特　集「知能ロボット」

プラニング

Planning

安部 憲広*
Norihiro Abe

* 大阪大学産業科学研究所
ISIR Osaka University.

1990年9月14日　受理

Keywords: planning, operator, goal interaction, execution, assembly.

1. はじめに

達成すべき目標(goal)の記述から，それを達成する計画(plan)すなわち一連の動作(action)を求め，実行するシステムの実現が望まれている．そのようなシステムに関する研究は人工知能研究の初期から行われており，さまざまなアイディアに基づくシステムが考案されている．それらは概ね対象分野を特定することなく一般的な計画問題を解こうとするものと，対象分野に依存した知識をうまく利用して計画問題を解こうとするものとに分類される．後者は，応用人工知能，特にエキスパートシステムの研究に属すると考えられるため，本稿では取り扱わないことにする．ただし，記号レベルのプラニングのみでは，例えばロボティクス関係の問題はほとんど解決したことにはならないため，ロボットによる組立作業問題やナビゲーションに関連した問題については取り扱うことにした．しかしながら，本稿の第一目的が，広範な分野に対して適用できるプラニングの知識表現や手法についての解説であることはいうまでもない．

これまでに多くのプランナがさまざまな分野を対象として開発されている．しかし残念ながら，実用レベルで使われるほどには完成度は高くない．実用的レベルで使用されているスケジューラ等のプログラムはエキスパートシステムとして開発されているものが多く，現状ではそうしたシステムのほうが性能が高いことも事実である．しかし，エキスパートシステムの弱点は経験のない事例を適切に処理できないことであり，深い知識に基づくエキスパートシステムが望まれている．この意味では，対象分野とは独立に開発されたプランナはエキスパート技法を用いて作られたプランナよりは将来的には適用性が高くなる可能性を有しているといえよう．

ほぼ30年間に行われてきたプラニングの研究の大部分は，計画立案に焦点が置かれていて，計画ができれば直ちに実行が可能との前提に立っていた．しかし旅行計画などの例を考えれば明らかなように，計画時と実行時とで状況(世界の状態)が変化しているのが普通であり，計画と実行の双方を考慮する必要がある．世界の状況が変化するということは，計画立案者以外に行動する者が存在することを意味し，マルチエージェント(multiagent)による計画立案と実行に関する研究も行われつつある．また，不測事態に対処し得るプラニングなどの研究も行われている．

しかし，世界が変化しない(静的世界)という仮定でのプラニングの問題がすべて解決されたわけではなく，既存のシステムの限界を考慮しつつ，動的世界でのプラニングを考える必要があろう．そのような意味でも，静的世界におけるプラニングの現状と限界について考えてみよう．

2. 計画と動作の表現

人間はさまざまな領域の問題や，初めて出合った問題を解く能力を有しているが，これと同等な能力を持ったプログラムの開発研究から一般問題解決プログラム(General Problem Solver : GPS)[1]が生まれた．GPSは与えられた目標(end)と初期状態との差異を減じる手段(means)のなかから適当な作用素(operator)を発見する(mean-end法)ことにより，状態間の差をしだいに縮めていくことを基本戦略とし

```
Pickup (X)
PRECONDITION: Ontable (X)
              Hand-empty
              Clear (X)
DELETE LIST:  Ontable (X)
              Hand-empty
              Clear (X)
ADD LIST:     Holding (X)

UNSTACK (T, U)
PRECONDITION: On (T, U)
              Hand-empty
              Clear (T)
DELETE LIST:  Clear (T)
              Hand-empty
              On (T, U)
ADD LIST:     Holding (T)
              Clear (U)
```

図1　STRIPSの代表的作用素

た．しかしGPSの作用素の表現はあまりに単純で現実問題の表現には適しているとはいえなかった．豊富で強力な表現法の一つとして述語論理がある．McCarthyは状況計算（situation calculus）を用いて計画問題を解くことを提唱した[2]．例えば，puton (A, B) によりon (A, B) が作られることを次のように記述した．

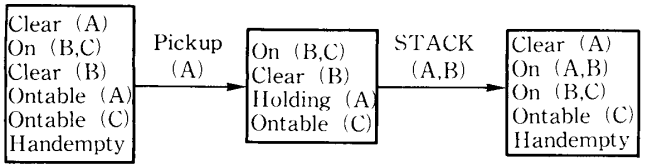
holds (clear (A), S) ∧ holds (clear (B), S)
→holds (on (A, B), result (puton (A, B), S))

(holds (p, s) は状況sでpが成立することを，result (p, s) は状況sでpを行った後の状況を指す)

