Design Knowledge Acquisition: Task Analysis and a Partial Implementation

Yoram Reich
Department of Civil Engineering
Carnegie Mellon University
Pittsburgh, PA 15213

Knowledge Acquisition: An International Journal of Knowledge Acquisition for Knowledge-Based Systems, 3(3):237-254, 1991

Abstract: Design is conceptualized as an ill-structured process that requires diverse knowledge that is hard to acquire. Systematic analysis of design and the knowledge requirements is has in general and in the context of bridge design shows that the knowledge needed can be semi-automatically acquired by using machine learning techniques. Although there are limitations to the approach, preliminary results in the bridge design domain are promising and can potentially transfer to other design domains.

1 INTRODUCTION

The process of knowledge acquisition for any type of expert system is time-consuming and tedious. This effort increases when dealing with design domains that are ill-structured by their nature. One approach that promises to alleviate the difficulty of the knowledge acquisition process is the introduction of learning into system development and maintenance stages (Reich and Fenves, 1989b; Shalin et al., 1988; Witten and MacDonald, 1988). For tasks that are relatively well understood and specified, the above promise can be realized by using a single learning method. For example, one can use the learning program ID3 (Quinlan, 1986) to create decision trees for classifying symptoms into malady classes. Single learning techniques have also been used for very restricted aspects of design (e.g., knowledge refinement (Mitchell et al., 1985) or the acquisition of simple design rules (Arciszewski et al., 1987)); however, no overall approach for the acquisition of design knowledge has been proposed. The goal of this research is to develop a framework for assisting in identifying learning methods that can automate or assist in design knowledge acquisition.

Understanding the issues involved in design knowledge acquisition requires a systematic analysis of the task, deriving the knowledge it uses, the knowledge dynamics and application (Gaines, 1987). The requirements formulated by this analysis are then mapped into specific knowledge acquisition methods which either exist or should be developed. These methods can be viewed as *generic tasks* (Chandrasekaran, 1986) for solving learning problems (Reich and Fenves, 1989a). An abstract account of a similar methodology based on system engineering is described by Rook and Croghan, 1988.

Two observations from the analysis described later are important to note.

1. *The nature of design process.* Design is conceptualized as an iterative process composed of five sub-tasks executed in sequence: problem analysis, synthesis, analysis, redesign, and evaluation. A framework for design knowledge acquisition should address each of these tasks. Clearly, this view is a simplification of real design; nevertheless, it still constitutes an important hard problem.

2. *The need to integrate diverse knowledge acquisition techniques.* The distinct nature of the five design sub-tasks entails the use of different techniques suitable for each. We identify machine learning techniques that can support the acquisition of knowledge for these tasks, and illustrate their integration. The three types of expertise transfer discussed by Gaines, 1987 are used in our approach: synthesis, redesign, and potentially analysis knowledge are acquired *by example*; redesign and potentially evaluation knowledge are acquired *by evaluation*; and redesign knowledge is also acquired as *an apprentice*.

The paper is organized as follows. The next section analyzes the design process to derive its knowledge requirements[1]. Section 3 identifies general learning approaches that can acquire design knowledge based on the above requirements and an analysis of the application domain. The identification of specific learning programs is entirely directed by the analysis of the application domain. Section 4 demonstrates the approach in the domain of bridge design by describing a system under development called Bridger. We concentrate on capturing synthesis and redesign knowledge and briefly illustrate the *optional* capabilities of acquiring analysis and evaluation knowledge. The problem analysis is the least understood task. We do not deal with the acquisition of knowledge for this task in this paper. Section 5 concludes with a summary and future work.

2 THE DESIGN TASK

In this study, design is conceptualized as a five-task process. Figure 1 illustrates these tasks with the knowledge each task requires and the corresponding name in AI terminology. We first describe these tasks and then presents design as a more complex process composed of several phases. We then limit the scope of the approach and provide an analysis of our target domain: the design of cable-stayed bridges.

2.1 Design tasks

Problem analysis. A design starts with a given problem statement containing some abstract requirements to be achieved. The first design task is a *problem analysis* of this statement, trying to understand its nature and to define it more precisely. This task involves the use of common-sense as well as domain

[1]Shalin et al., 1988 have provided a formal analysis of several tasks and their relevance to existing machine learning techniques. Their analysis of design is very simple, it does not allow a true appreciation of the task complexity and it prevents a detailed analysis of the prospects of using machine learning for knowledge acquisition. Consequently, they were not able to map any learning program onto design knowledge acquisition.

specific knowledge. Problem analysis might require an initial study and possible negotiation with the client who provided the problem to better understand his intentions. Ultimately, this "understanding" results in a detailed set of specifications and a definition of a design search space. This space is also an *intentional* description of all the possible designs.

Synthesis. The second design task is the *synthesis* of candidate designs that satisfy the specifications. Synthesis uses search techniques for exploring the design space. Control knowledge from diverse sources is used to guide this exploration. Synthesis can be viewed as a transformation from intentional to extensional description of designs, expressed by the pair *(specifications, design-description)*.

