8-15-1997

Application of AI Principles to Constraint Managementin Intelligent User Interfaces

Donald Day
The University of New South Wales, d.day@acm.org

Application of AI Principles to Constraint Management in Intelligent User Interfaces

Donald Day
School of Information Systems, The University of New South Wales, Sydney, NSW 2052 Australia
d.day@acm.org

Abstract

This paper describes the application of artificial intelligence principles to constraint management in intelligent user interfaces. Following a description of the problem area, I discuss reasoning models, design knowledge representation, and implementation. User acceptance issues and a current study that applies AI principles to constraints also are addressed.

The Problem Area

As the computerized decision support tools become more common in the design professions, increasing numbers of users must negotiate process constraints in their everyday work.

Prior research suggests that user resistance to design tools may be triggered by the high visibility imposition of constraints (Day, 1996). Artificial intelligence implementation of such constraints may offer one means to minimize this alienation among tool users.

Reasoning Models

There are three perspectives from which to consider constraints in design decision making. First, constraints can be viewed as elimination rules used in object selection. Also, constraints may be considered partial descriptions and commitments in plan refinement. Finally, constraints can be seen as a communication medium expressing interactions between subproblems (Stefik, 1981).

Least Commitment Solutions

The interpretation of constraints as commitments is central to the least-commitment strategy, in which object and process selection decisions are deferred for as long as possible. Stefik (1981) describes a "constraint posting" approach to least commitment for managing interactions among subproblems. Constraints are noted when recognized, but not imposed until necessary. Constraint posting uses constraints to anticipate potentially harmful interactions among loosely coupled subsystems.

A hierarchical, least-commitment system compares goals, identifies differences among goals, and selects operators to reduce differences. The system depends upon a hierarchical knowledge base, divided into objects and design operators. Design operators refine objects (and other operators), create and propagate constraints, simulate user actions, and identify differences.

Stefik notes that in test execution of one constraint posting system the total combinations of potential solutions decreased from 3456 to 4 as constraints were added. The largest number of candidate lookahead solutions the system needed to track simultaneously was 21.

A least-commitment model also was implemented by Mitchell, Steinberg and Shulman (1985) in VEXED, a knowledge-based system for VLSI design. The authors report that:

As implementation choices are made for one submodule in the design, additional knowledge of the behaviour of this submodule is acquired which must then be converted into additional constraints on the interface between submodules and therefore on the specifications of other submodules in the design.

The authors also note that a least commitment system must log enough information about its own operation that it can "replay" the process, showing the hierarchy of abstractions and implementation choices that led to the final design. This is important to backtracking, and could be used to automate repetitive design activities.

Day (1995) discusses least commitment principles in refining an earlier model of cognitive fit between task and technology. His paper describes in theory how least commitment could be applied as part of constraint network propagation within a CASE tool.

Propose and Revise Solutions

In the propose and revise model, a proposed design is built, constraints are identified, and the design is modified to satisfy the constraints.

Stout et al (1988) extended SALT, an AI knowledge acquisition tool, to design problem solving applications. As each procedure that had passed the constraint test was applied, a "dependency network" was built. The network what had occurred, to facilitate later backtracking if subsequent violations forced a "truth maintenance system" to remove design elements that were inconsistent with the preferred design.

Truth maintenance may reveal a high level of constraint antagonism among proposed fixes. Stout et al note that "...virtually every fix has the potential of causing several new constraint violations". Resolving constraint antagonism promptly can minimise error propagation. However, this requires that constraint lookahead (forward reasoning) be part of the system, to ensure that a local fix does not violate more fundamental global constraints.

Network Solutions

AirCAD, an object-oriented tool for the interactive design of airport manoeuvring areas, uses a hierarchy of object classes to reflect the relationships of all possible entities in a design (Rigopoulos and Schmitt, 1988).

Design objects in AirCAD are representations of physical objects (e.g., equipment), geometric objects (e.g., minimum separation distances), and database objects (airport data and tables). Intersections of runways and taxiways are treated as nodes; design objects that connect nodes are links. Links inherit properties (e.g., constraints) from entities to which they are connected, in addition to having properties of their own. Constraints are inherited from class variables that cannot be cancelled, deleted or changed.

A key feature of AirCAD is the hierarchical interdependence of objects. The system warns users when undesirable components or configurations are selected, then suggests changes. AirCAD verifies that every new entity avoids conflict antagonism, as it is inserted. AirCAD displays restrictions graphically, so they are obvious to the designer.

The Antecedent Reasoning System (ARS) was used by Stallman and Sussman (1977) to implement a set of rules for electronic circuit analysis. ARS threads deduced facts with justifications -- justifications that mention antecedent facts and the inference rule applied.

Dependency directed backtracking was developed by Stallman and Sussman to avoid combinatorial explosions. ARS scans backward from a contradiction (along its chain of deduction) to identify incorrect choices; it never tries those combinations again.

ARS uses its record of deductions to explain the derivation of facts to users, to find choices relevant to a contradiction, and to delineate facts believed to be true. Whenever an assumption is invalidated, the system scans forward from that choicepoint, marking all consequences of the discredited choice invalid. Since such

decisions are only marked invalid (not eliminated), it is possible to reinstate them and their consequences if later reasoning causes the system to reconsider the discredited decision.

