

TM-1071

法的推論における並列推論

前田 茂、小野 昌之、大嶽 能久
坂根 清和、大崎 宏、新田 克己

July, 1991

© 1991, ICOT

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

法的推論における並列推論

前田 茂 小野 昌之 大嶽 能久 坂根 清和 新田 克己

(財) 新世代コンピュータ技術開発機構

大崎 宏

(財) 日本情報処理開発協会

法的推論とは、法律を用いてなされる問題解決における推論方式をいう。法的推論においては、法律の条文を直接用いて問題解決を行うことは実際は困難である。なぜなら、法律の条文は抽象的過ぎるため具体的な問題に適用できないからである。そのため、事実認定や解釈を伴う推論が必要になる。この推論を実現するために、事例ベース推論 (Case-Based Reasoning: CBR) の技術が重要になる。つまり、現在の問題に対して類似の過去の事件の判決に含まれる判断を用いることによって、推論を行う。そして、その論理の補強として、条文を用いたルールベース推論を用う。両者が相補的に並列推論を行うシステム HELIC-II を開発したので報告する。

Cooperative Parallel Legal Reasoning System

Shigeru MAEDA Masayuki ONO Yoshihisa OHTAKE

Kiyokazu SAKANE Katsumi NITTA

Institute for New Generation Computer Technology

Mita Kokusai Building 21F, 1-4-28 Mita, Minato-ku, Tokyo 108

Hiroshi OHSAKI

Japan Information Processing Development Center

Kikai Syinkou Kaikan, 3-5-8 Sibakouen, Minato-ku, Tokyo 105

Legal reasoning is a kind of problem solving using statute law. However, it is difficult for it to directly use laws, because they are too abstract to apply to actual legal cases. Then problem of fact findings and statutory interpretation are arose and solving them are necessary. Case-Based Reasoning (CBR) is suitable for these kinds of problem. CBR uses judgments in them to make arguments for new case. On the other hand, Rule-Based Reasoning (RBR) makes logical reasoning using statute for justification of the arguments. We developed cooperative legal reasoning system HELIC-II using both CBR and RBR executing in parallel. Also We discuss efficiency of parallel execution.

1 はじめに

法的推論とは、裁判などの場面で、法律を用いて問題解決をするときに行う推論方式をいう。法的推論では、法の体系に従って三段論法などの論理的推論を行うことで問題解決を行っているように考えられるが、実際には法律の条文は、現実の問題に対して抽象的過ぎるため、直接適用することは困難である。そのため、与えられた事件に対する事実認定や、ある事実要素が法的概念に当てはまるかどうかを判定する解釈の問題を伴う推論を行う必要がある。

実際の裁判などでも、このような問題に対して過去の判例を利用することが行われる。すなわち、過去の類似の事件の判例の中で行われた判断を用いて新しい事件の判断を生成することである。したがって、このような推論は事例ベース推論 (CBR) によって行うのが適切である。

そして、CBR によって生成される判断が法的概念にまで抽象化された場合、法律の条文を用いたルールベース推論 (RBR) が CBR の論理の補強を行うために必要である。これは、CBR が最終結論を導けなかった場合やまた、条文による推論によって最終判断を行わなければいけない場合、CBR の結論の正当性の主張を行う場合などに有効である。したがって、法的推論では CBR と RBR の両者が相補的に動作する必要がある。

以下では、我々が開発した CBR と RBR との相補的並列推論システムである法的推論システムの概要について述べ、CBR, RBR それぞれの並列推論について述べる。最後に、法的推論システム HELIC-II を ICOT で開発された並列論理型言語 KL1 で記述し、並列推論マシン Multi-PSI 上で実行したので、その推論結果を評価する。

2 刑法における法的推論

刑法の分野においては、いわゆる罪刑法定主義の原則がある。これは、あらかじめ法律で規定されていない限り、いかなる行為も犯罪とされることはなく、したがって、それに対して刑罰を課せられることがないという原則である [阿部純二編 89]。そのため、最終的な判断を下すためには、刑法の条文による推論を行

わなければならない。

一方、条文の記述は抽象的な概念によって記述されているので、具体的事件を与えられた時に、これを直接適用することは一般に困難である。たとえば、刑法一九九条の殺人罪の記述は、

人を殺したるものは死刑または無期若しくは三年以上の懲役に処す

とあるが、現実の対象や行為が「人」や「殺す」という概念に当てはまるかどうかを判断する問題が残る。例として、胎児が人と考えられる時期、人を刺す行為の後、主体者の予期せぬ事件が起こった結果、その客体が死亡した時、それを殺人というか (因果関係、故意の問題) などの問題があげられる。

