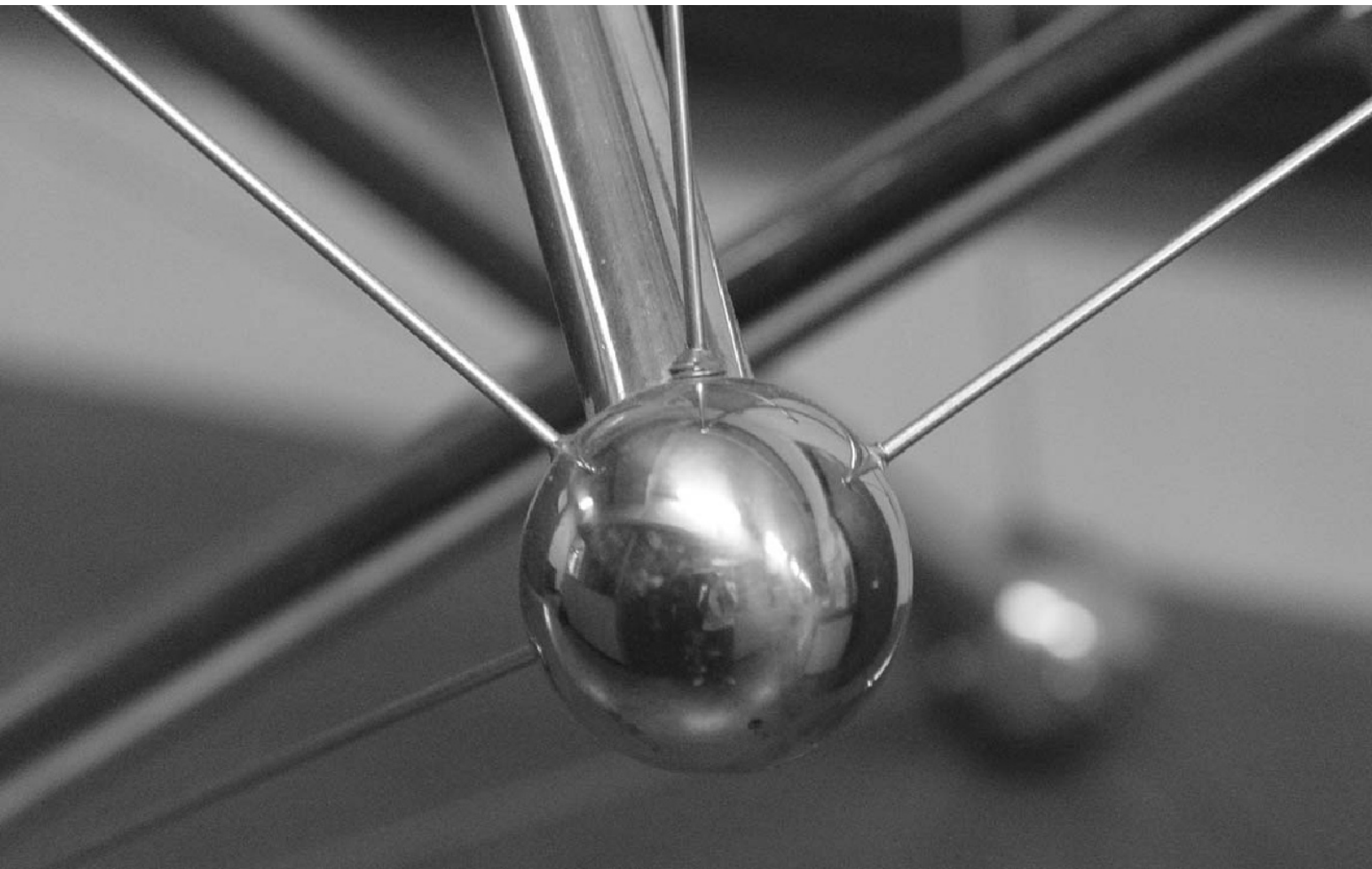
Chapter 6

Artificial intelligence for design process improvement

David C Brown
Worcester Polytechnic Institute



This chapter presents some ways in which artificial intelligence (AI) research can be used to improve the way that agents (people or machines) design things (i.e. design process improvement). There are a variety of definitions of AI, influenced by the goals of the researchers involved (Russell and Norvig, 2003). The best known is Marvin Minsky's statement that it is the science of making machines do things that would require intelligence if done by humans. This highlights the common AI paradigm of producing some theory about how a task might be done: in terms of specifying the knowledge and reasoning, and possibly also details of sensing, action and communication. The theory is then implemented in some computational form (typically a computer program) to see whether it can exhibit the appropriate intelligent behaviour. The tasks studied are usually those for which no efficient solution is known, and usually (but not always) those which intelligent beings can solve. Some researchers focus on a more cognitive point of view:

