Framework for AI-Based Nuclear Reactor Design Support System

Kazuo FURUTA and Shunsuke KONDO

Department of Nuclear Engineering, Faculty of Engineering, University of Tokyo*

Received October 22, 1991

Nowadays many computer programs are being developed and used for the analytic tasks in nuclear reactor design, but experienced designers are still responsible for most of the synthetic tasks which are not amenable to algorithmic computer processes. Artificial intelligence (AI) is a promising technology to deal with these intractable tasks in design. In development of AI-based design support systems, it is desirable to choose a comprehensive framework based on the scientific theory of design. In this work a framework for AI-based design support systems for nuclear reactor design will be proposed based on an explorative abduction model of design. The fundamental architectures of this framework will be described especially on knowledge representation, context management and design planning.

KEYWORDS: artificial intelligence, expert systems, design, support system, programming, knowledge base, consistency maintenance, ATMS, blackboard model, numerical methods, nonlinear planning

I. INTRODUCTION

Application of artificial intelligence (AI) has come to be a trend in various domains of engineering as basic research has progressed in these decades. While a lot of attempts are being made to apply AI techniques in nuclear industry, design is a promising field of the application. Computers are widely used in the nowadays design practice to perform numerical simulations, but designers are still responsible for the tasks that are not amenable to algorithmic computer processes. AI seems to be an effective method to free designers from some of these tasks so that they can concentrate more on demonstrating creativity. In nuclear industry, expert systems were developed for layout of plant equipment, pipe routing, logic design of control systems, and so on.

So far various architectures of design support systems have been invented and sophisticated for particular domains. Though very powerful within the original domains, most of them are too specific to apply to other domains. Consequently, one needs as many architectures as the number of domains. This problem can be alleviated, if design activities can be decomposed and each part can be performed independently. In this case, however, the interface between the experts in charge of different domains is a serious problem. A common platform of design support architectures which can accept domain-specific methods as well will be necessary to solve these problems.

It is sometimes argued that there is no generic methodology of design because of the mysterious and ill-structured nature. Many researchers, however, are studying design scientifically to formalize the general aspects of design, and remarkable progress has been made. The design theories proposed in these efforts are expected to enhance generality and prospects of AI-based design support systems; they will provide a good framework for the development. In the present paper a framework of AI-based nuclear reactor design support systems will be proposed based on an explorative abduction model of design, which is an attempt of design formalization.

The role of numerical methods as well as

* Hongo, Bunkyo-ku, Tokyo 113.
AI techniques is increasing in the present design practice, but the methodology to integrate both technologies seems still immature. Another aim of this paper, therefore, is to present a methodology to combine AI techniques and numerical methods in the same framework so that designers can use more computer powers.

The goal of this work is not to automate routine design but to provide various supports for the designers. The framework to be proposed primarily deals with the class of design for which the knowledge on solution candidates and their specification methodologies are already structured and represented. It is however open to the proposal of new ideas by the designers in more innovative design as long as the ideas can be expressed in a symbolic form.

II. OVERALL ARCHITECTURE

Design is defined as a human activity that design requirements are transformed into design specifications. This process is viewed often as a mapping from the function space onto the attribute space(3). Such an ideal and static model of design provides the most basic viewpoint of the issue, but it is not enough to feature the relevant aspects of design.

One of the aspects that are remarkable but overlooked in the static model is that design is a reversal problem solving. Though design is a process to obtain design descriptions from design requirements, the direct methodology to do so is usually lacking. Our scientific knowledge is useful for the analysis of a certain object to predict the expected behavior, but it is powerless for problem solving in the opposite direction. Design thereby needs a repetitive process of proposing a hypothetic solution, predicting the expected behavior, and comparing the prediction with design requirements. The design process can be modeled as abduction to obtain some consistent set of design specifications $S_d$ such that

$$K_d \cup S_d \models R_d,$$

where $K_d$ and $R_d$ denote design knowledge and design requirements, and $A \models B$ means $B$ is derivable from $A$.

The second remarkable aspect of design is its explorative nature. It has been proposed that human design can be modeled as a search in a large problem space(4). This view is based on the assumption that the problem space of design, which consists of state space of the problem, a set of operators, and the goal state are predefined before the design process starts. It is not always the case, since design often includes the process of discovering the structure and the goal of the problem. For example, the design requirements may be ambiguous and inconsistent at the beginning, and it is investigated in parallel with a design solution. If this is a rule rather than an exception, it is improper to model design as a search.