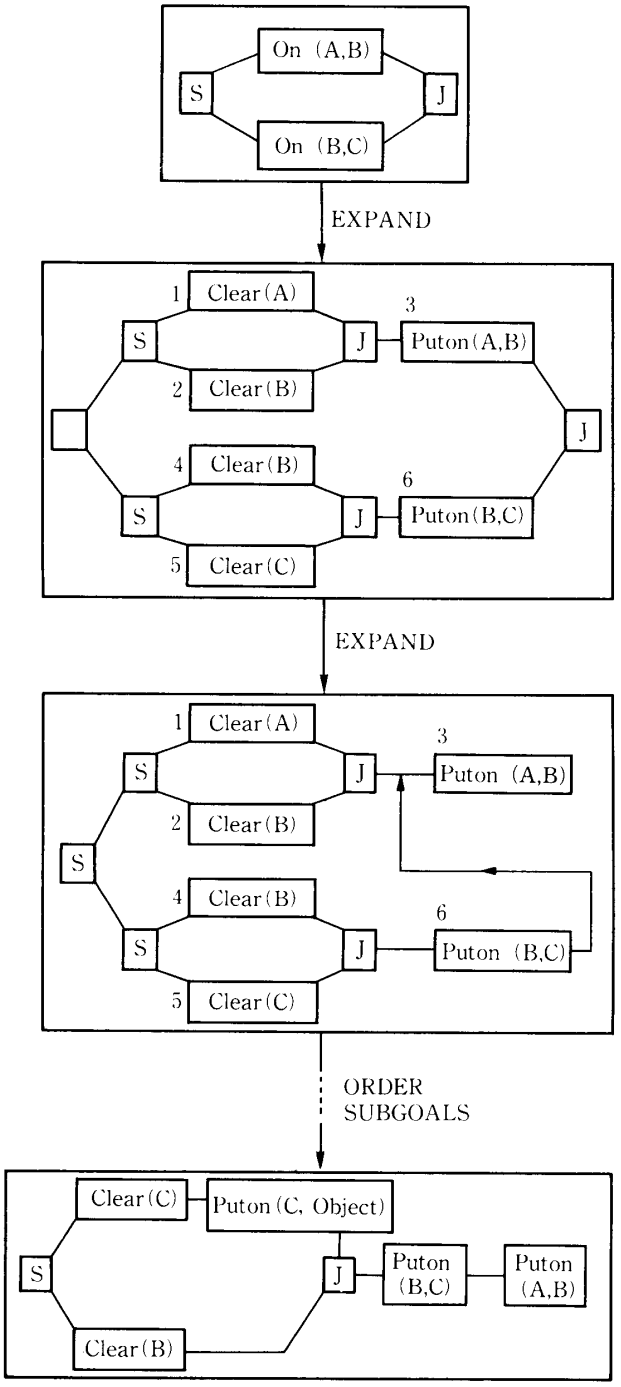
しかし，puton (A, B) という動作によって影響されないものを記述するフレーム公理の記述が導出過程の組合せ爆発を招く結果となった．そこでフレーム公理の記述と状況変化のたびにその時点で成立している物事を求める必要のない表現法がSTRIPSにより提案された[3]．図1にSTRIPSの代表適用作用素の例を示す．例えば，ロボットが物体を置くことができるのはputonの前提条件（precondition）が成立する場合であり，その行為によって変化するものは削除リスト（delete list）と付加リスト（add list）に記されたものだけであり，他は変化しないと考える（STRIPSの仮定）．この考え方はその後開発されたプランナでも一般的に採用されているが，現実性に欠ける仮定でもある．しかし，論理システムとしてのSTRIPSの正当性の条件が解明されているため，プラニングにおける学習の題材としてSTRIPSが利用されることも多い．その最初の試みはSTRIPSを拡張したMacropsによって行われ，マクロ作用素の獲得による計画能力の向上が報告されている．しかしながら，単純なチャンキングによる一般化は不必要なマクロ作用や，過剰に一般化されたマクロ作用素を生成したため，実際には計画能力を高めることに失敗した．これに対し説明に基づく一般化（explanation-based-generalization）技法を用いることによって，有効で過度に一般化されないマクロ作用素の獲得が行われるようになっている[4][5]．しかし，獲得された作用素は記号レベルにとどまっており，ロボティクスも含めたより現実的な問題解決への貢献はまだ見られない．

許される動作が決まれば，可能な動作の重複順列内で与えられた目標を満足させるものが求める計画となる．この意味では初期状態から始めてあらゆる動作を適用し，その結果が目標と一致するまで探索を続ければ，目標に矛盾がない限り解は発見できる．例えば，{on (A, B), on (B, C)} という目標を実現する計画は図2(a)のようになる．この場合探索しているのは状態の空間であり，1970年代中期までのプランナはこの型に属していた．しかし単純にこのような方式をとれば，簡単な積木の世界でも最適の計画を得る問題はNP-困難となることがわかっている（最適の計画とは冗長な動作を含まないプランを意味する）．そこで一般に状態空間型のプランナはGPSの手段目的法に似た作用素の適用，つまり未決の目標と照合する付加リストを持った作用素を適用するというヒューリスティックを用いる．しかしそのようにしても，例えばholding (X) という目標に対して図1のpickupとunstackが適用すべき作用素の候補として選ばれることになり，プランナは選択を強いられることになる．一般的には縦型探索に基づいて何れか一方が選ばれる．このように状態空間型のプランナでは，目標に至る中間状態を具体的に求める必要があるためバックトラックは避けがたい問題となる．

状態空間型のプランナとは異なり，部分計画の空間を探索するプランナがある．図3に示す簡単な問題を図1の作用素を持った手段目的型のプランナに解かせると，二つの目標をどの順に行っても，最初に達成した目標が2番目の目標達成時に破壊されてしまうことがわかる．つまり，目標間に干渉があって，単純にそれぞれの目標を順番に解けば元の目標が達成できるという線形仮説が成立しない例になっている．与えられた目標達成の失敗経験を利用して技術（skill）の獲得を目指したHAKERは結局この種の連言形の目標を一般的に解くことができなかった[6]．Waldinger[7]は最初の目標を解いた際に課せられた制約を保護しつつ，第2の目標を解いていくという方法を考案したが，



（a） 状態空間の探索



（b） 部分計画空間の探索

図 2 探索としてのプラニング

その手続きには曖昧な点があり一般性に欠けている．同様な試みを多くの研究者が試みたが，あまり良い結果が得られなかったため紹介は省略する．

この問題は Sacerdoti の NOAH[8] により解かれたが，その基本的な考え方を図 2 (b) に示す．図 3 に示した問題を解くために，まず与えられた目標を併置した後それぞれを展開する．puton (B, C) の結果は puton (A, B) の前提条件を侵さないため，puton (B, C) の後で puton (A, B) を行うように部分計画が変更される．しかし puton (A, B) の前提条件 clear (A) は初期状態では成立していないため，clear (A) の実現が試みられる．このように明確に順序付けが可能と判断されるまで決定を行わない考え方を最小拘束 (least commitment) 法と呼ぶ．部分計画の考え方を用いるプランナの利点は動作間の時間や因果関係の表現が柔軟であること，計画が最初から半順序関係を有していて，並列動作による実行にも適していることである．しかし，NOAH が計画変更用に用いた作用素の効果の一般性には疑問点もあり，その計画能力はロバストでないといわれている．

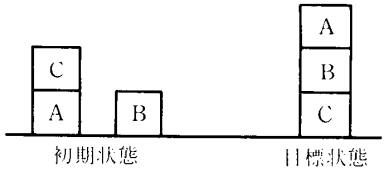


図 3 線形仮説の成立しない例

3．連言目標探索の改良

上述したように初期のプランナは連言形の目標を解く際に，どのような順に個々の目標を解くべきかを考えていないため，非常に冗長な計画を作ることを防止できなかった．1970 年中期以後のプランナのほとんどは，目標の重要性と干渉の回避を目指して考案された．その基本的アイディアは四つに大別される．