> *By 'artificial intelligence' I therefore mean the use of computer programs and programming techniques to cast light on the principles of intelligence in general and human thought in particular.*
>
> (Boden, 1977)

"By 'artificial intelligence' I therefore mean the use of computer programs and programming techniques to cast light on the principles of intelligence in general and human thought in particular."
(Boden, 1977)

while some seek to study AI in more absolute terms:

> *...studying the structure of information and the structure of problem solving processes independently of applications and independently of its realisation in animals or humans.*
>
> (McCarthy, 1974)

The area of 'AI in design' (AI-in-D) has flourished since the early 1980s. It attempts to use the techniques and approaches of AI to study design processes, most often engineering or architectural design. As it is so closely tied to AI, its researchers have also focused on different outcomes. The field has produced:

- software systems that design artifacts;
- software systems that provide assistance to designers (for example, by critiquing design choices);
- theories about how designers reason;
- studies and analyses of actual designer activities;
- models and descriptions of natural categories of design activity (for example, routine parametric design, or configuration);
- guidance about how to apply existing AI techniques to design problems.

Most AI-in-D researchers believe that engineering design is not a mysterious art and that there are core reasoning 'skills', and specific types of knowledge that apply to the same type of design task (e.g. component selection), even across domains.

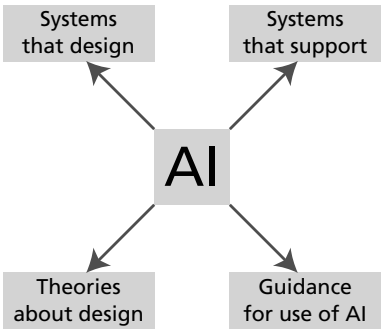
An overview of the history of the AI-in-D field can be found by looking at the following sources: the collective *Proceedings of the AI in Design* conferences; the *AI EDAM* journal (Cambridge University Press); the *IEEE Expert AI in Design* special issues (Brown and Birmingham, 1997); Stahovich's (2001) survey ; and the Encyclopedia of Artificial Intelligence article on Design (Brown, 1992).

The field has progressed over time by attempting to understand and replicate increasingly less-well understood design activities. Early work focused on parametric, routine and case-based design, moving gradually via configuration to functional reasoning and creative design, and from solo designers to teams.

There is a vibrant group of researchers active in the AI-in-D area worldwide. It has its own major conference, the *International Conference on Design Computing and Cognition* (until recently the *International Conference on AI in Design*), lots of related specialised workshops, and its own Webliography and list of AI-in-D books which can be found at

http://www.cs.wpi.edu/Research/aidg/AIinD-hotlist.html

http://www.cs.wpi.edu/Research/aidg/AIinD-books.html



6.1 AI in design

In this chapter, distinction will be made between:

- how AI has contributed to producing better theories about design processes;
- how AI can be involved in the process itself to help improve it;
- how AI can be used to produce better processes.

Artificial intelligence producing better theories

When expert systems (Jackson, 1999) were first introduced, it was quite quickly noticed that an immediate benefit of studying an expert's reasoning and knowledge, in enough detail that a software system could be built to replace him or her, was that a previously private process became public and understandable. Sometimes that yielded enough knowledge to improve the process without developing a system.

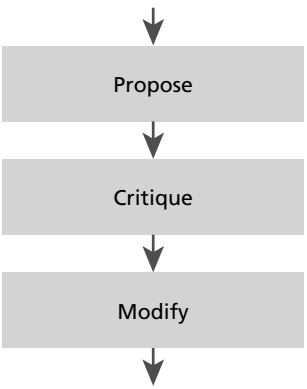
The field of AI-in-D has had a similar effect. Theories and models of design activities have been produced that make conjectures about exactly what kinds of knowledge and what kind of reasoning are necessary in different design situations. Once this is well understood, then this information can be taught, and design or design assistance systems built, all of which can improve design processes.

The rest of this section provides a brief introduction to some of this work, and points to where further descriptions might be found.