Smithers et al. has proposed an exploration model of design(5)(6). In this model, discovery of the problem structure by applying design knowledge is performed first, and then search within the space already structured follows. These activities are repeated in tandem till the problem space towards a design solution has all been discovered. Initially, the design requirements guide this process only weakly in search of a possible design space, and finally the complete design requirements as well as the design solution are obtained. Such a view seems more suitable for the model of design.

From the above discussion, the authors propose to model design as explorative abduction, and developing an AI-based nuclear reactor design support system based on such a view of design. Figure 1 shows the brief structure of this design support system, which consists of the following modules:

**Design Knowledge Base (DKB)** represents the knowledge used in design, which is organized hierarchically by two relationships: kind-of and part-of. Each module of the DKB declares a class of parameters and constraints on design objects in accordance with designers' perception of the design space.

**Design Description Document (DDD)** is a knowledge base representing the knowledge generated in the course of the design process.
The consistency of this knowledge base is maintained by the Assumption-based Truth Maintenance System (ATMS)(7).

Support subsystems carry out useful works to support the designer in the design process. General purpose support subsystems carry out tasks such as algebraic equation solving, geometric reasoning, table manipulation etc., which are common to several domains. Special purpose support subsystems carry out goal-directed tasks specific in a particular domain under close supervision of the designer.

A control system based on the blackboard model is adopted(8). It makes an agenda of the tasks that some support subsystem can do, selects the item with the highest priority from the agenda and executes the selected task.

A user interface is provided for the designer to communicate with the system. The user is thought as a knowledge source, who can supply additional knowledge to the system.

III. DESIGN KNOWLEDGE BASE

The primary knowledge which is useful to discover the structure of the design space is the knowledge on the objects which are candidates for a design solution. This knowledge base is usually organized as hierarchical class definitions, and instances of design objects are generated from the general object classes and operated on in the course of the design process. The attributes of the objects defined here will be used to parameterize the design space. Additionally, several kinds of constraints: equations, inequality relations, geometric relations, etc. are also declared in the class definitions, and they will be used to prune the space of possible design. Figure 2 shows an example of object class definition.

```plaintext
class fuel_pin has
    instance X;
    component
        clad isa cyl_pipe,
        fuel isa fuel_pallet;
    attribute
        diameter,
        length,
        ........
    composition;
    X&clad:inner_dim = X&fuel:diameter + 2*X&gap_width;
    X&fuel:diameter < X&clad:inner_dim;
    ........
end.
```

Fig. 2 Example of object class definition

The expressions X & Y and X : Y in the above representation denote object Y as a part of object X and attribute Y as a property of object X respectively.

Every numerical method used in design is based on certain principles, a mathematical model of the reality. The knowledge on mathematical models, which primarily consists of the following assumptions and approximations, is necessary for the system to support designers in analytical tasks with these methods:

1. Assumptions on the structure of the world where the process of interest occurs.
2. Assumptions on the relations among different parts of the structure.
3. Approximations necessary to simulate the process as a sequence of numerical operations.

It is possible to include these types of knowledge in the same framework as design objects, but mathematical models should be
defined separately from any particular design object, though each model is aimed to represent designer’s view of some design object. It is partly because a single mathematical model can be used to represent several realizable structures, but mainly because a mathematical model can operate sometimes as a conceptual substance independent of any design object. A particular design object and the mathematical model as its expression can be related by an interface class definition, where the constraints between them are declared.

IV. HIERARCHICAL CONTEXT MANAGEMENT

Observation of design instances reveals the following features of the design exploration process. Firstly, design is not completed instantaneously by a single step of task, but it is progressive work made up of many steps of tasks. Secondarily, several branches will grow gradually in this process when a new alternative of design specification or exploration strategy is adopted. Comparisons and selections among these branches are repeated until some plausible design solution is found. Thirdly, when the design requirements are refined or revised, the selections already made may become unfavorable and designer’s focus may be switched to another branch.

Figure 3 graphically shows an example of progressive design exploration history observed in a nuclear design of an axially heterogeneous fast breeder reactor\(^{(9)}\). In this example many design alternatives indicated by tree nodes were created in the course of parametric surveys on three attributes: blanket thickness, blanket aspect ratio and blanket position. Each set of sibling nodes was used in a parametric survey.

