人工知能学会誌 Vol. 3 No. 2 (March 1988)〔別刷〕

論 文 Technical Paper

問題解決における戦略知識学習システム：PiL

―1次方程式・不等式でのケーススタディー

PiL : A System to Learn Strategy in Problem-Solving (A Case Study in an Equation & Inequality of the First Degree)

山田 誠二* 安部 憲広* 辻 三郎*
Seiji Yamada Norihiro Abe Saburo Tsuji

* 大阪大学基礎工学部制御工学科
Dept. of Control Engineering, Faculty of Engineering Science, Osaka University, Toyonaka 560, Japan.

1987年9月17日 受理

Keywords : learning, problem-solving, explanation-based generalization, absolute operator, macro operator.

Summary

PiL (Paradigm-based inference Learner) is a learning system that can acquire strategy knowledge of problem solving. PiL has basic operators (primitive transformation rules) a priori, but can not solve a problem with only this knowledge. Because the conditions of basic operators are so general, if all possible operators are applied to a problem state, the search space gets too large to reach the goal state. So a teacher gives PiL solving examples and PiL generalizes the examples and specializes the basic operators by adding the generalized solving examples to the conditions as conjunction. PiL generalizes examples with the condition-propagation based generalization method that is a kind of explanation-based generalization.

PiL generally consists of two modules, problem solving module, and knowledge maintenance module. The problem solving module solves the problem with production system,and details the given (or solved by itself) solving example. The knowledge maintenance module generalizes a solving example, extracts macro operators and absolute operators from generalized operator sequence, and arranges knowledge base. Further, PiL's knowledge base has hierarchical structure consisting of finish-condition rules, macro operators, absolute operators, heuristic operators and basic operators to control inference.

When a problem is given, first PiL tries to solve it by itself, and if it can't, PiL requires a teacher to give a solving example. Next, PiL translates the given solving example to sequence of PiL's operators, and generalizes it with condition-propagation based generalization method. The condition-propagation based generalization is done by regressing the conditions of operators applied in the given solving example through operator sequence. Further more, from generalized operator sequence, PiL is able to extract absolute operators and macro operators that are powerful knowledge in problem solving.

In this paper, We will report an application of PiL in an equation & inequality of the first degree.

1. はじめに

コンピュータに人間と同様の学習機能を持たせようとする試みは，長い間人工知能研究の本質的かつ困難な課題であり，過去において種々の研究が行われてきた．その中の1つに，例題からの概念学習を問題解決の分野で行うものがある[1]．これは，問題の状態を変化させる基本的な書換えルールを始めに与えておき，システム自身がルールの不適当な適用を避けるよ

うにその条件部を精練することで，問題解決の戦略知識を学習させようというものである．ルールの条件部の精練は，教師より与えられた（あるいはシステム自身が解決した）解法例を解析することにより行われる．ここでは書換えルールが適用されるべき問題状態が概念学習されることになる．

2．問題解決における戦略知識獲得としての学習

一般に問題解決システムを構築しようとするユーザが，容易にかつ明確にシステムに与えることのできる知識は，問題の状態を変化させる公式的かつ基本的な書換えルール（以後，基本オペレータ）である．しかし，この基本オペレータはその条件部が一般的過ぎる場合が多く，これをそのまま用いていたのでは前向き後向きに限らず，まったく無意味な展開が数多く生じ，探索空間が爆発的に拡大してしまい解にたどり着けない場合が多い．また，基本オペレータは非常に断片的な知識なので細かいステップごとに，次に適用すべきオペレータの探索を行わねばならず，無意識にマクロオペレータを使っている人間の問題解決と比べると非常に効率が悪い．したがって，従来の問題解決システムではパフォーマンスの効率化のため，ユーザが，条件部を精練したオペレータやマクロオペレータなどの戦略知識を事前に与えていたわけである．しかし，ユーザにとってこのような戦略知識を与え，かつ知識全体として整合性を保つことはかなり困難なことである．よって，ユーザにとって入力が容易な基本オペレータと，さらにもう1つユーザにとって入力が容易なものである解法例の2つを与えるだけで，そこからシステム自身が，問題解決の戦略知識を獲得することができればユーザの負担は非常に軽減されるはずである（Fig. 1）．