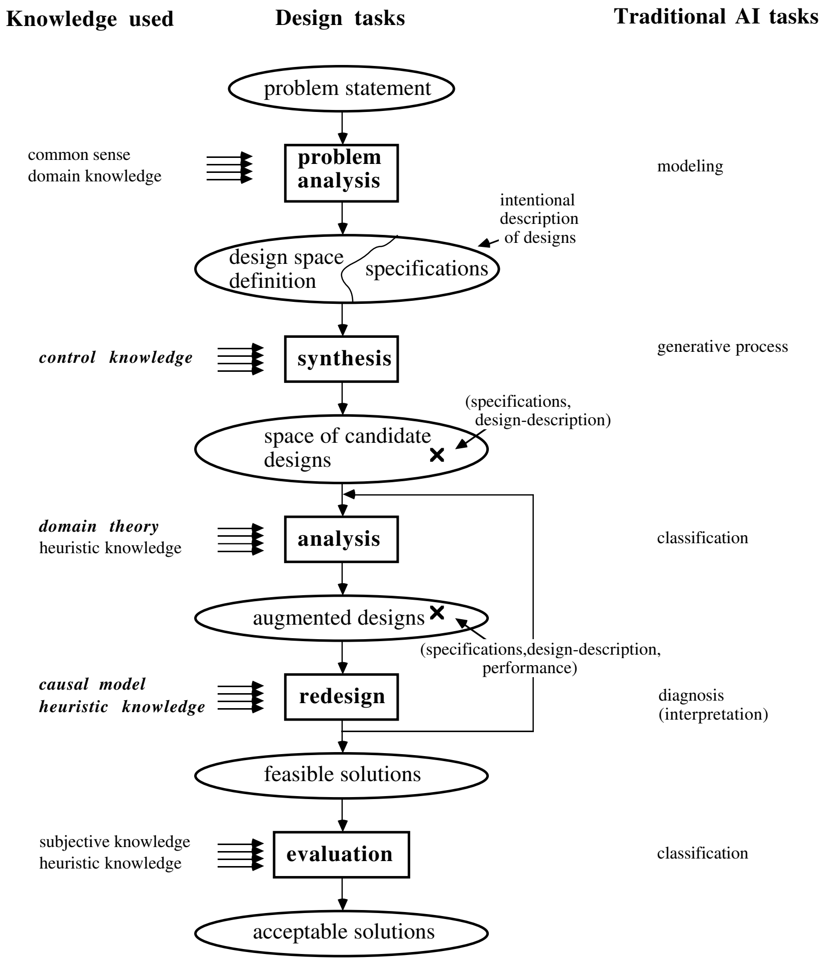

Figure 1: The structure of design

Analysis and redesign. Since not all the problem constraints are explicitly stated in the problem statement or detected by the problem analysis, and since the design space might be incomplete or contradictory, many candidate designs might be inadequate. Designs might violate hard constraints or be inefficient in using material, space, etc. Candidate designs are analyzed in the third design task – the *analysis*. Analyses vary in the way they model the artifact. Approximate techniques use coarse models but can still rely on theory (e.g., beam vs. finite-element analysis of a bridge deck). The analysis or performance measures calculated are augmented to each design description – creating the tuple *(specifications, design-description, performance)*.

Next, *redesign* resolves any inadequacies in the artifact. Redesign may use heuristic knowledge as well as causal models of the domain. Both suggest modifications and supply preferences for the best possible design modifications. Analysis and redesign may iterate until the design complies with all the requirements and satisfies all the constraints, henceforth called a feasible design.

Evaluation. Feasible designs are evaluated in the fifth design task – the *evaluation*. In real world problems there will be many feasible solutions that need to be ranked. Ranking designs can be done in several ways. One way of reducing the set of feasible designs is to only keep designs that are Pareto-optimal with respect to some criteria, for example, designs that have the least weight and are inexpensive to manufacture. Most evaluations, however, rely on heuristics or subjective criteria such as esthetics or simplicity. The highest ranked designs are the product of the overall design process.

Bridger, the system described later, only addresses the acquisition of knowledge that appears in italicized font in Figure 1.

2.2 Design phases

Most artifacts may be represented at several levels of detail, e.g., main components, sub-components, and individual parts. Such artifacts cannot be designed in one phase. Rather, the design process follows the different levels of abstraction of the artifact, leading to a top-down approach composed of several phases: the *conceptual*, the *preliminary*, and the *detailed* design phases. The five design sub-tasks re-occur in each of the phases, but with a different nature. Problem analysis becomes shorter and produces better defined specifications. Synthesis shifts from the overall configuration of components to the generation of parts within these components. Analysis changes from approximate and shallow into exact and detailed. Redesign and evaluation focus on the concrete parts that were recently generated and evaluated. Due to the hierarchical nature of this approach, overall backtracking is almost eliminated. The structure of the three phases is similar to one another, therefore we assume the existence of a single phase in the remaining of this paper.

2.3 Scope of design

Our view of design is similar to that of General Design Theory (Yoshikawa, 1981). In this approach, specifications and artifact descriptions are represented as sets in topological spaces. Design is a mapping from the specification topology to the artifact topology. The mapping can be based on heuristic associations or derived in stages by knowledge of the domain. The missing aspect in General Design Theory is that it does not show how such mapping can be acquired (Reich, 1990b).

Heuristic mapping is amenable to preliminary or conceptual design, where major design decisions are made without good quantitative measure of their tradeoffs. In these phases, there is no clear way of structuring the domain to allow for a direct mapping between the goals of the design to the functionality of available components; rather, many artifact properties contribute to achieve the design objectives. In particular, this view is appropriate to our target domain which is cable-stayed bridge design (Reich, 1990b; Reich and Fenves, 1992).

This approach differs with common approaches to VLSI design where top-down refinement is used to recursively decompose goals into subgoals and assign components that match the subgoals with their functionality (Steinberg, 1987).

In our design problem we make an assumption that artifacts are represented by lists of property-value pairs. This restricts the scope of design to domains where the topology of artifacts is pre-determined[2].

2.4 Cable-stayed bridges domain

The nature of the domain specific knowledge that needs to be acquired substantially influences the choice of learning methods. Figure 2b shows the types of knowledge used in the five tasks of cable-stayed bridge design. There is no theory of synthesis in our domain. Moreover, synthesis knowledge is virtually non existent for modern bridge design. Books on the subject (Podolny and Scalzi, 1986; Troitsky, 1988) provide a base-line for describing designs, and analyzing them – essentially the product of a simplified problem analysis task. Practically, catalogues of existing designs are the only source of synthesis knowledge. This entails the construction of knowledge from scratch, leaving machine learning techniques as the primary means for knowledge acquisition beside the tedious process of knowledge engineering with an expert designer.

Analysis of bridges is a well defined procedure in the form of a *finite-element analysis*. It can be easily coded completely. Approximate analyses techniques are also available for manual calculations. Redesign knowledge partially exists as a weak causal model that given some constraint violations, can point to possible design modifications for correcting the problems. In addition, design case studies have been compiled in the past into an unstructured collection of preferences that can be used to resolve conflicts in the causal model. The causal model and the heuristics provide enough substance to build a relatively competent redesign module.

A general statement about design as well as other domains is that knowledge is acquired incrementally by experts. In reality, experts need to continuously accommodate new experiences and technologies in their design domain. Similarly, learning techniques for design knowledge acquisition should be incremental.

3 MAPPING LEARNING ONTO DESIGN

The five design tasks are very different by their nature, utilizing knowledge of different forms that is used by different procedures. This observation is important for identifying the ways in which knowledge for these tasks can be acquired for a specific domain. First, the tasks are mapped into generic artificial intelligence tasks. This can further assist in identifying appropriate machine learning techniques for supporting knowledge acquisition. Next, a specific learning program is assigned to each task based on the characteristics of the application domain. Figure 2c summarizes the mapping between design tasks onto machine learning tasks for the domain of cable-stayed bridge design.

3.1 Problem analysis

Problem analysis is a process of modeling using diverse and unformalized knowledge. Such a process in general is not captured by any existing artificial intelligence paradigm. We assume that this process is performed manually by a knowledge engineer or a domain expert. This process not only provides a

[2]This restriction can be relaxed based on a recent research (Thompson and Langley, 1989).