その最初は Abstrips[9] が用いた，目標をその重要性や優先度によってレベル分けし，重要なものから順に解こうとする考え方である．最も重要な目標とは，達成が困難な目標や，計画によって特性の変化が困難な属性を持ったものに関する目標である．これに対し最も容易な目標とは，それを達成し得る代替計画が常に発見できるような目標をいう．Sacerdoti は移動ロボットの問題では次のようなレベル分けを与えている．

レベル 1　TYPE, COLOR
レベル 2　INROOM
レベル 3　PLUGGED-IN, UNPLUGGED
レベル 4　NEXTTO

しかし，所与の目標を達成するために，設定すべき階層を決定する一般的方法は不明である．またこのシステムでは上位階層へのバックトラックができず，階

層の設定によっては問題が解決不能になる場合もあった．これに対して NONLIN[10] では，上位階層へのバックトラックを可能にしている．その際単純なバックトラックではなく，計画立案に失敗した原因となる階層に戻って再計画することにより，失敗原因に無関係な目標に対する計画を保護している．NONLIN はリンクと動作の展開および矛盾解消用の順序付けを用いて部分計画の変更を行ったが，図3の問題を解く過程が文献(11)に詳述されているため，本稿では省略する．達成すべき目標の抽象化のアイディアは，XCON や RI-SOAR，さらには Unruh ら[12]でも検討されており，階層の自動設定を目指した研究が始められようとしており，興味深い．

2番目は，解くべき目標の選択を十分な情報が得られるまで引き延ばすアイディアである．MOLGEN ではネズミインシュリン遺伝子を持った媒体を内包する細菌の培養を得るという目標に対するプラニングにおいて，遺伝子の取込み操作法，取り込む媒体等の具体化をすぐに行わず制約(constraint)という形で蓄積していく[13][14]．その理由は，媒体の制約が後の培養作成過程になって初めてわかる場合があるため，不用意な媒体の決定はそれ以後の培養の全過程をむだにしてしまうからである．ただし注意すべきは，問題解決に必要な制約がいつでもすべて求め得るとは限らない点である．制約不足の場合にはデータベースの参照や生成・検査，領域に依存したヒューリスティクスの利用が不可欠である．しかしながら，この制約の生成・充足という考え方はこれからの問題解決の一つのパラダイムであると考えられ，SIPE[15] や後述する Spar にも採用されている．

MOLEN では与えられた目標記述から，遺伝子取込みの実験過程の探索と，対象の持つべき特性に関する制約とを同時に求めている．しかし遺伝子取込みの成功例が一つ既知であれば，説明に基づく一般化を利用することにより，成功例の持つべき条件(制約)を得ることができる．したがって，さまざまな場合に対する計画の成功例があれば，同じ計画が適用可能な条件を求めることにより，後の計画に利用することが可能となる．これは後述する事例に基づくプラニングそのものである．

図4に SIPE の PUTON の定義を示す．ブロックではない OBJECT1 と BLOCK1 上にあるものを並列的に取り除いた後，BLOCK 1 を OBJECT1 に積むことが，目標記述(GOAL)と手続き記述(PROCESS)の両方を用いて示されている．また RESOURCES によってその動作に必要な資源の型がわかるため，変

```
OPERATOR: Puton
ARGUMENTS: (Block1, Object1) IS NOT Block1;
PURPOSE: ON (Block1, Object1);
PLOT:
    PARALLEL
        BRANCH1:
            GOALS: Cleartop (Object1);
            ARGUMENTS: Object1;
        BRANCH2:
            GOALS: Cleartop (Block1);
            ARGUMENTS: Block1;
    END PARALLEL

    PROCESS
    ACTION: Puton, PRIMITIVE;
    ARGUMENTS: Object1;
    RESOURCES: Block1;
    EFFECTS: On (Block1, Object1);
END
```

図4 SIPE のオペレータ Puton

数を具体化しなくても他の動作との間で資源の競合が生じるか否かが判定できる．このため，Noah のように具体化して初めて資源の競合がわかるといった欠点がなくなっている．SIPE では制約記述言語によって望ましい変数の具体化等のヒューリスティクスの記述も容易になっている．さらに STRIPS 等のシステムで用いられていた STRIPS の仮定に従わない動作の効果の記述が，動作とは別の演繹ルールによって行えるようになっている．SIPE では EFFECT がその動作の真の結果を表しており，STRIPS の付加リストのように副作用と結果とが混同されることがない．

3番目は目標間の干渉が生じないように目標を順序付ける非線形のプランナである．その最初が NOAH であることはすでに述べた．NONLINE, SIPE, DEVISER[16] は，非線形プランナに時間や資源および矛盾解消法等を採用している．これらのシステムは計画立案の道具としては有用であるが，論理的なプラニングとしては必ずしも妥当でないといわれている．しかし，論理的に正しいといわれる TWEAK[17] は，制限された行動の表現しか許されないため，非常に狭いクラスの計画しか取り扱えない．また非線形のプランナでは，計画が完成するまで世界の状況が確定しないため(線形では各時点で世界が確定)共同効果[18](synergy effect)が問題となる．

状態空間型のプラニングでは与えられた目標に対して課す部分目標間の制約が強すぎるため，後でバックトラックが生じる．一方最小拘束法の場合には，逆に部分問題間に存在する固有な関係を考えずに，問題解決の過程を通じて部分目標の順序を決めようとするため，誤りを含んだ計画や冗長な計画が得られるという欠点がある．そこで与えられた問題固有の関係を利用

してあらかじめ部分目標間の順序関係を静的に求めておき，その関係を満たすコスト最小の解決順序を求めようとする考え方がある．例えば，目標g1を「箱1を部屋1に置く」，目標g2を「ロボットが部屋2にいる」として，G＝g1∧g2とすれば明らかにg1をg2よりも先に達成すべきである(g1＜g2)．また，与えられた目標g1, g2以外に明示的には述べられていないがg1とg2とを達成するために必要となる暗黙の目標がある．今の例では，「ロボットは部屋1にいる」，「ロボットは箱1にのみ近づく」，「箱1のそばには何もない」といった目標が，暗黙の目標となる．この強化された目標間の順序制約を満たすコスト最小の解を最良優先探索で求めればよい[19]．さらに，この関係を部分目標上の部分集合に拡張することによってより強い目標間の制約が求められる[20]．例えば「箱1を箱2の隣に置く」という目標g3が加えられた，G＝g1∧g2∧g3に対しては，g1＜g3, g1＜g2, g1∧g3＜g2という関係が得られる．ただこの方式の導入によるプラニングの高速化の評価はなされていないが，結果は楽観的ではない．

