AI APPROACH TO VLSI ROUTING PROBLEM

K. Mitsumoto, H. Mori, T. Fujita and S. Goto

C&C Systems Research Laboratories
NEC Corporation
Kawasaki 213,Japan

ABSTRACT

This paper describes a new interactive routing system based on artificial intelligence technique. In the VLSI routing problem, most of the time is spent on the interactive design operations to get the complete connection after automatic routing. An expert designer has to delete, modify or draw routing patterns on the display terminals by exploiting his own specific knowledge and experience. This interactive process is done in a try and error manner, and usually takes a lot of design time. The system proposed here, adopts a rule-based method and enables a designer to relieve the burden from complicated design operations. This system accepts the designer's knowledge in clausal form of the first order predicate logic, Prolog language. This rule-based routing system has been developed to solve a large scale real problem of a few thousand gate VLSI, and showed quite promising result.

1. INTRODUCTION

A routing problem is to determine a connecting path meeting all physical and electrical constraints for each given net. From a computational point of view, a routing problem is considered to be a hard combinatorial problem. It is far too difficult to complete the whole design by only using automatic programs. The total design process consists of automatic and manual design. In most practical cases, a large portion of the design time is spent on manual design, such as drawing, error checking and correction after automatic design.

In these years, highly interactive CAD systems have been developed (1,2,3) and the layout design time is reduced in a great deal. However, its successful operation is a fully dependent on the designer's ability. The expert designer carries out the design by exploiting his own specific knowledge. The knowledge specifies a certain operation in accordance with a specific situation, which is represented in a form of the design objects, their properties, and logical relations among them. Recently, several new CAD systems, based on artificial intelligence technique, have been reported (4,5,6). However, it seems that there remain some problems to solve the large scale real problems. The system proposed here is focused on applying the designer's specific knowledge into the design process for practical routing design problems. The rule is a set of knowledge acquired through the expert's design experience and it quickly leads to an appropriate design goal. Here, the rule is expressed in Prolog language and stored in the knowledge base. The Prolog interpreter (7) compiles the rule and makes inference on the knowledge. The inference result is transformed into a certain procedure sequence, a set of FORTRAN subroutines. In this way, the procedure sequence is executed in a high speed manner.

2. AI APPROACH TO ROUTING PROBLEM

In a large scale routing problem, a large portion of the design time is spent on manual design for unconnected nets, after automatic routing. In order to reduce the manual design time in interactive systems, it is important to classify which work a human being should be responsible for and which part a computer program should do. A human being is superior to a computer program in understanding and recognizing the design situation and planning strategy. On the other hand, a computer can achieve high speed processing in simple data manipulation. We have developed integrated CAD systems, where the optimum exploitation of human intelligence and the computer's high speed processing is realized by integrating automatic and interactive functions (2,3).

However, a large portion of the design time is spent on the interactive design operation in completing connection of unconnected nets. The design activity depends upon the designer's ability. Therefore, when a part of the interactive design process is replaced by computer programs, the design time will be reduced with more intelligent man-machine inter-face. In order to such an intelligent system, it is essential to have a mechanism

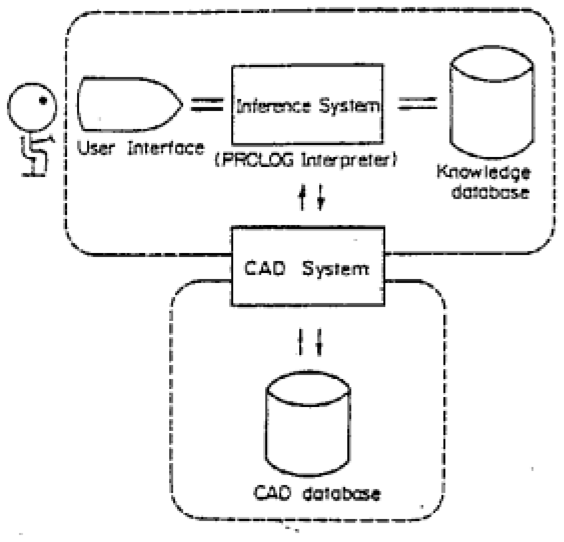


Fig.1 System Configuration

which represents and utilities the designer's knowledge. In the proposed system, the designer's knowledge is represented as rules in Prolog language, and the design goal is reached by exploiting those rules.