本論文では，以上のようなパラダイムの基に問題解決における戦略知識の学習を行うシステム PiL (Paradigm-based inference Learner) の基本的動作と1次方程式，不等式におけるケーススタディについて述べる[2][3]．なお，従来の学習システムにない PiL の特徴として，①説明に基づく一般化，②オペレータの重み付けによる絶対オペレータの抽出，③完全因果性によるマクロオペレータの抽出などがある．

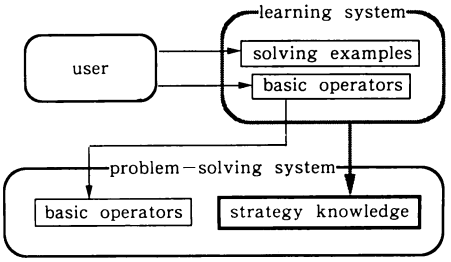


Fig. 1 Aquisition of strategy knowledge.

3．システム概要

他の問題解決の分野と同様，1次方程式・不等式においても適用可能な基本オペレータをすべて適用して前向き探索していたのでは，探索木が爆発的に展開されてしまい，解に到達することは不可能である[4]．また，この種の問題を STRIPS[5] 流に，後向きに解くことも不可能である．その典型的な例として，「等式の両辺から任意の整式を引く」という基本オペレータは，あらゆる等式に適用可能であり，かつその結果は無限個あるので，ここですでに探索木の爆発的展開が起こってしまう．このオペレータは，方程式を解くために不可欠な「移項」の際に必ず必要なため，その条件部を精練しなければならない．PiL で用いた1次方程式のための基本オペレータの一部を Fig. 2 に示す．

PiL は人間の教師に，問題解決のために十分な基本オペレータおよび模範的かつ誤りのない解法例を与えてもらい，その解法例を肯定例として一般化することにより，書換えルールの条件部を精練し戦略知識の学習を行う．なお，PiL は問題解決において基本的に縦型または横型探索を行い，競合解消はまったくやっておらず，オペレータの条件部の精練と後述する知識

```
rule010 : LS=RS → LS+A=RS+A, [], []
rule030 : LS=RS → LS/A=RS/A, [], [not_zero (A)]
rule210 : A*B+A*C → A* (B+C), [], []
rule220 : 0+A → A, [], []
rule230 : 0*A → 0, [], []
rule270 : R1+R2 → 0, [real_number (R1),real_number (R2),equal (R1,(−1) *R2)], []
rule280 : A/A → 1, [not_zero (A)], []
rule300 : R1+R2 → R3, [real_number (R1),real_number (R2)], [equal (R3,R1+R2)]
rule320 : R1/R2 → R3, [real_number (R1),real_number (R2),not_zero (R2)],
                      [equal (R3,R1/R2)]
rule340 : (A*B) /C → (A/C) *B, [], []
rule1000 : 1*AL=R → finish, [alphabet (AL),real_number (R)], []
```

Fig. 2 Basic operators.

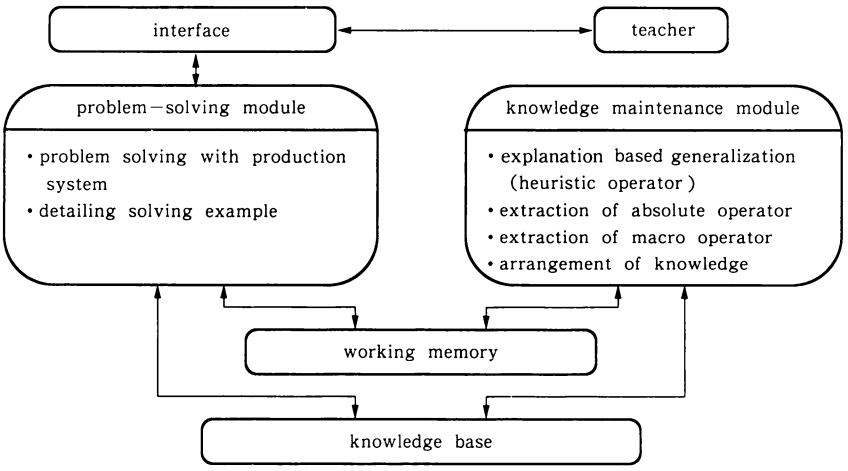


Fig. 3 The structure of PiL.

