修 士 論 文

集合的な状況の形式化とそれに基づく法的推論システム

指導教官　東条敏 助教授

北陸先端科学技術大学院大学
情報科学研究科情報処理学専攻

大森正則

1997 年 2 月 14 日

目 次

図 目 次

要 旨

法的推論システムにおいて, 適切な結論を推論するためには, 扱っている事例 を取り巻くさまざまな状況を考慮しなければならない. 法的な推論を行なう場合に, 推論主体 (主に人間) は推論に多様な側面から背景条件を持ち込む. また, 対象として時間状況を考慮しなければならない. 近年, この状況依存性を取り扱う研究がなされている. しかし, 今までの研究では, 状況をその中で成り立つ事実またはルールの集合としてモジュール的に扱うものがほとんどであり, 状況それ自身の性質について十分に考慮されていないと考える. そこで本研究の目的は状況の性質を扱うための形式的な状況のモデルを提案することである. 具体的には, 状況を単体で捉えず, より基本的な要素の集合として捉える. まず, 状況の素粒子なるものを仮定し (状況素と呼ぶ), その状況素の間の関係 (隣接関係と呼ぶ) で結ばれた状況素の集合として状況を表現する. このモデルの利点は 2 つある. 第 1 に, 状況がサポートするインフォンの集合 (サポートマップ) が状況素から構成的に計算によって求めることができることがある. 第 2 に状況素と隣接関係からなるグラフ上のパスであれば状況になるので, 非常に幅の広い世界構造を持つことができる.

このモデルを用いて, さまざまな状況が一様に表現できることを示す. その中で特に時間的な状況の表現を考察する. 時間情報の表現として, 我々は時間的な時点や時区間などを実体を表す状況を持ったモデルを与える. 状況はこのモデルにおいて, 相互に関係付けることによって時間的性質を保持する. このモデルの利点はインフォンが持っている時間情報を考えず, 状況間の時間関係だけを考慮すればよいことである. これにより, 有効な時間情報だけを推論に用いることができる.

我々はこのモデルに基づく言語・システムを作った. このシステムは導出機構だけでなく, 推論を行なう過程で動的に状況を決定する制約解消器を持っている. 最後に, このシステムが複雑な時間関係を持つ事例の上の推論においてうまく働くことを示す.

第 1 章

はじめに

1.1　背景・目的

1.1. 1　人工知能研究の立場から

人間の日常的な振舞いをできるだけ直観的にかつ簡潔に記述し, そこから導き出される事柄を推論できるようにしようとするのが, 人工知能研究の一つの目的である. そのための論理的基盤として, 命題論理, 述語論理があり, そのさらに発展として, 様相論理, 時制論理, 時間論理, 直観論理, 状況理論などが研究されてきた. 近年は, 様相論理での ‘世界 ‘や状況理論における ‘状況’ など考慮すべき場所によって命題の真偽が変化する ‘文脈依存’ や ‘状況依存’ を論理に取り込んでいく方向が主流になってきている. このような流れの中で情報や情報の流れとはどのようなものかということを正面から取り組もうとした状況理論の試みは称賛されるべきものである. この枠組で具体的なシステムを製作すれば, さまざまな状況を一様な形式で表現し, 記述の効率化, 推論機構の平坦化を達成することができる. このことは本研究の第 1 の目的にもなっている. すなわち, この一様性を失わない形式化を行なうことである. しかし, 状況理論は情報の捉え方は示してくれるが, 状況とは何か, サポート関係とは何かの明確な定義は与えていない. そのため, 近年, 状況理論の枠組で論理プログラミング言語を実装しようとする試みがなされているが, 状況理論の実装を行なう研究者それぞれの状況の解釈の違いから, さまざまな形式化が存在する. しかし, 今までの研究では, 状況をその中で成り立つ事実またはルールの集合としてモジュール的に扱うものがほとんどであり, 状況それ自身の性質について十分に考慮されていない

と考える. そこで本研究の第 2 の目的は状況の性質を扱うための形式的な状況のモデルを提案することである. 具体的には, 状況を単体で捉えず, より基本的な要素の集合として捉えることである.

1.1. 2　法的推論研究の立場から

法的推論の研究は人工知能のさまざまな技術要素を融合し統合するという人工知能研究の立場からも, 実社会へのコンピュータへの応用という意味からも注目されている研究分野である. この領域の中で法的知識の静的な側面と動的な側面, すなわち表現と推論の両方に対して近年, 状況依存性が重要な要素とみなされるようになってきた. 法的推論システムにおいて, 適切な結論を推論するためには, 扱っている事例 を取り巻くさまざまな状況を考慮しなければならない. 法的な推論を行なう場合に, 推論主体 (主に人間) は推論に多様な側面から背景条件を持ち込む. 例えば, その人間が属する領域における常識, 価値観, 彼がとる立場などである. また, 推論の対象とする事例においては時間的場面や, 空間的位置を状況としてみることもできる. 法的領域の事例の調査から, とりわけ時間的な状況を扱うことが必要であることが観察される. なぜなら, 時間的な情報は時間経過による知識の変化や, 事象間の時間的関係などが推論に影響を与える大きな要素となるからである. このような時間的要素を付加的に扱うのではなく, 直接的に扱うことが表現を明確にする上でも, 計算機に実装する上でも必要なことである. 本研究の形式的モデルに従って法的推論システムを構築する. このシステムの上で複雑な時間関係を持った事例を記述し, 推論を行なうことでモデルの有効性を示す

1.2　本論文の概要

本論文では,2章において状況理論の一般的概念と関連研究について説明する. さらに法的推論システムの概要と現状を説明し, 法的推論における種々の状況依存性について論じる 3章で, 2章で考察した状況の取扱いの考え方に基づいたシステムの形式的モデルをで与える. 特に, 状況による時間情報の表現について 3. 3節で述べる.

最後に, 3章の集合的状況モデルに基づくシステムの実装を行ない, 実際の事例を記述し, 推論の実験によってシステムを評価する (4章).

第 2 章

状況依存性と法的推論

この章では状況依存性がどのように法的推論のさまざまな側面において考慮されなければならないかについて論じる. はじめに状況依存性の表現として状況理論とその実装例について説明し, その理論を用いて法的推論における状況依存性について述べる

2.1 状況依存性の表現と推論

この節では状況依存性を扱う考え方としての基盤として状況理論を取り上げ, 扱っている諸問題について説明する. また, 本論文で時間の扱いを重点的に論じることから時間に関する論理についてもサーベイを行なった.

2.1.1 状況理論

状況理論は時と場合により発話や言明の意味が異なることを背景にある**状況**を陽に扱うことにより捉えようとする理論である. 状況理論は Barwise[1] ではじめて世に出て以来, さまざまな形式に分化した.

状況理論では状況 (論理学の世界で世界, 文脈と呼ばれるものと同等である) に依存して述語が成り立つ成り立たないが決まる. つまり述語それ自体には真偽を割り付けることはできない.

ここでいう状況は Situation Calculus([19]) における世界の時間的断面という捉え方とはことなる. Stuation Calculus は時間によって区切られる全体的な状態のみを扱っており, 時間的な側面しか捉えてなかった. また, ある時間に成り立っている事実と別の時間に

成り立っている事実との間の関係を記述すること (すなわち時間的にメタな関係の記述) や, 状況それ自身を関係の要素として表現することなども行なえない. 状況理論における状況はより一般的な存在である (実在するという意味ではない). 理論上個体や事実と同じように第 1 級の構成要素として状況を含む状況や, パラメータと呼ばれる制限された状況も扱える.

この節では状況理論の一般的概念を説明する. 状況理論における基本要素は**状況, 個体, 関係**である. 状況は正確に何であるかについては状況理論の発展を築いた Barwise や Devli ne([2]) も明確にしてはいない. というよりも, 選択を後の研究者に与えている. これは状況のみならず, あとで説明するサポート関係やインフォンの性質などに対してもいえることである. 個体は考えられている世界における実在である. 関係は個体や状況などの間の関係である. さらに, 情報の基本要素が存在すると考え, これを**インフォンと呼ぶ**インフォンは一階述語論理における述語に対応する. しかし, インフォンはただアーギュメントの間の関係をいうのみであって, それ単体で真偽値が決まるものではなく, 状況を伴ってはじめて真偽が定まる. インフォンは以下の形式で書かれる.

$$\ll r, a_1, a_2, \cdots, a_n; 1 \gg \text{ or } \ll r, a_1, a_2, \cdots, a_n; 0 \gg$$

ここで rは関係, a_iはアーギュメントである. またインフォンの最後の $1, 0$ は**極性**と呼ばれ, アーギュメントの間に関係がある, ないをそれぞれ表す.

ある状況 s においてインフォンσが成り立っていることを

$$s \models \sigma$$

と書く. ここで関係 $\models$はサポート関係と呼び, 上の言明は「s はσ をサポートする」と読む. サポート関係がどのようなものであるかも問題となる部分である. その中でもよく議論されているのはインフォンの**保存性**である. 状況 s_1より大きい状況 s_2において

$$s_1 \models \sigma \text{ ならば } s_2 \models \sigma$$

であるなら, インフォンσは保存的であるといわれる. また, 状況や個体を抽象化して**タイプ**を導入している. タイプは

$$[s|s \models \sigma] \text{ と } [x|s \models \sigma(x)]$$

のような形式で表現し, 状況, 個体の変数を表す. このタイプを用いて, **制約**を表現することができる. 制約は実世界の法則やルールを記述するために用いられる. 例えば,

$$[s|s \models \ll kissing, x, y \gg] \Rightarrow [s|s \models \ll touching, x, y \gg]$$

のように書かれる.