The basis

The initial AI-in-D research on parametric design, and routine (or near-routine) design resulted in several models of design processes. Two that shaped the way that researchers looked at design were Chandrasekaran's (1990) and McDermott's (1988).

In Chandrasekaran's paper, and in others subsequently, he proposes a task-method analysis where designing consists of picking an appropriate method from a set that will help address a task. Each method suggests its own sub-tasks, and they too have alternative methods. This process repeats. Methods can be selected on the basis of available knowledge, the likelihood of success, or a variety of other reasons. He points out that many design tasks can be characterised as having the sub-tasks 'propose', 'critique', and 'modify' (Figure 6.2). Proposing solutions might be done by case retrieval, constraint reasoning, or some other AI technique, or the problem might be reduced by some decomposition method into lower level sub-tasks.



6.2 Many design tasks can be characterised as having sub-tasks (Chandrasekaran, 1990)

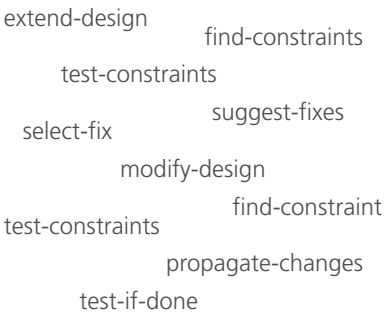
McDermott, in a similar proposal, focuses more on suggesting that we try to tease out what kinds of knowledge are needed, what roles they play, and how they can be represented. For example, he identifies design extension knowledge (propose), constraints, and fixes (to help correct constraint violations).

Methods for parametric design have been extensively studied by the AI-in-D community (Motta, 1999; Fensel, 2000). For example, 'propose and backtrack' starts by extending incomplete but consistent designs, and then restores consistency if constraints fail by backtracking to prior design extension choice points; 'propose and revise' starts by extending incomplete but consistent designs, and then revises them to restore consistency when necessary using special-purpose fixes; 'propose and improve' starts with a complete solution and then attempts to improve it. Unfortunately, these terms are not always used consistently.

Note that many parametric design problems can be viewed as constraint satisfaction problems. For example, see the discussion in Russell and Norvig (2003) of the min-conflicts heuristic for constraint satisfaction problems as an example of Propose and Improve.

The DSPL language (Brown and Chandrasekaran, 1989) was developed to allow the expression of design knowledge. It recognises distinct pieces of knowledge that represent what a designer did for major sub-problems, individual design decisions, groups of decisions, constraints, suggestions about what to do if failure occurs during designing, plans, plan selection, and several other aspects. Once these actions are captured and expressed in DSPL, it forms a design expert system.

The basis for this language was that during routine reasoning (Brown, 1996) the knowledge needed at every step is known, so that decisions can be made with essentially no searching or planning. For some problems the kinds of knowledge in DSPL will suffice. It can be used to do routine parametric design problems and routine configuration problems in a domain-independent manner. However, for many problems, designers move in and out of situations that are routine for them. They need to search and plan for a while until they get to a sub-task that they recognise and can treat as routine (Brown, 1996).



6.3 Basic methods with known knowledge types (Balkany *et al.*, 1991)

Balkany *et al.* (1991) studied several design systems, including the DSPL-based AIR-CYL system, attempting to compare and contrast them in terms of basic methods that have known knowledge types, such as extend-design, find-constraints, test-constraints, suggest-fixes, select-fix, modify-design, find-constraint, test-constraints, propagate-changes, and test-if-done (Figure 6.3). As the systems all contained a significant amount of parametric design, and a lot of routine reasoning, they found a large amount in common.

All the research mentioned above points out that identification of these basic reasoning 'skills' (the ingredients of designing) and the knowledge they need allows focused 'knowledge acquisition'. For example, pointed questions can be asked, such as "when the constraint on the size of part A fails, what is the best way to alter the design so that the constraint no longer fails?" Hence, formerly impenetrable processes can be understood in terms of these basic skills.

Once researchers in a field think something is well enough understood there is the inevitable move towards some kind of toolkit. In the area of problem-solving methods (PSMs; Motta, 1999; Fensel, 2000) there have been attempts to build catalogues of templates or abstracted modules for

different tasks. The reasoning is characterised in terms of patterns of inferences and general methods for carrying out the task. The goal is to provide reusable modules that, when completed with the right knowledge, can be used to build systems.

