

多重世界による時間論理システム

八田 匡史、 中川 裕志
横浜国立大学 工学部 電子情報工学科

本論文では、Uranusの多重世界記述をベースにした時間推移の表現を用いて、現実にある複雑な状態変化を柔軟に記述し、また推論するために必要な時間情報として逆向き半順序関係の導入と、それに伴う時間軸のループの発見のために、ユーザが注目する述語のリスト(Attention Predicate List)を限定する方法を提案している。また本システムへのテンポラル演算子の実装、さらに時間概念として時刻(Time Point)と時区間(Time Interval)の両方を扱う方法についても検討している。

WG AI 46-10 "Inheritance Based Temporal Logic System" (in Japanese)

Masashi HATTA and Hiroshi NAKAGAWA

Department of Electrical and Computer Engineering, Yokohama National University, Yokohama, 240, Japan.

Since some excellent models of temporal logic have been already known for us, we are now at a stage for efficient implementation. This paper proposes a representation and inference system based on an inheritance based logic programming language Uranus. In this system a temporal knowledge is given in the form of an extended partial ordered set. We make a time line from the given partial order relations set by a lazy construction method proposed in this paper. We also introduce both modal operations (ex. next time, always, eventually), and time interval representation in order for more powerful representation.

1. はじめに

自然言語の意味処理やエキスパートシステムなどにおけるさまざまな知識表現において、時間に関する知識を表現し、またそれらについて論証することの重要性が認識されている。しかし、planningなどの分野においては現実から遊離した抽象的な世界を対象にしているため、時間は直接の操作対象にはならず状態空間の変化という形でimplicitに扱われていた。そのため何もactionのない時間の推移の表現などの問題が残されていた^{2), 4)}。比較的古くから知られている様相論理(S4体系)を時間の推移に適用したtemporal logic³⁾は、現在の時刻から見て別の時刻におけるいかなるassertionが見得るか(いわゆる到達可能性)についての定式化を与えてくれる。しかしtemporal logicで表現された概念に関する推論はできるが、temporal logic自体比較的粗い表現力しかなく、現実の世界における時間の流れを精度良く表現することは困難である。時間概念の性質に関する厳密な議論はD. McDermottの論文⁵⁾において展開されている。彼は時間の連続性、事象(event)やプロセスの扱い、因果関係などについて述べ、さらに未来において可能性のある世界を表すtree状に分岐した時間軸についても言及している。我々はこのようなtree状に分岐した時間軸を扱うモデルとして、多重世界機構をベースにした時間論理システムを提案した⁷⁾。本稿ではこのモデルの拡張として、より効率的な時間情報の記述と時間軸上の推論に必要なAttention Predicate Listの導入について述べ、さらに時間インターバルを基本表現とする時間論理やテンポラル・ロジックとの対応について考察する。

2. 多重世界機構による時間軸の表現

本稿で提案するモデルは時間軸を離散的時刻によって構成する。この際、それぞれの時刻を表す世界はそれ以前の世界の状態に依存する点に着目し、Uranus⁶⁾の多重世界記述(within)を用いてこのような世界の状態を表現している。withinは第一引数として世界のネスティングを表わすリストを取る。リストの先頭が一番内側の世界で、それから順に外側へとネストする。第二引数には任意の述語呼び出しが書ける。例えば、

(within (T2 T1 T0) (P *X)) (2-1)

は、T2の世界にPの定義がない場合は外側の世界であるT1, T0へ定義の参照を行う。

ここで用いるモデルでは多重世界記述を用いて時間推移を次のように表現する。まず時間を世界の状態の変化ごとに区切り、区切られたそれぞれの状態をひとつの時刻として、時間の流れを状態の変化の積み重ねで表す。このときひとつの時刻にひとつの世界を対応させ、それぞれの世界を時間の流れに沿って時間の早い方が外側の世界になるようにwithのネスティングに展開すれば、継承(inheritance)によってある時刻ではそれまでに起こったことがすべて見えることになり、我々の持つ時間に対するイメージに合っている。