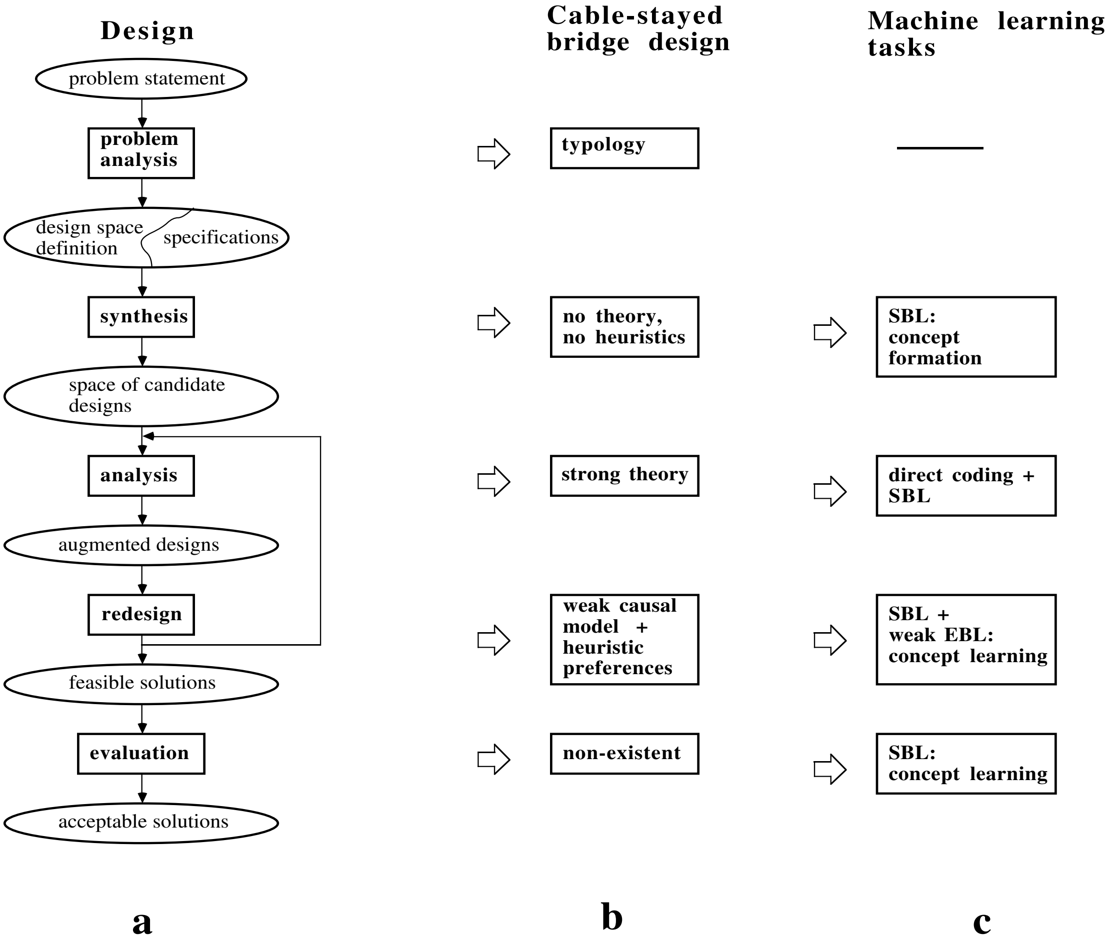

Figure 2: Mapping design tasks onto the domain of cable-stayed bridge design and onto machine learning tasks

language for describing artifacts and processes in a particular domain, but more importantly, it governs the specific machine learning tools chosen for each learning task identified.

3.2 Synthesis and concept formation

Since bridge synthesis theory does not exist, *similarity-based learning (SBL)* is a candidate methodology for the learning task[3]. Furthermore, statistical learning techniques can be used since many examples of bridges exist (approximately 100) or can be generated and analyzed. Within these limits, the best method for learning knowledge for a generative process is concept formation. The need for concept formation, rather than concept learning is argued next[4].

[3]In contrast, in VLSI, using decomposition and mapping primitive functions into components (Steinberg, 1987) is a powerful mechanism that with the addition of simulations forms a strong domain theory. This theory allows conceptualizing synthesis knowledge acquisition as knowledge refinement and permits the use of *explanation-based learning (EBL)* techniques (Mitchell et al., 1985).

[4]An elaborate discussion on this issue appears in (Reich and Fenves, 1991).

Suppose there was only one specification governing the design of an imaginary artifact and suppose that there were many possible designs for each specification. The problem of acquiring knowledge for this domain can be transformed into a concept learning task where the possible classes are the possible values of the specification and the examples of a certain class are the candidate designs satisfying the corresponding specification value. One could generate training examples by constructing arbitrary designs and evaluating them to identify which specification they satisfy. This process is illustrated in Figure 3a. In a design scenario, a new specification would be classified into one of the classes and candidates would be generated from the concept description of that class (see Figure 3b). The above simplification–designing for a single specification– is never appropriate for real artifacts that are specified by many specifications which are sometimes redundant or conflicting. Therefore this learning method is inadequate.

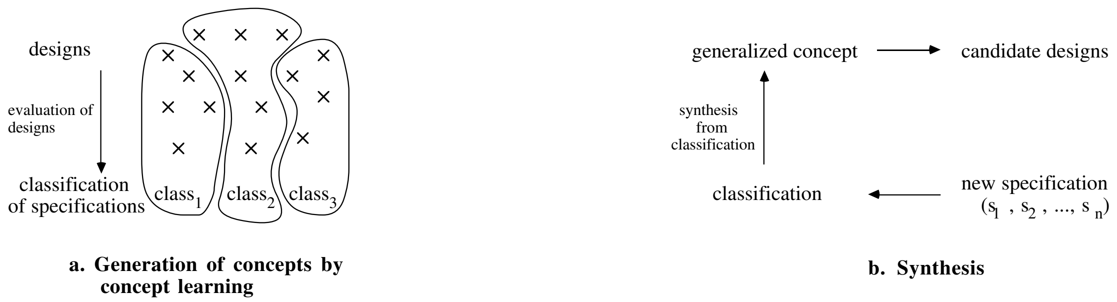

Figure 3: Concept learning and synthesis

An alternative method is the use of each property of the design as a separate independent design decision that can be carried out by observing the specifications. One way to implement it is to construct a decision tree for each such property (denoted by d_i) where decision nodes represent different specification properties. A design scenario in this approach consists of using each tree to select a single design description property based on the specifications. This approach is illustrated in Figure 4. The main problem of this approach is that information implicit in design examples by virtue of describing a single artifact is lost when each artifact descriptor is selected independently. In many cases this indispensable information represents the product of a complex constraint satisfaction process.