CommonKADS (e.g. Bernaras, 1994; Schreiber *et al.*, 1994, 1999) recognises many types of tasks, including synthesis, configuration design, assignment, and planning. The 'generic tasks' effort (Chandrasekaran and Johnson, 1993) is focused on diagnosis and design.

Smithers (1998) has a detailed formal 'knowledge-level' theory of designing in which he identifies the types of knowledge "involved in designing and the different roles they play in the overall process". He shows how these different types of knowledge are connected by processes, but does not specify the processes themselves, unlike in the PSM approaches.

Configuration and learning

Most of the work that focuses on routine and parametric design makes the assumption that the requirements are given and remain essentially the same throughout the design process. This is not always true. The AI-in-D work sharing this more general view has become known as exploration-based models of design (Smithers and Troxell, 1990). Both the candidate sets of requirements and the candidate sets of designs are specified, explored and refined during designing (Brazier *et al.*, 1994). A similar view of this process can be found in research on co-evolutionary design (Maher, 2000).

Configuration is the process of taking a set of predefined components and attempting to find a set of relationships between them that satisfies a set of constraints and meets some requirements.

Another more recent area of research has been on configuration (Mittal and Frayman, 1989; Brown, 1998; Wielinga and Schreiber, 1997). If you take a set of predefined components and attempt to find a set of relationships between them (for example, an assembly or arrangement) that satisfies a set of constraints and meets some requirements then you've done a configuration task. Simple depth-first (propose and backtrack) methods for configuration are very inefficient and require significant knowledge-based guidance.

As there are extensive commercial applications of configuration (Faltings and Freuder, 1998), this work has received a lot of attention, and powerful constraint-based and logic-based approaches have been developed (Soininen and Stumptner, 2003). Knowledge can help with the process, as, for example, if you know something about the physical or functional organisation of the desired configuration, perhaps at an abstract level, then this can be used to guide a refinement or instantiation process.

Another more recent area of study has been how to model the learning that takes place while designing. Designers accumulate all sorts of knowledge as a consequence of the process of designing, and that knowledge affects their present and future design activity.

Influenced by an analysis of how the learning that takes place in design might be classified (Grecu and Brown, 1998), Sim and Duffy (1998, 2000) took a simple model of generic design activity, in terms of the flow of knowledge, and coupled it to a model of learning. The models include eight kinds of knowledge and two kinds of process. By coupling the models in different ways they were able to characterise three kinds of design-related learning. 'Retrospective' learning occurs after the designing is completed; '*in-situ*' learning occurs during the designing activity; and 'provisional' learning takes place prior to design activity that will require the knowledge being learned.

Creativity

There is a very large body of literature on creativity, but less about creative design (Christiaans, 1992; Gero and Maher, 1993, 2001; Dasgupta, 1994, 1996; Goel, 1997). Creativity is seen as being relative to a standard of some kind, such as the designer's own previous designs, or the designs of some community (Boden, 1994). As a consequence, it is hard to come up with a model of this activity.

Analogical reasoning involves the retrieval of the solution to a similar design problem and the transfer of its relevant aspects for use in the solution to the original problem.

It is possible, however, to point to some situations and approaches that tend to produce results that are judged to be creative. In a situation where the design variables need to be determined and can change, and the ranges of their possible values can change, then the design process (and probably the result) will be creative as the designer will be taken outside his or her normal experience during much of the process. The approaches that tend to produce creative results include the use of analogy (Goel, 1997), functional reasoning (Umeda and Tomiyama, 1997), mutation (de Silva Garza and Maher, 2000) and reasoning from first principles (Williams, 1992).

Analogical reasoning involves the retrieval of the solution to a similar design problem and the transfer of its relevant aspects for use in the solution to the original problem. Analogy can be used to provide designs, methods, plans or other useful knowledge.

Analogical design research usually focuses on conceptual design, as more abstract descriptions are easier to work with. Representations of function are also at this level and can be used in analogical design, in conjunction with representations of structure and behaviour (Balazs and Brown, 2001).

Functional representations can also be used to guide the refinement of a design, as once the necessary functionality has been determined it can be refined into types of component that can provide it (Umeda and Tomiyama, 1997). They can also be used to provide high-level simulations of a proposed design. AI provides a richer set of representations than that currently found in engineering design views of function (Wood and Greer, 2001).

Mutation is used to explore the space of possible designs by making changes to a proposed (partial) design. This might then lead the designer into new sub-problems not faced before, or allow him to make unusual associations that lead to interesting analogies. Reasoning from first principles allows the designer to work from scratch and to be less biased by his existing knowledge.