3. 時間情報

本節では、時間軸の元になる時間情報について述べる。我々の時間に関する知

識は絶対時間とはあまり関係がなく、他の事象(event)に対する相対的な関係としてとらえられる事が多い。したがってすべての事象を時区間としてとらえると、人間の知識(記憶)はこれら一連の事象の連続と考えることが自然である。また事象に付随する時間情報は、すべての事象を完全に順序付けたものではなく、任意の二つの時刻の前後関係の集合と考えられる。このモデルではこの二つの時刻の間の前後関係を“半順序関係”と呼ぶ。ここで二つの時刻A、Bの半順序関係を次のように表すものとする。

$$(\rightarrow A B) \quad (3-1)$$

これはAの後にBが起こった、すなわち時間的にBがAよりも後であることを示している。時間軸はこれらA、Bのような離散的時刻から、半順序集合の要素を順番にたどることにより構成される。

3.1 未来の分岐と条件付き半順序関係

人間にとって、未来には様々な可能性がある。つまり未来に延びている時間軸は一本ではなく、多数に分岐したtree状の構造を持つと考えられる。このことを実現するために、withinの第一引数である時間軸を表すリストを複数本用意し、時間の進行に伴って適切な時間軸に対応するリストを選択していく方法を考える。このときtree状に分岐した時間軸から複数の時間軸を取り出すと、時間的に早い部分は同一だが未来方向に進むと異なる時間軸群が得られる。例えば、

$$(TA3 TA2 TA1 T2 T1 T0) \quad (3-2)$$

$$(TB3 TB2 TB1 T2 T1 T0) \quad (3-3)$$

という二本の時間軸を考えると、この場合T0からT2までは同じ時間軸だが、その後TA1とTB1に分岐する。この複数の時間軸に分岐する時点での時間軸の選択は、外部から与えられる条件によって判断される。つまりT2の後に起こるのがTA1とTB1のどちらであるかということは、外部から与えられる条件を含む半順序関係によって示されることになる。このモデルは条件付き半順序関係を導入することにより、このようにtree状にに分岐する時間軸を扱うことができる。上の例のように、ある時点Aの次に起こることが条件により二通りあるような分岐点においては、次のように半順序関係を表す。

$$(IF CONDITION1 (\rightarrow A B)) \quad (3-4)$$

$$(IF CONDITION2 (\rightarrow A C)) \quad (3-5)$$

これはAの時点でCONDITION1が真であれば($\rightarrow A B$)を、CONDITION2が真であれば($\rightarrow A C$)をAに関する半順序関係としてみなすという意味である。

3.2 逆向き半順序関係

カウンタのような順序回路の例を考えた場合、一回のクロックパルスの入力を時間の更新とみなすと、回路への入力が高の場合とlowの場合の二通りの条件分

岐が時刻が進むごとに常に存在する事になり、時間の推移とともに起こる状態の変化は将来に向かって無限の可能性が生ずる。しかしこの無限に続く時間軸も、未来において回路の取り得る状態は幾通りもなく、同じ状態の推移を何度も繰り返しているに過ぎない。

この状態遷移を時刻の半順序関係を用いて表そうとするとき、あらゆる入力に対して可能性のある時間軸を記述しておくことは非効率的で、当然限界がある。そこで逆向き半順序関係という、過去のある時刻と同名の時刻に再び戻ってくるような時刻の半順序関係の記述が必要となる。

ここで『過去のある時刻と同名の時刻に戻る』という表現をしたが、将来において同じ時刻名が再び現れたとしても、その時点での世界の状態が過去に現れた同名の時刻での状態と同一であるとは必ずしも言えない。つまり逆向き半順序関係によって『過去の時刻に戻る』ということは、『過去の時刻とまったく同一の時点に戻る』という意味ではなく、未来における時間の推移の到達可能性を示唆しているにすぎない。したがって半順序関係から過去と同じ名前の時刻に到達したとしても、その時点での世界の状態が過去の同名の時刻と異なっていれば、まったく異なる世界として処理しなければならない。

このように時間情報として時刻の半順序関係を考え、未来における時間軸の分岐と時刻の繰り返しに対応する条件付き半順序関係や逆向き半順序関係を記述することによって、潜在的に可能なすべての時間軸を効率的に表現することができる。

3.3 時間軸のループ

このモデルでは、より効率的な時間情報の記述のために、逆向き半順序関係を導入した。しかしこの逆向き半順序関係によって、同じ状態遷移が将来に向かって無限に起こることが考えられる。つまり逆向き半順序関係によって時間軸にループができてしまう可能性が生ずる。したがってシステム上で将来における事象の成立などを質問した際、時間軸にループができてることにより処理系が無限ループに入り込むことを防ぐために、その時間軸のループを発見し、その先の時間軸での探索を打ち切るといった制御が必要となる。本節では時間軸のループができる場合について、そのようなループの判断について考察する。

