人 工 知 能 89－4
（1993. 8. 18）

AND/OR 両並列性をいかしたプラン生成について

高橋 和子
〒661 尼崎市塚口本町 8-1-1
三菱電機株式会社 中央研究所
(TEL) 06-497-7141
(FAX) 06-497-7289
(E-MAIL) takahashi@sys.crl.melco.co.jp

AND/OR 両並列をいかしたプラン生成手法で得られた局所プランから全体プランを導き出す手法と全体プランの解釈について述べる。複数の可能性を持って行動するエージェント同志がメッセージを交換しながら各自が自分の局所プランを生成する手法は既に提案した。その結果得られる局所プランはメッセージの送受信を表すイベントの列として表される。局所プラン同志はメッセージによって一応対応付けられてはいたものの整合性や全体プランとしての意味が不明確であった。本稿ではこの局所プランの集合から全体プランを作る手法を示し、得られた全体プランに基づいて与えられた初期状態からゴール状態への遷移が可能であることを示す。

AND/OR-Parallel Planning

Kazuko Takahashi
Central Research Laboratory
Mitsubishi Electric Corporation
8-1-1, Tsukaguchi-Honmachi, Amagasaki, 661, JAPAN
(TEL) +81-6-497-7141
(FAX) +81-6-497-7289
(E-MAIL) takahashi@sys.crl.melco.co.jp

We describe the method of making a global plan from a set of local plans and its interpretation. We have already proposed the method of planning in which agents with multiple possible behaviors make their local plans by exchanging messages. As a result, a set of local plans are obtained. The local plan is represented by the sequence of events of sending or receiveing messages, but their consistency is not guranteed. In this paper, we describe the method of making a global plan from these set of local plans, and show that it is possible to transfer from the given initial state to the goal state, according to the obtained global plan.

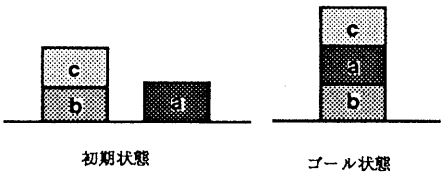

図 1: 積木ブロックの積み替え問題

1 はじめに

プラン生成とは与えられた初期状態とゴール状態をつなぐ一連の動作 / 操作を求めることである。並列性という見地からアプローチをかけたプラン生成は大きく 2 つに分けることができよう。1 つは分散 AI の枠組でとらえたものであり、もう 1 つは可能世界モデルで捉えたものである。

前者は集中的な管理者がシステム全体を見回して制御するのではなく、各エージェントが局所的な情報に基づいて独自に局所プランを生成することによって全体のプランをつくるという枠組で解こうというものである [Geo84][KR89][ZR91]。各エージェントを AND 関係にあるプロセスと見なし、黒板モデルを使ったりネットワークをはったりしてプロセス同志が何らかの形で連絡をとっている。この枠組でモデルを考える場合の問題は、複数のエージェントがとる行動に矛盾が生じた場合の扱いであり、交渉 (negociation) などで解決している [CW92][ZR91]。これらの手法では矛盾のない適当な解を見つけるものであり、全解を探索するものではない。

可能世界モデルに基づくプラニングでは全体を眺める supervizor の存在を仮定して全体を一つの状態と見なした状態遷移を考えている [Gin86]。状態遷移には複数の可能性があり、それぞれに対して異なる可能世界を対応させてすべての世界にわたって推論を行う。各可能世界は OR 関係にある。

[TTS90] では、両者をつなげた新しいアプローチとして AND/OR 両並列性をいかしたプラニング手法を提案した。この手法は全体を眺める制御プロセス不在の状況下で、局所状況のみを知る各エージェントが互いに通信し合いながら各自の局所プランを生成し、局所プランの集合として全体のプランを得るというものである。この手法は以下の特徴を持つ。

1. エージェント が自分に関する知識のみで動く。
2. エージェント 同志はメッセージを送り合って通信する。
3. 各 エージェント は非決定的に動く。
4. エージェント 同志は同時に動く。