したがって、具体的事実と法的概念を結び付ける操作 (法的解釈の問題) が必要になる。法的推論では、この問題を解決するために、過去の判例を利用することが多い。過去の類似の事件でなされた判断を、現在の事件に適用することで、現在の事件の判断理由を生成し、法的概念と結び付けることができる (図 1)。これは、類似の事件からは類似の判断がなされるはずであるという経験的観測からなりたつものである。

このように、法的推論では、条文による推論と判例を用いた推論の両者が相補的に動作する必要がある。この推論モデルにしたがって、ICOT で開発された CBR による労働災害を扱う法的推論実験システム [新田ほか 90] を拡張し、条文を用いた論理的な推論も扱えるシステムを開発したので、概要を 4 章で述べる。

3 知識表現

[新田ほか 90] では、労働災害の事件の表現は、平面的な意味ネットワークで表されていた。ところが、刑法で扱う実際の事件は、ある時間の流れに沿って起こるものである。つまり、時間の流れの中のある時点によってイベントが起こったり、また、そのイベントによってある状態が引き起こされる。たとえば、以下のような問題が与えられた時に、この事件の記述を時系列に並べると図 2 のようになる。

問題:

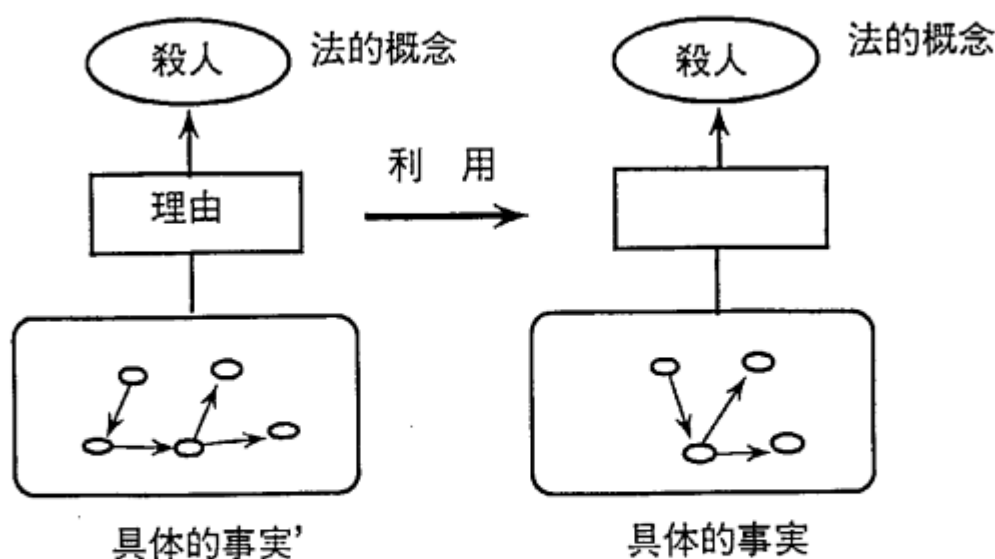


図 1: 判例を利用した法的解釈の生成

(司法試験問題 刑法 昭和 60 年度より改変)
([早稲田司法試験セミナー])

甲はまだ幼児の実子太郎の養育に疲れ、厳寒期のある夜、歩道上に太郎を捨てた。そこを乙が発見し、警察に送り届けようとして、自動車に乗せて運転中、誤って事故を起こし、太郎に瀕死の重症を負わせた。乙は、太郎が死んだものと思い、その場に太郎を置き去りにして逃走した。太郎はその夜凍死した。甲と乙の罪責を述べよ。

この時系列による問題表現を可能にするために、事件は、ある物理的・時間的に近接する状況を表すステージと、それらの間の区間をベースとする時間関係で表すことにした。たとえば、問題を表現すると以下のようなになる。

```
case 問題 has
  stages
    置き去り #1, 発見, 運ぶ, 事故, 置き去り #2;
  story
    after(置き去り #1, 発見),
    after(発見, 運ぶ),
    during(運ぶ, 事故),
    after(事故, 置き去り #2);
```

```
comment
  "Comments";
end.
```

ここで、stages の後に記述されるのがステージであり、story の後に、それらのステージ間の時間関係を記述する。ステージの記述は、一つの意味ネットワークで表す。意味ネットワークは、ノードとリンクからなり、ノードはイベントとオブジェクトからなる。dg ステージ 置き去り #1 を例にとると、以下のように表現する。

```
stage 置き去り #1 has
  events
    置き去りにする;
  objects
    甲, 太郎;
end.
```

