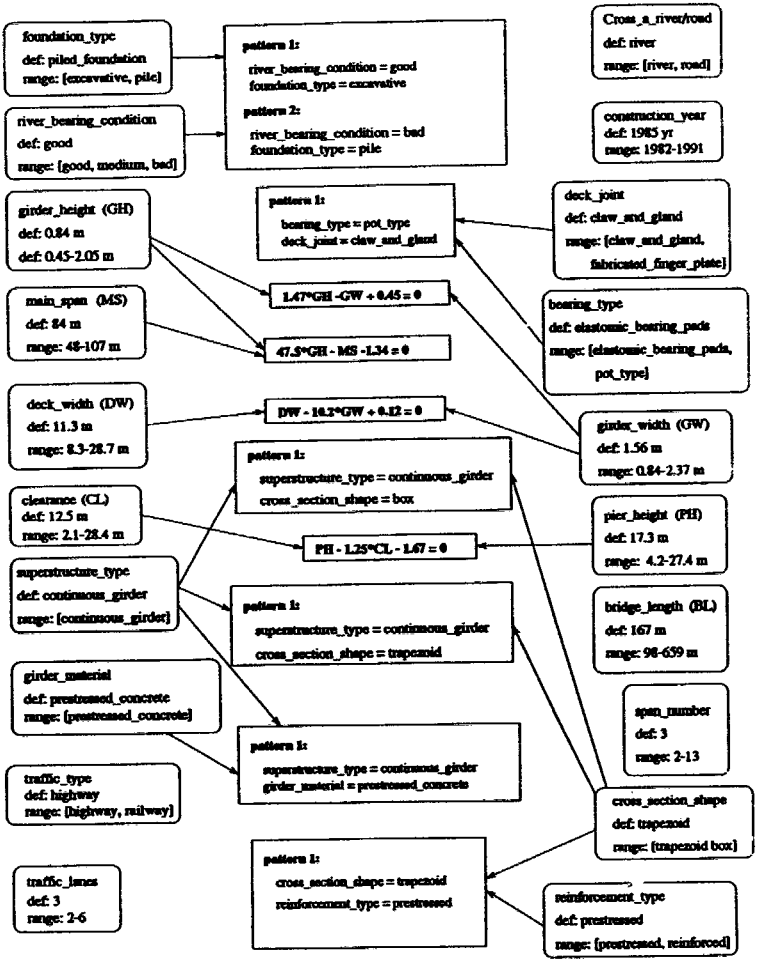

Figure 7. A Design Concept for Continuous Bridge Design

Given these attributes, the empirical network for continuous bridges is selected. By using the empirical network illustrated in Figure 7, following design decisions may be made.

height of girder (GH) ----> (GW - 0.45)/1.47 = 1.94 m
main span (MS) ----> 47.5*GH - 1.34 = 97.2 m
deck_width (DW) ----> 10.2*GW -0.12 = 21 m
height of pier (PH) ----> 1.25*CL + 1.67 = 11.4 m
type of foundation ----> no-pile foundation

Those attributes whose values cannot be directly derived from using empirical formulas or design patterns may be given default values. As such, a conceptual design can be produced.

CONCLUSION

The presented model for automatically learning design concepts from project data is an aggregation of machine learning and numerical analysis techniques. Each of the learning methods are, in isolation, simple approaches to generalising data. The major contribution of the model lies in the combined use of the methods where the input as design examples and the representation of the learned design concepts establish the relevance of the learning approach. The focus in developing this learning model is to use project data to learn design concepts to be part of a knowledge base for preliminary design.

References

Arciszewski, T, Mustafa, M and Ziarko, W (1987), A methodology of design knowledge acquisition for use in learning expert systems. In International Journal of *Man-Machine Studies*, 27, pp. 23-32.

Fisher, D (1987), Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, Vol 2, pp 139-172.

Montanari, U (1974), Networks of constraints, fundamental properties and applications to picture processing. *Information Science* Vol 7 pp 95-132.

Rao, R B, Lu, S, C-Y and Stepp, R E (1991), Knowledge-based equation discovery in engineering domains in machine learning. In L.A. Birnbaum and G.C. Collins (eds) *Proceedings of the Eighth International Workshop in Machine Learning (ML91)*, CA, USA.

Reich, Y (1991), *Building and Improving Design Systems: A Machine Learning Approach*. PhD thesis, Department of Civil Engineering, Carnegie Mellon University, Pittsburgh, PA.

Winston, P H (1975), Learning structural descriptions from examples. In Winston, P. (ed) T*he psychology of computer vision*, McGraw-Hill.

Woods, W A (1975), What's in a link? Foundations for semantic networks, in *Representation and Understanding*, D. Bobrow and A. Collins (eds), Academic Press pp 35-82.

c. 1993, Management of Information Technology for Construction, K. Mathur et al (Eds), World Scientific Publishing Co., Singapore.