制約

自然摂理や常識などの法則を記述する手段として制約が導入されている. 制約は, 以下のような形式をしている

$$s \models I \leftarrow s_1 \models I_1, s_2 \models I_2, \cdots, s_n \models I_n || B$$

ここで I_iはインフォンの集合, Bは状況間の制約である. 例えば,

$$s_{after} \models \ll dead, x \gg \quad s \models \ll die, x \gg || \ll meet, s, s_{after} \gg$$

は s_{after}が時間的に s の直後であり, s において x が死亡したなら, s_{after}において x は死んでいるという常識を表現する.

2.1.2 状況理論の実装例

PROSIT

PROSIT([18]) は状況理論から導かれる独特な要素を実現するために開発された. その要素として以下の 4 つをあげている

- 部分的に記述されたオブジェクトの使用と部分情報の一般的扱い
- 理論の一級構成要素としての状況
  状況を単なる言明の集合として扱うのみでなく, 状況それ自身をインフォンの引数として扱うことができる.
- 情報の制約の形式的扱い
  前向き制約で知識の追加を, 後向き制約によって推論を行なえるようになっている.
- 自己参照表現の適当な扱い
  言明ににラベルをつけることにより, 言明それ自身をその言明の中で引数として用

いることができるというもの. 例えば, “この文は偽である” ということを

$$p : s \models \overline{p}$$

として表現できる.

PROSIT の状況理論に対する解釈の特徴をまとめると以下のようになる

- 制約も状況に依存する
- 状況はソート論理における型のような機構としても使える. これにより, その状況で常識の知識はそれより上位の状況からの継承によって効率的に導かれる
- インフォンに含まれるアーギュメントは状況の位置に移行することができる.
  例えば, 時間を扱う推論の場合, $l \subseteq s$ として,

  $$s \models kissing(John, Mary, l)$$

  $$l \models kissing(John, Mary)$$

  2 番目の式は時間のパラメータを含んでいない. このような状況を考えることはあるタイプの演繹推論の効率をあげる.
- パラメータと変数の違いはパラメータは知識全体に対して同じ意味を持つものであり, 変数は制約の中にだけ現れる局所的なものである. 変数は制約が適用された時点でパラメータに具体化される. パラメータも定項に具体化される.
- 状況間の関係として上位下位関係があり, 以下の性質が成り立つ
  - 状況をインフォンの集合と見ると, $sit1 \supseteq sit2$
  - 上の結果として, *sit1* のすべての制約は *sit2*のインフォンにも適用される.

Quixote

Quixote([10]) はもともと状況理論の実装という目的で製作されたわけではなく, 演繹オブジェクト指向データベースというカテゴリで発表された. 状況理論の諸概念と Quixote のデータ構造の相性がよいことから東条による状況理論による法的推論の形式化の研究([14]) において用いられている.

Quxote はモジュール, オブジェクト, モジュール間の包含関係, ルールなどをデータ構造として持つ. モジュールはオブジェクトの集合を入れる箱として機能する. 東条の研究では, 例えば, オブジェクトがモジュールに含まれるという関係をサポート関係だと解釈するように直接的な対応から状況理論を実装している. その他の諸概念の対応を図 2.1に示す ([17] からの引用).

状況理論		$\mathcal{QUIXOTE}$
状況	-	モジュール
インフォン	-	複合オブジェクト項
ロール	-	ラベル
支持関係	-	モジュール・ルールの包含関係

図 2. 1: 状況理論と Quixote の概念の対応

その他のモジュール付システム

論理プログラミング言語において, モジュールシステムが導入されるのは大きなプログラムを組む場合の部分プログラムの独立性 (すなわち他の部分に対して表層しか見えないブラックボックスにすること) を高めるためであることが多い. そのため, モジュールシステムを状況依存性を表現するための道具として見ると, モジュール間の関係, モジュール内部のオブジェクトの継承などに関して十分な機能を備えているとはいえず, PROSIT で達成されたモジュールを第 1 級構成要素として扱うことができないなどの制限が多い.

2.2 時間情報の表現と状況

事例の記述に対して, 時間情報をどのように記述するかは大きな問題である. 最も単純な表現法は述語の引数として時間を表現する方法である. 例えば, t で犬が寝ているという情報は

$$\text{寝ている}\,(\text{犬}, t)$$

というように直観的に書ける. しかし, この表現法をとると, 事象の時間的な側面に対する普遍的な性質を記述することができなくなる. 例えば, ある時間で成り立っている関係は次の時間でも成り立っているという制約を記述することができない. そこで, t を述語の

外へ出し, $true(t,$寝ている (犬)) の形式でメタな述語を用いて表現することができる. このような形式で表現すれば, さきほどの問題は解決することができる. この述語 $true$ の時間を表す引数を可能世界と見立てて, 様相論理の応用として時間に関する論理にしたものが時制論理 (temporal logic)[7] である. 時制論理のもとになっている様相論理では世界は真偽値を決定するための単なる指示子であったが, 状況理論では内部の個体や状況を存在として扱って, 引数として状況を取る関係について形式化が試みられている. このため, 本研究では時間情報の表現として状況理論の枠組を用いる.

ただし, 状況理論では時空間を通常状況には含めないとしているから, 時間を状況で表現する場合には若干の修正が必要になるが, それは 3.3節で述べる.

2.3 法的推論における状況依存性

この節では法的推論におけるさまざまな状況依存性を既存の形式またはシステムがどのように扱っているかについて述べ, 次章で説明する状況の形式化のアイデアを論じる.

2.4 法的推論システムの概要と現状

人工知能関連の研究者にとって, 法的推論という研究分野は現実世界の事象をどう表現するかやその上でどのように推論を行なうかを研究するにあたって多くの題材を提供してくれ, 研究の必要性を強調してくれるものである. 以下にその概要と現状を示す.

Prdcg のような論理型言語は一階言語をその論理的基盤においた形式言語である. このような言語では「A ならば B」のような規範・ルールを記述することができる. また, 法例文は見かけ上このような形式を持ったルールである. この両者の相似から人工知能の研究の一分野として法律上の事件を取り扱ったら興味深い結果が得られるのではないかと思ったのがそもそもの始まりであろう. では具体的にはどのように法的推論システムの研究は進められるものであろうか. 原口 ([20]) は以下のような項目を提示している.

1. 法的概念や事象をできる限り直観的で効率的にかける知識表現法. 言語.

2 基本的な推論システム

3 大規模知識に対応した規範の解釈システム

4. 論争システム

このうち 1,2 はどのようなシステムにも必要な項目であってシステムの基幹をなす要素であり, 後で詳しく述べる. 3 は人間の書いた法令文が計算機で扱うには曖昧であり, さまざまな解釈ができるために実際の場面でも拡大解釈や縮小解釈が頻繁に行なわれていることを鑑みると重要な要素となる. 4 は実際の裁判を想起すると実感できる. 例えば刑事事件では検察と被告という 2 つのまったく正反対の立場から自分の立場に有利な推論が行なわれている. 違う立場から主張をぶつけあっていくのが論争システムである. これも多くの研究者が研究を行なっている.

1 に関して, 古典論理プログラミングにおける項では記述の効率性, 概念の整理力を向上するために LOGIN,Hellic II などに見られるψ項がある. これはオブジェクト間のさまざまな上下関係, 例えば is-a 関係や包含関係などの関係を定義して, これを項の表記に応用したものである. この関係を包摂関係と呼ぶ. さらに, Hellic II ではこのψ項を述語にも階層関係を記述できるように拡張した H 項が提案されている. 述語 (関係) もオブジェクトと同じように扱うことで非常に簡潔な概念階層のラティスによって法的概念を整理することができる.

2 に関して法的推論システムは推論方式により大きく 2 つのカテゴリに分けられる. ルールベース推論 (RBR) と事例ベース推論 (CBR) である. RBR は法律の持つ形式的なルールから例外も含めた結論の証明を演繹的に行なういわば上から下への推論である. 逆に CBR は既存の判例の集合から類似性により, 現事例の証明を行なう下から上への推論である. しかし, これらは方針としての区別であり, どちらに重点をおいてるかという区別でしかない. CBR にもルールの記述は存在するし, 多くの判例に裏づけされた事柄は学説などのルールとして一般化されるだろう.

上で述べたように, 法例文は‘見かけ上’ルールの形式を持っているが, あくまで見かけ上の形式であり, これを一階言語の形式だけで書くには自然言語がもたらす意味の曖昧性があり, また法例文には必ずといっていいほどその法律が適用される背景条件があり, 一階言語の枠組だけで記述・推論を行なうのは困難である. これを克服するために, 近年はさまざまな拡張がなされてきたが, 知識表現においてはソート, 素性構造の導入 (Order-sorted Logic), 推論機構として, 仮説推論機構, 非単調論理, モジュール機構を導入する試みがなされている.