の階層構造で推論の制御を行っている．Fig. 3 に PiL の構成図を示す．

まず，教師（ユーザ）により問題が与えられ，最初は問題解決モジュールが独力でその問題を解こうとする．そして，一定の探索空間を調べ尽くしても解に至らないときは教師に解法例の入力を要請し，教師の助力により問題を解決する．こうして解法例が得られると知識管理モジュールがそれを一般化していき，その結果がヒューリスティックオペレータとして蓄えられる．さらに，得られたヒューリスティックオペレータに重み付けを行うことにより絶対オペレータを抽出し，次に完全因果性という基準を用いて特定のオペレータシーケンスをマクロオペレータとして取り出し，最後に冗長な知識を取り除いて知識の整理をするというサイクルを繰り返す．各々のモジュールの詳細を以降に述べる．

4．オペレータの記述

Fig. 2 での基本オペレータの例のようにオペレータはルール名，条件部，結論部，条件の拘束部，結論の拘束部から成り（Fig. 4），マクロオペレータを含む

```
rule−name :
    condition → action,
     [condition−restriction],
     [action−restriction]
```

Fig. 4 Discription of operator.

すべてのオペレータはこの形式で表現される．また，オペレータ中の大文字のアルファベットは変数を表す．条件部，結論部ではリスト構造に変換された数式の構造的なマッチングのみを行い，そのマッチングされた各要素についての拘束条件を拘束部が表す．拘束部の要素には，対象がある概念に属しているか否かを返す一項の概念述語（real_number, alphabet など）と，第 2（または 1）引数の数式を評価した結果を第 1（または 2）引数に単一化する二項の述語（equal）などがある．概念述語間には包含関係があるが，ここでは包含関係を考慮した簡潔な記述は避け，冗長ではあるがその述語の表す概念を包含している，より一般的な概念述語はすべて記述する（たとえば，minusp (X) とはせずに real_number (X) &minusp (X) と書く）．このとき，概念ハイアラーキを必要とするが，これはオペレータ個々についてのもので LEX [1][6][7] で用いられていたようなすべてのオペレータについて必要と思われる概念全体を統一的にまとめたものとは性質が異なる．

5．問題解決モジュール

問題解決モジュールでは，Prolog によって書かれたプロダクションシステムによる問題解決および解法例の詳細化が行われる．

5・1 問 題 解 決

各種オペレータで構成されている知識ベースは，Fig. 5 のように階層構造になっており，問題解決モジュールはこの階層を (1) から (4) の順序で適用可能なオペレータを探索していく．階層 (2) での探索のみ縦型で，他の各階層では横型に探索が行われ，現在の階層で適用されるオペレータがなくなったとき下位の階層に制御が移る．探索中に 1 つでもオペレータが適用されると，再び (1) に戻り最初から探索が行われる．

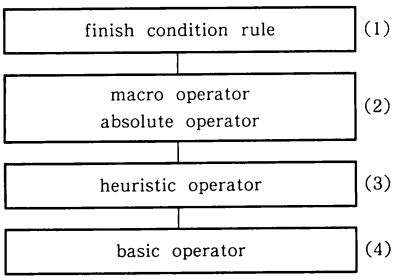


Fig. 5 Hierarchy of knowledge.