The reason underlying the inadequacy of concept learning techniques is the nature of synthesis. Synthesis is a *many-to-many* mapping (e.g., specifications to design description properties). This mapping cannot be divided into several *many-to-one* mappings to be handled by concept learning since design description properties are not mutually independent. Furthermore, synthesis cannot be modeled by a mapping from specifications to classes of designs since such classes are not known a priori, except for some types of routine design.

A method that can capture a many-to-many mapping is concept formation. The approach to synthesis knowledge acquisition by hierarchical, possibly overlapping, concept formation is described in Figure 5. The main idea is that specification and solutions are correlated together and forming classes with their intentional description allows for an appropriate generation of solutions given similar specifications. Ideally, one would form concepts from the final acceptable designs, reducing the need to perform evaluation and redesign in the future. Since the concept formation process is performing an inductive

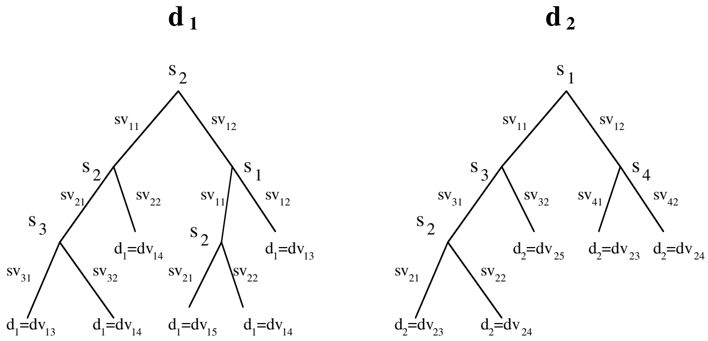

Figure 4: Synthesis with multiple decision trees

leap (overgeneralization) over the data and cannot guarantee the generation of perfect designs for new specifications, evaluation and redesign tasks cannot be eliminated.

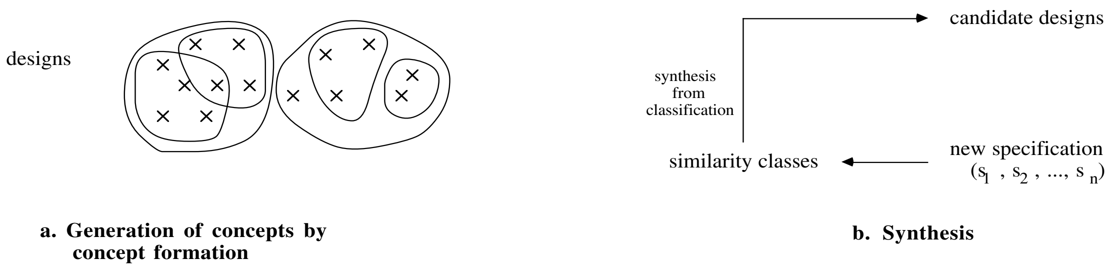

Figure 5: Concept formation and synthesis

There is another dimension to synthesis that constrains the selection of the concept formation method. Namely, the output of synthesis is rarely one alternative; usually, several candidates are produced. The concept formation method used should use knowledge representation that supports such generation.

3.3 Analysis and concept learning

A single analysis is a process of observing an artifact and stating whether it conforms to some criterion. A complete analysis consists of many such checks and consequently can be conceptualized as a set of classification processes[5]. Exact analysis is relatively easy to capture by direct coding. The ability to perform heuristic analysis can be acquired by generalizing over exact, tedious analyses. Each analysis can serve as a training example where the concept class is the analysis result. Given enough experience, a new artifact can be heuristically classified.

Heuristic analysis never replaces exact analysis; nevertheless, it is useful in many of the iterations of the design process when coarse measures are sufficient to make decisions. In the last iterations, exact analysis is used to verify the final design. Figure 6 summarizes analysis knowledge acquisition.

[5]Analysis is also a many-to-many function (e.g., design description to performance properties); but it can be modeled by several many-to-one functions.

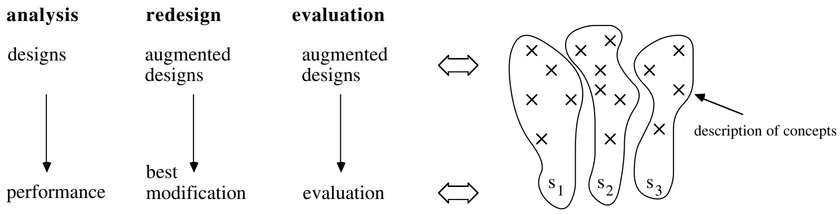

Figure 6: Concept learning for analysis, redesign, and evaluation

3.4 Redesign and concept learning

Redesign is a process of identifying the causes (e.g., design decisions or artifact descriptors) for a specific behavior of the design and determining the changes that these causes should undergo to correct the behavior. This is a diagnosis task augmented by the ability to alter the causes for correcting the inappropriate behavior.

Heuristic knowledge for diagnosis can be captured by a concept learning process (see Figure 6). For satisfying our purpose, examples should consist of a list of violated constraints and the concept classes should be the possible modifications. In some domains (e.g., our target domain), causal models can be acquired as well. This can be done by performing a sensitivity analysis. In general, this process will be costly and should be avoided. It is hoped that coding heuristic causal models with the help of an appropriate knowledge elicitation tool is still a reasonable task compared to the acquisition of general heuristic knowledge from an expert.

Note that acquiring diagnosis knowledge by concept learning assumes that there are only small interactions between candidate modifications; otherwise, concept formation is more suitable for gradually assimilating the interactions as it does in capturing synthesis knowledge.

3.5 Evaluation and concept learning

Evaluation is a heuristic and subjective classification of an artifact. Acquiring such knowledge can be done in an apprentice mode or by extracting implicit judgment from existing designs. In our approach, part of the evaluation knowledge is captured as synthesis knowledge. We do not make any further attempt to acquire subjective evaluation, although such acquisition can be performed by a concept learning task in a similar way to the acquisition of heuristic analysis knowledge (see Figure 6).