2.5 法的推論システムの実例に見る状況推論の実現

2.5.1 Helic-II

Helic II は法的推論に使用されるために開発された言語であり, 法令 (RBR) と判例 (CBR) とのハイブリッド化を押し進めさらに拡張したものである. 事実の表現として H 項を用いる. H 項は述語とオブジェクトを区別せずに同じソート (型) を用いて表現する 1 階論理の項の拡張である. すべてをソートで表すために, 個体概念を持たない. 例えば, 哺乳類や鳥などはソートとして直観にあう表現であるが, 太郎とかコロなどの通常はある特定の個体を指し示すラベルもソートとして扱う. すなわち, 太郎も集合である. このようにする利点はクラスとインスタンスや上位概念と下位概念のなどの関係をすべて同一の包摂関係 (集合の包含関係) で表すことができることである.

また推論機構としては, 法例文, 学説, 判例などの間の矛盾を解決するために論駁推論を採用している. これはルールに優先順位を与えて, 優先順位の高いルールを用いて, 論証を求める推論方式である. 具体的にはルールを確定ルールと論駁され得るルールに分け, そこから論証を求める. ルールの記述としてはルールの集合を一つのカテゴリに区切り, カテゴリ間の優先順位を定義できるようになっている.

ソートを用いることによって, 知識は概念階層というひとつの世界 (状況) に依存した記述を行なうことができている. また, 論駁推論機構については, ルールの集合としてのカテゴリひとつひとつを状況であるとみなすことができる. そして, 状況間の上位下位関係によって優先度を表せる. しかし, Helic-II では各々の状況依存性に対して別々の形式を定義しており, 表現が一様ではないことが問題だと考えられる

2.5.2 その他の法的推論システム

法的類似性を用いた推論システム ([21]) は以下の目的・方法を持つ.

目的 型の**類似性**の階層をゴールに依存して自動的に作って推論に多様性を持たせる.

方法 あるゴール G が与えられた時に, はじめに GDA(ゴール依存アブストラクション) を用いて複数の既存概念を下位概念に持つ新たな概念を構成する. この新たな概念階層を用いてその下位概念どうしを類似するものとして同じ rule を適用できるように

することで推論の幅を広げようとする. 推論の過程中に概念階層が変化することはない.

このシステムでは与えられたゴールに依存した適当な状況が推論実行時に (または推論の前処理として) 生成されるという点で動的な状況の生成を目指しており, 注目に値する.

その他のシステムの特徴を以下にあげておく. HYPO は事例ベース推論を基本にしたシステムである. 判例一つ一つをフレームのようにひとまとまりにしており, 新たに入力された事例に対して, あらかじめ抽出されている法的要素に基づいて類似性を評価する. GREBE の特徴はルールベース推論と事例ベース推論を統合した点にある.

2.6 法的推論における状況推論の必要性

2.4節でも述べたように, 法例文にはその法が適用される背景条件, すなわち状況が存在する. 例えば, 違法行為を行なったとき, その行為が合法的な理由を持たない場合, 行為者はその行為に対応する責めを負うが, 例えば行為者が危険な状況にいるなどの合法的な理由を持つ場合, 刑は軽減されるか無罪になる. ここで危険な状況にいるというのは, 行為者を x という変数で表して,

$$[s|s \models\ll \text{危険な状態}, x \gg]$$

のような状況のタイプで表すことができる. また, 時間的場面の推移によって知識が変化するような仕組みも状況により表現できると考える. さらに, 推論主体の立場, 価値観をも状況によって表現することができる. HELIC-II においては立場や価値観をルールの集合間の優先度を付与することにより実現していたが, これらは各ルール集合を状況として定義して, 状況間の優先順位をつければ状況により表現できる. このように法的推論ではさまざまな状況を扱う必要があるが, その種類ごとにことなる論理を導入することなく一様な論理体系により表現することは有効である. そのため本研究では状況理論の考え方をもとに状況の論理を構築する.

法的知識・ルールと事例が与えられた時, 法的推論システムに求められる質問は何であろうか? 刑事事件であれば被告が有罪か無罪か, さらにはどのような罪に問われ, どれくらいの刑が請求され得るのかというようなことであろう. しかし, それだけではない. なぜなら, 法律のルール体系というものは曖昧性があり, 立場や価値観の違いによって容易

に判決, すなわち質問に対する答が逆転するものであるからである. それゆえ, ある言明が成り立つためにはどのような知識体系, 立場や価値観が必要かということ, より一般的には, どのような状況でその言明が成り立つかということを答えることができることも必要であると考える. また, 推論がどのような経過で行なわれたかということも重要である. 状況理論の表現を用いれば

$$s \models \sigma$$

に対して, s においてσが成り立っているという見方は前者の要求を満たし, σが成り立つ状況 s という見方は後者の要求を満たす

さらに, Quixote による法的推論の形式化では与えられた事例や判例をひとつの状況としてみていたが, 事例中の時間的な場面や, 空間的な位置によっての知識の相違を表現するのにも状況の考え方を導入することができるだろう. 以上考察したことをまとめると法的推論において必要な状況は以下のようになる.

1. ルールの集合としての状況
   法令文, 学説, 憲法などそれぞれ別の状況として記述する

2. 事例としての状況 : 判例
   その判例だけで成り立つルールや事実を記述する

3. 事例の内部の状況 : 時間状況など
   事例内部で知識が変化する, またはことなることを表現でき, 制約により, 情報の流れを定義することができる

4. 様々な状況間の関係を定義している状況 : 推論主体
   ここでは立場や価値観を表現できる

本研究では上記の 3. と 4. を取り扱う. 特に 3　事例内部の状況を重点的に扱う. 次章では特に時間的状況を考えるにあたって, 事例の記述において複雑な状況の関係が言及されることから, 状況自身の性質を扱うために, 状況をより基本的な素粒子の集合として構成的に生成されるものとして捉えた形式化を行なう.

第 3 章

集合的な状況の扱い：集合的状況モデル

状況理論は状況, インフォン, その間のサポート関係などに直観に基づく抽象的定義のみを与えている. このことは応用力の広さを示すための利点にもなっている. しかし, 状況理論に基づくシステムの実装を行なうためには明確な定義に基づくモデルを構築しなければならない. 我々は状況理論の一般性をできる限り損なわない形式化を行なった.

この章で展開する論理は様相論理の変種である. この論理の特徴は状況 (様相論理における世界) を単体で捉えず, 集合として扱うことである.

3.1 準備

この節ではこの章で用いられる表現と用語について解説する. 2.1.1節でインフォンは

$$\ll kill, taro, jiro; 1 \gg$$

のように表現されると述べたが, この章からはアーギュメントロールを明示し, 関係を前に出して,

$$kill(agt = taro, obt = jiro)$$

のように表現することにする. agt は行為者 (agent), obt は目的物 $object$ を表している. これは極性が 1 のインフォンの例だが, 極性が 0 のインフォンは関係の前に '–' をつけることで表現する. 一般には

$$\ll r, a_1, a_2, \cdots, a_n; 1 \gg \text{ or } \ll r, a_1, a_2, \cdots, a_n; 0 \gg$$

を

$$r(l_1 = a_1, l_2 = a_2, \cdots, l_n = a_n) \text{ or } -r(l_1 = a_1, l_2 = a_2, \cdots, l_n = a_n)$$

と表記する. ここで l_iを**ラベル**, a_iを l_iに対する**値**と呼ぶ. このインフォンの表現はLOGIN([9]) で定義されたψ項から来ている. ψ項はさらにソート階層を導入しているが, 本論文ではソートの導入を今後の課題としている. 上記のようににラベルを導入して, アーギュメントを素性のように扱うのは関係を n 項関係に限定せず, 拡張性を保つためである. この表現において, 各アーギュメントの位置は意味上の違いを生じない. すなわち,

$$kill(agt = tar\ o, dat\ = jiro) \text{ と } kill(dat\ = jiro, agt = tar\ o)$$

は同一のインフォンであると解釈する. ラベルの仕様を統一すれば, 大規模な知識を記述しようとする時に, どのような関係も引数をいくつとるか, 何を意味する引数かに気を使わなくて済む.

3.2 一般的状況のモデル

まず, さまざまな状況の種類, 例えば, 巨人と阪神の第一戦の状況とか, 太郎が次郎と討論している状況などに普遍的な状況の性質に関する形式化を試みる.

状況の演算は一つの状況の種類に対してのみ行なえるものとする. これは通常, 人の信念状況とある時間状況を一緒に考えないことを反映している.

状況とは一つの世界であり, ある言明が成り立つ状況を求める手続きはその言明が成り立つような公理系, すなわち制約の集合を求めることである.

3.2.1 状況

まず, 状況の素粒子となるべき**状況素**を考える. 状況素は推論主体 (たいていは人間) によってある**因子**に関して分割された (individuated) 存在である. ここで, 因子とは, 例えば, 時間, 空間, 信念などである. 状況元素は推論主体にとってそれ以上分割されない, すなわち最小の世界である.