分散 AI に基づく手法では一般に エージェント のとる行動に複数の可能性がある場合、重み付けやランダムな選択や他の エージェント との交渉などによって取るべき行動を決定する、という手法がとられるが、我々の示した手法では、各エージェント の行動とメッセージの届く順番に非決定性を導入して全解探索を施し（時に制約をいれなかえれば）すべての可能な解を求める、という方式をとっている。解として得られるのは局所プランの集合であり、局所プランはメッセージの送受信を表すイベントの列として表される。局所プラン同志はメッセージによって一応対応付けられてはいたものの、互いにつながりをもったメッセージの列の集合だけから実際のブロックの動きや全体的な状態の遷移を把握するのは難しく、これらの整合性や全体プランとしての意味が不明確であった。

本稿では、この手法で生成された局所プランの集合から全体プランを作る手法を示し、得られた全体プランに基づいて与えられた初期状態からゴール状態への遷移が可能であることを示す。

本稿の構成は以下のとおりである。まず、第 2 章で AND/OR 並列プラン生成手法に基づく局所プランの生成方法を示す。第 3 章では得られた局所プランの集合から全体プランを合成する手法を示し、全体プランに沿って矛盾なく状態遷移が起こっていることを示す。第 4 章では AND/OR 並列プラン生成手法における正当性について論じる。最後に第 5 章で結論および今後の課題を示す。

2 AND/OR 並列プラン生成手法

以下では [TTS90] で論じた積木の積み替え問題を取り上げる。これは、図 1のような 3 個の積木ブロックを初期状態 (a) からゴール状態 (b) に積み替えるプランを求める問題である。

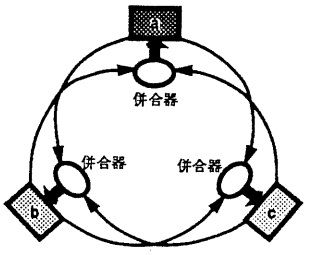

図 2: 通信モデル

2.1 モデル

各ブロックは自分の上に何もない時に限り動くことができる。テーブルはブロックを無限個置くことができる場所と考える。これはブロックの一時的な退避場所として使われる。

このモデルではブロックの動きに関して以下のような仮定をしている。

1. 各ブロックは自分の現在およびゴール状態における自分の局所的な位置（すなわち自分の上下）を知っており、ゴール状態に到達するように動こうとする。

2. 上に何ものっていないブロックが複数個ある場合、それらは同時に動くことができる。

3. 移動には有限時間を必要とする。

ブロック B の局所的な位置は $(On, Under)$ で表される。ここで $On, Under$ はそれぞれ B の上下にあるブロックである。上に何もない場合 On は $clear$ になり、下がテーブルの場合は $Under$ が $table$ になる。各ブロックの間には図 2に示すようにチャネルがはられ、ブロックの動きや要求を伝えるメッセージがその動きが起こった瞬間に送られる。メッセージには $move(B)$, $clear$, $remove(B)$ の 3 種類がある。ただし B はブロックの名前である。各ブロックは送信用と受信用の 2 種類のチャネルを持つ。送信用チャネルは他のブロックに自分の動きや要求を伝播し、受信用チャネルは他のブロックから送られてきたメッセージを併合 (merge) して受信する。

このモデルでは、2 つの非決定的な要素を導入している。1 つはブロックのとる行動の非決定性であり、各ブロックは自分のゴールを満たすための能動的行動 (active action) とメッセージとして受信した別のブロックの要求を満たすための受動的行動 (passive action) の 2 通りの行動を取り得るものとする。もうひとつはメッセージの到着順序であり、1 つのブロックに複数のブロックからメッセージが送られて場合、到着順が非決定的になる。これらの非決定性があるためにブロックの行動には複数の可能性が生じるが、このモデルではすべての場合に対して局所プラン生成を行う。従って、ある場合は局所プランが得られるが、ある場合では無限ループやデッドロックに陥ってしまうこともある。しかし、そのように解の得られない場合があっても、可能な解はすべて集めることができる。

