ICOT Technical Memorandom: TM-1073

TM-1073

A Legal Reasoning System on the Parallel Inference Machine

by
K. Nitta, K. Sakane, Y. Ohtake, S. Maeda,
M. Ono & H. Ohsaki

July, 1991

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03)3456-3191～5
Telex ICOT J32964

Institute for New Generation Computer Technology

A Legal Reasoning System on the Parallel Inference Machine (Extended Abstract)

Katsumi Nitta† Kiyokazu Sakane† Yoshihisa Ohtake†
Shigeru Maeda† Masayuki Ono†
Hiroshi Ohsaki‡

† Institute for New Generation Computer Technology
‡ Japan Information Processing Development Center
Tokyo, Japan

nitta@icot.or.jp

1 Introduction

A law consists of legal rules. As legal rules are given as logical sentences, they are easily represented by logical formulae, and legal conclusions can be drawn by deductive reasoning. However, legal rules also consist of legal concepts. Legal concepts are ambiguous and their strict meanings are not fixed until the rules are applied to actual facts. Therefore, to apply legal rules to actual facts, rule interpretation and matching between concepts and facts are both needed. To interpret legal rules and to match between concepts and facts, precedents (old cases) are often referred, and based on explanations of similar cases, an explanation which combines both legal concepts and facts is often constructed. We consider this step as a kind of case-based reasoning.

The **HELIC-II** system aims to draw all possible legal conclusions of a given case, and to show their explanations. It consists of two inference engines, such as a rule-based engine and a case-based engine. These two engines are implemented on the Multi-PSI developed by ICOT (Institute for New Generation Computer Technology) and draw conclusions in parallel.

In this paper, we introduce an overview of the HELIC-II system, and show how quickly this system reasons.

2 Multi-PSI and PIM

The experimental parallel inference machine **Multi-PSI** was developed in 1988. It consists of 64 processing elements (PEs) each of which is composed of a processor, a router

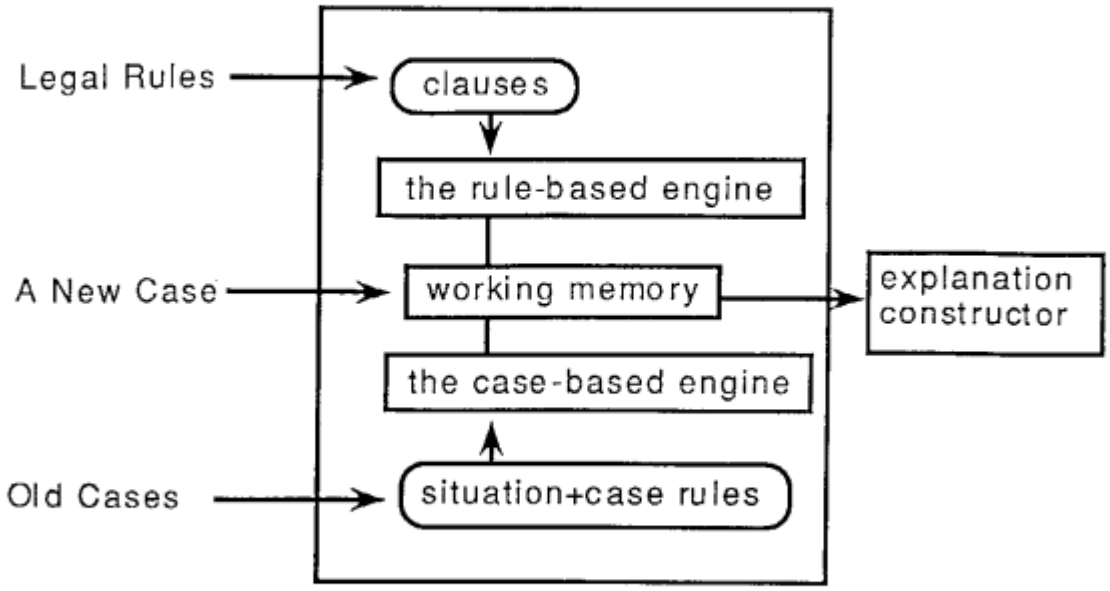

Figure 1: Two engines of HELIC-II

and its own memory. These PEs are loosely connected to construct a two dimensional mesh network.