逆向き半順序関係によって過去と同名の時刻に到達した場合、将来において再び同じ状態遷移が起こることによって、時間軸にループができることは容易に想像される。この時間軸のループのできかたには、大きく分けて次の二通りがある。

- ①ループの中に条件分岐点を含まない場合。(図1)
- ②ループの中に条件分岐点を含む場合。(図2)

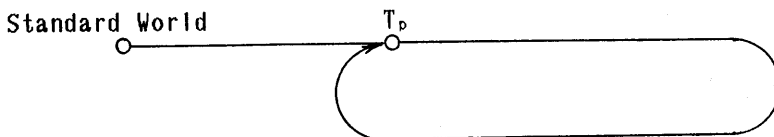


図1 条件分岐点を含まない時間軸のループ

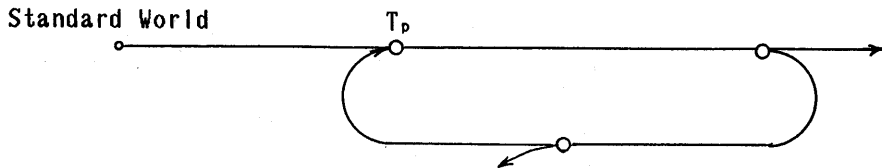


図2 条件分岐点を含む時間軸のループ

図1, 2で、時刻 T_p が時間軸のループポイントになるわけだが、区別するために時刻 T_p に到達した回数で $T_{p,0}$, $T_{p,1}$, $T_{p,2}$, ...、と表すことにする。つまり①の場合、Standard Worldから時間軸を構成していき最初に到達した時刻 T_p を $T_{p,0}$ 、一回のループで再び現れる T_p を $T_{p,1}$ とする。時間軸にループができていようかどうかは、このループポイント $T_{p,0}$, $T_{p,1}$, $T_{p,2}$, ...、の世界の状態を比較することによって判断する。

①の場合、逆向き半順序関係によって再び T_p に戻って来た時は、この $[T_{p,0} \sim T_{p,1}]$ の時間軸がループを成しているという事は一概には言えない。なぜなら $T_{p,0}$ において例えば述語 P の値が $(P A)$ だったとしても、 $[T_{p,0} \sim T_{p,1}]$ の時間軸上で状態が変化し $T_{p,1}$ では $(P B)$ となっていることも考えられるからである。したがってこの $T_{p,1}$ の世界の状態が $T_{p,0}$ と同一であるかどうかは、すべての述語についてその状態を比較する必要がある。しかし二周目からは、一周目の $[T_{p,0} \sim T_{p,1}]$ と同じ状態遷移が起こるので、再び T_p に戻っても $T_{p,2}$ における各述語の値は $T_{p,1}$ と同じであるから、 $[T_{p,1} \sim T_{p,2}]$ の状態遷移が無限回繰り返される時間軸のループであると見なすことができる。

②の場合、ループの中にある分岐点での条件が $T_{p,0}$ 以前の状態ですべて決定されていれば①の場合と同じように考えることができるが、ユーザからのINPUTに任されていたり、ループするごとに分岐点で選択される条件が変わる可能性があるような場合、将来再び現れる $T_{p,i}$ が必ずしも同じ状態になるとは言えず、またいつかはループから抜け出す様な条件判断が起こらないとも限らないので、時間軸の構成を途中で打ち切ることができない。

このようなループの判断には各世界における状態の推論が必要となるのであるが、あらゆる事象に対してその状態の変化をチェックすることは効率の面からも大変困難である。またすべての条件分岐点について『将来どの方向に分岐して行くのか』といった推論も必要となる。そのため世界の状態の推論を最小限に抑えるためにユーザが状態の変化を注目すべき述語名を限定してやる必要がある。そしてシステムはその述語に関してのみ推論を行うようにする。本稿ではこの注目する述語名の集合をAttention Predicate Listと呼ぶ。このAttention Predicate Listはシステムの初期設定の際にユーザから与えられ、システムが時間軸のループを発見する際に参照される。すなわちAttention Predicate Listの要素の状態が $T_{p,i}$ と $T_{p,i+1}$ においてすべて等しければ $[T_{p,i} \sim T_{p,i+1}]$ がループを成しているとみなす。また推論する際に問題となる将来における条件分岐点に関しては、条件述語にdefault値を与えることによって、ユーザの人力を無視して時間軸の構成を行うことにする。