2.2 行動規約

ブロック B の現在の状態を $(On, Under)$、ゴール状態を $(FOn, FUnder)$ とすると、B の行動規約は以下の通りである。

能動的な行動

$On = FOn$ かつ $Under = FUnder$ (ゴールに到達した) ならば
　停止する。
$Under \neq FUnder$ ならば
　$On = clear$ ならば
　　$FUnder$ に $move(B)$ を送り
　　$Under$ に $clear$ を送り
　　現在の状態を $(clear, FUnder)$ に更新する。
　$On \neq clear$ ならば
　　On に $remove\_from(B)$ を送り

現在の状態を $(clear, Under)$ に更新する。

受動的な行動

$move(B')$ を受け取ると、
　$On = clear$ ならば
　　現在の状態を $(B', Under)$ に更新する。
　$On \neq clear$ ならば
　　On に $remove\_from(B)$ を送って
　　現在の状態を $(B', Under)$ に更新する。
$remove\_from(B')$ を受け取ると、
　$Under \neq B'$ ならば
　　そのメッセージは無視する。
　$Under = B'$ ならば
　　$On = clear$ ならば
　　$table$ の上に移動して
　　現在の状態を $(clear, table)$ に更新する。
　　$On \neq clear$ ならば
　　On に $remove\_from(B)$ を送って
　　$table$ の上に移動し、
　　現在の状態を $(clear, table)$ に更新する。
$clear$ を受け取ると、
　現在の状態を $(clear, Under)$ に更新する。

この規約に基づいてプログラムを書いて実行すると可能な局所プランの集合がすべて求められる。

3 プラン生成

3.1 局所プランの集合の解釈

各ブロック の局所プランはブロック名とその局所プランの組 $(B, LocalPlan)$ で表される。ここで、$LocalPlan$ は $[start, (E_1, k_1), (E_2, k_2), \ldots, (E_n, k_n), end]$ で表される列である。ただし、(E_i, k_i) はイベントとその識別子の組である。イベントは $move\_to(B)\&clear, remove(B), accept(B), cleared, escape, ignore$ のいずれかである。ただし B はブロック名である。これらのイベントはメッセージの送受信に対応する。

$move(B)$ というメッセージの送信は $move\_to(B')$ というイベントとして記録され、目標 B' に向かって出発するということを表す。受信は $accept(B)$ というイベントとして記録され、B を受け取ったことを表す。

$clear$ というメッセージの送信は $clear$ というイベントとして記録され、自分がその位置から離れたことを表す。受信は $cleared$ というイベントとして記録され、自分の上にブロックがなくなったことを表す。

$remove\_from(B)$ というメッセージの送信は $remove(B')$ というイベントとして記録され、B' を除去するということを表す。受信は $escape$ としてイベントとして記録され、テーブルの上に退避したことを表す。

$move\_to(B)$ と $clear$ は全く同時に発せられるため、一つのイベントとして捉える。

メッセージを出してから相手に届くまで時間がかかることから要求メッセージを受け取った時に既にその要求が満たされてしまっている時がある。その場合は $ignore$ というイベントで記録される。

あるメッセージを送信するブロックがあると、必ずそれを受け取るブロックがある。メッセージは識別子を伴って送受されるため同一メッセージを送信するイベントと受信するイベントには同一の識別子がつく。

たとえば、以下のような局所プランの集合が求められる[1]。

```
% Plan 1:
( a, [start,(move_to(b),#96),(clear,#96),(accept(c),#1015),end] )
( b, [start,(accept(a),#96),(remove(c),#297),end] )
( c, [start,(escape,#297),(move_to(a),#1015),(clear,#1015),end] )
```