This year, ICOT will complete the development of five kinds of **PIMs** (parallel inference machines). They are PIM/p, PIM/m, PIM/k, PIM/c and PIM/i. ICOT is developing these PIMs using different hardware technologies. For example, PIM/p consists of 64 clusters each of which is composed of 8 PEs and shared memory, and each cluster is connected by a hypercube network. PIM/m consists of 256 PEs, which are connected with a two dimensional network as the Multi-PSI.

3 A Parallel Legal Reasoning System: HELIC-II

3.1 Overview of HELIC-II

The target domain of HELIC-II system is the Penal Code of Japan. When a new case is given as a set of facts to this system, it lists up all possible crimes with their explanations by referring to both the Penal Code and old cases.

HELIC-II consists of two inference engines (Fig.1). The first one is a rule-based engine which handles legal rules. The legal rules of the Penal Code are represented as clauses. When legal concepts are given to the engine, it acts as a production engine and calculates all possible crimes.

The second engine is a case-based engine which retrieves similar cases from a case base, and produces legal concepts based on them. Each old case is represented as a situation and case rules. A situation is a sequence of actions and related information such as agents

and objects. A case rule represents the arguments of both parties. The LHS (left hand side) of a case rule is part of a situation, and the RHS (right hand side) is a legal concept insisted on by one of parties. Case rules are different from production rules because they don't contain variables and not fired by strict matching but by similarity based matching.

When a new case is given to HELIC-II as a form of situation, the system starts to reason. Usually, the original facts contain very few legal concepts. Therefore, at first, only a case-based engine reasons and produces legal concepts. Next, the rule-based engine works and produces legal conclusions. These results are gathered by an examination constructor, which then produces inference trees.

3.2 Representation of Legal Knowledge

The legal rules of the Penal Code are interpreted and represented as clauses. The following is a part of the rule related to the *crime of abandonment by a person responsible.*

$$\{Crime1, super, criminalAbandonment\},$$
$$\{Crime1, agent, A\}, \{Crime1, object, B\},$$
$$\{Resp, super, responsibilityToTakeCare\}, \{Resp, agent, A\}, \{Resp, object, B\},$$
$$\{Resp, time, Tr\}, \{Resp, time, Ta\}, \{Tr, during, Ta\}$$
$$\rightarrow$$
$$[\{Crime, super, criminalAbandonmentByPersonResponsible\},$$
$$\{Crime, agent, A\}, \{Crime, object, B\}, \{Crime, action, Abondon\}],$$
$$\{\{newAtom(Crime)\}\}.$$

Old cases are represented as both situations and case rules. The following is an example of the representation of a situation.

case $trafficAccident112$ *has*
  stages $act1, act2, act3$;
  story
    $during(act1, act2), after(act2, act3)$;
end.

stage $act1$ *has*
  events $drive1$;
  objects $john$;
end.

event $drive1$ *has*
  super $drive$;
  attributes
    $agent = john$;
end.

The meaning of this example is that the case "traffic accident 112" consists of three scenes such as $act1$, $act2$ and $act3$. $Act1$ occurred during $act2$, and $act3$ occurred after $act2$. During the stage of $act1$, one event "$drive1$" and one object "$john$" appear. The event "$drive1$" is a lower concept of "$drive$", and its agent is "$john$".

In this case, the prosecutor insisted that John is guilty, and the attorney insisted that he is not guilty. In the lawsuit, both parties tried to construct explanations to persuade the judge. Such explanations are represented as case rules. The following is a simplified example of a case rule.

$(rule001, penalCode218, [theoryRelatedToResponsibility],$
$\quad "comment",$
$\quad [\ [drive1,\ \ (agent = john, trivial),$
$\qquad\qquad\ (object = sportsCar1, trivial)],$
$\qquad [accident,\ \ (agent = john, trivial),$
$\qquad\qquad\ (cause = injury1, important)],$
$\qquad [injury1,\ \ (agent = ken, trivial)]]$
$\quad \rightarrow$
$\quad [make, [responsibilityToTakeCare1,$
$\qquad\qquad (nature = responsibilityToTakeCare),$
$\qquad\qquad (agent = ken)]]).$

The meaning of this case rule is that a traffic accident caused by John's driving injured Ken. Therefore, John had a responsibility of care to Ken.
The situation and LHS/RHS of case rules are compiled into a form of semantic networks before reasoning.