定義 3.1 状況種 *(species of situation)*

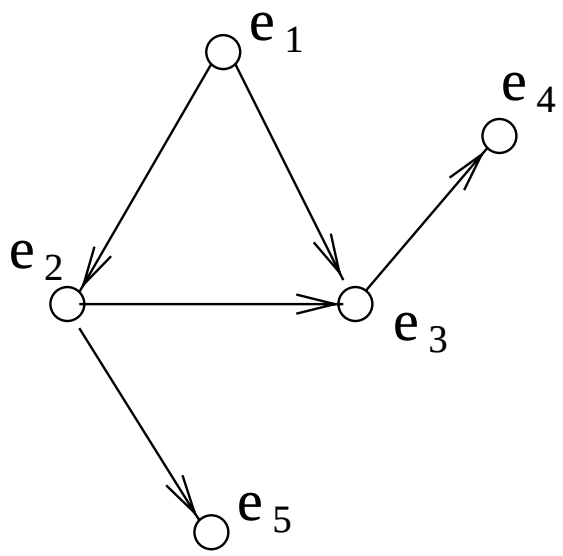


図 3.1: 状況グラフ

1. ひとつの因子 ωについて, 分割されたれた状況の集合を状況種 *(species of situation)* Γ_ω と呼ぶ.

2. ωに関して分割された状況素の集合を E_ωとする.

3. $\Gamma_\omega \subseteq 2^{E_\omega}$

ωは時間, 空間, 信念などである. 例えば, $E_{空間}$の一つの要素は東京や神奈川などになる.

次に状況素間の到達関係として, 以下の関係を考える.

隣接関係 Rを状況素間の 2 項関係とし, 隣接関係と呼ぶ. $(R : E \times E)$

Rがどのような性質を持つかはωに依存して決定される. とにかく人間の直観に合うような性質を持たなければならない.

Eと Rからグラフを構築することができる. 図 31 において状況素は頂点, 関係 Rは辺で表現されている.

このグラフ表現を用いて状況を以下のように定義する.

定義 3.2 状況

Eと Rによって作られる有向グラフ $G =< E, R >$において, $\forall s_1 \forall s_2$の任意のパス中のすべての頂点 *(*状況元素*)* の集合が状況である.

図 3.1から, $\{e_1, e_2, e_5\}$ は状況である. しかし, $\{e_1, e_3, e_5\}$ は状況ではない. このすべてを含むパスが存在しないからである. これは多軸の時間構造を考えた時に, ある時点からの複数の可能な世界の集合を状況にはしないことなどを反映している. 人の共通知識などを表現する場合にはその領域で対象としている人間ひとりひとりを一つの状況素とみなして, すべての人間状況の間に隣接関係を定義すればよい.

もうひとつ例をあげる. 先ほどの例をまたあげて, $E_{空間}$ = { 東京, 千葉, 神奈川 }, 東京 R千葉, 東京 R神奈川, 神奈川 R千葉, かつ Rは反射的だとすると, { 東京 }, { 東京, 神奈川 }, { 東京, 千葉, 神奈川 }, などはすべて状況になる.

3.2.2 サポート関係

次に曖昧な概念であるサポート関係を定義する.

定義 3.3 サポート関係

1. 扱う状況種の注目している要素ωに関して大きさの定義を行ない, インフォンσ, 状況 s に対して $[\sigma]_\omega$, $[s]_\omega$ をそれぞれωに関する大きさ とする. サポート関係とは,

   保存的 $s \models_p \sigma \ \mathit{iff}\ |s|_\omega \geq |\sigma|_\omega$

   非保存的 $s \models_{np} \sigma \ \mathit{iff}\ |s|_\omega \leq |\sigma|_\omega$

   等価的 $s \models_{eq} \sigma \ \mathit{iff}\ |s|_\omega = |\sigma_\omega|$

   の *3* 種類である.

2. サポートマップΔとは状況がサポートするインフォンの集合である
   $\Delta s = [\sigma | s \models \sigma]$

3. $s = s_1 \cup s_2$であるとき,
   σが保存的:$\forall\sigma[s_1 \models \sigma \vee s_2 \models \sigma \rightarrow s \models \sigma]$
   σが非保存的:$\forall\sigma[s_1 \models \sigma \wedge s_2 \models \sigma \rightarrow s \models \sigma]$

“太郎は東京で生まれた” は保存的なサポート関係でサポートされるインフォンの例である. このインフォンは時間的にも空間的にも広げた状況において真である. しかし “太郎は眠っている” は空間的には保存的であるけれども, 時間的には非保存的である. すな

わち時間的に状況を広げると太郎は起きてしまう. また, “犬が 4 匹いる” というようなインフォンは時間的にも空間的にも等価的なサポート関係で状況と関係付けられる. (状況を小さくしても大きくしても成り立たなくなる) このような考察から, 状況種毎にサポート関係を定義する必要があることがわかる.

例えば, { 東京 } |=太郎のコンサートが開かれたという言明は保存的なサポート関係であり, { 東京 } |=犬が 4 匹いる, というのは非保存的である. なぜなら, 前者は { 東京, 神奈川 } 状況でも成り立つが, 後者は { 東京, 神奈川 }, 状況では成り立たない.

このようにサポート関係を定義すると, サポートマップは状況が状況素の集合として表現されているために構成的に計算することができる.

3.2.3 状況の性質・状況間の関係

状況の性質はその状況がサポートしているインフォンの集合, すなわちサポートマップにより決定する. 例えば, $s \models running(agt = taro)$ ならば s は太郎が走っている状況であるし, さらに $s \models running\ (agt = hanako)$ ならば s は太郎と花子が走っている状況となる. このような性質は状況を抽象化して, 状況のタイプとすることで, 制約に使用される. $[x|x \models running(agt = taro)\ ]$という表現は太郎が走っている状況のタイプを表す.

状況の無矛盾性

$\exists\sigma \in INF : s \models \sigma \wedge s \models \bar{\sigma}$であるとき状況 s は矛盾であるという. このようなσが存在しない時, s は無矛盾であるという.

状況は状況素の集合として表現されるため, 集合論的な関係はは定義するまでもなく成り立つ. すなわち部分集合, 和集合, 積集合などはそれぞれ, 部分状況, 和状況, 積状況の関係を定義していることになる.

部分状況は関係 $\sqsubseteq$ で表す. 状況 s_1, s_2の間に $s_1 \sqsubseteq s_2$の関係があるとき, 以下のことがいえる.

$$s_2 \models_{np} \sigma \rightarrow s_1 \models_{np} \sigma$$

$$s_1 \models_p \sigma \rightarrow s_2 \models_p \sigma$$

3.3 時間状況のモデル

3.2章のモデルにおいて, ω=時間とした場合, 状況はある時間的長さを持った時区間と捉えることができる.

- 時間状況種においてωは時間である.
- $E_{time} = E_{time}^{interval} + E_{time}^{point}$
  時間状況種の状況素は時区間の集合 $E_{time}^{interval}$ と時点の集合 E_{time}^{point}の直和である. 時区間は 0 でない時間的長さを持った状況素であり, 時点は時間的長さが 0 の状況素である. 実際のモデル上に時間的長さは明示的に表現しないので, 時区間も時点も本質的に同じ状況素である. その区別は隣接関係の接続状態によって決定される.
- 時間状況種において Rを特別に $|(\mathrm{meet})$ と書く.
  $|$ は反対称的である. また $|$ は $E_{time}^{interval}$ 上のみの関係である.
- $|\sigma| = 0$ のインフォンをイベント, $|\sigma| > 0$ のそれをステートと呼ぶ.
- イベントは時間に関して保存的なインフォンであり, ステートは非保存的なインフォンである. それゆえ, 時区間にイベントが保存的にサポートされる表現も許される. ただし, その場合その状況素内に成り立つ他のインフォンはそのイベントの影響, すなわち変化される要素とはなり得ない. ステートに関して言えば,
  $t \models_{time} \sigma$ iff $|t|_{time} \leq |\sigma|_{time}$
  というサポート関係の定義になる.
- 状況 s_1でσが成り立っていて, その直後の状況 s_2(すなわち $s_1|s_2$) でσの否定が成り立っていないなら s_2でσが成り立つ.
  $s_1|s_2$ implies $\forall\sigma[s_1 \models \sigma \wedge s_2 \not\models \overline{\sigma} \rightarrow s_2 \models \sigma]$
  これを**保存制約と呼ぶ**
- その他の状況間関係
  $pre(e,t), post(e,t)$

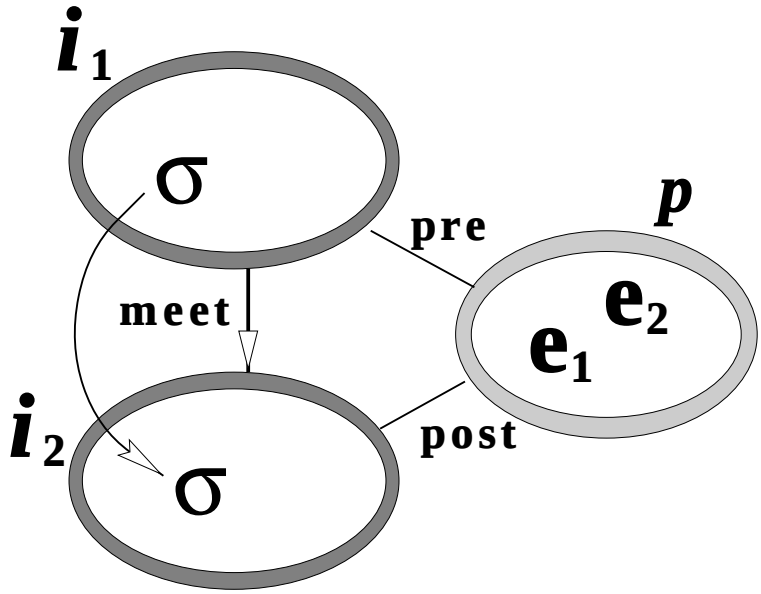


図 3.2: 時間関係

これらは時間状況特有の関係として定義する. $pre, post$ とも $E^{interval}_{time}$ と E^{point}_{time} の間の 2 項関係である. pre(e,t) はイベントの集合としての状況素 e の直前の状態が t であることを表し, post(e,t) は t が e の直後であることを表現する.

