3A−3

制約充足機構を用いた作業プランナの設計*

谷口 憲　大和田 勇人　溝口 文雄†

東京理科大学 理工学部‡

1 はじめに

作業プランナには，作業目標からそれを達成するための計画すなわち一連の操作を自動生成する能力が要求される．我々は，領域-独立 (domain-independent) な AI プランナ [1] に制約充足機構を付加するというアプローチにより，それを記号レベルから幾何レベルまでより柔軟に推論できるよう拡張した作業プランナの実現を目的としている．その際，問題解決における有効な手段として位置づけられている制約論理プログラミングの枠組みでプログラムを記述する．本論文では，その宣言性，モジュール性に着目して設計された作業プランナの計算機構を提案する．システムは制約論理型ロボット言語 PRL(Prolog-based Robot Language) [2] 上で構築されている．

2 作業プランナの設計指針

2.1 制約論理プログラミングによるアプローチ

従来の手続き型言語による作業プランナでは，作業-依存部と制御部のプログラムが混在しているため，開発者は作業に固有な制約の他にその制約を処理する手続きまでも記述しなければならなかった．このようなプランナは一般性があるとはいえない．我々は制約論理プログラミングの宣言性，モジュール性に着目し，できるだけ制御部を変更しないで作業-依存の制約を充足させるような計算システムを設計する (図 1参照)．また，制約論理プログラミングを用

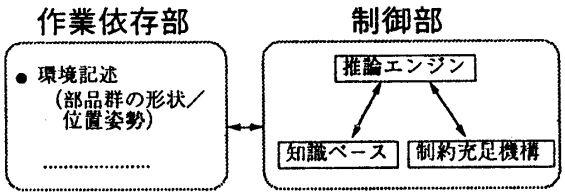

図 1: システム構成

いることによって次のメリットが得られる．

- 記号と数値が混在した記述を 1 つの言語上で統一的に扱える．
- 計算の方向性がなく，入出力が自由に選べる．
- 記述の追加，削除が容易である．

2.2 プラニングの方法

従来の AI プランナが対象としていたブロックワールドは非常に抽象化されており，部品の記述や部品間の関係記述を容易に与えることができる (例えば，block(a),block(b),on(a,b) のように)．しかし，現実性の高い組み立て作業では，目標状態での部品群の関係記述を抽象化した表現で直接記述することは困難である．そのため，部品群を組み立てるプランを生成する前に，目標状態でのそれらの接続関係 (組み合わせ方) を求めておく必要が生じる．そこで，我々は以下の 2 つのプロセスを行なえるようにプランナを設計する．

- **組み合わせプロセス** (configuration process):
  必要な部品群の姿勢記述から，次の制約を充足させる組み合わせプラン (部品の位置姿勢を変換する写像系列) を生成する．

  non_overlap : 複数の部品を空間的に重なりなく配置するための制約．

  connected : 2 つの部品を結合 (接触) させるための制約．

  変換写像として，次の**回転移動写像** (rotational mapping)f_r と**平行移動写像** (translational mapping)f_t を考える．

  $$f_r \in \{r \mid 0° \leq r \leq 360°\}, f_t \in \{t \mid 0° \leq t\}$$

  ただし，r,t は離散的な値をとるものとする．

- **組み立てプロセス** (assembly process):
  必要な部品群の 3 次元位置と組み合わせプランから，次の制約を充足させる組み立てプラン (ロボットの動作系列) を生成する．

  grav : 部品を重力に反しない姿勢で配置するための制約．

  removable : ロボットが完成品を組み立てる際の部品配置方向が，物理的に可能であるための制約 (候補となる部品組み合わせが取り外し可能かチェックする)．

  動作として，**把握動作** (pickup) と**配置動作** (set) を考える．

2.3 対象部品

我々は，モデルベーストロボティクスの考え方に基づき，次のような比較的簡単な幾何学的形状の部品群を対象とする．すなわち，部品の形状はその最小単位となる**要素立体** (1 つの面が m 角形の n 面体) が結合したものであるという前提を設ける．このような部品の $360/m$ 度きざみの姿勢は $m \times n$ 個存在する．

3 プラニングの基本的な考え方

状態S において命題P が成り立つことを`holds(P,S)` で，状態S から写像/動作A の結果生じる新しい状態へと遷移させる関数を `do(S,A)` で表す．このとき，状態に関する次の実現可能性規則によって状態を遷移させることができる．

```
holds(P,do(S,A)) :- causes(A,S,P).
holds(P,do(S,A)) :- holds(P,S),not(disabled(A,S,P)).
```

ここで，`causes/3` は写像/動作の実行可能性規則を，`disabled/3` はフレーム規則を表す．実行可能性規則の例を次に示す．

*Design of a Task-planner with Constraint Satisfaction Mechanisms
†Ken TANIGUCHI,Hayato OHWADA,Fumio MIZOGUCHI
‡Faculty of Sci. and Tech., Science Univ. of Tokyo

```
<1> causes(pickup(Obj),S,holding(Obj,Es)) :-
        holds(on_floor(Obj),S),
        consist_of(Obj,Es).
<2> causes(rotate(Obj,Ro,Es1),S,locate(Obj,Es2)) :-
        holds(holding(Obj,Es1),S),
        rotational(Ro,Es1,Es2).
```