ここでは，競合解消はまったく行っていない．以下に各オペレータについて述べる．

(1) 終了条件ルール

終了条件ルール (Fig. 2 では rule 1000) は，教師により与えられるもので，このルールが適用されれば問題は解決されたことになる．このルールは解法例の一般化伝搬の根本になるもので，その条件部は必要十分に一般的に記述されなければならない．

(2) マクロオペレータ

強い因果関係のある特定のオペレータのシーケンスを1つにまとめたオペレータ．複数個のオペレータの適用を迅速に行う．

(3) 絶対オペレータ

適用可能なとき優先的に必ず適用される単独のオペレータ．

(4) ヒューリスティックオペレータ

一般化されたオペレータのうち，絶対オペレータ以外のもの．

(5) 基本オペレータ

最初に与えられる基本的な書換えルール．PiL では問題解決のために十分かつ完全な基本オペレータが事前に与えられているものとする．

5・2 解法例の入力およびその詳細化

教師が自然に与える解法例の行間は，1つのオペレータで埋めることができない場合が普通である．これは，教師が無意識のうちにオペレータの省略を行っているためである．システムは解法例を自分の持っているオペレータのシーケンスで表す必要があるので，教師により省略された部分を補わなければならない．

問題解決モジュールは，解法例の次の行をサブゴールとして5・1で述べたようにオペレータを適用していき，省略された書換えルールを補う．このとき，同じステップ数で，結果も同じ展開は，すべて一般化伝搬の候補として保持しておく．もし一定の探索空間を

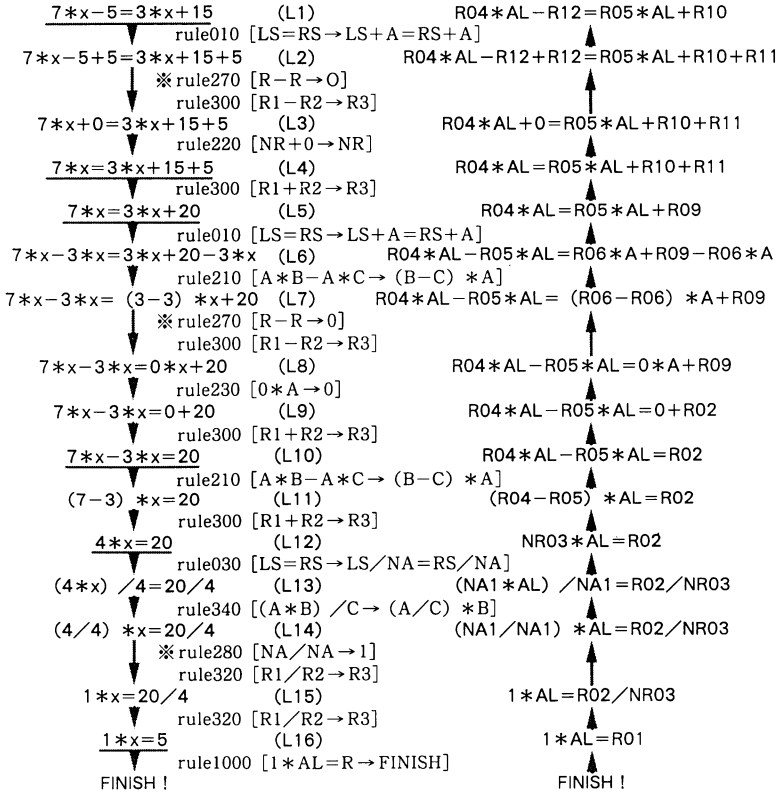


Fig. 6 Solving example.

Fig. 7 Propagation of generalization.

調べ尽くしてもサブゴールに到達できないときは，今のサブゴールよりもさらに現在の問題状態に近い解法例を教師に要求し，それを新しいサブゴールとして探索を繰り返す．Fig. 6 の下線部が教師により自然に入力された解法例，そして，Fig. 6 全体が省略されたオペレータを補い詳細化したものである．なお，PiL では教師は理想的であるとし，誤った解法例の教示は行われないものとする．

このようにして得られたオペレータのシーケンスの各行は，解法の過程で使用されたオペレータの条件部の肯定例であると考えられる．しかし，これらは肯定例の個々のインスタンスであり，これらをこのまま持っていたのでは次にまったく同じインスタンスがきたときにしか使えない．よって，これらの肯定例を一般化する必要がある．