In order to deal with these aspects of the design exploration process, the design support system should have the functions as described below:

1. Different instances of design objects should be represented and investigated concurrently in the design description document, and they should be distinguishable each other, including outcomes of system functions on them.
2. The structure of a design exploration history should be reflected in the knowledge representation of the design description document.
3. The focus can be fixed onto any context in a design exploration history by designer’s request, and the all internal states should be recovered as they were when the focus returned to an old context.

An assumption denotes an undecided premise for inferences in problem solving, and a context is determined by a consistent set of assumptions. ATMS proposed by de Kleer is an intelligent database, which knows what proposition is valid in what context. With the help of ATMS problem solvers can carry out reasoning over multiple contexts, among which some inconsistency may exist. In ATMS each proposition is kept together with its foundation of validity, which consists of two types of information: justification and environment. Justification is a set of other propositions from which the proposition has been derived. Environment, which is calculated from the justification, is a minimal set of assumptions supporting the validity of the proposition. A disjunctive set of environments is called a label, and labels are recalculated when a problem solver creates a new justification. It is relatively easy to judge whether or not each proposition is justified in a particular context by investigating the label.
ATMS provides the capability of representing multiple contexts, handling contradiction, switching the focussed context, comparing data between different contexts, and avoiding redundant calculations.

Since no specific method of context management is presupposed originally in the basic architecture of ATMS, one needs some proper management method for design. In the design support system of this work, contexts are named and maintained in a hierarchical structure like the UNIX file system. This method is useful to represent a design exploration history such as illustrated in the example of Fig. 3. A new context is created as a child of some other context, when the design exploration process proceeds to the next design stage. It is proper to think that creation of a new design alternative or refinement of design specifications normally starts a new stage. The root view is provided by the system as the common ancestor of every context. The context on which designer’s focus is fixed at the moment is called the current view. Every support subsystem works just on the propositions in the current view.

The designer can create, delete and copy contexts and move the current view to another context, if s/he wishes. When the current view has been changed, only the propositions justified in the new context are made visible to the design support subsystems, and the suspended tasks in the new current view are reactivated. Particular propositions of different contexts can be made visible to the designer at the same time to make comparison.

The context management architecture is also useful to repair the consistency of the design description document. When a problem solver has found some contradiction and informed it as a nogood node, ATMS calculates its label. Then the current context is split into two contexts so that the assumptions in the nogood environment do not appear in the same context. Distribution of the assumptions can be determined automatically for a binary nogood, which is supported by just two assumptions, but designer’s choice is required otherwise.

V. DESIGN PLANNING

In the conventional use of computers, the problem is decomposed into several subproblems and programs are developed and used to solve some subproblems which are amenable to numerical and algorithmic operations. Such an application program is called a function oriented system. Since the use of computers in this manner is practical and intensive in nuclear reactor design, it is inevitable to incorporate function oriented systems into the nuclear reactor design support system. The simplest way is to add application programs as special purpose support subsystems and to prepare the user commands to execute them on the specified data in the design description document. In this case, the designer is responsible for selection of a proper subsystem and judgement of execution timing. In this work an attempt was made to support these activities by AI techniques.

The purpose of using a function oriented subsystem is to obtain some attributes of design objects for the critique stage in propose-critique-modify cycle, which is a relevant method of design. Consequently, usefulness of a subsystem depends on what attributes are required and whether or not the required attributes are obtainable with the subsystem. The answer to the former question is related to design requirements and exploration strategy, which the designer will finally decide. Selection of a proper subsystem will be constrained also by several contextual factors such as trade-off between accuracy and cost.

Function oriented subsystems are normally used in the order: preparation of particular input data, execution of algorithmic procedure, and then collection of output. Since the input data should be ready before starting the subsystem, execution of other subsystems in advance may be necessary for preparation of the input data. In nuclear reactor design, for example, the power distribution is required before the heat removal is evaluated. Such a prerequisite determines hierarchical order of task executions.