4．計画の再利用

ここまでに議論してきたプランナは目標間の干渉に難渋した．干渉を防ぐために要する計算時間も問題となる．こうした問題点を解消する一つの方法は，過去に作られた計画を利用して現在与えられている問題を解くことであり，事例に基づくプラニング(case based planning)の考え方を導入することである．この考え方はプラニングだけでなく，立案された計画が実行時に破綻した場合の再計画(replanning)にも適用することができる．そのためには過去の計画がどのような条件下で作成されたか，また目標を実行する動作間にはどのような依存関係が存在するかを記録した注釈構造(annotation 構造)が必要となる．Priar[21]は計画された各動作に対し，動作の効果内で計画の残部で利用されているもの(e-cond)，動作の実行中に保存されるべき物(p-cond)，動作の前提条件(e-precond)を記録しておく．問題が与えられるとPriarは現在の問題と最も近い問題に対する計画を選出し，その計画に従って問題解決を行うが，前提条件が一致しないと適用不能な動作が出現する．その動作の代替案が複数個あるとき，①保存するe-condの数，②p-condの保存数，③e-preconditionの保存数，の順に代替案を順序付けることにより，最適の代替案で計画を補修する．例えばサンフランシスコの誕生会へ出席する計画として次のような計画がすでに作られていたとする．

GO-BY-CAR (Airport) → BUY (Ticket)
→ GOTO-SHOP (Giftshop, Airport)
→ BUY-FROM-SHOP (? Gift)
→ TAKE-FLIGHT (sf)

与えられた問題の解法にこの計画を利用しようとしたころ，空港には土産物店がなく，GOTO-SHOP (Giftshop, Airport)が成立しないことがわかった．そこで他のGOTO-SHOPとして次に示す二つの方法がわかっているとしよう．

A : GOTO-SHOP (Candyshop, Airport)
B : GOTO-SHOP (Giftshop, Plaza-near-home)

今適合しなかったGOTO-SHOPの各条件が

e-cond : at (? shop), sell (? shop, ? Gift)
p-cond : at (Airport), poss (Ticket), poss (Money)
e-precond : at (Airport)

であるとすれば，Aはp-condのat (Airport)やe-precondのat (Airport)を保存するが，Bはこれらを保存しないため，Aが代替案として選ばれる．

Plex[22]もこれとほぼ同様な考え方を用いて，ニューヨークの地下鉄を利用した際の計画を用いて，サンフランシスコの高速鉄道利用の計画を立てているが，切符の購入，自動改札口で切符の回収の差異の解消法の選択基準が明解でない点で問題がある．しかし上例において，空港を離れて近所の店まで戻るとむだな経費を費やすことになるが，空港へ行く前に近所の店で贈り物を購入するのがいっそう適切な計画であるのは明らかである．このような再計画はPriarでは困難と思われる．事例に基づくプラニングで問題となるのは，類似した事例の発見方法である．この点に関しては，説明に基づく一般化によって得られた制約を利用できるが，膨大な事例の組織立ては今後の研究に待たねばならない．

Priarの考え方をより一般化したアイディアとして，生成—検査—テスト法(Generate & test & debug : GTD)がある[23]．これは問題解決に必要な修復の候補を生成し，その候補による修復の結果を評価することによって候補生成器を修正して生成器の能力を向上させようとするものである．その効果は同質の問題に対する同様な失敗に限定されるが，学習能力がある点でPriarより優れている．

GTDより優れたプランナとして説明に基づくものがある[3]．これは対象領域に関する知識を用いたシミュレータが，立案された計画の評価を行う．計画が

悪い場合は失敗を導き出すが，その際に用いられたシミュレーションルールを一般化し，それをプランナに与えることによって同種の失敗を防止しようとするものである．この考え方はGTDよりも強力であるが，事前に与えられるシミュレーションルールの量の問題と，関連するルールだけを取り出して適用できるか否かでその有効性を決定される．

5．計画と実行

これまでに述べたプランナは世界に関して完全な知識を有しているとの前提に立っていた．したがって，作られた計画は誤りなく実行された．しかし，ロボットが計画を実行してみると，指定された動作が行えなかったり，行った後の状態が計画と異なり以後の動作が実行不可能となることが生じる．前者の原因としては，モデルと現実との誤差のほかに予測外の事象による世界の変化が考えられる．動作の効果についても，ロボットによる操作誤差のほかに，STRIPS仮定が成立しない状況などが考えられる．例えば他のロボットの動作が世界の状態を変化させることもあり，計画通りの実行は保証されない．

静的プランナの代表ともいえるSIPEでも時間や動的なプロセスは取り扱えない．また既存のプランナは立案に時間をかけすぎるうえに，その方法も固定されていて，状況に合わせた目標の実現，目標順序の変更，あるいは目標の中断，棄却等の臨機応変な動作が行えない．

これらの問題に対処する方策は次の四つに大別される．

① 動作の前提条件が成立しないときに再計画を行う．

② 予想される失敗に対処する再計画を与えておく．

③ 計画と実行とをインタリーブする．

④ 複雑な計画立案をやめてあらかじめ与えておいた計画の中から状況に適合する計画を選択する．

①については，計画の再利用が望ましく，再利用ができないときのみ再計画すべきである．②については，想定される状況に対してあらかじめ対処法を与えておいたり，実行時に何を監視すべきかを決め，失敗が生じたと判断された場合に対する再計画を与えておく方法がある．Shoppersは考えられ得るすべての場合に対して計画を生成するuniversal plan[24]を提案しているが，その実用性については議論がある[25]．網羅的な計画立案は，あまりに非効果的である．また，予測される事象よりも予測不能な事象の生じる可能性が高く，この方法は実用性にも疑問がある．これに対して③は，計画の一部がすぐに実行されるため，世界の変化の間隔に比して計画と実行の切換えが短ければ，計画通りに事が運ばれるうえ，常に正確な情報の確保が保証されることになる[26]．しかし，変化の激しい状況への対応はやはり困難である．これに対して④は急激な変化にも追従可能なことが期待されている．これは時間のかかるプラニングを棄て，センサ値と動作との対から適切な計画を選択し，リアクティブなシステム構築を目的とする．