Artificial intelligence involved in the process

There are plenty of places in the design process where a designer can use a little more assistance. Intelligent tools, for example, can provide assistance with generating design ideas, can critique design proposals, and can actually do some of the designing to relieve the designer of the more mundane tasks.

A finer grained analysis of roles that knowledge-based techniques and systems can play during design can be found in Brown (1992). These design sub-tasks include: abstraction; analysis; conflict resolution; criticism; decomposition; estimation; evaluation; interpretation; learning; negotiation; patching; prediction; redesign; refinement; retraction; suggestion making; and selection.

There are a wide variety of AI techniques that can be used to assist with these sub-tasks to help improve the process (Brown, 1992; Stahovich, 2001). These include expert systems, genetic algorithms (GAs), case-based reasoning (GBR) and formal grammars. Other examples include neural networks, qualitative reasoning, heuristic search, planning and multi-agent systems.

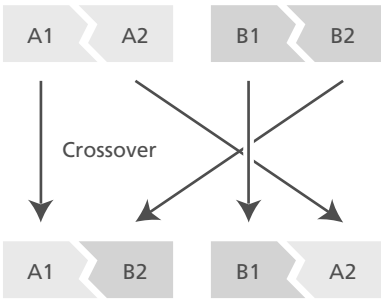
Expert systems are computer programs that solve problems or give advice about some specialised subject by heuristic reasoning using representations of knowledge.

Expert systems (David *et al.*, 1993; Jackson, 1999) are computer programs that solve problems or give advice about some specialised subject by reasoning using representations of knowledge. They are normally associated with tasks that would be done by an expert, such as diagnosis or design. They usually involve heuristic reasoning, structuring the reasoning the way the expert would, and using knowledge based on the experiences of an expert. The majority of such systems have been built using rules. Each rule describes some key aspect of a situation (for example, the partial diagnosis and the symptoms available) and suggests what action to take in that situation. They can be used 'forwards', recognising and acting, or 'backwards', where the

system hypothesises that the result exists (for example, a disease, for diagnosis) and reasons back through the rules to see whether all the necessary conditions for its existence are actually present. Rule-based systems are appropriate for a very wide variety of applications.

GAs (Bentley, 1999; Bentley and Corne, 2001; Lee *et al.*, 2001) attempt to mimic evolution. It is a very flexible and efficient technique for searching a large space. In this approach, descriptions (of designs or design processes, for example) evolve and improve. A whole 'population' of descriptions is considered at once.

In order to make progress towards better and better solutions, the descriptions are evaluated in every generation to determine their fitness, as fit solutions are worth keeping. Fitness might be, for example, how well the design represented by the description meets the requirements. You'd also like to keep the qualities of the best items and propagate them to the next generation. To enable this, randomly selected fit parents are used to generate offspring descriptions by doing a 'crossover' between their two descriptions (Figure 6.4). Typically the two parts of description A (A1, A2) are mixed with the two parts of description B (B1, B2), to generate two new descriptions (A1, B2) and (B1, A2). Sometimes new descriptions are also generated by random mutations of a few descriptions. The fittest are kept and passed to the new generation.



6.4 GAs, an attempt to mimic evolution

This process of generating offspring, evaluation, and passing fit members of the population to the next generation is repeated until some stopping criteria are reached – for example, the best description does not improve over several generations.

CBR (Maher *et al.*, 1995; Maher and de Silva Garza, 1997; Maher and Pu, 1997) is a method for reusing experience. Cases, past design solutions for example, are collected and 'indexed' by key features. For design cases, the features used to index them might include the requirements of the problem for which that design was the solution. Hence, when a new design is needed the index is used to 'recall' the stored cases that appear to be the most likely to be solutions (because the new requirements are similar to the old ones). Once those candidates are retrieved and evaluated, either one case is the perfect answer, or one or more recalled cases need to be 'adapted' to improve them. Typically, CBR systems rely on many cases and small adaptations. But if a human designer is doing the adaptation then the cases can be presented to him or her as starting points for the design process, encouraging him or her to build on well-tried results, and also to try things that would not necessarily have been thought of.