3. SYSTEM CONFIGURATION

The Figure 1 shows a system configuration. The system includes two kinds of databases, knowledge database and CAD database. The knowledge database stores a certain amount of design knowledge about specific routing situations : properties of design objects, relations among the objects, and rules for guiding problem solving. The CAD database, which is a conventional database, contains physical information about the design objectives.

At present, the inference system is Prolog interpreter. Prolog language has inference mechanism in itself. And the designer's knowledge is written directly in Prolog language. In this language, rules can be considered as a set of knowledge and they are represented as a sequence of predicates. For linking to the existing CAD system, our Prolog interpreter allows predicates that correspond to procedures in the CAD system. The procedures manipulate physical data on the CAD database and execute design primitives. High speed processing is realized in this way.

Rules in Prolog programs make a hierarchical structure, and rules at the top level of the hierarchy can be considered as commands to the system. As the designer inputs a rule name, the interpreter compiles the program and produce a certain procedure sequence. This sequence simulates the designer's operation. These mechanisms enable the designer to solve a practical routing problem within a reasonable time.

4. PROLOG INTERPRETER

In case of Prolog, the form of a term is expressed by the Horn clause subset of logic. Prolog is one of languages suitable for representing knowledge and developing rule-based systems. Prolog is characterized by the following three points ; the first point is to have a basic inference mechanism in itself. The second point is to be able to manipulate structured data objects, such as lists and trees, applying pattern matching. The Third point is to access program and data in the same way and be mixed together.

In general, the mechanism of a rule-based system is considered to a simple repetition of selecting and executing rules. The rule-based system must operated pattern matching process whenever such a repetition would occur. Therefore, the more rules the rule-based system has in the knowledge database, the slower it becomes in inference. In Prolog programming, data are expressed by clauses with empty body(unit clauses). The rule-based system has to store programs and many rules in the knowledge database to solve problems with a large amount of data. This causes the slower inference speed.

To cope with this problem, procedural knowledge should be written in more adequate language such as FORTRAN. A lot of knowledge in the design process is procedural knowledge, that is, how to manipulate physical data and which design primitives should be executed. We have developed Prolog interpreter which has the additional function so as to link to FORTRAN language. Design data in the CAD database of an existing CAD system can be accessed by FORTRAN routines. This is an efficient method to take advantage of accumulated programs in an existing CAD system. Predicates corresponding to FORTRAN routines can be defined in the Prolog interpreter. Those routines are procedures in the CAD system, and they can access data in the CAD database. These predicates are called external functional predicates . This additional function enables the system to realize high speed processing. The interpreter also provides standard built-in predicates.

5. DESIGN KNOWLEDGE REPRESENTATION

Automatic routers cannot always give a complete routing, since a net routing is often blocked by a wiring pattern which has been routed early. In conventional CAD systems, the designer does two kinds of manipulations in order to accomplish 100% routing after automatic routing. One is to modify existing wiring patterns ; altering wire routes, removing detour patterns, eliminating via holes. Fig. 2 shows these examples. The other kind of manipulations is to connect unconnected nets.

The designer's operations are based on his knowledge aquired through his experiences. In order to express the designer's knowledge as sequence of rules, we must analize the designer's operation.

In case of completing unconnected nets, the designer will find the reason why nets are failed in connectivity. The reasons are classified into the following two cases.

(1) A signal net is blocked near or around its pins by other nets.

(2) A signal net is blocked on the routing channel area far from the pins.