このようにして、Attention Predicate Listを導入することによって推論する述語を限定すれば、ユーザの必要な情報に関して推論させることができ、計算量の効率化と共に、②のようなループの場合にも対処できることになる。

4. テンポラル演算子の実装

本節では、本システムへの時間に関する様相オペレータの導入と、その実行について述べる。まずその際に最も重要となる直後の時刻の概念(next time operator)について考察する。

テンポラルロジック³⁾に定義されているnext time operatorは $\bigcirc P$ (next P)と表現され、 $\bigcirc P$ は次の時刻にPが真であることを示す。本システムで \bigcirc (next)を実行する場合注意しなければならないのは、現在評価中の時刻が時間軸のtreeの末端で、 \bigcirc を評価しようとしても次の時刻が見つからないために評価できない場合が起こることである。ある時刻で $\bigcirc P$ を評価する場合、まず現在評価している時刻から半順序関係をたどることにより次の時刻を求める。この際、次の時刻がまだ時間軸上に構成されていない場合は時間軸の更新を行う働きをし、また時間軸のループの検出などもこの時に行われる。次の時刻が求めればその時刻において実行述語Pの評価を行うのであるが、次の時刻が見つからない場合にはtreeの末端の世界の状態がその先の将来においても永久に保存されると考え、treeの末端までの世界の状態に依って判断する。さらにこの \bigcirc (next)は次の(4-1), (4-3)に見られるように、 \square (always)、 \diamond (eventually)の終了条件を満たさなければならない。つまり \bigcirc による次の時刻がtreeの末端であるか、時間軸のループポイントであれば、その時刻以降の世界の状態遷移はその時刻以前と同様であると見なして、その時刻でPが成功すれば $\square P$ は常に成功し、その時刻でPが失敗すれば $\diamond P$ は常に失敗することになる。

次にこの \bigcirc (next)を用いた \square (always)、 \diamond (eventually)の定義を示す。

① $\square P$ (always P)

$$\square P \equiv P \wedge \bigcirc \square P \quad (4-1)$$

$$\begin{aligned} &(\text{assert } ((\text{always } *P) \\ &\quad *P \\ &\quad (\text{next } (\text{always } *P)))) \end{aligned} \quad (4-2)$$

② $\diamond P$ (eventually P)

$$\diamond P \equiv P \vee \bigcirc \diamond P \quad (4-3)$$

$$\begin{aligned} &(\text{assert } ((\text{eventually } *P) *P)) \\ &(\text{assert } ((\text{eventually } *P) \\ &\quad (\text{next } (\text{eventually } *P)))) \end{aligned} \quad (4-4)$$

5. 事象の成立区間の表現

本節では、事象の成立区間の扱いについて説明する。この事象の成立区間(Time Interval)と成立時刻(Time Point)の関係について考えてみよう。一般に人間は時間の概念をTime PointとTime Intervalの両方で把握していると考えられる。例えば、

We found the letter at twelve noon.
we found the letter yesterday.

という表現において、前者の“at twelve noon”とは事象の起こった正確な時刻 (Time Point)を表しており、後者の“yesterday”とは事象の起こった時間上の区間 (Time Interval)を表している。この表現から我々はTime PointとTime Intervalの二つの時間論理システムを併用していると考えるよりは、むしろ時間の表現に対する人間のあいまいさを表しているものと考えることが自然であろう。特に、先に述べたように時間概念のほとんどは、ある事象と他の事象との時間関係に関する記述により表現される。また、事象の起こった時間に関する概念は、あいまいな時区間で認識されることの方が多い。Allenはこの時間インターバルを基本表現とする時間論理について議論を展開している^{1), 2)}。また篠原等は時間関係と因果関係を融合して扱うシステム⁵⁾において、時間情報にはAllenと同様に時間区間を用いている。