時間的な状況を考える場合に, 状況はイベント (の集合) によって分割されると考えるのが相当であろう. イベントをある時間的長さを持った時区間に保存的なサポート関係でサポートされるというモデルも構築することができるが, その場合そのイベントの前提状態と結果状態が同じ状況元素の中に記述されることになり, イベントの影響を推論することができなくなる. したがって, 状況元素はイベントの発生によって分割されているべきであり, その前後の状況元素の集合としての状況を考えることにより, そのイベントの発生の影響を受けないステートの成り立つ状況を扱うことができる.

時間状況間の関係のイメージは図 3. 2のようになる. ここで $i_1, i_2 \in E^{interval}_{time}$, $p \in E^{point}_{time}$ であり, 3 つの関係 $meet$, pre, $post$ によって接続されていると考えて良い. また, σはステート, e_1, e_2はイベントである. i_1でサポートされるインフォンは保存制約より強い制約がなければ i_2でもサポートされることを矢印によって表している.

この時間状況の形式化は Situation Calculus の状況の概念と類似しているが, Stuation Calculus の状況が絶対的時間軸の上の断面であるのに比べて, 相対的, 主観的分割であり,

また, あらかじめ与えておくものである. また, 状態の変化によって瞬点を求めるのに比べて, あらかじめ与えておき, かつその瞬点も状況であり, 複数のインフォンをサポートできる.

3.4 推論機構

3.4.1 推論主体

何か推論を行なおうとする時は推論の対象となる領域 (対象領域) と推論を行なうために必要なルール群の領域 (推論領域) が必要である. そのどちらも推論主体の内部に存在していなければならない. 例えば, 太郎が, 次郎は花子に暴行したので有罪だという推論を行なう場合, 次郎が花子に暴行したという事実も, 暴行したら有罪だというルールも太郎の頭の中に存在しなければならない. すなわち推論主体は内部にある対象領域の事実から推論領域のルールを用いて何らかの結論を導くという操作を行なっている. この直観的推論機構の解釈はどのような推論であっても当てはまるものである. それゆえ, この仕組みは固定とする.

対象領域はその推論主体によって分割された状況元素のグラフで表現できる. これは前節までに述べてきた状況のモデルを使用する. 次に推論領域であるが, これはひとつの世界に固定であるという見方もできるが, 実際の推論を鑑みるに, 推論主体には複数の衝突を起こすルールが存在し, 推論を行なう時には衝突を起こすルールの強さによって選択的に適用しているように思われる. このような直観は同じレベルの強さを持つルールの集合を状況とみなして, その状況間の強さを考えることにより, 実現できる. 実際には, 推論主体の状況 $world$ にルールを輸入することができるルールを持つ状況間に優先順位をつけるインフォンが $world$ にサポートされているという形式化を考えれば良い.

上述の推論機構のイメージを図3.3に示す. 推論主体は$World$, 対象領域は$TargetDomain$, 推論領域は $ReasoningDomain$ で表されている. 推論領域において, 上の方が強い状況である. ここでは状況 s_4でインフォンσが成り立っているということを証明するのに, 推論領域には矛盾するルールがあるので強い c_1中のルールが採用され, ゴールは証明できることになる.

上記の考えに基づき証明手続きを形式化する.

3.4.2 変数

変数は個体, 状況, インフォン, 項を表現するのに用いられる. この論文では変数を大文字から始まるレターで表す. 変数は知識として用いられる場合には任意限量される. 例えば,

$$s \models alive(a\text{-}obt = X)$$

は s という状況ですべての人は生きているという意味になる. 一方, 変数が質問の中に現れる場合は任意限量される. 上の項で質問がなされた場合, X はある特定の人に束縛される. さらに質問として

$$?X \models alive(a\text{-}obt = taro)$$

が与えられた場合には X はある状況に束縛される. この例のように状況に対する変数を**状況変数**と呼ぶ. 状況変数は $G_\omega =< E_\omega, R_\omega >$から生成できる状況のひとつに束縛されうる.

3.4.3 否定の取り扱い

状況一つだけを対象とすると, ある状況 s がインフォン$\sigma, \overline{\sigma}$が成り立つかどうかということに直観論理のように次の 3 つの場合が考えられる.

1. $s \models \sigma$
2. $s \models \overline{\sigma}$
3. $s \not\models \sigma \wedge s \not\models \overline{\sigma}$

対象を広げて他の状況も対象とすると, 3 .の場合には制約によって結ばれた状況に真偽を求める. さらに対象領域全体では失敗としての否定 (negation as failure) である. このことは質問がなされる状況が対象とする世界の範囲を決定しなければならないことを示唆する. マクロに見れば推論は対象世界の中だけの事実から行なわれる閉世界仮説に基づく.

事例記述において論理的な否定と失敗による否定の両方を記述する場合が生じるので, この両者ともを扱う.

3.4.4 単一化

インフォンの単一化ここではインフォンの種類イベント, ステート, プロパティに対して単一化の定義を与える. インフォンの単一化について論じるために, まずインフォンの**一般化**の概念を定義する.

定義 3.4 インフォンの一般化

2 つのインフォン

$$s = a(l_1 = a_1, l_2 = a_2, \cdots, l_n = a_n)$$

$$t = b(l_1 = a_1, l_2 = a_2, \cdots, l_m = a_m)$$

が与えられた時, s のラベルの集合 L_s と t のラベルの集合が $L_s \subset L_t$ であり, L_s 中のすべてのラベルに対して, s, t ともに同じ値を持つならば, s は t の**一般化**であるという.

ゴール $s = a(l_1 = a_1, l_2 = a_2, \cdots, l_n = a_n)$ と
事実または仮説 (ルールの頭部)$t = b(l_1 = a_1, l_2 = a_2, \cdots, l_m = a_m)$ が与えられた時,

イベント・ステート s が t の一般化であるならば, 単一化できる.

プロパティ t が s の一般化であるならば, 単一化できる.

例を挙げると, イベントの場合は

事実 : 打つ ($agt =$ 太郎, $obt =$ ボール, $inst\,r =$ バット)
ゴール : 打つ ($agt =$ 太郎, $obt =$ ボール)

ならが単一化可能, 逆にプロパティの場合

事実 : 飛ぶ ($agt =$ 鳥)
ゴール : 飛ぶ ($agt =$ 鳥, $place =$ 皇居上空)

のとき単一化可能となる.

状況の単一化

- 基底状況であるなら, それ自身としか単一化しない
- 変数であるなら制限 (どんなインフォンをサポートするか) を満たす**矛盾のない**状況を求める操作.

状況の単一化は導出機構の実行過程と不可分の手続きであり, 3.4.5節で詳しく述べる. 状況を抽象化して変数として与えることの意味は, 2.6節でも述べたように, 与えられた言明が成り立つ公理系を求めることである. これはゴールに依存して前提となる知識を動的に決定することに他ならない. これを集合的操作を用いて制約を解消することにより決定する.

3.4.5 証明手続き

この節では全節までに述べてきた状況モデルに基づく推論 (証明) 手続きを説明する.

証明手続きは図 3.5の手続き $SOLVE$で与えられる. 証明されるべきゴール $G = s \models \sigma$ が与えられた時, 手続き $SOLVE$が行なう操作を概略的に示すと以下のようになる.

1. G の状況 s の探索範囲 IN_sを求める. IN_sは$< f_s, l_s, N_s, R_s >$ で表現される. f_s, l_sは状況素で, 状況の始点, 終点をそれぞれ表す. N_sは必須状況素の集合で, R_sは取り得る状況素の最大集合である.

2 すべての $e \in IN_s$に対して $e \models \sigma$の証明を試みて (手続き $SOLVE_e$によって行なう, 図 3 . 参照) ,証明できた e を OUT_sに加える.

3 . OUT_sから手続き $CONSTRAINTSOLVER$によって状況間の制約を解消する.

さらに, 上の 2 .で使用される $SOLVE_e$の動作を以下に示す (アルゴリズムは図??). ゴール $G = e \models \sigma$(e は状況元素) に対して,

1 e がσと単一化できるσ'をサポートしているか調べる. サポートしているなら 3 へ.

2. ゴール $e \models \sigma$ と単一化できるヘッドを持つルールを求めて, ボディ部をゴールとして $SOLVE$ により証明を行なう.

3. 1,2 をゴール $\overline{G} = e \models \overline{\sigma}$ に対しても行なう.

4. G の言明の強さが $\overline{G}$ よりも強いならば, 証明は成功する. 逆か等しいならば, 失敗する.

最後に $CONSTRAINTSOLVER$ は $SOLVE$ により得られたそれぞれの s に対する OUT_s と状況間の制約から整合性のある状況を求める関数である (図 3.7参照). 制約の解消は下方制約解消と上方制約解消がある. 一般的には, ゴールの状況が基底であるなら下方制約解消を用い, 変数であるなら, 上方制約解消を用いる. ただし, 実際の手続きとしてはどちらの場合も下方上方両方の制約解消手続きを実行する. 下方制約解消を用いるのは主に処理の効率のためである. 制約解消のイメージ図を図 3.8に示す.