Another type of network solution was described by Jain and Maher (1988). Evaluator, an expert system for the evaluation of structural designs for buildings, applies hierarchically organized design criteria to form a tree that includes inferred and implied nodes. Inferred nodes take their values from system rules; derived nodes are assigned values using their subordinates' values and associated importances. The importance of any node signifies the extent to which a descendant can influence the values of its parent in the hierarchy.

Design Knowledge Representation

Since both text and graphics are important to design, a useful interface should include semantic as well as syntactic representation of design knowledge. According to Oxman and Gero (1987), " Semantics -> syntax and syntax -> semantics transformations are fundamental to any interactive graphics interface of a design expert system."

However, Jain and Maher (1988) note problems in combining graphics and expert system techniques. Drawings are a syntactical component in which entities have contextual dependence. By contrast, reasoning functions of an expert system are semantic components that combine variables, value ranges, decisions, evaluations and justifications.

Implementation Issues

Research in the use of expert systems for design tasks has identified a range of implementation issues.

Mitchell, Steinberg and Shulman (1985) cite (1) questions regarding granularity (the size or impact of commitments), (2) questions of which constraints to include in rule pre-conditions and which to consider as subgoals), and (3) questions about whether and how the machine should learn new rules from users as they override constraints (a so-called "learning apprentice" system). Also of concern is the need for recovery methods if constraint conflicts go undetected.

User Acceptance Issues

Regardless of the solution model applied, the effectiveness of AI design applications depends upon a minimum level of trust by professional users. As Bonnie Muir (1987) observes

The concept of trust is a critical one in the design of decision support systems. If forced to use an aid which he does not trust, a decision maker may use any means available, even very demanding and time consuming ones, to direct the output of the aid toward his own decisions.

Buchanan and Feigenbaum (1978) suggest that system acceptability can be enhanced by making the scope and limitations of problem solving methods known to the user, and by providing estimates of problem size and of the amount of work remaining, at any point in the design process.

Application in Current Research

This paper has described how artificial intelligence principles can be applied to design decision making. These principles are being applied currently in CADPRO, a study of software engineers' responses to process constraints (Day, Ahuja & Scott, 1997). This three-year examination of Constraints And the Decision PROcess includes specification and construction of a constraint-variable computer-aided systems engineering tool, to be used in laboratory observation of user behavior.

CADPRO structures the design problem space as a semantic network in which each decision node is a frame. Constraints occupy a slot within each decision frame. Constraint propagation is viewed as the probabilistic firing of links between nodes, forming a decision pattern limited by the characteristics of slot constraints.

This model provides a cognitive mapping framework for experimental design and for the interpretation of study results. Specific constraints will be assigned to tasks in user scenarios. Constraint selection will apply least commitment and propose-and-revise principles. During analysis, user decision patterns will be mapped into a node structure by applying network solution principles and the frame-and-slot approach described above.

Ultimately, findings of this study may be used to specify an intelligent user interface for design tools -- an interface which uses adaptive discovery (Day, 1995) to dynamically adjust levels of constraint to match the varying skills of individual users.

References

Buchanan, B. & Feigenbaum, E. (1978). Dendral and Meta-Dendral: Their applications dimension. *Artificial Intelligence* **11**, 5-24.

Day, D. (1995). Adaptive discovery and least commitment: An extension of cognitive fit. In Hasan, H. & Nicastri, C. (Eds.), *HCI: A Light into the Future*, 256-261. Proceedings, 4th conference of the Computer Human Interaction Special Interest Group of the Ergonomics Society of Australia (OZCHI'95), Nov. 27-30, Wollongong.

Day, D. (1996). User responses to constraints in computerized design tools: An extended abstract. *Software Engineering Notes* **21**(5), 47-50 (September 1996).

Day, D., Ahuja, M. & Scott, L. (1997). Constraints in design engineering: a report of research in progress. Submitted to 8th Australasian Conference on Information Systems (29 September - 2 October 1997, University of South Australia, Adelaide)

Jain, D. & Maher, M.L. (1988). Combining expert systems and CAD techniques. In Gero, J. & Stanton, R. (Eds.), *Artificial Intelligence Developments and Applications*, 65-81. North-Holland: Amsterdam.

Mitchell, T., Steinberg, L. & Shulman, J. (1985). A knowledge-based approach to design. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **PAMI-7**(5), 502-510.

Muir, B. (1987). Trust between humans and machines, and the design of decision aids. *International Journal of Man-Machine Studies* **27**, 527-539.

Oxman, R. & Gero, J. (1987). Using an expert system for design diagnosis and design synthesis. *Expert Systems* **4**(1, February), 4-15.

Rigopoulos, D. & Schmitt, G. (1988). AirCAD: A tool for CAD of Aerodromes. *Proceedings of the Fifth Conference on Computing in Civil Engineering*, 635-644. American Society of Civil Engineers: New York.

Stallman, R. & Sussman, G. (1977). Forward reasoning and dependency-directed backtracking in a system for computer-aided circuit analysis. *Artificial Intelligence* **9**, 135-196.

Stefik, M. (1981). Planning with constraints (MOLGEN: Part 1). *Artificial Intelligence* **16**, 111-140.

Stout, J., Caplain, G., Marcus, S. & McDermott, J. (1988). Toward automating recognition of differing problem-solving demands. *International Journal of Man-Machine Studies* **29**, 599-611.