Georgeff と Lansky の PRS (Procedural Reasoning System)[27]は，プランを最初から作るのを止めるとともに繰返しや再帰的動作を含んだ計画の実行を可能とするため，プリコンパイルされた手続き的知識を有している．そして状況に対処し得る手続き的知識の中から，メタプラニングの知識を用いて最も妥当と思われる手続きを選択するシステムを目指している．すなわち，KA (Knowledge Area) に手続き的知識とメタプラニングの知識が与えられていて，例えば，(＊GOAL (!(AT-ROOM PERSON1))) という目標が与えられると，PRSはこの起動パターンにマッチするKAを選び出す．そのようなKAの例を図5に示す（ここで (!P) はPを達成せよ，(?P) はPが成立するか，(⇒P) はデータベースにPを記録する，を意味する）．図からわかるように，KAは，達成すべき副目標の系列を与えられており，必要に応じて次々と別のKAが起動されていく．PERSON 1の部屋 (troom) が位相な地図上でどのウィングの廊下に面しているか，ロボットの現在位置 (froom) がどこかを求めた後，その間の経路を Planned-path で計画し，部屋を出て (room-left) 計画された道を followed-plan に従って辿る．詳細は略すが，階層的に組織付けられたKAが適用されて経路が求められる．このナビゲーションロボットのKAは，①途中の経路を計画するKA，②実際にステアリングや速度を決めるKA，③障害物回避などの緊急事態対処用のKAからなり，この三つの並列実行が可能となっている．特に③のKAは，特定の事実に反応した起動が許されている．しかし，競合するKAの並列実行によって目的外の結果が発生しないよう，メタレベルの知識によって調停をする必要があり，目的通りにシステムを機能させるのは容易ではない．実際にメタレベルのプラニング知識をどのように用いるかが重要な課題となろう．

現状では特に優れたメタレベルの知識が用意されて

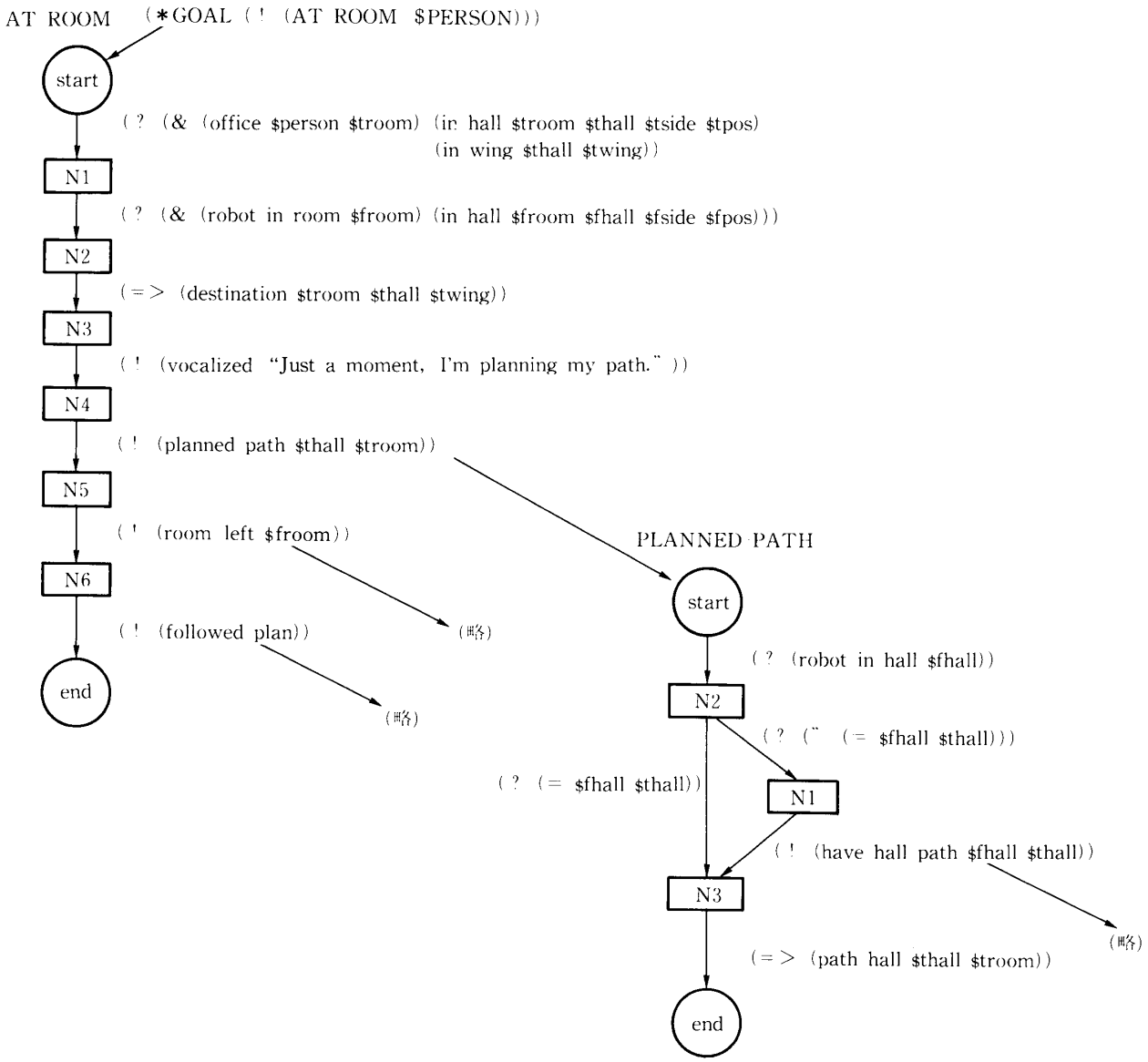


図 5 ナビゲーション用ロボットのKAの例

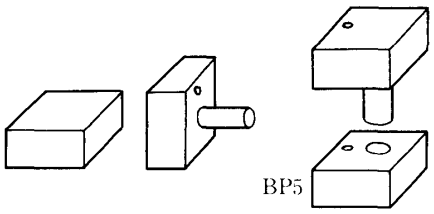
いるとはいえない．最初からプリコンパイルされた知識を与えることによりすべての状況に対処できるのかといった点にも疑問がある．実際に作られたシステムでは，以上の問題以外にリアクティブな振舞いも実現されているとはいえず，これからがいかにしてその理想とするシステムを実装するかの正念場となろう．