In order to control design activities con-
considering the factors described so far, the knowledge on the task which each function oriented subsystem can do is to be declared in the design knowledge base. An example in Fig. 4 shows the representation scheme of this type of knowledge.

```plaintext
method burnup_2D(X,C) has
  purpose
  C:burnup_reactivity <- C isa fbr_core;
  .......... procedure
  run(cit(burnup),X);
  component
  [C,X] isa core_2D_mapping;
  precondition
  X:sec_file,
  .......... X&zone(1):xsec_set for (1=1,X:zones);
  effect
  C:excess_reactivity,
  .......... X:ave_power_den;
  control
  priority=120, accuracy=50, cost=200;
end.
```

Fig. 4 Example of design task definition

Execution of design support subsystems is controlled basically by the blackboard model of control. The design description document is separated into two parts, object DDD and control DDD, to keep the knowledge on design objects and on system control respectively. The problems of system control are solved in the control DDD. Consistency maintenance and context management are applied on the control DDD as well as the object DDD. When the control system finds a task achievable by some design support subsystem, a record called KSAR (Knowledge Source Activation Record) is created on the control DDD. In the conventional blackboard system, an agenda of KSARs is made and the task with the highest priority is executed in each execution cycle. In the present system, however, a nonlinear design planner will generate more complicated execution plans referring to the task definitions of subsystems\(^{(11)}\). The tasks of the design planner are as follows:

Selection of a method by means-ends analysis, where the purposes of the defined tasks are matched with the design goal claimed by the designer or by the system. Whether or not the proposed tasks are applicable in the current context is investigated. Trade-off between accuracy and cost is considered to determine the priority of the proposals. This action generates a KSAR for the selected task.

Creation of a model object which is necessary to carry out the invoked task. An instance of the model object is created, and it is specified in accordance with the design object under evaluation.

Declaration of new goals to achieve the prerequisites of the invoked task. This action generates a hierarchy of the design plan.

Arrangement of the tasks in the same level of the hierarchy. The least commitment strategy is adopted in design planning, i.e. the system is allowed to arrange the tasks only when sufficient information for the arrangement has been obtained.

Elimination of redundancies in the design plan. Redundant tasks are detected and eliminated by the design planner.

Execution of the tasks whose prerequisites have all been satisfied. An agenda is made by collecting the KSARs only in the lowest level of the design plan hierarchy, and the task with the highest priority is executed. If some of the prerequisites are left unachieved due to lack of knowledge, the task is suspended till the designer will supply sufficient assumptions for achieving them. The system executes the computational procedure by either internal or external computer process.

There are three possibilities that a design support subsystem is activated: the designer issues an execution command, the designer claims a preliminary design goal, or the system makes a subplan. Designer's intention on design exploration strategy is informed to the system by creating assumptions in the control DDD. The design support subsystem possessing the functions described above can support the designer in solving considerably the problems of design planning. Since it is impossible, however, to represent every planning knowledge in a symbolic form, the sys-
tem cannot always make proper decisions on planning. To supplement this inability, the system follows the least commitment strategy and it is ready for designer’s intervention.

VI. IMPLEMENTATION AND EXAMPLE

A nuclear reactor design support system based on the framework proposed in this work was implemented on a UNIX workstation using K-Prolog language. A design knowledge base for nuclear design of fast reactors was constructed. Some instances of LMFBR design from literatures were reproduced using this system to demonstrate that the architecture adopted here can represent the principal aspects of the design process.

The context tree which represents the design exploration history already shown in Fig. 3 was successfully constructed by the multiple context management architecture. The design planner was able to schedule design activities of nuclear design codes properly. Figure 5 shows an example of the design plans created by the system, which is for evaluating control rod worth by the perturbation theory.

Fig. 5 Design plan for evaluating control rod worth (KSAR identification numbers, method names and arities are shown as a procedural network.)

VII. CONCLUSION

A scientific theory of design will provide a good framework for development of practical design support systems. The authors proposed an architecture of AI-based nuclear reactor design support systems based on an explorative abduction model of design. The primary features of this architecture are as follows:

1. The design knowledge on mathematical models as well as design objects was represented as hierarchical class definitions.
2. Multiple context management is necessary in design, and ATMS was extended so that it can represent the progressive structure of design process.
3. Nonlinear design planning method was adopted in addition to the blackboard model of control for planning design activities using the knowledge on tasks.

The approach outlined in this paper is not yet complete for full utilization of AI techniques in design, and the system is still under improvement. The approach relies on the assumption that designers can always externalize their intention to set a preliminary design goal. Since it is not always the case, prediction of their intention from some evidence expressed implicitly is left for the future research. A method to interpret numerical results in qualitative terms based on mathematical models is another research topic.

ACKNOWLEDGMENTS

The authors would like to thank Tim Smithers of University of Edinburgh for valuable discussions. This work has been supported by the Japan Atomic Power Co. and the Japan Atomic Energy Research Institute.

REFERENCES

(4) CHANDRASEKARAN, B.: ibid., 11[4], 59 (1990).