リンクは、個々のノードに対する記述に含める。たとえば、イベント 置き去りにするは以下のように表す。

```
event 置き去りにする has
  super
    遺棄行為;
  attributes
```

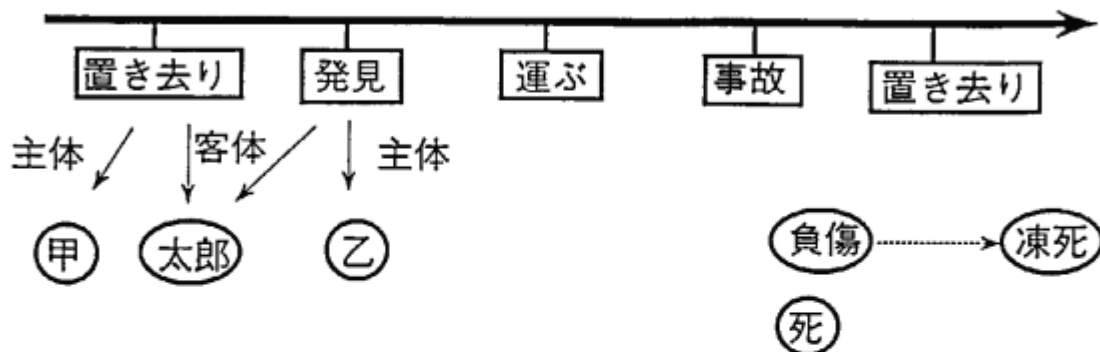


図 2: 事件の時系列による表現

主体 = 甲,
客体 = 太郎;

end.

刑法の分野では、しばしば問題になる重要な論点は、因果関係や故意、錯誤等の問題である。因果関係とは、ある行為とある状況の間に法的な因果関係が認められるかどうかの問題¹であり、故意は犯罪の意思があったかどうかの問題、錯誤はあらかじめ意図していた結果と実際の結果が異なる場合の問題である。

そのため、リンクには、意図、認識、因果などを用意した。これらのリンク先にはステージを取るが、それは、あらかじめ与えられるか、推論から導かれる。

4 システムアーキテクチャ

全体システムの概要を図 3 に示す。

システムは大きく分けて CBR と RBR の二つのモジュールからなる。今、システムに新しい事件が入力されると、それは二つのモジュールが共有するワーキングメモリー (WM) に書き込まれる。

CBR モジュールは、過去の一つの事件に対して、事件の記述と、その事件の判例から抽出した判断を事例ルールの形で記述したものを組にしたものを事例として多数事例ベースに蓄えている。

事件の記述としては、以下のようなものが入力されている。

¹物理的因果関係や社会常識的因果関係とは異なり、法的判断のために用いられる因果関係

殺意を持って麻縄で首を絞めると、被害者が身動きをしなくなったので、死亡したものと思った被告人が、被害者を海辺の砂の上に放置したところ、被害者が砂を吸い込んで死亡した。

(判例: 大判大一二・四・三〇集二 - 三七八)

この事件の争点は、被告人が首を絞めた行為と死亡との間の因果関係の認定であり、検察側と弁護側の主張が、後に述べるような事例ルールとして記述されている。

CBR モジュールでは、新しい事件が入力されると、まず、その事件と類似の事件を検索する。次に、その類似の事件に対してなされた判断を記述した事例ルールを、類似度を用いて検索し、それを与えられた事件に適用する。そして、その結果を WM に書き込む。以下は一般のプロダクションシステムとほぼ同様に、認知-行動サイクルを繰り返す。ここでは、一般に、推論が進むほど生成されるデータの抽象度は上がってくる。

RBR モジュールは、RBR のルールが適用できるようなデータが WM に書き込まれた時点で起動される。そして、条文ルールにしたがって論理的な推論を行う。

CBR モジュールと RBR モジュールは、それぞれ、並列に相補的に動作する。問題では、先の判例との類似性により、交通事故と太郎の死亡との間に因果関係が認められるかもしれない。しかし、それだけで