メタレベルの知識はプロダクションシステムでも提案された考え方であるが，そのアイディアが高度であればあるほど実際にはあまりうまく機能しなかった．このシステムが同じ轍を踏まないことを期待したい．

行動するロボットやエージェントが複数になると問題はより複雑である．他のエージェントの動作からその目標や意図を知るのはアブダクション（abduction）であり，その解決は非常に困難である．それらを既知としても競合と協調の取扱いは今後の問題であり，この議論については本誌ですでに取り上げられている[28][29]．

6. 組立と計画

計画し行動するロボットの例として組立ロボットの可能性を考えてみよう．そのためには視覚や障害物回避が必要だが，ここでは組立作業の自動化をプランナに行わせる際の問題点だけを考えよう．問題はAIプランナが非力なことと，プランナに目標記述を与えることが困難な点にある．例えば，模型のエンジンの構造を記述してみれば，部品の記述と部品間の関係記述の複雑さに驚かされる．そこで可能な手段として，完成品を見せたり[30]-[32]，完成品を作る手順を示すことにより完成品の記述を求めさせる方法がある[33]．後者の場合組立手順を与えているように見えるかもしれないが，示した手順がロボットに適したものとは限らない点に注意されたい．



（a） 初期状態　　（b） 作業指令

図 6 作業指令と初期状態

前者の一例として大嶽らの研究を考えよう．作業スキーマはSIPEに類似した記述形式で与えられ，非線形プランナによってまず目標構造の組立手順が決定される．次にアーム，ハンド，カメラ，および部品形状，安定把持等を考慮した動作が立案される．このシステムで実際に問題となったのはこの際に用いる世界モデルの記述や作業の副作用に対する処置であり，それらが組立対象に依存せざるを得なかった点である．つまり，プランナが用いる知識や事象の表現の抽象度が高すぎる．そのため作業と動作の両プラニングの分離を招いている．現実的には，視覚を用いて推定した対象の構造を作業目標とすること自体に限界があると思われる．

後者の例では，部品間の結合関係を簡単な言語で与えることにより，組み立てる部品間にどのような座標変換が必要かを計算させて完成品の状態を決める．次に完成品の状態から可能な分解手順をすべて生成し，その中から組立容易性を評価することにより組立手順を求める[34)-(36)]．この手法は本稿の意味ではプラニングとはいえないが，分解の因果関係を利用した組立へのアプローチとして現実的ではある．しかし，実際の組立システムは未完成である．現在，プランナと組立との最適な組合せは，SPAR (simulutaneous planner for assembly robots)[37)]が実現している．これは組み立てるべき部品の相対的位置に基づき，まず記号レベルで組立計画を立てた後，計画を実行に移すとき必要となる幾何学的制約を算出し，ハンドを用いて目標状態を達成する．SPARで用いられる作業指令と初期状態を図6に示す．

```
assembled (peg, block, [[7, 8]],
  tm ([[−1, 0, 0], [0, 0, 0], [0, 0, −1]
  [3, 0, 3.25]]), [−4, 0, 0])
```

これらに対して，図7(a)に示すテンプレートが与えられていて，記号レベルの前提条件が展開される．図7(a)のG2は与えられた初期状態で成立するが，G1のholding (Obj1, Grasp)がPickupにより展開され，図7(b)に示すような結果が得られる．図中の[PG1, …, PGn]はpegの把持できる状態のリストであり，BP5はblockの安定状態内で組み合わせるべき特徴面（今の場合二つの穴7, 8）が隠されない状態であり，これらの記述は対象ごとにあらかじめ与えられている．グリッパが開いていればこの状態で記号レベルでの問題解決は終了し，幾何レベルの制約充足に処理は移る．reachable (Grasp−1, Pos−2)はアームとpegとの位置関係がpegを把持できる状態にあるか，in-position-class (Pos−1, [BP5])はPos−1が組立可能な状態にあるかどうかを表す幾何学的制約である．図6の状態では前者は成立するが，後者は成立しないため（与えられたブロックの初期状態がBP5ではない），記号レベルに戻ってgoal_2を付加リストに持つ動作putdown (block, Pos_1)を探し出すことにより，blockをBP5の状態に置き得ることが発見される．このようにSPARはまず記号レベルで計画した後，幾何学的レベルで制約充足を試み，それが不能なときは記号レベルにバックトラックする．しかし，この例のように計画途中で制約充足不能なことが判明する場合もあるため，もう少し柔軟な制御構造を持たせたほうがよいと思われる．また，プランナが状態空間型なので複雑な組立計画時には問題が生じるだろう．この例などは「ペグをブロックに，両方の穴が一致するように挿入せよ」と指令すればよいと思われる[33)]．SPARは計画と動作を一体化した点で評価できるが，より現実的な組立問題への対処が今後の課題となろう．

以上のシステムのほかに，対象の把持法も自動算出し対象との衝突を回避する計画を立てるHandey[38)]や制約を用いて実世界の不確実さを減らすためのセンシングを計画するTwain[39)]などのシステムがある．これらのシステムは目標について推論するという点で本稿の目的から外れるため省略したが，不完全な世界の情報をより正確に獲得しつつ，計画し動作するプランナがこれから研究されねばならないことは明らかである．

7. むすび

プラニングの発展の経緯と問題点について述べた．人工知能の多くの問題が結局は探索問題に帰着されることを考えると，何らかの方法で探索空間を制限しない限り，良い計画を短時間で得ることは難しい．この意味では，さまざまな目標を達成する計画のライブラリを利用したプラニングが有望ではないかと考えられる．

一般にプランナは記号で表現された世界でプラニン

```
action-id:      ActionId,
action:         Assemble(Obj1, Obj2,[M1,M2],Transform, Mvector),
preconditions:
        operational:
                op(G1, ActionId, Holding(Obj1, Grasp))
                op(G2, ActionId, part_location(Obj2,Pos))
        geometric:
                geo(G3, ActionId,
                        member(Grasp, GraspList),
                        Holding(Obj1, Grasp))
                geo (G4, ActionId,
                        in_position_class(Pos,PositionList),
                        part_location(Obj2, Pos))
                geo (G5, ActionId,
                        mate_reachable(Grasp, Pos, Transform),
                        Holding(Obj1, Grasp),
                        part_location(Obj2, Pos))

add-list:
        Assembled(Obj1, Obj2, [M1, M2], Transform, Mvector)
        Gripper(open)
delete-list:
        Gripper(closed),
        Holding(Obj1, Grasp)
```