4 BRIDGER AS A MODEL OF LEARNING FOR DESIGN

Bridger is a learning system for knowledge acquisition and performance improvement that is currently under development. In its intended domain – the design of cable-stayed bridges – it is expected to perform a detailed preliminary design given a set of specifications. The problem analysis task is compiled into a language for describing specifications, constraints, design examples, and analysis methods. Bridger is not fully integrated: the synthesis and evaluation modules are implemented and tested and the redesign module is in its initial stage of implementation. In the next section the name Bridger refers to the synthesis module only.

4.1 Supporting synthesis by concept formation

Bridger is built on the foundations of the learning program COBWEB (Fisher, 1987). The decision to choose a COBWEB-like mechanism as the underlying concept formation process for learning synthesis knowledge is the product of the problem analysis task. This task determined that for the design of bridges, specifications, artifact descriptions, and even evaluations can be represented by a list of property-value pairs.

Bridger extends COBWEB along several dimensions. It can handle entities described by a combination of nominal and continuous property types; it has a correcting-hierarchy module; it has a richer set of learning operators; it can forget undesired knowledge; it can perform directed experimentation that increases the utility of its knowledge; and it can also perform a simple form of constructive induction. Such extensions can improve the ability to master design rather than simple diagnostic domains[6].

The learning approach. Bridger uses an incremental learning scheme for the creation of hierarchical classification trees. Bridger accepts a stream of designs described by a list of property-value pairs and builds a classification hierarchy in the following way. When a new design is introduced, Bridger tries to accommodate it into the existing hierarchy starting from its root. Bridger can perform one of several operators to accommodate the new design into the hierarchy and/or modify the hierarchy. If the design has been accommodated into an existing sub-class (i.e., using one of the learning operators), the process recurses with this class as the root of a new sub-tree. Bridger uses a category utility function (Gluck and Corter, 1985) to select between its available operators. The best operator is the one that results in a new classification that maximizes the category utility function.

Synthesis scenario. Bridger synthesizes using a mechanism similar to learning. Bridger sorts the new specification through the tree to find the best host node for the new specification. Synthesis progresses by assigning the new artifact characteristic property-value pairs describing the nodes traversed (see example in the next section). Characteristics of a node are represented by property values that describe most of the designs in the node and almost none in the other nodes. This strategy, which is different than the original method employed by COBWEB, can be viewed as a least-commitment, top-down refinement strategy (Reich, 1990b).

We now illustrate how Bridger uses a hierarchy of bridges to synthesize a new bridge in the domain of Pittsburgh bridges[7]. In this domain, a bridge is described with 12 properties of which seven are specification properties (three continuous and four nominal): the river and location of the bridge, the period it was constructed, the purpose of the bridge, the number of lanes and length of the bridge, and whether a vertical clearance requirement was enforced in the design. Five design description properties are provided for each design: the material used, the span of the bridge, the type of the bridge, the location of the road with respect to the bridge, and the relative size of the span to the channel width. Figure 7 illustrates some of these properties.

Figure 8 shows the hierarchy generated by Bridger from 60 examples. The nodes at the hierarchy are described using their characteristics only. The reference symbol and the number of examples below each node (in parentheses) are also provided. Bridger is required to synthesize a *highway* bridge on one of the rivers where *vertical clearance governs*. Bridger starts designing by classifying the design with

[6]For additional details on Bridger, including statistical performance tests on other domains, see (Reich, 1989; Reich, 1990b; Reich, 1990a).

[7]This domain is different that our target domain, but is similar in nature.

Figure 7: A partial description of a bridge

the hierarchy. The current state of Bridger's knowledge is knowing how to design *through* bridges. This property-value is assigned to the new design. Since vertical clearance governs the design, the bridge description should be refined using class G130. The new design will be of a *simple-truss* configuration made of *steel*. Since G130 consists mainly of bridges with 2 lanes, this specification property refines the bridge specification of the new design. Class G138 is chosen next since it represents *highway* bridges. The overall length of the bridge (1088 ft) also refines the bridge specifications. At this stage the design terminates since all the specifications have been met. The resulting bridge is partially specified since the main span and the relative length of the bridge are not determined. This is expected since the specifications are abstract.

Bridger can maintain the abstract design as the solution or use other refinement strategies to complete the synthesis (Reich, 1990b). The latter is necessary for the execution of the exact analysis described in the next section. Furthermore, Bridger can generate a ranked set of alternatives from the subtree whose root is G138 – a necessary requirement of synthesis.

4.2 Supporting analysis by direct coding and concept learning

Analysis of bridges is performed by a finite-element procedure. This analysis can be viewed as a perfect domain theory for this task. The results of the analysis are compared with design codes, that are compiled, mandatory limits on behavioral functions, for identifying the critical parts of the design. A part of analysis which identifies the critical loads on a bridge (i.e., calculates influence lines) is seen in Figure 9. We do not make any attempt to compile analysis results into heuristic analysis because we expect that the synthesis module will absorb some of these heuristics when successful, acceptable designs are learned by Bridger.

Although we do not intend to use heuristic analysis knowledge, we performed an experiment to check the potential of this approach. Five hundred designs of cable-stayed bridges were randomly generated and analyzed by the finite-element procedure. Examples were generated by concatenating the design description with the analysis results of each design. In the experiment, examples were gradually used to generate a classification hierarchy with the same technique that learns synthesis knowledge. The hierarchy was used to heuristically analyze the remaining examples based only on their design description part. Each heuristic analysis was checked against the actual finite-element results for the particular bridge. The following *average* results were obtained. The heuristic results continuously improve when additional examples are learned. Improvements range from a match of 1000% after learning 100 examples to a match of 30% after learning 400 examples. Four out of twelve of the results of the heuristic analysis converge to within 8% from the exact values after learning 400 examples. One