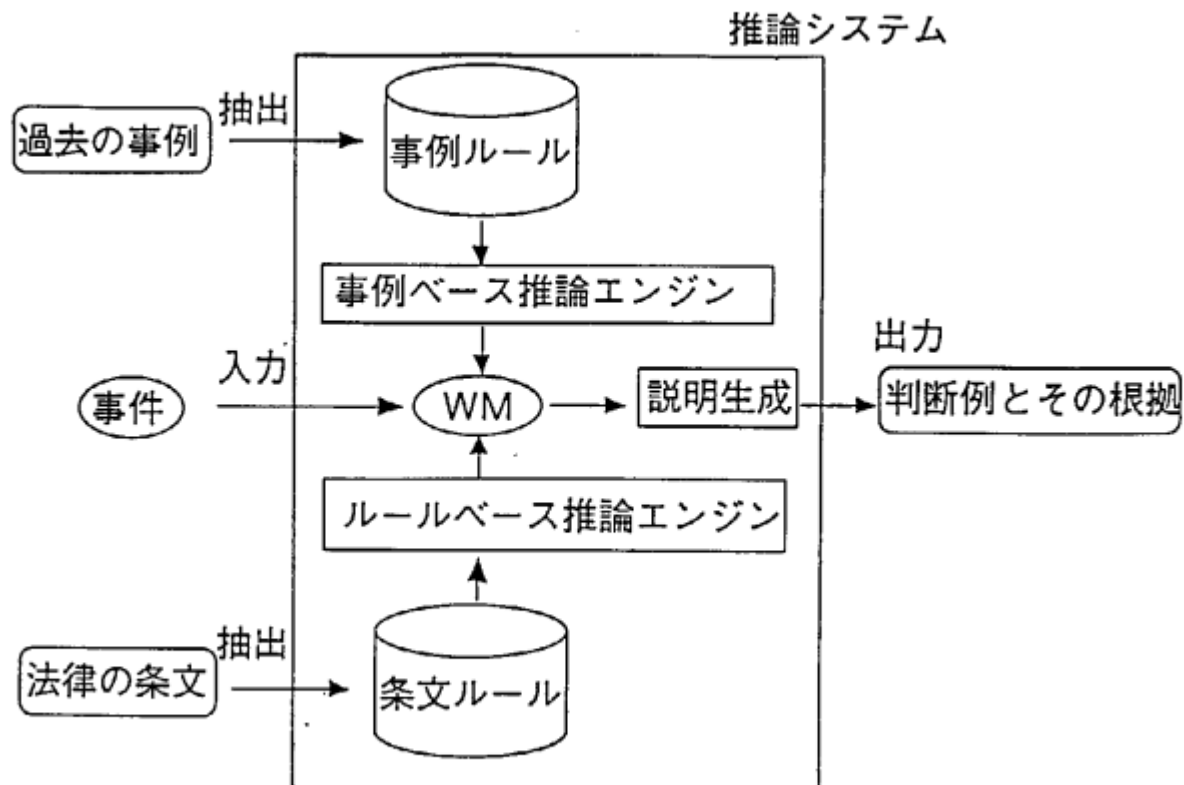


図 3: 法的推論システムのアーキテクチャー

は、罪責までは決定できない。したがって、この様な場合、RBRの推論により、以下のルールが発火し、罪責が決定する。

刑法第211条: 業務上過失致死傷罪

業務上必要なる注意を怠り、よって人を死傷に致したるものは、五年以下の懲役もしくは禁固または八千円以下の罰金に処す

最後に、両者がそれ以上新しいデータを作成しなくなった時点でシステムは動作を終了する。出力結果は、考えられる全ての罪責の候補と、それに対する説明、理由づけを推論木の形で表現したものである。

推論木の一例を図4に示す。黒いノードはCBRが生成した判断であり、白いノードはRBRが生成した判断である。本システムでは、先に述べたように、罪責の決定は、RBRが行ったものを採用する。図では、業務上過失致死罪が得られた。

5 事例ベース推論とその並列実行

事例ベース推論部の構成を図5に示す。事例ベース推論部は、類似事例照合部と類似論理構築部の二段階からなる。まず、類似事例照合部で事件の時間的流れの類似性によって類似の事件を選ぶ。次に、選ばれた事件のうち、事件が類似している部分で行われた判例の判断を用いて、新しい事件の判断を生成する。以下にその各々について詳しく述べる。

5.1 類似事例照合部

事例ベースの一事例は、事件の記述およびその事件に対して裁判でなされた判断(事例ルールで表現)で表されている。類似事例照合部は、そのうち、事件の記述だけを参照し、与えられた問題に類似の事件(の部分集合)を検索する。そして、類似論理構築部に、類似の事件(の部分集合)に当たる範囲でなされた判例の判断(事例ルール)を出力する。この方式の利点は以下の通りである。