[1] これは並列論理型言語 ANDOR-II[TTS90] でプログラムを記述し実行した結果得られてもののひとつである。

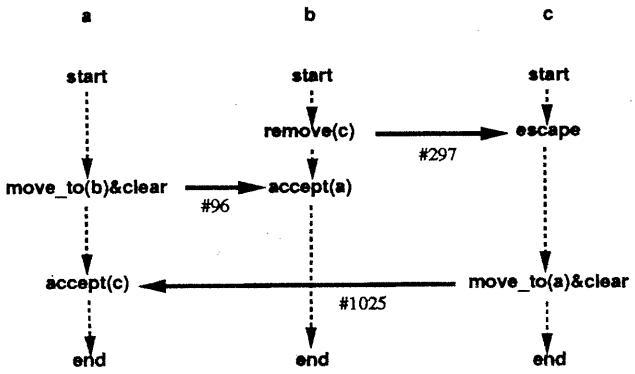

図 3: Plan 1

各ブロックの局所プランの意味は以下の通りである。ブロック a はまず、下のテーブルに自分の移動を告げながら b に向かって移動し、次に c を受け取る。ブロック b は c をのけて a を受け取る。ブロック c は まず a に移動を告げてテーブルの上に避難し、次に テーブルに移動を告げながら a に向かって移動する。

縦軸を時間、横軸を各ブロックとして この局所プランの集合から以下のようにして図 3 を書くことができる。各ブロックの局所プラン $((E_1,k_1)\ldots,(E_n,k_n))$ に対して、 上から順に縦軸にそって $start, E_1,\ldots,E_n, end$ を置く。次に、同一の識別子を持つ各メッセージに対して 送り手側から受け手側に向かって矢線を引く。この矢線はその識別子のついたメッセージがブロックの間で送られたことを示す。この図の全体プランとしての解釈は複数通り考えられる。なぜならイベントの間には半順序しか付けられていないので、同時に起こったり部分的にオーバーラップして起こるイベントに関する制約が甘く、制約を満たすことのできる様々な全体プランが考えられるからである。たとえばブロック b によるブロック c の除去とブロック a の ブロック b へ向けての出発に順序的な制約はない。従ってこの 2 つのイベントはどちらが先に起こってもかまわないし同時に起こってもかまわない。また、ブロック b がブロック a を受け取る前にブロック c がブロック b に向けて出発してもかまわない。

たとえば以下のような解釈が可能である。

解釈 1：

1. ブロック a はブロック b へ向けて出発し、ブロック b は自分の上からブロック c をテーブルの上に除去する。

2. ブロック a はブロック b の上に到着し、ブロック c はブロック a に向けて出発する。

3. ブロック c はブロック a の上に到着する。

全体プランは $\sigma_1,\sigma_2,\ldots,\sigma_n$ で表される列である。ただし、各 σ_i は ブロック B とイベント E の組 (B,E) の集合である。

すると、この解釈に対応する全体プランは $\sigma_1,\sigma_2,\sigma_3,\sigma_4$ となる。ただし、

$$\sigma_1 = \{(a, move\_to(b)\&clear), (b, remove(c))\}$$
$$\sigma_2 = \{(b, accept(a)), (c, escape)\}$$
$$\sigma_3 = \{(c, move\_to(a)\&clear)\}$$
$$\sigma_4 = \{(a, accept(c))\}$$

別の解釈をすると、

解釈 2：

1. ブロック b は自分の上からブロック c をテーブルの上に除去する。

2. ブロック c はテーブルの上に避難し、ブロック a はブロック b へ向けて出発する。

3. ブロック a はブロック b の上に到着し、ブロック c はブロック a に向けて出発する。

4. ブロック c はブロック a の上に到着する。

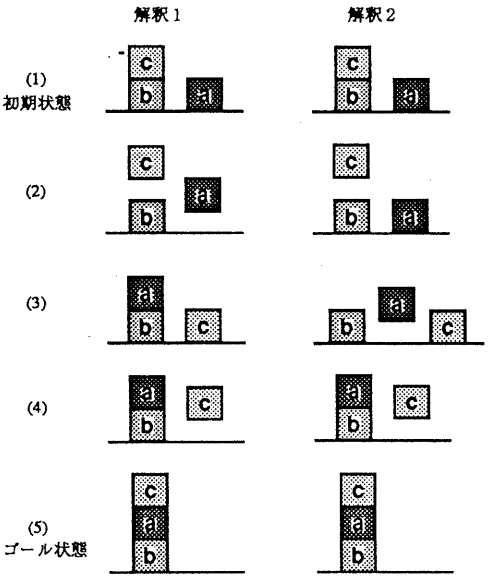

図 4: メッセージに基づくプランの解釈

この解釈に対する全体プランは $\sigma_1', \sigma_2', \sigma_3', \sigma_4'$ である。ただし、

$\sigma_1' = \{(b, remove(c))\}$
$\sigma_2' = \{(a, move\_to(b)\&clear), (c, escape)\}$
$\sigma_3' = \{(b, accept(a)), (c, move\_to(a)\&clear)\}$
$\sigma_4' = \{(a, accept(c))\}$