3.3 Reasoning by HELIC-II

The HELIC-II system consists of two inference engines - a rule-based engine and a case-based engine. When a new case is given to this system, each engine reasons in parallel on the Multi-PSI or the PIM.

3.3.1 A Rule-based Engine

The rule-based engine is an extension of the theorem prover MGTP [3]. The MGTP solves range restricted non-Horn problems by generating models. For example, let's take the following clauses.

C1: $p(X), s(X) \rightarrow false.$
C2: $q(X), s(Y) \rightarrow false.$
C3: $q(X) \rightarrow s(f(X)).$
C4: $r(X) \rightarrow s(X).$
C5: $p(X) \rightarrow q(X); r(X).$
C6: $true \rightarrow p(a), q(b).$

At first, the proof starts with null model $M0$. By applying C6, $M0$ is divided into $M1 = \{p(a)\}$ and $M2 = \{q(b)\}$. Then, by applying C5, $M1$ is divided into $M3 = \{p(a), q(a)\}$ and $M4 = \{p(a), r(a)\}$. Using C3, $M3$ is extended into $M5 = \{p(a), q(a), s(f(a))\}$. As $M5$ is discarded by C2, this branch is proven to fail. As all other branches are also proven to fail, this problem cannot be satisfied.

In MGTP, each branch of this proof tree is distributed to other PEs and is executed in parallel. Therefore, this engine is useful when the given problem generates lots of branches. In HELIC-II, the different theories of interpreting legal rules generate different branches.

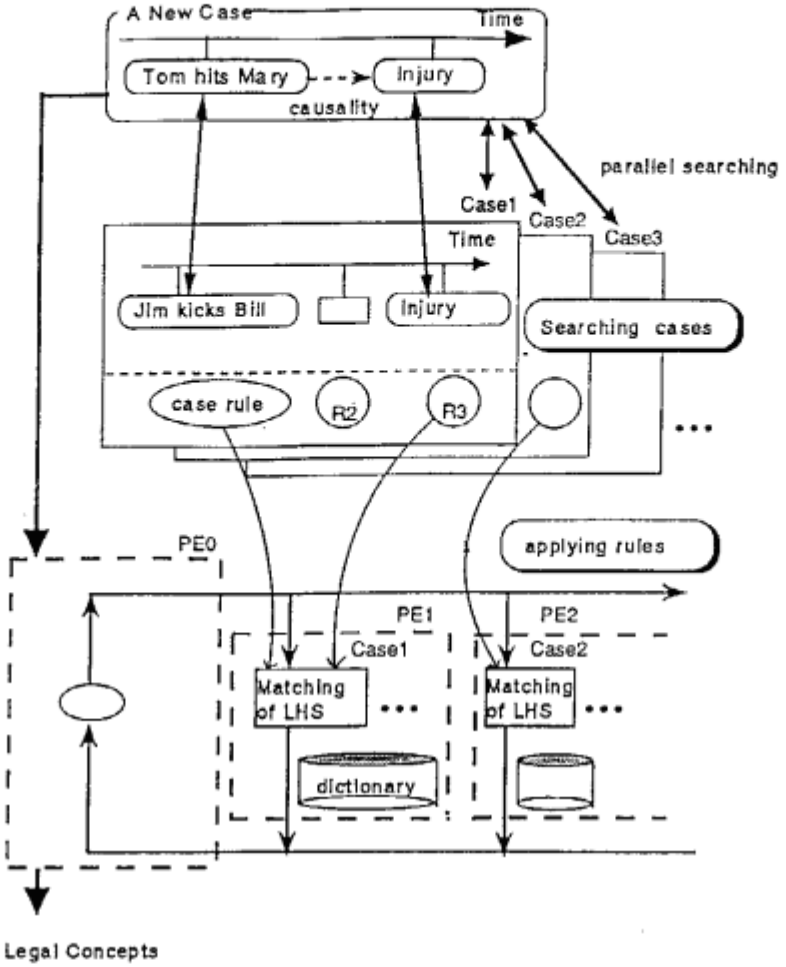

Figure 2: Reasoning using the case-based engine

3.3.2 A Case-based Engine

The reasoning of the case-based engine consists of two stages (Fig.2). When a new case is given as a form of case situation, as the first stage, the case-based engine searches for similar situations in a case base. Old cases in a case base are distributed to different PEs and the searching is executed in parallel. The similarity between two situations is measured by comparing the sequences of events of each situation. If two sequences have a long common subsequence, the situations are judged as similar situations. To obtain the common subsequence, the hierarchical relation among events is considered. Therefore, even if two events are different, they are judged as similar if their upper concept is the same.