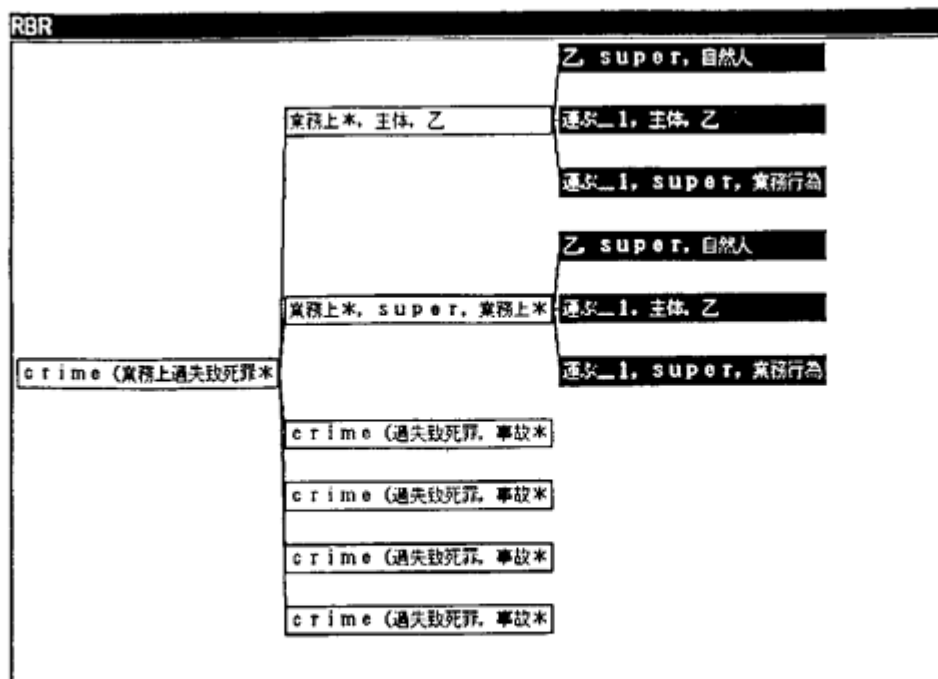


図 4: 推論結果の推論木

1. 多数の事例の中から事件の類似性によって類似の事件を選び、そこで判断に使う事例ルールを絞り込むことで、事例ルールの照合のコストを減らす。
2. 陽に事例ルールに記述されない条件を考慮に入れることができる。
3. 事件の時間的流れを考慮することによって、事例ルールの正当性を高めることができる。

2. は、判例の理由づけには陽に現れないが、暗黙の内に考慮されている条件までも含めて事例ルールを適用できることを表す。たとえば、先の問題では、事例ルールには、太郎が幼児であるから置き去りにされた時に凍死したというような条件が書かれていなかったとしても、類似事例照合部では、そのような条件も考慮に入れて類似の事件を選び、適切な事例ルールを選択することができる。

3. は、事件を時系列上でとらえることで、事件の時間的状況変化による判断を行うことができる。それは、ある同様に表現された行為でも、事件の流れの中では別の意味を持つという場合に有効である。たとえば、走るという行為でも、窃盗行為の後では逃走の意

味であり、殺人行為の前では追走かも知れないからである。

それでは、先の問題と判例を例にとって類似事例照合部の動作を詳しく説明する。判例を時系列にそって表すと図 6 のようになる。

まず、与えられた問題に対して、事例ベース中の事件のそれぞれに対して、事件の時系列が類似するかを並列に検索する。これは、各々の行為、状態を概念辞書を用いて類似照合し、問題、事件の中の照合された行為、状態の個数の大きいもの、および時間的に密接に照合したものを選ぶ。