(a) 作業指令に対するテンプレート．オペレーショナルなレベルで目標が展開される

```
OPERATIONAL GOAL STACK
  op (goal_6, action_2, Gripper (open))
  op (goal_7, action_2, part_location (peg, Pos_2))      } Pickupによる展開
  op (goal_2, action_1, part-location (block, Pos_1))
GEOMETRIC GOAL STACK
  geo (goal_8, action_2,
      reachable (grasp_1, Pos_2),),                      } Pickupによる制約
      part_location (peg, Pos_2))
  geo (goal_3, action_1,
      member (grasp_1, [PG1, ···, PGn]),
      Holding (peg, grasp_1))
  geo (goal_4, action_1,
      in_position_class (Pos_1, [BP5]),
      part_location (block, Pos_1))                      } Assembleによる制約
  geo (goal_5, action_1,
      mate_reachable (grasp_1, Pos_1,
      tm ([[-1, 0, 0], [0, 1, 0], [0, 0, -1], [3, 0, 3.25]]),
      part_location (block, Pos_1),
      Holding (peg, grasp_1))
PLAN ACTIONS
  action (action_1, Assemble (
                          peg,
                          block,
                          [[7, 8], [7, 8]],
                          tm ([[-1, 0, 0], [0, 1, 0], [0, 0, -1], [3, 0, 3.25]]),
                          [-4, 0, 0]))
  action (action_2, Pickup (peg, grasp_1))
```

(b) オペレーショナルなレベルの目標が達成されると幾何的制約の充足が調べられる．goal_4 の制約不成立によりオペレーショナルな goal_2 が再計画される

図7 SPAR による組立作業

グを行う点で，人間が頭の中で行うプラニングと同じである．しかしその後が異なる．私たちは行動することによってより多くの異なった種類の情報を取捨選択する．人工知能のプランナは，抽象化された情報からの計画に終始し，抽象化以前の情報を無視することが多い．したがって計画が失敗したとき，あらかじめ準備された語彙を用いてその原因が記述できないなら，的確な再計画が行えなくなる．人間は抽象化された世界だけでなく，位置や力といった物理量が係わる世界についても推論できる．力までいかなくとも，もう少し具体化された情報をプランナが扱い得るようにすべきであろう．

解説者の独断と偏見により，機会主義的プラニングや時間推論は省略した．解説者自身は組立・分解問題を研究してはいるが，プラニングの専門家ではない．そのため，解説が不十分であったり，批評が的を得ていないかもしれないことを断っておきたい．

◇参考文献◇

(1) Newell, A. and Simon, H. A. : GPS : A Program That Simulates Human Thought, In *Computers and Thought,* eds. E. A. Feigenbaum and J. Feldman, McGraw-Hill, New York [GPS] (1963).

(2) McCarthy, J. and Hayes, P. J. : Some Philosophical Problems from the Standpoint of the Artificial Intelligence, *Machine Intell.,* Vol. 4, eds. B. Meltzer and D. Michie, Edinburgh University Press (1969).

(3) Gupta, A. : Explanation-Based Failure Recovery, *Proc. of 6th AAAI,* pp. 606-610 (1987).

(4) Minton, S. : Selectively Generalizing Plans for Problem-Solving, *Proc. of the 9th IJCAI,* pp. 596-599, Menlo Park, Calif. [Prodigy/ebl] (1985).

(5) 山田, 辻：完全因果性によるマクロオペレータの選択学習, 人工知能学会誌, Vol. 4, No. 3, pp. 321-329 (1989).

(6) Sussman, G. A. : A Computational Model of Skill Acquisition, *MIT AI Lab Memo,* AI-TR-297, AI Lab., Massachusetts Institute of Technology [HACKER] (1973).

(7) Waldinger, R. : *Achieving Several Goals Simultaneously,* SRI AI Center, Menlo Park, Calif. (1975).

(8) Sacedoti, E. D. : Non-Linear Nature of Plans, *Proc. of the 4th IJCAI* [NOAH] (1975).

(9) Sacedoti, E. D. : Planning in a hierarchy of Abstraction spaces, *3rd IJCAI* [ABSTRIPS] (1973).

(10) Tate, A. : Generating Project Networks, *Proc. of the 4th IJCAI,* pp. 215-218, Menlo Park, Calif. [NONLIN] (1977).

(11) Vere, S. : Planning, *Encyclopedia of Artif. Intell.* (Edited by Stauart C. Shapiro) pp. 748-758 (1987).

(12) Unruh, A. and Rosenbloom, P. : Abstraction in Problem Solving and learinig, *Proc. of the 11th IJCAI,* pp. 681-687 (1989).

(13) Stefik, M. J. : Planning and Meta-Planning, *Artif. Intell.,* Vol. 16, pp. 141-169 [MOLGEN] (1981).

(14) Stefik, M. J. : Planning With Constraints, *Artif. Intell.,* Vol. 16, pp. 111-140 [MOLGEN] (1981).

(15) Wilkins, D. E. : Representation in a Domain-Independent Planner, *Proc. of the 8th IJCAI,* pp. 733-740, Menlo Park, Calif. [SIPE] (1983).

(16) Vere, S. : Planning in Time : Windows and Durations for Activities and Goals, *IEEE Transaction on Pattern Analysis and Machine Intell.,* Vol. PAMI-5, No. 3, pp. 246-267 [DEVISER] (1983).

(17) Chanpman, D. : Nonlinear Planning : A Rigorous Reconstruction, *Proc. of 9th IJCAI,* pp. 1022-1024, Menlo Park, Calif. [TWEAK] (1985).

(18) Chanpman, D. : Planning for Conjective Goals, *Artif. Intell.,* Vol. 32, pp. 333-377 (1987).

(19) Irani, K. and Cheng, J. : Subgoal Ordering and Goal Augmentation for Heuristic Problem solving, *Proc. of 10th IJCAI,* pp. 1018-1024 (1987).

(20) Cheng, J. and Irani, K. : Ordering Problem Subgoals,. *Proc. of 11th IJCAI,* pp. 931-936 (1989).

(21) Kambhampai, S. and Hendler, J. : Control of Refitting during Plan Reuse, *Proc. of 11th IJCAI,* pp. 943-948 (1989).

(22) Altermann, R. : Adaptive Planning, *Cognitive Science,* Vol. 12 [PLEUS] (1988).