When similar cases are selected from a case base, reasoning goes into the second stage. The second stage is an extension of a production system. At first, the engine compares the situation of a new case to the LHSes of case rules of selected cases. If similar LHSes are found, their RHSes are executed. The case-based engine repeats the cycle of matching and firing without conflict resolution.

To measure the degree of similarity, we use the hierarchical relation among concepts again. The degree of similarity is represented by the number of triplets ($\{object, attribute, value\}$) which are common to both semantic networks. To realize this mechanism, LHSes of case rules are compiled into a Rete-like network with a dictionary of hierarchical concepts in advance.

Case rules are distributed to different PEs and the matching of LHSes is executed in parallel. When LHSes of some case rule match the input data, such information is reported to the control process in PE0, and their RHSes are executed in the process.

4 An Example

The following is an example problem handled by HELIC-II.

> On a cold winter's day, Mary abandoned her son Tom on the street, because she was very poor. Tom was just 4 months old. Jim found Tom crying on the street, and started to drive Tom by car to the police station. However, Jim had an accident on the way to the police, and Tom was injured. Jim thought that Tom had died of the accident, and left Tom on the street. Then, Tom was frozen to death.

The problem is judging the crimes of Mary and Jim. The hard issues are the following two points.

1. Is there causality between Mary's action (abandonment) and Tom's death?
2. Is there causality between the traffic accident and Tom's death?

HELIC-II searched similar cases, and enumerated 6 crimes by Mary and Jim.

5 Performance of Parallel Execution

We measured the effectiveness of parallel execution on the multi-PSI. In this experiment, the number of clauses was about 200. The numbers of cases were 130 (table 2) and 16 (table 3). In table 2, we could not measure the speed where the PE number is less than 8 because of memory shortage. On the contrary, in table 3, the performance was not so good because the cost of data transfer becomes relatively expensive.
In the case of the case-based engine, the matching information is gathered by the control process in PE0. However, the concentration of traffic is not so serious, because the matching cost is very large comparing to the normal production system.

number of PEs	1	2	4	8	16	32	64
time(sec)	271	268	163	127	109	97	82
speed up	1.0	1.0	1.7	2.1	2.5	2.8	3.3

Table 1: Speed Up by the rule-based engine

number of PEs	1	2	4	8	16	32	64
time(sec)	-	-	-	1182	516	309	170
speed up	-	-	-	1.0	2.3	3.8	6.9

Table 2: Speed Up by the case-based engine (large data)

number of PEs	1	2	4	8	16	32	64
time(sec)	805	731	264	206	115	-	-
speed up	1.0	1.1	3.0	3.9	7.0	-	-

Table 3: Speed Up by the case-based engine (small data)

6 Conclusion

We presented an overview of the HELIC-II system. This system draws legal conclusions using legal rules and old cases with two inference engines. The rule-based engine based on MGTP is suitable for experimenting logical aspect of legal rules. The case-based engine is suitable for searching cases that are similar.
In the legal domain, as the numbers of rules and cases increase, a reasoning time often becomes a serious problem. We showed that HELIC-II reasons quickly by parallel execution on the Multi-PSI or PIM.
To proceed with this research, we would like to try several problems such as representation of situations, reasoning using several cases at a time, and theoretical analysis of performance.

References

[1] L.K.Branting : "Representing and Reusing Explanations of Legal Precedents" International Conference on Artificial Intelligence and Law(1989)
[2] L.K.Branting : "The Role of Explanation in Reasoning from Legal Precedents", Workshop on Case-Based Reasoning(1988)
[3] H.Fujita, et al : " A Model Generation Theorem Prover using Ramified-Stack Algorithm", ICOT TR-606(1990)
[4] K.Nitta, et al : "Legal Reasoning using Precedents and its Parallel Execution", SIGAI of IPSJ (1990) (in Japanese)