6. 知識の管理

知識管理モジュールでは，①解法例の一般化，②ルールの重み付け，③マクロオペレータの抽出，④知識の整理などが行われる．

6・1 説明に基づく解法例の一般化

各ルールの肯定例を得るために，詳細化された解法例の各行を一般化していく．PiL が行う一般化は，この分野で従来よく行われてきた帰納的一般化[1][8]ではなく，展開された各ルールの条件部の適用範囲が最後の終了条件ルールから解法例を後向きに伝搬していくことにより，各行の一般化が行われるという，近年注目されている「説明に基づく一般化」(Explanation-Based Generalization)[9]の手法である．よって，帰納的一般化に必要な全体的かつ統一的な概念ハイアラーキを必要とせず，無根拠な一般化を避けることができる．たとえば，整式に同じ数字が含まれていたとき，従来の一般化ではこれを根拠もなく同一の変数に一般化するが，説明に基づく一般化では，後々同一であることを必要とする条件部を持つオペレータが適用されているものだけを同一の変数として一般化する．さらに，1つの解法例から一般化が可能であり非常に効率がよいという利点がある．

説明に基づく一般化を具体的な例で説明する．Fig. 7 に Fig. 6 に示した解法例を一般化していく過程を示す．図中で，Rn は任意の実数（n の等しいものは同一の実数），A, B, C は任意の整式，NA, NB は 0 以外の任意の整式，LS, RS は左辺・右辺，NR は 0 以外の任意の実数，AL は任意のアルファベット（変数）を表す．一般化の伝搬は終了条件から始まる．まず，終了条件 rule 1000 の条件部の左辺は 1 と任意のアルファベットの積，右辺は任意の実数でよいので，Fig. 6 での L 16 の左辺の“x”は AL に，右辺の“5”は任意の実数 R 01 に一般化され，Fig. 7 の L 16 のようになる．次に rule 320 を逆にたどることにより，L 15 の右辺の“20”と“4”は R 02, NR 03 に一般化される．さらに，L 14 では rule 280 により左辺の“4”と“4”が等しい任意の 0 でない整式 NA 1 に一般化され，L 14 は rule 340 により L 13 のようになる．そして，次は rule 030 により伝搬が行われるわけであるが，rule 030 は結論部で 0 でない同一の整式（NA）で両辺を割っているので，これにより L 13 の両辺の分母である NA 1 と NR 03 が単一化され，その結果 L 12 の左辺の AL の係数が NR 03 になっている．あとは同様に L1 まで一般化が行われていく．ここで重要なのは，十分な一般化を行うために，一般化の伝搬は完全に後向きで行われることである．このことにより一般化は以前に適用されたオペレータの拘束条件の影響をまったく受けない．例として，L6 を見てみる．L6 の上の rule 010 の結論部で，両辺に同じ整式 A を足しているので，L6 で rule 010 の整式とマッチする R 05＊AL と R 06＊A が単一化されそうであるが，rule 010 は L6 よりも先に適用されているので，それらは単一化されない．そして，L5 において R 06＊A は R 05＊AL に単一化されている．

なお，この一般化の伝搬は具体的には，演算子以外の各要素（変数，定数）を変数に変換したワーキングメモリの各レベルを単一化していくことにより実現される．ただし，ここでは rule 300 のような実数どうしの演算を行うルールでの一般化伝搬のために，「実数は加減乗除について閉じている」という知識を付加的に与えている．こうして一般化された解法例は，肯定例と考えられるので，これらの肯定例を各オペレータの元の条件部に連言として付加することにより条件部の精練が行われる．これがヒューリスティックオペレータである．

また，PiL の「説明に基づく一般化」は，基本的には LEX 2[7]における「もの」と同様の手法であるが，LEX 2 はあくまでも全体的な概念ハイアラーキにおけるバージョンスペース理論に基づいているのに対し，PiL は肯定例の一般化だけを行い，さらにオペレータの階層化により一般化された肯定例（ヒューリスティックオペレータ）を最も一般的なオペレータ（基本オペレータ）よりも優先させているところが異なる．また，システム全体でみると LEX 2 はマクロオペレータをまったく考慮しておらず，後述するようにマ