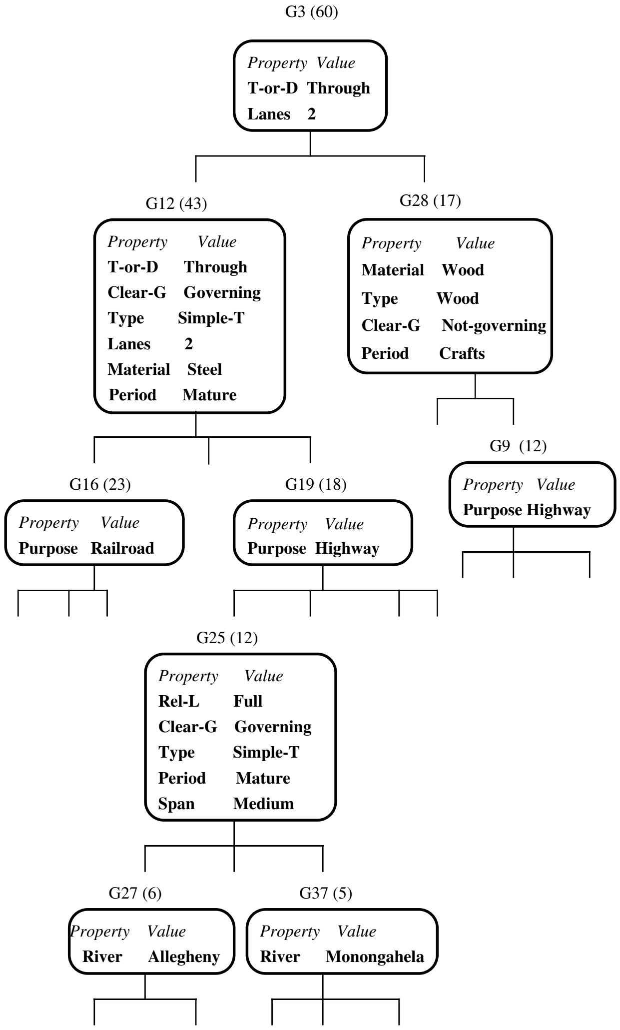

Figure 8: Design concepts hierarchy after learning 60 examples

result remains far from the exact analysis (200% mean relative error), but this is caused by only a small number of major errors. We view this experiment as supporting the claim that heuristic analysis can be learned as well.

Bridger's ability to augment synthesis with analysis knowledge is also the reason for not trying to learn evaluation knowledge separately from synthesis. Evaluation is used as a means to obtain the user feedback on the designs produced by the system. The feedback is then used to enhance synthesis. An example of an aesthetic evaluation that can be augmented to synthesis is: avoid generating bridges that have two pylons or more, with pylon height more than one-forth of the main span.

4.3 Supporting redesign by concept learning

A simple causal model containing redesign knowledge in the form of qualitative relations is shown in Figure 10. A plus sign on an arrow denotes a positive influence of the tail on the head whereas

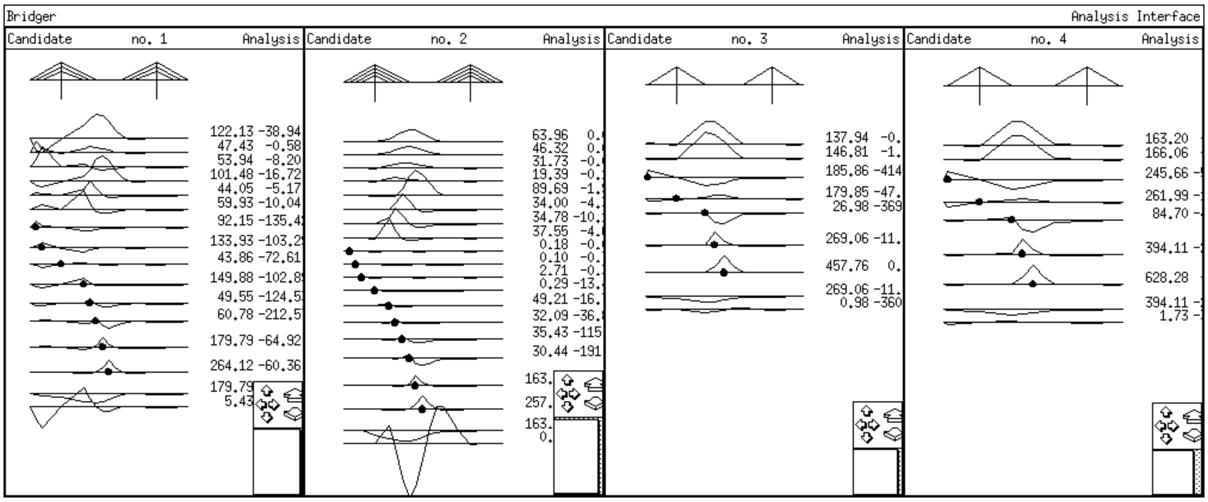

Figure 9: Theory-driven evaluation

a negative sign denotes the opposite. Such a graph can be generated systematically for our domain, although it might be incomplete. Note that the graph has two problems: (1) conflicting paths between *cable-spacing* and *bridge-stiffness* and (2) conflicting paths between *pylon-height* and *bridge-stiffness*. Such conflicts can be resolved by performing a sensitivity analysis for the modification in question or by using experience (e.g., increasing pylon-height leads to an increase of bridge-stiffness). Sensitivity analysis is an expensive procedure compared to the use of experience. Consequently, we try to acquire and organize experience for the redesign task of Bridger.

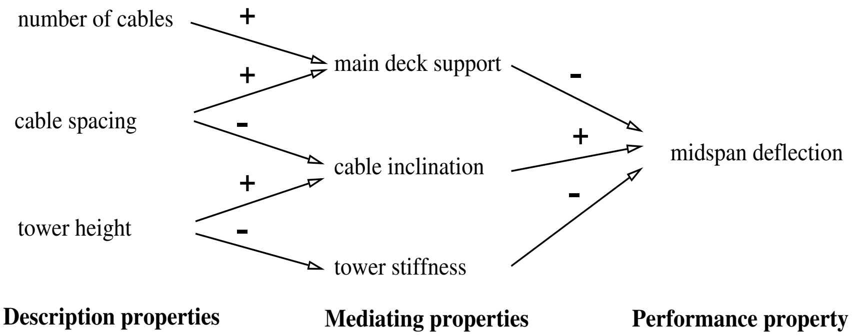

Figure 10: A partial causal model

Two of the main tasks of learning are: (1) the acquisition of heuristics that support the identification of the *best* modification which is usually indeterminate from the causal model, and (2) the refinement of the causal model. At first, such heuristics can be found in technical references. Learning can be used to structure them and resolve conflicts that exists between heuristics from different references. A *case-based reasoning* scheme suitable for this task has been proposed by Clark, 1988.

Some of the assumptions made by Clark are not appropriate for our domain, particularly the assumption that conflicts that are once resolved in a certain way, will be similarly resolved in the future. In our domain, heuristics abstract many details to become only approximations; thus, resolving conflicts depends on the context of the specific bridge design scenario. Consequently, learning must be used continuously to further refine the heuristics.

Another system that does not make the above assumption but has a similar case-based representation and reasoning is the Protos system (Bareiss, 1989; Porter et al., 1990). Protos is a system for the acquisition of concepts and their use in future classifications. Protos makes use of an *exemplar-based* representation of categories. In classification, the features of a new case remind Protos, in a process of heuristic matching, of the most similar exemplar in the category structure. The category of the new case is the category of this similar exemplar.