(23) Simmons, R. and Davis, R. : Generate, Test, and Debug : Combining Associational Rules and Casual Models, *Proc. of the 10th IJCAI,* pp. 1071-1078, Menlo Park, Calif. [G-T-D] (1987).

(24) Schoppers, M. : Universal Plans for Reactive Robots in Unpredictable Domains, *Proc. of the 10th IJCAI,* Menlo Park, Calif. (1987).

(25) Ginsberg, M. : Universal Planning : An (Almost) Universally Bad Idea, *AI Magazine,* Vol. 10, No. 4, pp. 40-44 (1989).

(26) McDermotto, D. V. : Planning and Acting, *Cognitive Science,* Vol. 2 [NASL] (1978).

(27) Georgeff, M. and Lansky, A. : Reactive Reasoning planning, *Proc. of the 6th AAAI,* Menlo Park, Calif. [PRS] (1987).

(28) 石田 亨：知識表現と動的世界—最近のプランニング研究から, 人工知能学会誌, Vol. 5, No. 2, pp. 146-153 (1990).

(29) 石田 亨：分散人工知能の技術と応用, 人工知能学会誌, Vol. 5, No. 4, pp. 441-448 (1990).

(30) Fahman, S. E. : A PLANNING System for robot Construction Tasks, *Artif. Intell.,* Vol. 5, pp. 1-49 (1974).

(31) 松原, 岡部, 井上：作業目標レベルのロボット言語の設計と試作, 日本ロボット学会誌, Vol. 3, pp. 48-55 (1985).

(32) 大嶽, 隅田, 水谷：組立知能ロボット ARI のプランニングシステム, 情報処理学会論文誌, Vol. 31, No. 3, pp. 491-499 (1990).

(33) 安部, 石川, 辻：組立説明文からの組立手順の生成, 人工知能学会誌, Vol. 3, No. 5, pp. 60-68 (1988).

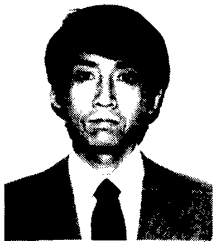
(34) Homen de Mello, L. S. and Sanderson, A. C. : and/or Graph Representation of Assembly Plans, *Proceduring of 5th AAAI,* pp. 780-785 (1986).

(35) Homen de Mello, L. S. and Sanderson, A. C. : Representation of Assembly sequence, *Proc. of 11th IJCAI* (1989).

(36) 山田, 安部, 辻：電機ドリル分解・組立コンサルタントシステム, 人工知能学会誌, Vol. 1, No. 1, pp. 116-123

(1986).
(37) Hutchinson, S. and KaK. A. : Spar : A Planner That Satisfies Operational and Geometric Goals in Uncertain Environments, *AI Magazine*, Vol. 11, No. 1, pp. 31-61 (1990).
(38) Lozano-Perez, T., Jones, J. L., Mazer, E., O'Donnell, P. A., Grimson, W. E. L. Tournassound, P. and Lanusse, A. : A Robot System that Recognizes Plan & Manipulators, *Proc. of IEEE Robotics & Automations* (1987).
(39) Lozano-Perez, T. and Brooks, R. A. : An Approach to Automatic Robot Programming, *AIM 842*, MIT (1985).
(40) 安部憲広：作業プランニング，日本ロボット学会誌, Vol. 5, No. 6, pp. 480-486 (1987).
(41) Chanpman, D. : Penguins Can Make Cake, In *AI Magazine*, Vol. 10, No. 4, pp. 5-50 (1989).
(42) Dean, T., Firby, J. and Miller, D. : Hierarchical Planning Involving Deadlines, Travel Time, and Resources, *Computational Intell.*, Vol. 3 [FORBIN] (1989).
(43) Doyle, J. : A Truth Maintenance System, *Artif. Intell.*, Vol. 12, pp. 231-272 (1979).
(44) Fikes, R. E. and Nilsson, N. J. : Strips : A New Approach to the Application of Theorem Proving to Problem Solving, *Artif. Intell.*, Vol. 2, pp. 189-208 (1971).
(45) Georgeff, M. : Communication and Interaction in Multi-Agent Planning Systems, *Proc. of 3rd AAAI*, Menlo Park, Calif. (1982).
(46) Greeen, C. C. : Theorem Proving by Resolution as a Basis for Question Answering, *Machine Intell.*, Vol. 4, eds. B. Meltzer and D. Michie, Edinburgh : Edinburgh University Press (1969).
(47) Hendler, J., Tate, A. and Drummond,M. : AI Planning : Systems and Techniques, *AI Magazine*, Vol. 11, No. 2, pp. 61-77 (1990).
(48) Miller, D., Firby, J. and Dean, T. : Deadlines, Travel Time, and Robot problem solving. *Proc. of the 9th IJCAI*, pp. 1052-1054 Menlo Park, Calif. [FORBIN] (1985).
(49) Rosenschein, S. J. : Plan Synthesis : A Logical Perspective, *Proc. of the 7th IJCAI*, Menlo Park, Calif. (1981).
(50) Rosenschein, S. J. : Synchronization of Multi-Agent Plans, *Proc. of the 2nd AAAI*, Menlo Park, Calif. (1980).
(51) Schank, R. C. and Abelson, R. P. : *Scripts, Plans Goals, and Understanding*, Hillsdale, N. J. : Lawrence Erlbaum (1977).
(52) Stallman, R. M. and Sussman, G. J. : Forward Reasoning and Dependency Directed Backtracking, *Artif. Intell.*, Vol. 9, pp. 135-196 (1977).
(53) Tate, A. : Interacting Goals and Their Use, *Proc. of the 4th IJCAI*, pp. 215-218, Menlo Park, Calif. [INTERPLAN] (1975).
(54) Wilensky, R. : Meta-Planning : Representing and Using Knowledge about Planning in Problem Solving and Natural Language Understanding, *Cognitive Science*, Vol. 5, pp. 197-233 (1981).

著 者 紹 介



安部 憲広（正会員）

1969年大阪大学基礎工学部電気工学科卒業．1974年同大学大学院博士課程修了．同年同学部制御工学科に勤務．現在，同大学産業科学研究所助教授．工学博士．人工知能，特に組立・分解システム，言語理解と画像理解の融合に関する研究を行っている．著者：「Prologプログラミング入門」（共立出版），「LISP」（培風館）など．情報処理学会，電子情報通信学会，AAAI各会員．