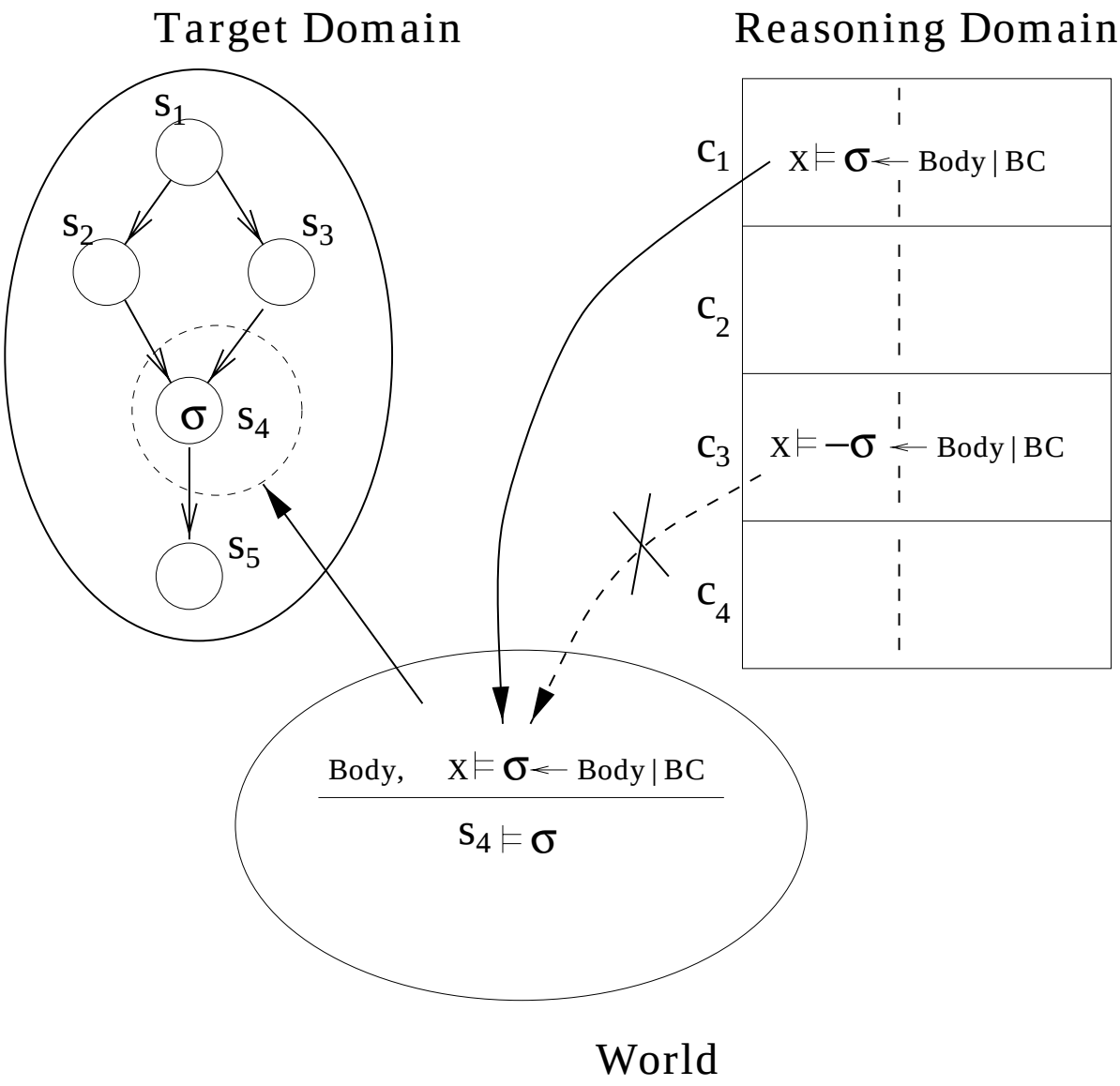
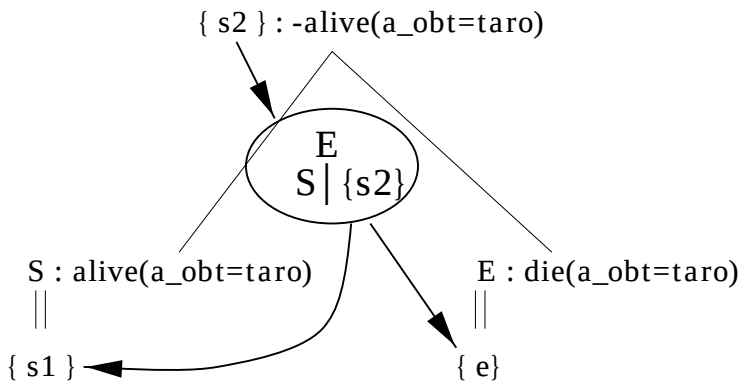


図 3.3: 推論機構のイメージ



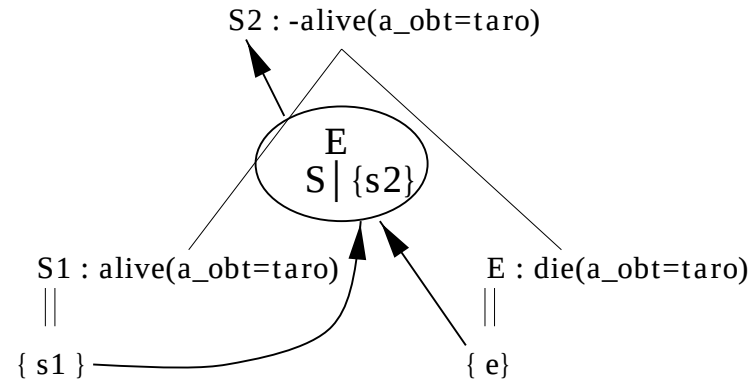


図 3.4: 制約解消のイメージ図

```
function SOLVE(goals, BC);
return variable − bind,
    begin
    GOALS=goals;
    while GOALS ≠ ∅ do
        begin
        remove {< situation, infon >} from GOALS ;
        for each e in elements(situation) do
            begin
            BINDS_e ← SOLVE_e(e, inf);
            BINDS ← merge(BINDS, BINDS_e);
            end
        BINDS;
        end
    BINDS ← CONSTRAINTSOLVER(BINDS, BC);
    end
return (BINDS)
end
```

図 3.5: 関数 $SOLVE$

```
procudure SOLVE_e(e, inf);
  begin
  if SUPPORT(e, inf)
    then return;
    else
      begin
      < head, body, bc > ← RULES;
      if UNIFY(< e, inf >, head)
        then SOLVE(body);
        else return(false);
    end
  end
```

図 3.6: 手続き $SOLVE_e$

```
procudure CONSTRAINTSOLVER(binds, bc);
return variabed − bind,
  begin
  BC ← bc;
  foreach c in BC do
    begin
    if c is given a variable-set vl
      then for all v ∈ vl, return all domain;
      else execute CS_in(bindsc)
    end
  end
```

図 3.7: 手続き $CONSTRAINTSOLVER$

第 4 章

実装と評価

4.1　システムの構成

3章のモデルに基づいて, Sicstus Prolog 上で実装を行なった.
システムの構成を図 4.1に示す.

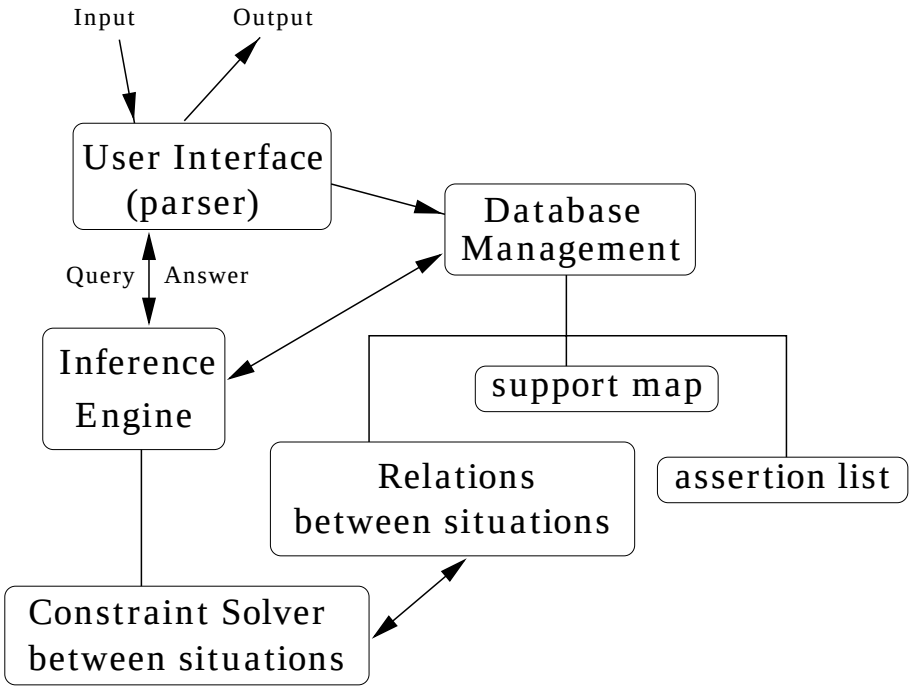


図 41　システム構成

システムのモジュール

入出力 ユーザからの入力を内部形式に変換, 推論エンジンからの出力を外部形式に変換する

推論エンジン 導出原理に基づく推論を行なう. サブモジュールとして状況間制約解消器を持っており, これを用いて導出の過程で状況変数をアサインする.