このような事象の成立する区間(interval)の表現について考えてみる。『ある状態PがAである区間』というのは、時間軸上で(P A)がassertされてからそのassertionに対する否定知識によって時間軸上のinheritが切られるまでを指す。つまり、時間軸上で(P A)がassertされた時刻を述語(P A)の成立時刻、(P A)が否定された時刻を不成立時刻と見なすことができる。この成立時刻と不成立時刻から、ある事象の成立区間を述語intervalを用いて次のように表すことにする。

(interval 成立する事象 ((成立時刻1 不成立時刻1)
(成立時刻2 不成立時刻2)
.....)) (5-1)

このintervalを作成する方法には、次の二通りが考えられる。

- ①ある事象(P A)が成立する区間を発見する必要がある時に、時間軸上をひとつひとつ調べ、(P A)の成立している区間をリストにして返す。
- ②ある事象(P A)が成立した時点でその成立時刻をintervalにassertしておき、(P A)が打ち消されれば、その時点の不成立時刻としてintervalにassertする。この場合、不成立時刻が時間軸上に存在しない場合はNILとしておく。

このとき、①に比べ②の方法はあらゆる事象に対してintervalを登録しなければならないが、時間軸を構成する述語CONSTRUCTの実行の際に並行して登録しておくことが可能なので、探索の効率の面などに問題のある①の方法よりも有利である。しかし、バックトラックなどにより時間軸の再構成を行う場合に、登録された成立時刻や不成立時刻が時間軸から消滅する様なことも起こるので、注意が必要である。

この述語intervalの導入により、事象の成立区間が陽に表現されるようになった。このことによって、推論上の問題となる事象の成立区間の発見や、事象の時間関係や因果関係についての推論を、効率的かつ積極的に行うことができ、述語intervalを用いることによって時間区間を基本表現とする推論システムにも対応することが可能である。

ここで interval を用いた推論の例を示す。事象(Q W)が(P1 V1),(P2 V2)によってその状態を表現されている時、(Q W)の成立区間は(P1 V1),(P2 V2)の成立区間から求めることができる⁵⁾。

(Q W) :- (P1 V1) and (P2 V2).

↓

(interval (Q W)) := (interval (P1 V1)) and (interval (P2 V2)). (5-2)

(Q W) :- (P1 V1) or (P2 V2).

↓

(interval (Q W)) := (interval (P1 V1)) or (interval (P2 V2)). (5-3)

6. おわりに

本稿では Uranus の多重世界機構を利用した時間論理システムを提案した。本システムは時間情報として時刻間の半順序関係を用いることにより、tree 状に分岐する時間軸を柔軟に記述することができ、またその時間軸上での様々な推論を時刻と時区間の両方で行うことが可能である。

本システムは時刻の半順序関係が定義されている場合の時間に関する知識表現及び推論を扱っており、その応用として各種マニュアル等に現われる時間推移を含む手続きや操作系列に関する知識のような、状態の推移をあらかじめ半順序関係で定義することのできるシステムでの知識表現及び推論が挙げられ、各種エキスパートシステムや OS におけるマンマシン・インターフェースとしての応用に最適であろう。

【参考文献】

- [1] Allen, J.F.: "Maintaining Knowledge about Temporal Intervals",
Comm. ACM, Vol.26, No.11, pp. 832-843, November 1983
- [2] Allen, J.F.: "Towards a General Theory of Action and Time",
Artificial Intelligence, Vol.23, No.2, pp. 123-154, July 1984
- [3] Manna, Z.: "Verification of Concurrent Programs, Part1: Temporal
Framework", STAN-CS-81-836, 1981
- [4] McDermott, D.: "A Temporal Logic for Reasoning About Processes and
Plans", Cog. Science, Vol.6, pp. 101-155, 1982
- [5] 篠原, 寺野: "時間関係と因果関係を融合して扱うプロダクション・システム",
情報処理学会 知識工学と人工知能研究会資料 41-7, 1985
- [6] 中島秀之, 他: "Prolog/KRからUranusへ—多重世界機構の拡張—",
情報処理学会 知識工学と人工知能研究会資料 36-2, 1984
- [7] 柳田, 中川: "多重世界による時間概念の表現法",
情報処理学会 ソフトウェア基礎論研究会資料 12-2, 1985
- [8] 米崎直樹, 他: "時間論理プログラミング言語 Templog",
日本ソフトウェア科学会第1回大会 1E-4, 1984