Formal grammars (Cagan, 2001; Brown, 1997) are a way of representing the structure of things precisely. Grammars for languages include rules such as "a sentence (S) is a noun phrase (NP) followed by a verb phrase (VP)", written "S → NP VP". Rules such as these can be used to recognise (by seeing an NP next to a VP and reporting that they form an S), or to generate (by seeing an S and replacing it by NP followed by VP). Rules describing an NP, a VP and their components allow recognition and generation of sentences.

This example operates in one dimension (the row of words), but grammars for design need to be for two or three dimensions. Shape grammars allow representations of shapes to be on both the left- and right-hand sides of a rule. Hence, more complex shapes (*i.e.* designs) can be generated by replacing examples of the shape from the left-hand side of a rule with the (usually more complex) shape from the right-hand side of the rule. Even more flexibility can be gained by adding parameters to the shapes in the rules. Semantic checking can also be added by allowing attributes to be attached to portions of the shape being generated, and constraints on them to be associated with the rules.

Artificial intelligence producing better processes

AI has had a lot of indirect impact on design processes. Models have been developed that have affected the way we describe design processes; types of knowledge and reasoning have been identified, and AI has led to new tools and new processes (for example, the use of GAs). In general, AI has affected our ways of thinking about designing, so that we now see it as a potentially understandable and rational information-processing task that requires lots of knowledge, and lots of different reasoning skills.

AI has affected our ways of thinking about designing, so that we now see it as a potentially understandable and rational information-processing task that requires lots of knowledge, and lots of different reasoning skills.

The expert systems, PSM and modelling efforts have raised the awareness of the importance of knowledge acquisition (not only of expert knowledge), and have led to studies of knowledge sharing (including the use of ontologies).

But there are opportunities for AI to affect the design process directly. The following sections present three examples: agent learning; methodology generation; and planning.

Agent learning

First, some work that shows ways in which learning can occur during designing will be presented (Grecu and Brown, 1996, 2000). Our experiences greatly affect how we design, as past successes and failures, for example, change the way we do things and change our evaluations of potential design decisions.

Sometimes during designing an attempt is made to decompose the problem so that decisions can be made more independently. Even for an individual designer, one decision can affect another in unanticipated ways. Separation of decision makers, in design teams for example, makes it even harder for one decision maker to know what impact his decisions might have on another. Usually, however, sub-tasks and decisions really are not completely independent, and the composition of partial solutions that follows from decomposition of the problem reveals conflicts.

In Grecu and Brown's single function agent-based design system small knowledge-based programs, known as agents, interact to solve a spring design problem. Each agent has a particular 'role' to play, dictated mainly by its function (what kinds of input it needs and output it produces).

For example, one agent might select a value for a parameter, while another acts as a critic, offering an explanation of why the chosen value isn't good. The selector agent builds a model (i.e. learns) of which selections to avoid in which design situations, based on feedback from critics. This reduces the number of interactions between agents by about half, thus reducing communication overhead and speeding up the whole process.

An experienced designer can use expectations to predict from available information what will happen later in the design process.

Another form of knowledge that can be learned is expectations (Grecu and Brown, 2000). An experienced designer can use expectations to predict from available information what will happen later in the design process: for example, that a constraint may fail. This can be done despite having only part of the information required to know this definitely.

In Grecu and Brown's LEAD system, "causal attribution" is used to collect together all the factors that might be an input to the expectation in the situation for which an expectation is desired. This might include relevant constraints, design decisions already made, or the state of other agents. Once those features are collected, an inductive learning phase, known as covariational analysis, uses data collected from past design processes to filter out those features that do not contribute. It builds a description of the expectation, which is validated prior to use in the system. The expectations change the design process, allowing better design choices, and reducing failures.

Methodology generation

The robot designer (RD) system (Shakeri and Brown, 2004) is concerned with methodology generation for multi-disciplinary design (for example, robot arm design). A design methodology is a scheme for organising

reasoning steps and domain knowledge in order to construct a solution effectively and efficiently. A good methodology guides a designer towards a successful design.

The RD system was built as a collection of agents, where each did a very small portion of the design from one disciplinary point of view. By breaking disciplinary knowledge up into many small pieces it becomes possible to interleave decisions from different disciplines, and even make them in parallel. Agents were opportunistic and were able to contribute when their pre-conditions were met.