図 4にこれら 2 つの解釈に対応する図を示す。解釈 1 の第 3 段階から第 4 段階でブロック c は冗長な動きをしている。c はもともとテーブルの上に避難するつもりで出発したのだが、途中で進行方向を変更して直接 a の上にのってもかまわないと思われる。しかし、このアルゴリズムではメッセージを送る（目標に向かって出発する）と必ず満たされないといけないため、途中で目的地を変更することはできない。一方、局所プランの集合は Plan 1 以外にも求まるが、c が最初に能動的な動きを選択するような局所プランでは、図 4における解釈 1 の第 3 段階を通らず、第 2 段階から第 4 段階へ直接遷移するものが得られる。

3.2 全体プランの合成

局所プランの集合から全体プランを合成する手法について述べる。

イベント E が イベント F よりも時間的に先に成り立つ時に $E < F$ と表記するとする[2]。全体プランを合成する際には以下の 2 つの制約条件がある。

1. 局所プラン内での順序に対する制約
   局所プランを $[start, (E_1, k_1), \ldots, (E_n, k_n), end]$ とすると、 $E_1 < E_2 < \ldots < E_n$ が成り立つ。

2. メッセージの送受信順序に対する制約
   異なるブロック の局所プランに存在する同一の識別子を持つイベント E_i, F_j に対して E_i がそのメッセージの送信者、 F_j が受信者である時、 $E_i < F_j$ が成り立つ。

第 2 の条件はメッセージの受信が送信より前に行われてはならないことを表す因果関係保持の制約である。ただし、メッセージを出す原因となったものと実際の行動との間の因果関係は必ずしも保持されていない。

[2]本稿では時間はすべて全ブロックに共通な絶対的な時間をさす。

ブロック B の局所プランを $[start,(E_1,k_1),\ldots,(E_n,k_n),end]$ とする。全体プラン $\sigma_1,\ldots,\sigma_m$ は以下の手続きで合成される。σ_j は各 B に関しては高々1つのイベントしか含まない。また、どんなイベントも同時に2つの σ_i,σ_j $(i\neq j)$ に含まれる事はない。メッセージ X の送信を表すイベントを $S(X)$ で、受信を表すイベントを $R(X)$ で表す。

全体プランの合成

$E_1,\ldots,E_{i-1}$ は、すでに $\sigma_1,\ldots,\sigma_{j-1}$ のいずれかに含まれるものとする。

集合 σ_j $(j=1,\ldots,m)$ は以下の規則に沿って決定される。

1. E_i が $S(X)$ の時、$\sigma_1,\ldots,\sigma_{j-1}$ に $R(X)$ が含まれないならば E_i を σ_j の要素とする。

2. E_i が $R(X)$ の時、$\sigma_1,\ldots,\sigma_{j-1}$ に $S(X)$ が含まれれば E_i を σ_j の要素とする。

$S(X)$ には $move\_to(B)\&clear$, $remove(B)$ の2つがあり、$R(X)$ には $accept(B)$, $cleared$, $escape$, $ignore$ の4つがある。任意の X に対して $S(X)$ は唯一存在するが $R(X)$ は2個存在する場合がある。これは $move\_to(B)\&clear$ というイベントが2つのメッセージをそれぞれ異なる ブロック に送っているためである。

この手順で得られた列 $\sigma_1,\ldots,\sigma_n$ は上記2つの制約条件を満たしている。また、任意のメッセージの組 X,Y についてブロック B_1 の局所プランで $R(X)<S(Y)$ かつ ブロック B_2 の局所プランで $R(Y)<S(X)$ となることはない。