状況間制約解消器 推論エンジンからの要求に応じて, 状況変数にアサインする状況の候補と状況間の制約から制約を満たす解を生成する

データベース管理 推論エンジンへ効率良くデータを渡せるデータ形式で保存する.

4.2 諸概念の実装

状況理論の諸概念は以下の通り表現される.

インフォン : $(-)rel(l_1 = v_1, l_2 = v_2, \cdots, l_n = v_n,)$

サポート関係 : $sit : \sigma$

制約 : $s : \sigma <- s_1 : \sigma_1, \cdots, s_n : \sigma_n | BC$

ここで BCは状況間の制約

4. 3 推論アルゴリズム

システムに対してゴール $s : \sigma$が与えられると, 以下のアルゴリズムに基づいて証明が行なわれる.

1. 証明

   - s が基底である場合:

     (a) s がσと単一化できるσ'をサポートすることを証明する.

     (b) ルールによる導出によってサブゴールを求める. 2 へ.

   - s が変数である場合:

     (a) σが成り立つ状況の候補を求める. すなわち, すべての状況元素に対してσが成り立つかどうか調べる.

(b) ルールによる導出によってサブゴールを求める. 2 へ

2. 導出・制約解消

ルールによってサブゴールが得られた後は, 各サブゴールに対して証明を試みる. すべてのサブゴールが求まった後は, ルールに付随している状況間の制約に基づいて正しい状況を求める.

4.4 例

この節ではシステムの推論の仕組みを実際の事例を用いて説明する. 以下の事例は [24] から引用した.

事件の概要:

「A は不仲である B の家に刃物を持って暴れ込んできた. A を見て, 身の危険を感じた B は家から逃げ出し, 駐車中の車に隠れていたが, A がこれに気づいて乗用車に乗って追いかけようとしたため, やむなく B は飲酒運転で逃げ出し, N 警察署に到着して助けを求めた. 」

事件の焦点は B が飲酒運転を行なったことが緊急避難・過剰避難またはどちらでもないかである.

刑法の緊急避難は

刑法 37 条 自己または他人の生命, 身体, 自由もしくは財産に対する現在の危難を避くる為め已やむことを得ざるに出でたる行為は, 其行為より生じいたる害其避けんとしたる害の程度を越えざる場合に限りこれを罰せず

と定められている. 緊急避難が成立するためには

- 現在の危難

  法益に対する侵害か差し迫った危険のことをいう

- 避難のための行為

  行為は避難の為の行為でなければならない

- 補充性

  他に避ける方法がないこと

- 法益均衡性
  避難行為が危難に対して相当であること

が必要である. 法益均衡性に対して, その程度を越える行為は, 過剰避難となり, 違法性は失わないが, その刑を軽減または免除する ([23]).

この例題で問題にするのは現在の危難と法益均衡性の時間的側面についてである. 図?? に事件の時間的な流れを表している. ここで現在の危難は危険な状態にあるというインフォンを持つ状況として表現される. 法益均衡性は危険な状況において飲酒運転は違法行為ではあるが, 害を避けるための相当な行為であるとして法益均衡性が保たれている. しかし, 危険な状況でなくなった時に, 時間的に過剰にその行為が行なわれていれば, その行為の全体としては過剰避難行為とみなされる.

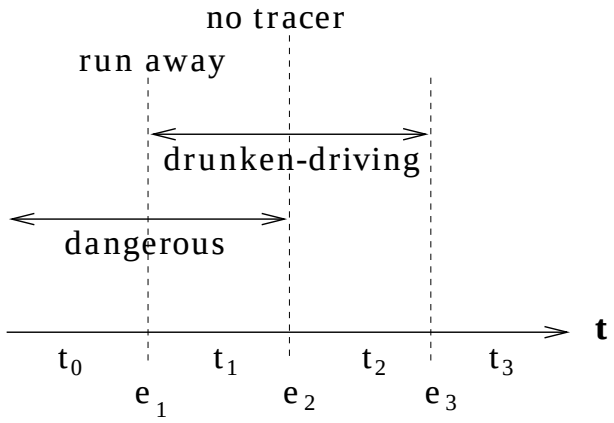


図 4.2 事例の記述イメージ

[ルール]
緊急避難行為 (a-obt=T1:違法行為 (agt=X))<-
T1:違法行為 (agt=X),
T:危険な状態 (a-obt=X),
T1:P.
| T1 ⊆ T.

% X が危険な状態にある状況のうちに行なわれた違法行為は

```
% 緊急避難行為とみなされる

過剰避難(a-obt=T:違法行為(agt=X))<-
    T:違法行為(agt=X),
    緊急避難行為(a-obt=T1:違法行為(agt=X)),
    not 緊急避難行為(a-obt=T2:違法行為(agt=X)).
  | T1|T2, T1 ∪ T2=T.

% X が違法行為を行なったことは緊急避難行為と認められるが,
% それ以上の時区間においても行なったならばその行為を行なった
% 全時区間に対しては過剰避難行為とみなされる.

違法行為(agt=X)<-飲酒運転(agt=X).

% 飲酒運転は違法行為である



[Query & Answer]
? 緊急避難行為(a-obt=X:飲酒運転(agt=b))

    yes X={t1}

? 過剰避難(a-obt=X:飲酒運転(agt=b))

    yes X={t1,t2}
```

1 番目の質問に対する証明木を図 4.4に示す. この質問は緊急避難と判断される飲酒運転を行なっている状況が変数として与えられているので, 証明木の最下段で得られた状況の候補から上方制約解消によって解を求めている.

本システムにおける実際の記述と推論のトレースを付録 A. に示す.

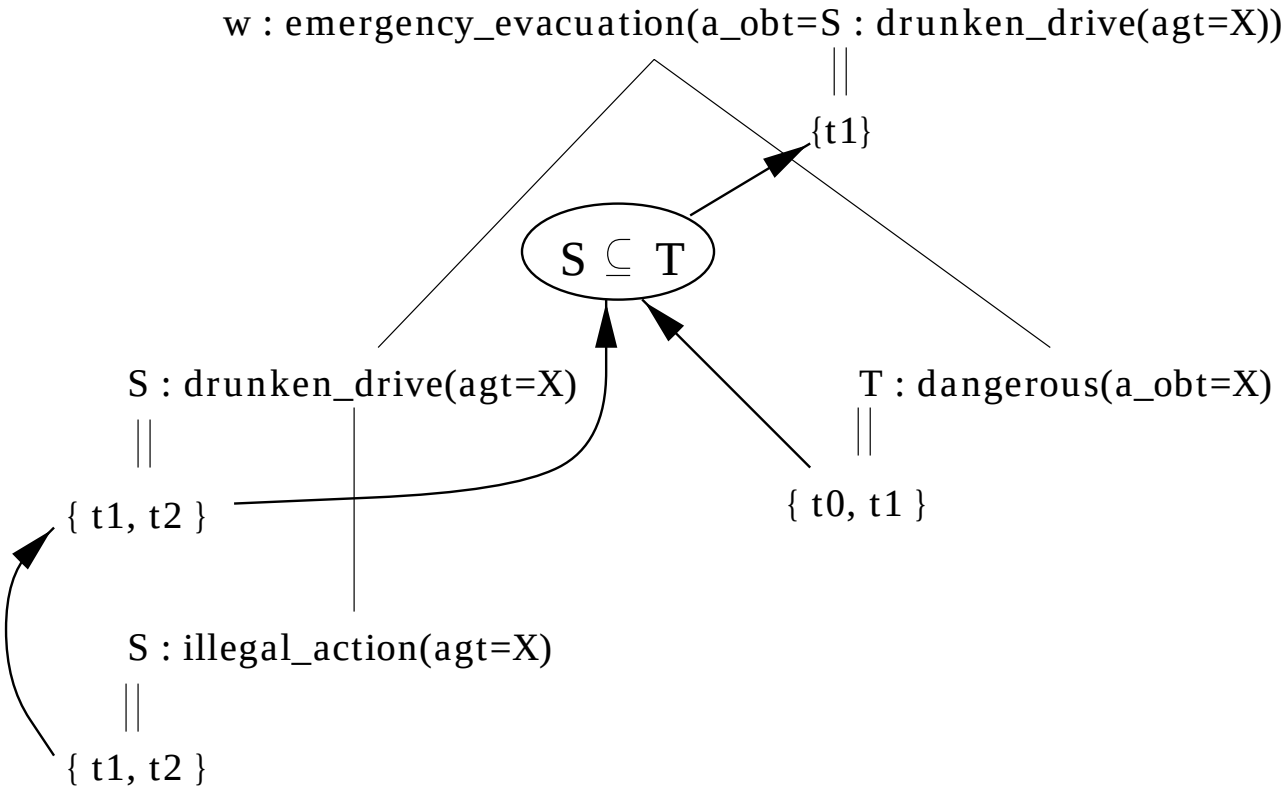


図 4.3: 緊急避難に対する証明木

第 5 章

おわりに

5.1 まとめ

本研究では集合的な状況のモデルを状況理論に基づいて形式化した. さらに, このモデルに基づいて推論システムを構築した. このシステムにおいて複雑な時間関係を持つ事例を直接的な記述で表現でき, 時間関係を考慮した推論を行なえることを示した. 本研究の利点は以下の通りである.