上の規則<1>,<2>は，把握動作，回転移動写像に関するもので，それぞれ次のような宣言的意味を持つ．

1. 状態Sにおいて，部品Objが作業環境内の床面上に存在し，そのときの姿勢がEsならば，その部品に把握動作pickupを適用可能である．適用の結果，ロボットが部品を把握しているという新しい概念holding(Obj,Es)が成り立つ．

2. 状態Sにおいて，部品Objがロボットに把握されており，そのときの姿勢Es1が回転Roによって姿勢Es2に変換できるならば，その部品に回転移動写像rotateを適用可能である．適用の結果，その部品が姿勢Es2をとるという新しい概念locate(Obj,Es2)が成り立つ．

これに初期状態s0で成り立つ概念holds(on_floor(a),s0)を与えれば，次のような問い合わせが可能である．

```
?- holds(locate(a,Es),S).
S=do(do(s0,pickup(a)),
          rotate(a,rm(z,pi/2),[e1,e2,...]))
Es=[e1',e2',...]     %最終姿勢: 要素立体の集合で表現
```

この解Sはlocate(a,Es)が成り立つ状態を，Esはそのときの部品aの姿勢を表している．Sは，部品aの初期状態に把握動作，回転移動写像(rm(z,pi/2): z軸まわりに$\frac{\pi}{2}$)を適用した新しい状態であることを宣言的に意味する．また手続き的には，部品aを把握し，グリップの回転を行なうという，locate(a,Es)を成り立たせるための操作手続きを証明していると解釈できる．以下の章で，このような考え方に制約の概念をとり入れたプラニングの方法について述べる．

4 制約概念に基づくプラニング

4.1 形状表現

部品や完成品を構成する要素立体の相対的な配置関係を用いることで，形状モデルを宣言的に表現できる．例えば図2の部品aを考えるとき，要素立体の相対位置は述語p/3(第1,第2,第3引数にそれぞれx,y,z方向の符号付き距離の値をとる)で表せる．このとき，

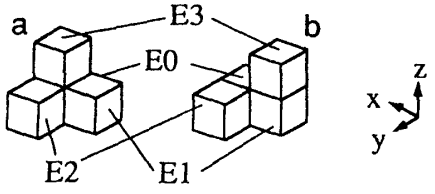

図 2: 要素立体の位置関係

要素立体間に成り立つ制約(接続関係)を宣言的に表現することによって，その制約をすべて充足するような部品の姿勢を求めることができる．部品aの姿勢記述を次に示す．

```
consist_of(a,[E0,E1,E2,E3]) :-
   left_of(E0,E1),    %左 (x 成分)
   beyond(E0,E2),     %奥 (y 成分)
   on(E3,E0).         %上 (z 成分)

left_of(p(Ax,Ay,Az),p(Bx,By,Bz)) :-
   Ax = Bx + 1,Ay = By,Az = Bz.
   ......................
```

次の問い合わせで，制約解消系が部品aの姿勢を返す．

```
?- consist_of(a,Es).
Es=[p(0,0,0),p(-1,0,0),p(0,1,0),p(0,0,1)]
yes
```

4.2 プラン生成

複数の部品の組み合わせ/組み立てでは，2.2章で挙げた制約を充足する部品の位置姿勢を求めなければならない．例として，次のように表現される制約non_overlap,connectedを充足させるプランについて考える．

```
non_overlap(p(Ax,Ay,Az),p(Bx,By,Bz)) :-
   diff(Ax,Bx);     %Ax と Bx は異なるという制約
   diff(Ay,By);
   diff(Az,Bz).

connected(E1,E2) :-
   left_of(E1,E2);left_of(E2,E1);
   beyond(E1,E2);beyond(E2,E1);
   on(E1,E2);on(E2,E1).
```

例えば，次の問い合わせによって，2つの部品a,bを結合するためのプランが生成できる．

```
?- holds(locate(a,Es1),S),
   holds(locate(b,Es2),S),
   forall(E1,Es1,forall(E2,Es2,non_overlap(E1,E2))),
   exist(E1,Es!,exist(E2,Es2,connected(E1,E2))).

S=do(do(do(do(do(do(s0,pickup(b)),
  rotate(b,rm(z,pi/2),[p(0,0,0),...])),
  translate(b,tm(3,3,1),[p(1,0,1),...])),
  pickup(a)),
   ..............  )).
Es1=[p(3,3,1),...]     %部品 a の最終姿勢
Es2=[p(2,3,0),...]     %部品 b の最終姿勢
yes
```

このプランは，状態に関する実現可能性規則と制約解消系の双方によって充足されるもので，3章で述べたように宣言的意味と手続き的意味は等価である．

5 おわりに

本論文では制約論理プログラミングの宣言性，モジュール性を利用した領域独立な作業プランナの設計について述べた．論理式と幾何学的数値データを統一的に扱う計算機構を実現できることから，制約論理プログラミングによるアプローチは強力であるといえよう．

参考文献

[1] Seth A. Hutchinson, Avinash C. Kak : Spar - A Planner That Satisfies Operational and Geometric Goals in Uncertain Environments,*AI MAGAZINE*,SPRING 1990.

[2] 谷口, 大和田, 溝口 : 論理型ロボット言語を用いた Hand-Eye システムの設計, 人工知能学会第6回全国大会,1992.