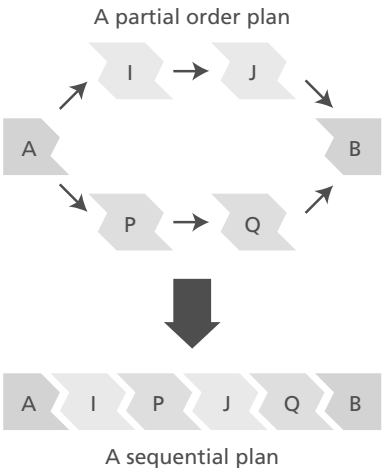
The agent-based system was given many sets of systematically slightly varying requirements. The resulting design traces were analysed. In the 960 experiments run, only 4% of the 2304 possible traces generated successful designs. In addition, some sets of traces were similar. By clustering them, it was possible to identify families of traces and describe to what situations they applied. By generalising these clusters it was possible to produce methodologies.

Planning

A large sub-area of AI is planning (Allen *et al.*, 1994; Russell and Norvig, 2003), which involves producing a set of actions, in advance of their execution, that is expected to achieve a goal. Clearly, planning is used in designing, as the design process itself needs to be ordered and structured.

Planning techniques involving searching for a plan in a space of possible plans, or proving a plan using logic, are not efficient. Actions are typically represented using a description of what must be true before the action can be used (preconditions), a list of what is no longer true after the action, and a list of what new conditions are true after the action.

A good approach to planning is to commit to where actions should go in the plan as little as possible until it is absolutely necessary. Hence a "partial order plan" might determine that the action sequences (I then J) and (P then Q) are between actions A and B, but makes no further commitment. This results in six possible sequential plans; for example, (A, I, P, J, Q, B) (see Figure 6.5). The process of building a partial order plan typically works back from the goal, finding actions that can satisfy outstanding preconditions for actions already selected. In addition, actions that invalidate other pre-conditions are moved or replaced. More complex planning techniques are required for realistic planning situations involving resources (scheduling), uncertainty, and multiple agents planning together.



6.5 AI in planning

Conclusions

In this chapter, some of the areas in which AI can impact the design processes have been revealed. The study of AI, and subsequently AI-in-D, has led to a number of theories and models of the design process, and of subareas such as configuration, learning and creativity. There are many uses during designing for AI techniques such as expert Ssystems, GAs, CBR and formal grammars. They can be used in tools to aid the designer. Finally, AI can be used to produce better processes themselves, e.g. via agent learning, methodology generation, and planning.

It's evident that AI has contributed to producing better theories about design processes, that AI can be involved in the process itself to help improve it, and that AI can be used to produce better processes.

References

Allen J, Tate A, Hendler J eds (1994) Readings in planning. Morgan Kaufmann

Balazs ME, Brown DC (2001) Design simplification by analogical reasoning. In: Knowledge intensive computer aided design. Kluwer Academic

Balkany A, Birmingham WP, Tommelein ID (1991) A knowledge-level analysis of several design tools. AID'91, Edinburgh, UK

Bentley PJ ed. (1999) Evolutionary design by computers. Morgan Kaufmann

Bentley PJ, Corne DW (2001) Creative evolutionary systems. Morgan Kaufmann

Bernaras A (1994) Problem-oriented and task-oriented models of design. AID'94, Lausanne, Switzerland

Boden M (1977) Artificial intelligence and natural man. Basic Books

Boden M (1994) What is creativity? In: Dimensions of creativity. MIT Press

Brazier F, van Langen P, van Ruttkay P, Truer J (1994) On formal specification of design tasks. AID'94, Lausanne, Switzerland

Brown DC (1992) Design. In: Encyclopedia of artificial intelligence. John Wiley

Brown DC (1996) Routineness revisited. In: Mechanical design: theory and methodology. Springer

Brown DC (1998) Defining configuring. AI EDAM, special issue on Configuration, 12(4): 301–305

Brown DC, Birmingham WP (1997) IEEE Expert, special issues on AI in Design, 12(2) and 12(3)