The first case is explained in detail. Fig. 3 shows topological relations between blocking wiring patterns and pins. Unconnected nets are indicated by chain lines. Fig.3 (a) shows a situation wherein a signal pin is placed on upper or lower side of a routing area, and it is blocked by other net wiring pattern on the neighbor grid. Fig.3 (b) shows the situation wherein a signal pin is on the right or left side.

The designer represents his knowledge to solve the situation of Fig.3 (a), as follows.

```
BLOCKING(*NET_U,*NET_B) :-
  GETPIN(*NET_U,*LAY,*XS,*YS),
  EQ(*LAY,1),
  DIRECTION(*YS,*DIR),
  $GRIDSEARCH(*DIR,*LAY,*XS,*YS,*NET_B).
```

Variables are represented as capital letters headed by '*', and the external functional predicates are headed by '$'. *NET_U is an unconnected signal net number, *NET_B is a blocking one. The first predicate takes out two coordinates of an unconnected pin and a layer number. *XS is a X-coordinate, *YS is a Y-coordinate, *LAY is a layer number. The second predicate succeeds when a pin is on the first layer. The third predicate determines a direction where a blocking wiring pattern may pass through. *DIR is a direction. The last predicate denote a procedure which searches for a blocking wiring pattern in a given direction and finds a net number of the blocking wiring pattern.

In case of the situation wherein a net routing is blocked like shown in Fig.3, in most cases, the designer can connect an unconnected net by a following sequence of operations : deleting a blocking wiring pattern, connecting an unconnected net, and connecting the deleted net. Such a process are described by a clause as follows.

```
RULE30(*NET_U) :-
    BLOCKING(*NET_U,*NET_B),
    $DELETE(*NET_B),
    $CONNECT(*NET_U),
    $CONNECT(*NET_B).
```

The second predicate executes deleting a specific part of a wiring pattern, where each end point of the deleting part is either a pin or a branch. The last predicate executes finding an error free path always guaranteed without any redundancy.

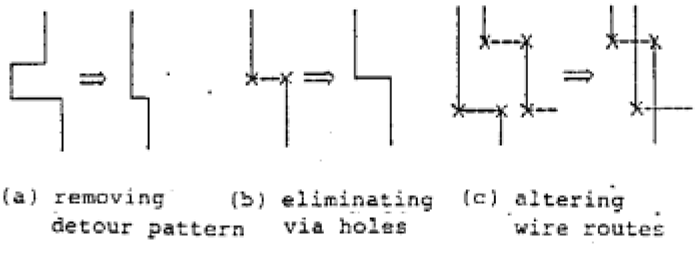


Fig.2 Modifying Wiring Segment

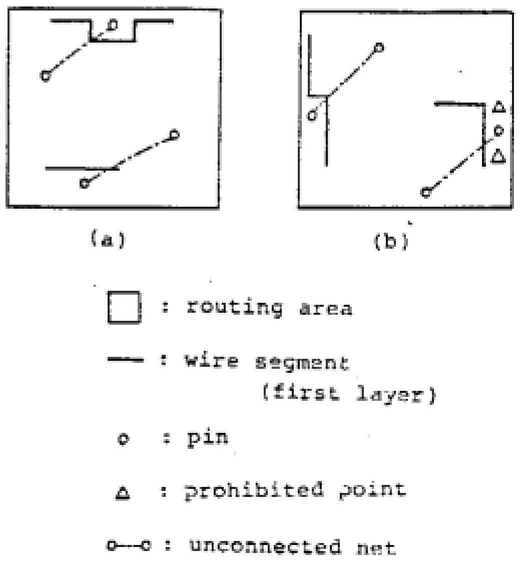


Fig.3 Blocking Wiring Situation

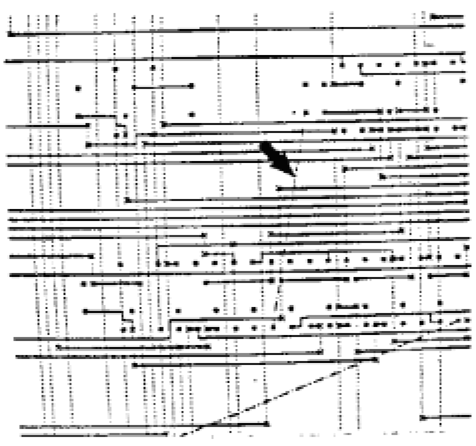


Fig.4 Unconnected net

:-RULE30(NET_U).