例では、各々の行為、状態の照合は以下のようになったとする。

問題: 置き去り #1 事故 気絶 凍死
置き去り #2
事件: 置き去り 絞首 負傷 窒息死

この時、時系列の照合では以下の二つの可能性が得られたとする。

照合 1:
問題: 置き去り #1 → 気絶 → 凍死
事件: 置き去り → 負傷 → 窒息死

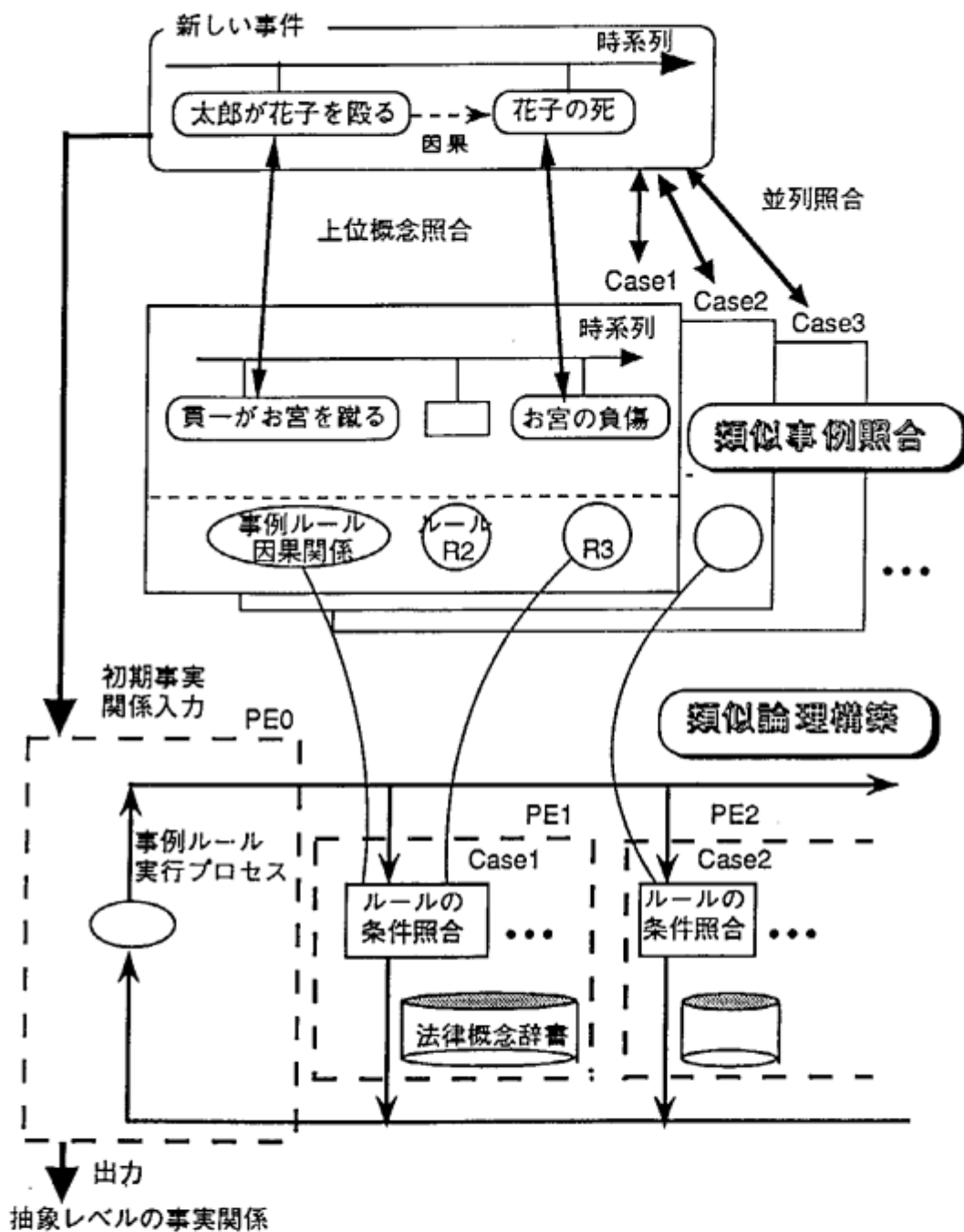


図 5: CBR の構成

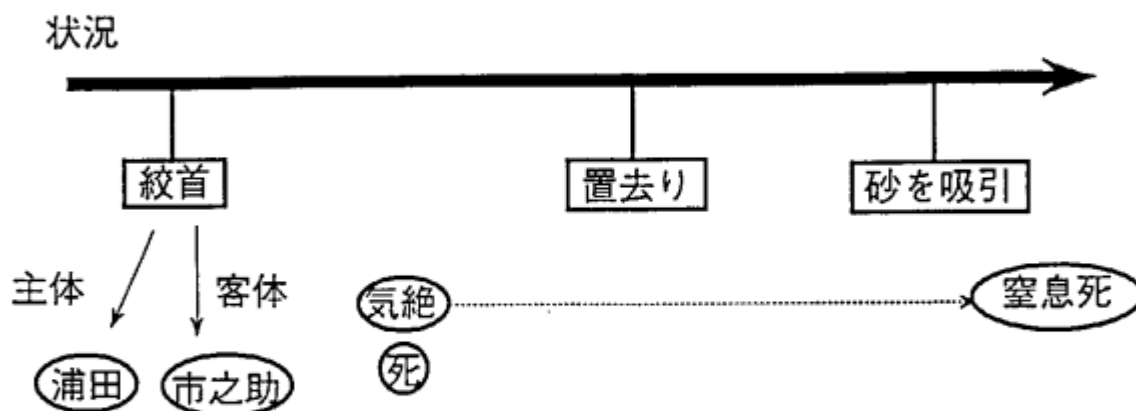


図 6: 判例の時系列による表現

照合 2:

問題: 事故 → 置き去り #2 → 気絶 → 凍死

事件: 絞首 → 置き去り → 負傷 → 窒息死

ここで、照合 2 の方が各々の照合する行為、状態の数、および、時系列の密接度が高いので選択される。したがって、絞首から窒息死までの行為に対して下された判断(事例ルール)が、問題の乙の行った行為の範囲に適用されるべく類似論理構築部に送られる。

ところで、事例は一般的なものではなく具体的事実を表すので、獲得は容易だが、適用範囲が狭い。したがって、推論能力を上げるためには、事例ベースに大量の事例が必要である。それにともない、類似事例照合部でも大量の事件の照合を行わなければならない。そのため、照合は、各事件に対して並列に行い、効率化をはかっている。

5.2 類似論理構築部

類似事例照合部で検索された事例ルールは、ストリーム並列で類似論理構築部に送られる。事例ルールは以下のような記述からなっている。

(rule001, 第 199 条, [因果関係条件説], "因果ありルール",

[[放置する,

(主体 = 被告, trivial),

(客体 = 被害者, trivial),

(結果 = 吸引する, important)],

[吸引する,

(主体 = 被害者, trivial),

(結果 = 窒息死, exact)],

[窒息死,

(主体 = 被害者, trivial)],

after(放置する, 窒息死)]

==>

[(make, [放置する, (因果 = 窒息死)])].

最初の 4 要素は、それぞれ、ルール ID、条文名、学説、コメントである。左辺部は、類似照合すべき行為、状態の記述または、時間関係の記述である。そのシンタックスは事件の記述のシンタックスにほぼなっている。さらに、ルールには、たどる上位概念を制限する limit、二つの照合した事実が異なることを記述する different が記述できる。

limit は、たとえば、人を殺すのは殺人であるが、犬を殺すのはそうではないというような場合を記述する時、limit(殺すの客体 #1, 自然人) の様に記述する。

認識と結果が異なる場合、錯誤の問題を問われることが多いが、この場合、different(認識 #1, 結果

#1) と記述すれば良い。また、他人のものをとると罪になるが、自分のものをとっても罪にならないなどという場合の記述にも用いることができる。

行為、状態記述の各要素の exact, important, trivial は、重要度で、判断に重要な要素を区別して記述できる。exact は、必ず類似照合しなければならないもの、important, trivial はそれぞれ、ルールが発火するのに重要なもの、それほど重要でないものである。右辺部は新しく付け加えられる判断を表す。ここでは、放置すると窒息死の間に因果関係が認められることを表す。

類似照合は、概念辞書を用いて行う。これは、二つの概念が同じ上位概念を持つ場合照合したとする。そして、二つの概念間の距離、および左辺部要素の重要度を考慮し類似度を計算する。そして、類似度がある一定の閾値を越え、時間関係が満たされたルールを発火させる。なお、時間関係の推論は、Allen の時間論理 [Allen 84] を非常に単純化したものを用いている。

概念辞書は PE (Processor Element) ごとに分散させて持ち、PE 間通信を抑えている。また、事例ルールは、判例を単位として複数の PE に分散させておき、照合は各 PE で並列に行われる。

判例を単位としたのは、判例内のルールはルールの連鎖を行いやすいので、実行サイクルごとのルールの発火数は均一になると予想されるためである。最後に、生成された判断は、WM に蓄えられる。

6 ルールベース推論とその並列実行

RBR では、条文に基づく推論を並列に行う。ここでは、ICOT で開発された並列定理証明システム MGTP (Model Generation Theorem Prover) [Fujita et al. 90] をルールベースエンジンとして用いている。MGTP とは、モデル生成法に基づく一階述語論理のための並列定理証明システムである。ここでは、MGTP の詳しい説明は省くが、CBR が生成するデータは、全てコンスタントであるので、MGTP との整合性は良い。

今、以下のようなルールが与えられたとする。

C1: true $\rightarrow$ p(a); q(b).

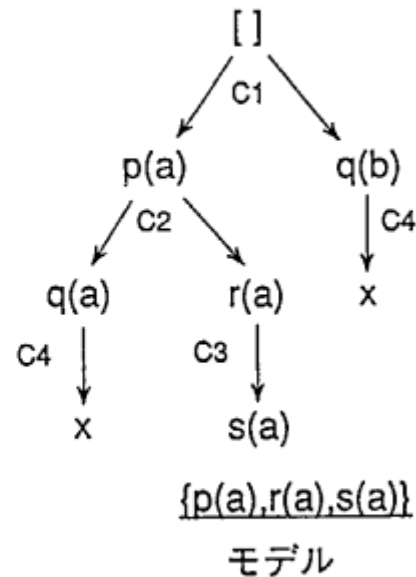


図 7: MGTP のモデル生成過程

C2: p(X) $\rightarrow$ q(X); r(X).