3.3 状態遷移

局所プランの直接表すものはイベントすなわちメッセージの送受信の列であり、上で述べた解釈はいわばメッセージの送受信に沿ったプランの解釈である。しかし、これでは実際に全体の状態がどのように遷移していくのか明確でない。以下では、局所プランの集合から全体の状態遷移が導出可能であり、初期状態から状態遷移を繰り返してゴール状態に辿り着くことができることを示す。

まず、状態 (state) を fragment と呼ばれる要素の有限集合と定義する。fragment には以下のものがある。

- $on(X,Y)$　　ブロック X はブロック Y の上に直接のっている。
- $clr(X)$　　ブロック X の上には何ものっていない。
- $air(X)$　　ブロック X は空中を移動中である。

ただし、X はブロック、Y はブロックまたは $table$ である。

全体プランを $\sigma_1,\ldots,\sigma_n$ とする。f_E を σ_i $(i=1,\ldots,n-1)$ の要素をすると、ある状態 S において f_E が起こることによって状態が S' に変わることから、f_E は集合に対する操作と見なすことができる。以下の規則に従うと f_E によって S から次の状態 S' が導き出される。操作 f_E によって状態 S が S' になる場合、$S'=S-S_1+S_2$ と表すことができる。ただし、S_1, S_2 は fragment の集合であり、空集合のこともある。$S-S_1$ は 集合 S から集合 S_1 の要素を取り去る操作を示し、$S+S_2$ は 集合 S と集合 S_2 の和集合をとる操作を示す。

(1) $f_E=(X,move\_to(Y)\&clear)$ の場合
$$S'=S-\{clr(X),on(X,Z)\}+\{air(X)\}$$
(2) $f_E=(X,cleared)$ の場合
$$S'=S+\{clr(X)\}$$
(3) $f_E=(X,accept(Y))$ の場合
$$S'=S-\{clr(X),air(Y)\}+\{on(Y,X),clr(Y)\}$$
または
$$S'=S-\{air(X),air(Y)\}+\{on(Y,X)\}$$
後者は $(X,accept(Y))$ の特別な場合であり、空中ドッキングを表す。3つ以上のブロックのドッキングも可能だが、ドッキングしたものが次の時刻にどこかに移動することは禁じられている。

(4) $f_E=(X,remove(Y))$ の場合
$$S'=S-\{on(Y,X),clr(Y)\}+\{clr(X),air(Y)\}$$
(5) $f_E=(X,escape)$ の場合
$$S'=S-\{air(X)\}+\{on(X,table),clr(X)\}$$
(6) $f_E=(X,ignore)$ の場合

$S' = S$

状態 S に対する操作 f_1 によって状態が $S_1 = S - S_{11} + S_{12}$ になり、状態 S に対する操作 f_2 $(f_1 \neq f_2)$ によって状態が $S_2 = S - S_{21} + S_{22}$ になるとする。この時、f_1 と f_2 の積操作を S から $S' = S - S_{11} \cup S_{21} + S_{12} \cup S_{22}$ を導き出すものと定義する。個々の操作によって変化する部分は1つのブロックに関する fragment のみであることに注意。同一の状態に対する操作は異なるブロックに対するものであるから、S_{11} と S_{12}、S_{21} と S_{22} はそれぞれ共通部分を持たない。

f_i を $f_E \in \sigma_i$ のすべての f_E の積操作とすると、f_i によって S_i から次の状態 S_{i+1} が導き出せる。以上により、状態の列 $S_0, S_1, \ldots, S_n$ を得ることができる。

3.4 全体プランの正当性

このようにして得られた状態の列 $S_0, S_1, \ldots, S_n$ に対して、各 S_i が以下の性質を満たす時に $\sigma_1, \ldots, \sigma_n$ が正当な全体プランであると見なす。

各状態は無矛盾でなければいけない（無矛盾性）。従って各状態は以下の性質を満たす必要がある。

1. $\forall Z.X \neq Y \wedge Z \neq table \supset \neg(on(X,Z) \wedge on(Y,Z))$
   ブロックが同時に2つのブロックの真上になることはできない。