In the cable-stayed bridge domain, categories are possible design modifications, for example, modify the pylon height. Essentially, each design descriptor is a category name since its modification can influence the performance. Other categories can be generated for modifications of design description properties that are tightly coupled. For example, the category *increase the number of cables while reducing their spacing* is a specialization (i.e., conjunction) of the categories increase number of cables and reduce cable spacing. Exemplars in the category structure are abstract bridge descriptions with their performance. Figure 11 depicts an abstract partial category structure consisting of three categories and two exemplars.

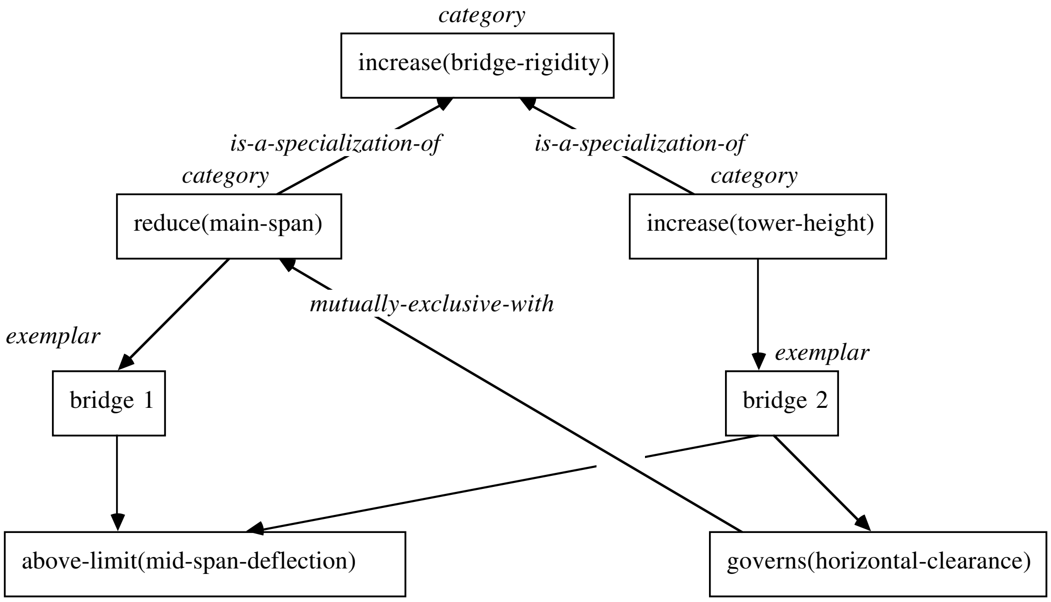

Figure 11: An abstract category structure

In redesign, the performance properties of a candidate design (especially those that violate constraints) serve as features for accessing an existing exemplar in the category structure. The category of the chosen exemplar is the best redesign modification. For example, bridge 1 in Figure 11 will be most likely chosen as a match for a new case whose middle span deflection exceeds the allowable limit, unless the horizontal clearance requirement does not allow it.

Bridger analysis module can provide immediate feedback on the redesign performance. If all the violated constraints of the design have been eliminated, the redesign was successful. Otherwise, a relative measure of the improvement can be calculated and stored in exemplars to improve future indexing. This might entail a modification to the hypothesis formation of Protos in that remindings to exemplars will be scaled by the utility stored in them.

The use of Protos is based on the hypothesis that a reasonable causal model without many conflicts can be constructed. Existing conflicting paths, such as those described before, can be resolved by exploration and refinement. This hypothesis remains to be confirmed by testing the complete Bridger system. If it is refuted then the redesign can be augmented by a concept formation system similar to the one used for learning synthesis knowledge (or the one experimented with learning heuristic analysis knowledge). In redesign, the concept formation system will be consulted first to retrieve the best fix of the design. This suggestion can then be confirmed by the Protos system. The concept formation system will gradually absorb the knowledge implicit in the apprentice system and in redesign experiences. We do not start with using concept formation since it requires many examples and do not currently use the available redesign knowledge.

5 CONCLUSIONS AND FUTURE WORK

Design is a complex activity. Structuring it and understanding the knowledge it requires for performing a particular task (section 2 of this paper), allows to define specifications of learning techniques that can acquire the domain knowledge needed (section 3 of this paper). Taking account of the specific knowledge sources available in the application domain allows to select specific machine learning programs to match the above specifications (section 4 of this paper). A partial implementation, called Bridger, exemplifies this methodology in the domain of bridge design. A summary of the process is shows on Figure 12.

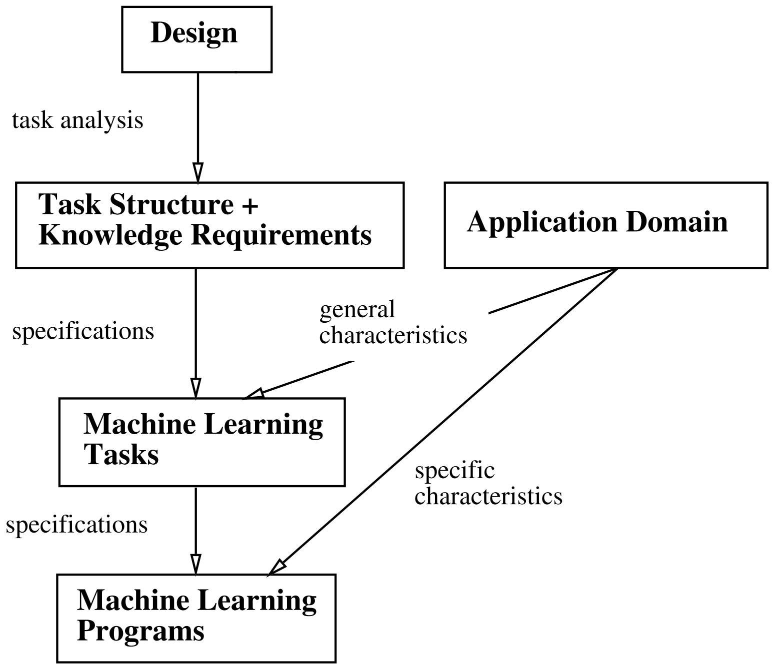

Figure 12: A summary of overall methodology

Since Bridger's modules integrate performance with continuous learning, machine learning techniques are used for all knowledge acquisition aspects possible (Shalin et al., 1988): initial system construction, refining of the existing knowledge base, and adaptation of the knowledge base to a specific style.

The methodology presented is suitable for other design domains. When following the steps in Figure 12 for different domains, different machine learning methods may emerge depending on the particular characteristics of these domains.