Brown DC, Chandrasekaran B (1989) Design problem solving: knowledge structures and control strategies. In: Research notes in artificial intelligence series. Pitman
Brown KN (1997) Grammatical design. IEEE Expert, special issue on AI in Design, 12(2): 27–33
Cagan J (2001) Engineering shape grammars. In: Formal engineering design synthesis. Cambridge University Press
Chandrasekaran B (1990) Design problem solving: a task analysis. AI Magazine, 11(4): 59–71
Chandrasekaran B, Johnson TR (1993) Generic tasks and task structures: history, critique and new directions. In: Second generation expert systems. Springer
Christiaans HHCM (1992) Creativity in design: the role of domain knowledge in designing. Lemma
Dasgupta S (1994) Creativity in invention and design. Cambridge University Press
Dasgupta S (1996) Technology and creativity. Oxford University Press
David JM, Krivine JP, Simmons R eds (1993) Second generation expert systems. Springer
De Silva Garza AG, Maher ML (2000) A process model for evolutionary design case adaptation. AID'00, Worcester Polytechnic Institute, MA, USA
Faltings B, Freuder E eds (1998) IEEE Intelligent Systems Engineering, special issue on Configuration, 13(4): 32–33
Fensel D (2000) Problem-solving methods: understanding, description, development, and reuse. Lecture Notes in AI, LNAI 1791. Springer
Gero JS, Maher ML eds (1993) Modeling creativity and knowledge-based creative design. Lawrence Erlbaum
Gero JS, Maher ML eds (2001) Computational and cognitive models of creative design. HI'01, Heron Island, Queensland, Australia
Goel AK (1997) Design, analogy, and creativity. IEEE Expert, 12(3): 62–70
Grecu DL, Brown DC (1996) Learning by single function agents during spring design. AID'96, Stanford, CA, USA
Grecu DL, Brown DC (1998) Dimensions of machine learning in design. AI EDAM, special issue on Machine Learning in Design, 12(2): 117–121
Grecu DL, Brown DC (2000) Expectation formation in multi-agent design systems. AID'00, Worcester Polytechnic Institute, MA, USA
Jackson P (1999) Introduction to expert systems. Addison Wesley Longman

Lee C-Y, Ma L, Antonsson EK (2001) Evolutionary and adaptive synthesis methods. In: Formal engineering design synthesis. Cambridge University Press
Maher ML (2000) A model of co-evolutionary design. Engineering with Computers, 16(3/4): 195–208
Maher ML, de Silva Garza AG (1997) Case-based reasoning in design. IEEE Expert, 12(2): 34–41
Maher ML, Pu P eds (1997) Issues and applications of case-based reasoning in design. Lawrence Erlbaum
Maher ML, Balachandran M, Zhang DM (1995) Case-based reasoning in design. Lawrence Erlbaum
McCarthy J (1974) Review of "Artificial intelligence: a general survey". Artificial Intelligence, 5: 317–322
McDermott J (1988) Preliminary steps towards a taxonomy of problem-solving methods. In: Automating knowledge acquisition for expert systems. Kluwer Academic
Mittal S, Frayman F (1989) Towards a generic model of configuration tasks. IJCAI'89, Detroit, MI, USA
Motta E (1999) Reusable components for knowledge modelling. IOS Press
Russell S, Norvig P (2003) Artificial intelligence: a modern approach. Prentice Hall
Schreiber G, Wielinga B, de Hoog R, Akkermans H, Van de Velde W (1994) CommonKADS: a comprehensive methodology for KBS development. IEEE Expert, 9(6): 28–37
Schreiber G, Akkermans H, Anjewierden A, de Hoog R, Shadbolt N, Van de Velde W, Wielinga B (1999) Knowledge engineering and management: the CommonKADS methodology. MIT Press
Shakeri C, Brown DC (2004) Constructing design methodologies using multi-agent systems. AI EDAM, 18(3)
Sim SK, Duffy AHB (1998) A foundation for machine learning in design. AI EDAM, 12(2): 193–209
Sim SK, Duffy AHB (2000) Evaluating a model of learning in design using protocol analysis. AID'00, Worcester Polytechnic Institute, MA, USA
Smithers T (1998) Towards a knowledge level theory of design process. AID'98, Lisbon, Portugal
Smithers T, Troxell W (1990) Design is intelligent behaviour, but what's the formalism. AI EDAM, 4: 89–98
Soininen T, Stumptner M (2003) AI EDAM, special issue on Configuration, 17(1): 1–2

Stahovich TF (2001) Artificial intelligence for design. In: Formal engineering design synthesis. Cambridge University Press
Umeda Y, Tomiyama T (1997) Functional reasoning in design. IEEE Expert, 12(2): 42– 48
Wielinga R, Schreiber G (1997) Configuration-design problem solving. IEEE Expert, 12(2): 49–56
Williams BC (1992) Interaction-based design: constructing novel devices from first principles. In: Intelligent computer aided design. Elsevier Science
Wood KL, Greer JL (2001) Function-based synthesis methods in engineering design. In: Formal engineering design synthesis. Cambridge University Press