2. $\forall X.X \neq Y \supset \neg(on(X,Y) \wedge on(X,Z))$
   ブロックが同時に2つのブロックの真下になることはできない。

3. $\forall Y.\neg(on(X,Y) \wedge clr(Y))$
   ブロックが同時にあるブロックをのせかつ何ものせないことはない。

4. $\forall X,Y.\neg(on(X,Y) \wedge on(Y,X))$
   どんなふたつのブロックも互いをのせあっていることはない。

5. $\forall Y.\neg(on(X,Y) \wedge air(Y))$
   $\forall X.\neg(on(X,Y) \wedge air(X))$
   ひとつのブロックは空中にいる時は他のどのブロックとも切り離されている。

6. $\forall X.\neg on(X,X)$
   どのブロックも自分自身の上にのることはない。

また、各状態ではすべての ブロック の位置が決っていなければならない（極大性）。従って各状態は以下の性質を満たす必要がある。

$\forall X.((on(X,Y) \wedge clr(X)) \vee (on(Y,X), on(X,Z)) \vee air(X))$

さらに、ゴール状態に達することが保証されなければならない（ゴール到達性）。従って S_n はゴール状態に等しくなければならない。

4 例

第2章で述べた2つの全体プランから対応する状態の列を導く。

まず、初期状態 S_0 とゴール状態 S_G は以下のようになる。

$$S_0 = \{clr(c), on(c,b), on(b,table), clr(a), on(a,table)\}$$
$$S_G = \{clr(c), on(c,a), on(a,b), on(b,table)\}$$

解釈1に対する状態の列は以下の通りである。まず、S_0 に対して σ_1 を適用する。

$(a, move\_to(b)\&clear)$ という fragment は

$$S_0 - \{clr(a), on(a,table)\} + \{air(a)\}$$

という操作に対応し、

$(b, remove(c))$ という fragment は

$$S_0 - \{clr(c), on(c,b)\} + \{clr(b), air(c)\}$$

という操作に対応する。

従ってこの2つの積操作

$$S_0 - \{clr(a), on(a,table), clr(c), on(c,b)\} + \{air(a), clr(b), air(c)\}$$

を S_0 に施すことによって

$$S_1 = \{on(b, table), air(a), clr(b), air(c)\}$$

が得られる。

同様にして、S_1 に σ_2 の 2 つの fragment の積操作

$$S_1 - \{clr(b), air(a), air(c)\} + \{on(a, b), clr(a), on(c, table), clr(c)\}$$

を施す事によって

$$S_2 = \{on(b, table), on(a, b), clr(a), on(c, table), clr(c)\}$$

が得られる。

同様にして、

$$S_3 = \{on(b, table), on(a, b), clr(a), air(c)\}$$
$$S_4 = S_G = \{on(b, table), on(a, b), on(c, a), clr(c)\}$$

がをそれぞれ得られる。

各段階では極大性および無矛盾性が成り立っており、S_4 はゴール状態になっている。各状態は図 4の解釈 1 における各段階に相当する。

解釈 2 に対しても同様にして、以下の状態の列を得る。各状態は図 4の解釈 2 における各段階に相当する。

$$S_1' = \{on(b, table), clr(a), on(a, table), air(c), clr(b)\}$$
$$S_2' = \{on(b, table), clr(b), air(a), clr(c), on(c, table)\}$$
$$S_3' = \{on(b, table), clr(a), on(a, b), air(c)\}$$
$$S_4' = S_G = \{on(b, table), on(a, b), clr(c), on(c, a)\}$$

5 議論

AND/OR 両並列性に基づく手法によって局所プランの集合をつくるアルゴリズムの正当性に関して議論すべき点がいくつかある。

5.1 冗長性の除去

局所プランの集合をつくる規則は冗長性に対して何も行なっていないので、同じ動作を繰り返して行なうブロックが出現する場合がある。このような冗長な行動を除去できれば無駄な解を求める必要がなくなる。しかし、一般に、全体プランを見つけてから繰り返しを除去するのは可能だが、局所プラン生成時に繰り返しを単純に除去してはいけない。なぜならば、全体プランを達成するためにいったん満たされたゴールを再び崩す必要があるのもしれないからである。