Future work includes completing the implementation and fully integrating the design modules. As we do not anticipate the full automation of design, we intend to provide a full interactive capability between the designer and Bridger. The designer will be able to direct learning and the use of knowledge in the synthesis module (e.g., select the learning operators manually). The designer will be able to monitor the redesign process and modify and refine the redesign knowledge. Finally, the designer will evaluate feasible designs and submit the best as training examples to Bridger.

Another work involves the extension of the description language to more complex artifact descriptions, especially descriptions with varying topologies. This will allow using the approach in additional types of domains such as mechanical and layout design.

The life cycle of artifacts continues long after the design is completed. Conceivably, the five-task model of design can be extended beyond the evaluation task to include the manufacturing task and the performance of the artifact itself. Consequently, feedback on actual performance of the artifact can be used to acquire evaluation, redesign, and synthesis knowledge. This allows viewing design as supporting the complete life-cycle of artifacts.

Finally, it is useful if Bridger will learn while not used by a designer. Currently, Bridger can heuristically select problems that are expected to enhance its synthesis knowledge. Completing the cycle to analysis, redesign, and heuristic (although not subjective) evaluation, will allow for continuous improvement of knowledge.

ACKNOWLEDGMENTS

This work has supported in part by the Engineering Design Research Center, a National Science Foundation Engineering Research Center, and the Sun Company Grant for Engineering Design Research. I would like to thank Steven J. Fenves for his support, comments, and invaluable discussions on this research and to Julian Winsor for his comments on a draft of this paper.

REFERENCES

References

Arciszewski, T., Mustafa, M., and Ziarko, W. (1987). A methodology of design knowledge acquisition for use in learning expert systems. *International Journal of Man-Machine Studies*, 27(1):23–32.

Bareiss, R. (1989). *Exemplar-Based Knowledge Acquisition*. Academic Press, Boston, MA.

Chandrasekaran, B. (1986). Generic tasks in knowledge-based reasoning: high-level building blocks for expert system design. *IEEE Expert*, 1(3):23–30.

Clark, P. (1988). Representing arguments as background knowledge for constraining generalisation. In Sleeman, D., editor, *Procedings of the Third European Working Session on Learning*, pages 37–44, Aberdeen. Pitman.

Fisher, D. H. (1987). Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2(7):139–172.

Gaines, B. R. (1987). An overview of knowledge-acquisition and transfer. *International Journal of Man-Machine Studies*, 26(4):453–472.

Gluck, M. and Corter, J. (1985). Information, uncertainty, and the utility of categories. In *Proceedings of the Seventh Annual Conference of the Cognitive Science Society, Irvine, CA*, pages 283–287, San Mateo, CA. Academic Press.

Mitchell, T., Mahadevan, S., and Steinberg, L. (1985). LEAP: A learning apprentice for VLSI design. In *Proceedings of The Ninth International Joint Conference on Artificial Intelligence, Los Angeles, CA*, pages 573–580, San Mateo, CA. Morgan Kaufmann.

Podolny, W. and Scalzi, J. B. (1986). *Construction and Design of Cable-Stayed Bridges. Second edition*. John Wiley and Sons, New York.

Porter, B. W., Bareiss, R., and Holte, R. C. (1990). Concept learning and heuristic classification in weak-theory domains. *Artificial Intelligence*, 45(1-2):229–263.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1(1):81–106.

Reich, Y. (1989). Combining nominal and continuous properties in an incremental learning system for design. Technical Report EDRC-12-33-89, Engineering Design Research Center, Carnegie Mellon University, Pittsburgh, PA.

Reich, Y. (1990a). Constructive induction by incremental concept formation. In Feldman, Y. A. and Bruckstein, A., editors, *Proceedings of the Seventh Israeli Symposium on Artificial Intelligence and Computer Vision (Ramat-Gan)*, pages 195–208, Amsterdam. Elsevier Science Publishers.

Reich, Y. (1990b). Converging to "Ideal" design knowledge by learning. In Fitzhorn, P. A., editor, *Proceedings of The First International Workshop on Formal Methods in Engineering Design*, pages 330–349, Colorado Springs, Colorado.

Reich, Y. and Fenves, S. J. (1989a). Integration of generic learning tasks. Technical Report EDRC 12-28-89, Engineering Design Research Center, Carnegie Mellon University, Pittsburgh, PA.

Reich, Y. and Fenves, S. J. (1989b). The potential of machine learning techniques for expert systems. *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing*, 3(3):175–193.

Reich, Y. and Fenves, S. J. (1991). The formation and use of abstract concepts in design. In Fisher, D. H. J., Pazzani, M. J., and Langley, P., editors, *Concept Formation: Knowledge and Experience in Unsupervised Learning*, pages 323–353, Los Altos, CA. Morgan Kaufmann.

Reich, Y. and Fenves, S. J. (1992). Inductive learning of bridge design knowledge. In Arciszewski, T. and Rossman, L. A., editors, *Knowledge Acquisition in Civil Engineering*, pages 169–189. American Society of Civil Engineers, New York, NY.

Rook, F. W. and Croghan, J. W. (1988). Knowledge acquisition for AI systems requirements analysis: A system engineering methdology. In *Proceedings of The First International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems*, pages 798–804, Tullahoma, Tennessee. ACM.

Shalin, V. L., Wisniewski, E. J., Levi, K. R., and Scott, P. D. (1988). A formal analysis of machine learning systems for knowledge acquisition. *International Journal of Man-Machine Studies*, 29(4):429–446.

Steinberg, L. I. (1987). Design as refinement plus constraint propagation: The Vexed experience. In *Proceedings of Aaai-87, Seattle, Wa*, pages 830–835, San Mateo, CA. Morgan Kaufmann.

Thompson, K. and Langley, P. (1989). Incremental concept formation with coposite objects. In Segre, A. M., editor, *Proceedings of the Sixth International Workshop on Machine Learning*, pages 371–374, Ithaca, NY. Morgan Kaufmann.

Troitsky, M. S. (1988). *Cable-Stayed Bridges: An Approach to Modern Bridge Design. Second edition*. Van Nostrand Reinhold, New York.

Witten, I. H. and MacDonald, B. A. (1988). Using concept learning for knowledge acquisition. *International Journal of Man-Machine Studies*, 29(2):171–196.

Yoshikawa, H. (1981). General Design Theory and a CAD system. In Sata, T. and Warman, E., editors, *Man-Machine Communication in CAD/CAM, Proceedings of The IFIP WG5.2-5.3 Working Conference 1980 (Tokyo)*, pages 35–57. North-Holland, Amsterdam.