モデル

- 集合的状況モデルは様相論理の拡張である. 状況 (世界) は状況素の集合として表現されるので, 状況間の関係や演算を集合的関係や操作によって構成的に生成することができる.
- 時間論理などの時間に関することに特化した論理ではなく, さまざまな種類の状況に用いることができる.

システム

- 固定化されたモジュールを持つシステムに比べ, モジュールを抽象化することができるので, 状況を固定して事実を証明するだけでなく, ある事実が成り立つ状況を求めることが可能になった.
- 状況は集合で表現され, 状況間の制約解消は集合間の演算と同等の演算を使用することができる. 例えば, 状況の包摂関係は単純に部分集合かどうかを判定するだけで済む.

5.2 他のシステムとの比較

↔ モジュールつきシステム (Quixote)

推論の過程において状況を集合的演算によって動的に計算する分処理は遅くなるが, それ以上の表現力と柔軟性を得ることができた.

↔ 法的推論システム (Helic-II)

時間関係の制約を伴った法例文や事例の記述が直観的で理解しやすい形式で書けるようになった. さらにこれらの記述からユーザが工夫することなく状況に依存した推論を行なうことができるようになった. Helic-II がソートを持っているために本システムはその分記述の効率性に劣るが, ソートの概念は集合的状況モデルと矛盾するものではないので, 簡単に導入できる. これは今後の課題になる.

5.3 今後の課題

- 時間的状況だけでなく空間的状況, 推論者の立場や価値観といった状況を統一的に扱えることを記述実験によって示す.
- 空間的・計算的な効率の向上と, 法的概念の整理のために Helic-II と同等のソート階層の導入とそれに伴う未解決の意味論の問題を整理する.
- 状況の分割の細かさの変換
- システムの入力の前処理として, 時系列に沿ったイベントの記述から状況表現への変換機構の導入. これは事象間の時間関係の自動的生成にに相当する (参照 [15])

参考文献

[1] J. Barwise, The Situation in Logic, CSLI Lecture Notes No. 17, University of Chicago Press, 1989.

[2] K. Devline, Logic and Information I: Situation Theory, Cambridge University Press, 1991.

[3] H. Nakashima, H Suzuki, P-K Hvorsen, S. Peters, "Towards a Computational Interpretation of Situation Theory", *The International Conference on Fifth Generation Computer Systems*, pp. 489-498, 1988.

[4] J. F. Allen, "Towards a general theory of action and time," *Artificial Intelligence* 23: pp. 123-154, 1984.

[5] D V. McDermott, "A temporal logic for reasoning about processes and plans," *Cognitive Science*, 6: pp101-155, 1982.

[6] Y. Shoham, "Reasoning about change," The MIT Press, 1988.

[7] N Rescher and A Urquhart, "Temporal Logic", Springer Verlag, 1971.

[8] W. Chen, M Kifer, and D S. Warren, "Logical Foundations for Object-Oriented and Frame-Based Languages", Technical Report 90/14(revised), June, 1990.

[9] Ait-Kaci and R. Nasr, "LOGIN: A Logic Programming Language with Built-In Inheritance", *Journal of Logic Programming*, pp.185-215, 1986.

[10] Yokota, K. and Yasukawa, H., "Towards an Integrated Knowledge Base Management System", *Proc. Int. Conf. on FGCS*, ICOT, Tokyo, pp.89-112, 1992.

[11] S. Tojo and H. Tsuda and K. Yokota and H. Yasukawa and Y. Morita, "Quixote as a tool for Natural Language Processing", *Proc. of International Conference on Tools with Artificial Intelligence (ICTAI '93)*, 1993.

[12] K. Nitta and S. Tojo and et al, "Knowledge Representation of New Helic II", *Workshop on Legal Application of Logic Programming, ICLP '94*, 1994.

[13] K. Nitta and Y. Ohtake and S. Maeda and M. Goto and H. Ohsaki and K. Sakane, "HELIC-II: A Legal Reasoning System on the Parallel Inference Machine", *FGCS '92*, pp.1115-1124, 1992.

[14] 東条敏, SHELLEN, WOONG, 新田克巳, 横田一正, "状況理論による法的推論の形式化", 情報処理 *vol. 36, No.* 1, 1996.

[15] 東条敏, "プロセス, 状態, イベントの区別から導かれる事象間の時間関係の生成", 人工知能学会誌 *vol. 10, No.*, 6, 1995.

[16] Robert KOWALSKI and Marek SERGOT, "A Logic-based Calculus of Events", *New Generation Computing*, *4*, pp.67-95, 1986.

[17] 東条敏, 津田宏, 安川秀樹, 横田一正, 森田幸伯, "言語情報処理の枠組としてのQuixote", 人工知能学会誌, *vol.9, No.6*, 1994.

[18] Hideyuki Nakashima, Hiroyuki Suzuki, Per-Kristian Halvorsen, and Stanley Peters, "Towards a Computational Interpretation of Situation Theory", *FGCS '88*, pp489-498, 1988.

[19] McCarthy, J. and Hayes, P.J., "Some philosophical problems from the standpoint of artificial intelligence," in *Machine Intelligence4*, Edinburgh University Press, 1969.

[20] 原口誠, "第五世代コンピュータプロジェクトの成果と残された課題 5. 法的推論", 情報処理 *vol.37, No5*, 1996.

[21] 原口誠, 角田篤泰, 大久保好章, "ゴールに依存した抽象化を用いた法的推論の研究", 平成 7 年度科研費重点領域研究「法律エキスパート」研究成果報告書, pp205-209, 1996.

[22] 萩谷昌己, "ソフトウェア科学のための論理学", 岩波書店, 1994.

[23] 前田雅英, "刑法総論講義", 東京大学出版, 1988.

[24] "刑法判例百選 I 総覧", 別冊ジュリスト, *No111*, 1991.

謝辞

東条敏助教授には主指導教官として研究方針, 研究の進め方について御指導いただきました. また研究者として興味深い題材を提示していただいたことに感謝致します. 奥村学助教授には研究の方向性についての助言と, 調査すべき文献を教唆いただきました. 感謝致します. 國藤進教授には学外での発表の機会を与えていただき, また適切なアドバイスを与えていただいたことに感謝致します. 東条研究室の輪講に参加して, 助言や議論をしていただいた佐藤研究室の小野哲雄氏と國藤研究室の村川賀彦氏に感謝致します. 東条・奥村両研究室の皆様には研究上様々な支援をいただきました. 最後にここには挙げ切れなかった方も含めて, 本研究を支えて下さった皆様に感謝致します.

付録 A

事例記述と推論トレース

```
/* Rules */
/* Temporal Propagation */
c0::S2:X<-S1:X | meet(pre=S1,post=S2).
/* Legal rules */
c1::w:emergency_evacuation(a_obt=S:illegal_action(agt=X))<-
        S:drunken_drive(agt=X),
        T:dangerous(a_obt=X)
        | subset(sub=S,super=T).
c1::w:superfluous_evacuation(a_obt=S:illegal_action(agt=X))<-
        S:drunken_drive(agt=X),
        w:emergency_evacuation(a_obt=T:illegal_action(agt=X)),
        w:-emergency_evacuation(a_obt=U:illegal_action(agt=X))
        | meet(pre=T,post=U), union(s1=T,s2=U,s3=S).
c2::S:illegal_action(agt=X)<-S:drunken_drive(agt=X).

/* Facts */
t0,t1:dangerous(a_obt=taro).
t2:-dangerous(a_obt=taro).
t1,t2:drunken_drive(agt=taro).
t3:-drunken_drive(agt=taro).
/* Neighborhood relation */
link(from=t0,to=t1).
link(from=t1,to=t2).
link(from=t2,to=t3).
```

```
/* temporal relation */
time(pre=t0,post=t1,event=e1).
time(pre=t1,post=t2,event=e2).
time(pre=t2,post=t3,event=e3).
/* strength of situations */
c2 > c1 > c0.

/* trace 1 */
w> emergency_evacuation(a_obt=S:illegal_action(agt=taro)).

'w:emergency_evacuation(a_obt=t1:illegal_action(agt=taro))'<-
   't1:illegal_action(agt=taro)'
   't0,t1:dangerous(a_obt=taro)'

't1:illegal_action(agt=taro)'<-
   't1:drunken_drive(agt=taro)'

S = [s1]

/* trace 2 */
w> superfluous_evacuation(a_obt=S:illegal_action(agt=taro)).

'w:superfluous_evacuation(a_obt=t1,t2:illegal_action(agt=X))'<-
   't1,t2:illegal_action(agt=taro)'
   'w:emergency_evacuation(a_obt=t1:illegal_action(agt=X))'
   'w:--emergency_evacuation(a_obt=t2:illegal_action(agt=X))'

'w:emergency_evacuation(a_obt=t1:illegal_action(agt=taro))'<-
   't1:illegal_action(agt=taro)'
   't0,t1:dangerous(a_obt=taro)'

't1:illegal_action(agt=taro)'<-
   't1:drunken_drive(agt=taro)'

S = [s1,s2]
```