As the designer gives the system this goal clause, the system can connect an unconnected net NET_U. Fig.4 shows an unconnected net and Fig.5 shows the result which the system executes the goal clause.

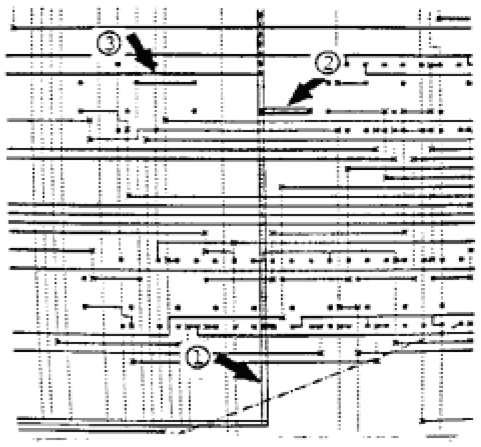
The above clause is a rule which represents one of the designer's knowledge. The rule simulates the designer's operations to achieve complete net connectivity. When the designer hits upon a good strategy to solve the situation, he can add new rules to the knowledge database and use them, while he is designing.

By accumulating more rules representing the designer's knowledge, corresponding to various design situations, more complex routing problems can be solve in a short time.

6. CONCLUSION

This paper presents a new interactive routing system based on artificial intelligence technique. The expert designer can store his own specific knowledge in the system and use them in the design process. The proposed rule-based system achieved an acceptable interaction speed and a reasonable result. For 2100 gate Master-Slice LSI routing problem, this system could make the complete net connection automatically within half an hour by applying several rules, whose paths could not be found by automatic routers.

By accumulating more rules, the system will become more intelligent and be able to solve more difficult problem. An essential difference between human beings and a computer program is considered to be an intuitive ability for reaching an appropriate goal. In order to achieve fast and high quality design, a CAD system should have a mechanism with the designer's knowledge from now.



① blocking wiring pattern
② connecting path of an unconnected net
③ connecting path of the deleted net

Fig.5 Execution Result

AKNOWLEDGEMENT

The authors would like to thank Drs. Y. Kato, T.Mikami and M.Naniwada for their helpful encouragement, as well as thanking everybody working on the development of the system for his help.

REFERENCES

(1) F.D.Kinner,"Interactive wiring system", Proc. of 17th Design Automation Conference, pp.296-308: 1980.
(2) H.Mori, T.Fujita, M.Annaka, S.Goto and T.Ohtsuki, "Advanced Interactive Lay-out Design System for Printed Wiring Boards", in "Hardware and Software Concepts in VLSI", ed. by G. Rabbat, pp.495-523, Van Nostrand Reinhold Company Inc.,: 1983.
(3) S.Goto, T.Matsuda, K.Takamisawa, T.Fujita, H.Mizumura, H.Nakamura and K.Kitajima, "LAMBDA, an integrated master-slice LSI CAD system", INTEGRATION, vol.1, no.1, pp.53-69, North Holland Pub. Co.,: 1983.
(4) G. L. Steel, Jr. and G. J. Sussman, "Constraints", Artificial Intelligence, vol. 14, pp.1-39 :1980.
(5) J.mcDermott, "Domain Knowledge and the Design Process", Proc. of the 18th Design Automation Conference, pp.580-588: 1981.
(6) H.Brown and M.Stefik, "Palladio : An Expert Assistant for Integrated Circuit Design", Heuristic Programming Project Report HPP-82-5, Stanford: 1982.
(7) W.F.Clocksin and C.S.mellish, "Programming in Prolog", Spring-verlag: 1981.