5.2 停止性

能動的な行動は自分のゴールに到達すれば停止して他のブロックに自分の停止を伝える信号 ('eos' など) を送る。受動的な行動は他のブロックがすべて停止したことがわかれば ('eos' を受け取ると)、自分はゴール状態を達成していなくても停止する。最終的にプランは常に能動的な行動の方で生成されるので受動的な動きは停止状態を考慮にいれる必要はない。

また、この行動規約ではプロセスがゴール状態でもないのに動かない場合も出てくる。たとえば $(clear, FUnder)$ に対しては、$FUnder$ がゴール状態のものになっていれば、たとえこれがゴールでなくてもこれ以上能動的な行動はしない。しかし、他のブロックから要求メッセージを受け取るとこのプロセスは受動的な動きをする。自分がゴール状態であるべきブロックの上に既にのってしまうとこのブロックにとっては自ら動く必然性はないし、自分のゴール状態を満たすために他のブロックを呼び寄せるのもこのブロックのすべきことではないからである。

各 ブロック に対するプロセスはプランができるまで停止しない。また、可能世界の数は無限に増える可能性がある。以上 2 つの理由から全体的な停止性は保証されない。

6 おわりに

本稿では AND/OR 並列をいかしたプラン生成手法によって得られた局所プランから全体プランを求め、与えられた初期状態とゴール状態をつなぐ矛盾のない状態遷移が得られることを示した。

局所プランを合成して全体プランをつくるものの多くはいったん局所プランをつくってしまってからそれを合成する際に矛盾が生じないように適当な合成方法を見つけるものである [All91][Dea86][?]。本稿で述べた手法では、局

所プラン作成時に空中で待機することによって暗黙の内に矛盾を回避するので、得られる局所プランの集合からは必ず矛盾のない全体プランが生成できる。

今後の課題としてまず本稿で述べた形式化に基づいてこのプラン生成法の正当性の厳密な証明を行う必要がある。また、本稿では1つの局所プランの集合に対する議論であったが、AND/OR並列をいかしたプラン生成手法では複数の解が得られるので、複数解の比較を行うことによって最短プランや同一解の除去に対しても考察する必要がある。

参考文献

[All91] J. F. Allen. Temporal Reasoning and Planning. In *Reasoning about Plans*, pp. 1–68, 1991.

[CW92] M. K. Chang and C. C. Woo. Negotiation and Goal Relaxation. In *Decentralized A.I.-3*, pp. 31–54. North-Holland, 1992.

[Dea86] T. L. Dean. Interactability and Time-Dependent Planning. In *reasoning abount Actions and Plans*, pp. 245–266. Morgan Kaufmann Publishers, Inc., 1986.

[Geo84] M. Georgeff. A Theory for Multi Agent Planning. In *Proceedings of AAAI-84*, pp. 121–125, 1984.

[Gin86] M. Ginsberg. Possible World Planning. In *Resoning about Actions and Plans*, pp. 213–243. Morgan Kaufmann Publishers, Inc., 1986.

[KR89] M. J. Katz and J. S. Rosenstein. Plans for Multiple Agents. In *Distribute Artificial Intelligence, Volume II*, pp. 197–228. Morgan Kaufmann Publishers, Inc, 1989.

[MK89] M.Dummond and K.Currie. Goal Ordering in Partially Ordered Plans. In *Proceedings of International Joint Conference on Artificial Intelligence*, pp. 960–965, 1989.

[TTS90] A. Takeuchi, K. Takahashi, and H. Shimizu. A Parallel Problem Solving Language for Concurrent Systems. In M. Tokoro, Y. Anzai, and A.Yonezawa, editors, *Concepts and Characteristics of Knowledge-Based Systems*, pp. 267–296. North-Holland, 1990. Also appearing as ICOT Technical Report TR-418.

[ZR91] G. Zlotkin and J. S. Rosenschein. Negotiation and Goal Relaxation. In *Decentralized A.I.-2*, pp. 273–286. North-Holland, 1991.