C3: r(X) $\rightarrow$ s(X).

C4: q(X) $\rightarrow$ false.

ここで、 $\rightarrow$ は implication を表し、; は or を表す。MGTP は、まず、implication の左辺に true を持ち、右辺に変数が現れない節を実行する。右辺に or が現れると、そこでモデルを分岐させ、並列に実行する。図 7 で枝別れしている部分がモデルの分岐であり、root から fail しない終端ノードまでがモデルとして証明される。

RBR では、まず、法律の条文を論理式で表す。現在、約 200 ルールが入力されている。そして、新しい判断が与えられるとその学説を調べ、異なる学説環境ごとにモデル分岐を行う。異なる学説環境では並列に推論が行われ、全ての学説環境での推論結果を出力する。

たとえば、因果関係条件説(行為と結果の間に、その行為がなかったら結果が生じなかったという関係があれば、因果関係があるとする説)をとり、A と B に結果が認められれば、因果が認められるというルールは以下のようなものである。

{ 推論環境, 因果関係条件説 }, {A, 結果, B}
 → {A, 因果, B}

また, RBR では, 簡単な矛盾の検出も行っている。同じ環境から矛盾する事実が導き出された場合, その環境を否定する。これは以下のようなルールによって行っている。

{A, 因果, B}, {A, 因果なし, B} → false.

最終的に矛盾の生じない学説環境で得られた罪責の候補全てが, 推論結果として出力される。

7 評価

実験システムを, ICOT で開発された並列論理型言語 KL1 で記述し, 並列推論マシン Multi-PSI 上で評価を行った。計測の都合上, CBR の計測結果を示す。まず, 事例を 16 用意し, PE 数 16 までの台数効果を計測した。台数効果が単純には PE 数に比例しないのは, 図 5 に示す通り, PE0 は事例ルール実行プロセス, 他の PE は類似事例照合プロセスという異なったプロセスが走っているためと, 1PE に割り当てられる事例数が少ないため, PE 間の処理量のばらつきが大きいと思われる。

PE 数	1	2	4	8	16
時間(sec)	804.7	730.7	264.2	205.5	114.9
台数効果	1.0	1.1	3.0	3.9	7.0

表 1: 少数事例での計測結果

次に実際の事例約 130 を入力した。メモリー容量の関係から, 1PE ではすでに実行できない規模であったので, 8PE から 64PE までで行った。このことから, 大規模な問題での並列処理の有効性が示された。

PE 数	8	16	32	64
時間(sec)	1181.7	516.2	309.1	170.4
台数効果	1.00	2.29	3.82	6.93

表 2: 大量事例での計測結果

ここでは 1 とは異なり, 事例数が多いので, PE 間の処理量のばらつきが少なくなり, ほぼ台数に比例する効果が現れている。なお, performance meter(各

PE の稼働状況を示す)を見た結果では, PE0 以外の類似照合部分の PE はほぼ 100% で稼働している。つまり, 照合のコストが処理の大部分を占めることが分かった。

8 まとめ

法的推論においては, 条文を用いた推論と, 判例を用いた推論の両者が相補的に行われる必要がある。また, 条文を用いた推論では, 対立する学説環境ごとの推論の必要性, 判例を用いた推論では, 大量の判例を扱うための手法の必要性がある。本システムではこれらの推論を並列に行うことにより, 効率的かつ大規模な問題も扱えるシステムになっている。

9 謝辞

本研究の機会を与えて頂いた ICOT 淵所長および内田研究部長に感謝致します。また, システム開発に対して, 数々の助言を頂いた ICOT の他研究室の諸氏に感謝致します。

参考文献

- [新田ほか 90] 事例を用いた法的推論とその並列化: 情報知識工学と人工知能研究会 69-5, 1990.
- [阿部純二編 89] 別冊法学セミナー 基本法コンメンタール [第四版] 刑法, 日本評論社, 1989.
- [早稲田司法試験セミナー] 司法試験 論文過去問シリーズ 刑法, 早稲田司法試験セミナー編, (株)早稲田経営出版, 1989.
- [Allen 84] "Towards a General Theory of Action and Time", Artificial Intelligence Vol.23 No2, 1984.
- [Fujita et al. 90] "A Model Generation Theorem Prover Using A Ramified-Stack Algorithm", ICOT TR-606, 1990.