クロオペレータを効率よく抽出し，それを問題解決に用いるPiLのほうがパフォーマンスにおいて優れていると考えられる．

6・2　一般化伝搬経路の選択

オペレータの適用過程において条件部および結論部の重複により，複数のオペレータから同一の結果が得られることがある（Fig. 6 の L2, L7, L 14）．PiL は一般化伝搬の際にそれらの候補から，伝搬する適用範囲が最も一致しているものを最適な経路として選択する．たとえば，Fig. 6 のL 14 では rule 280 と 320 が競合しているが，L 15 はルールの結論部に“1”を求めているので，rule 280 のほうが選択される．他の場合においても同様に，※印のルールが選択される．この選択は具体的には単一化前後での変数の個数の変化や概念述語を調べることにより行われる．

6・3　オペレータの重み付け（絶対オペレータの抽出）

次にヒューリスティックオペレータに重み付けを行う．ここで問題となるのは，どのような基準で重み付けを行うかであるが，PiL では，次のような基準で行う．ある基本オペレータの適用範囲（構造パターンおよび拘束条件）をA，その基本オペレータを一般化して得られたヒューリスティックオペレータの適用範囲をBとすると，BとAの比の値［B/A］が大きいヒューリスティックオペレータほど，重い重みを付ける．しかし，この適用範囲間の演算であるB/A の値を求めることは難しく，ここでは特殊な場合のみを扱う．それはA/B=1となるとき，つまりA=Bとなるときで，このときに限りそのオペレータは「絶対オペレータ」としてヒューリスティックオペレータよりも高い階層に移される．その結果，絶対オペレータは単独のオペレータではあるが，ヒューリスティックオペレータよりも優先的に適用されることになる．直観的には，絶対オペレータとは，適用可能なときに必ず適用される単独のオペレータ（たとえば，実数どうしの加減乗除）であると言える．

6・2で述べたような競合が起こっているところで絶対オペレータが抽出された場合，当然それ以後は競合した他のオペレータ（OP 1）よりも絶対オペレータとして抽出されたオペレータ（OP 2）のほうが優先的に適用されることになる．しかし，将来場合によってはOP 2 よりもOP 1 のほうが一般化の伝搬の経路として適当である場合も出てくると考えられる．よって，そのような場合に対処するために競合したオペレータの1つが絶対オペレータになったときは，その他の競合したオペレータもすべて絶対オペレータとして抽出することにする．

6・4　マクロオペレータの抽出

2節でも述べたように，基本オペレータは断片的な知識の集合であるから，それを一般化したヒューリスティックオペレータや絶対オペレータもまた断片的な知識に過ぎない．よって，これらだけでは問題解決においてオペレータを展開する際に細かいステップごとに次に展開すべきオペレータを探さねばならず非常に効率が悪い．実際，5・2でも触れているように，人間はある特定のオペレータのシーケンスをマクロオペレータとして持っており，それを用いて問題解決を行っている．そこでPiL も解法例から特定のシーケンスをマクロオペレータとして抽出することにより，オペレータ展開をより迅速に行い，効率を上げる必要がある．しかし，ここで問題となるのはどのような基準でマクロオペレータを選別して抽出するかである．なぜなら，解法例におけるオペレータシーケンスの任意の部分的なシーケンスが，マクロオペレータの候補として考えられ，これらをすべてマクロオペレータとして持っていたのでは解法例が与えられるごとにマクロオペレータが著しく増加してしまい，遂にはマクロオペレータの探索空間がヒューリスティックオペレータの探索空間より大きくなってしまう危険性があるからである[10]．これは，マクロオペレータの候補の中にはまったく無意味なものが多く含まれているためで，何らかの基準に基づいて候補の中から必要と思われるものだけを抽出する必要がある．そこでPiL では，「人間は適用されたオペレータ間に閉じた強い因果関係があるときに，それらのオペレータのシーケンスをマクロオペレータとして取り込む」という仮定に基づいた**完全因果性**という基準でマクロオペレータの抽出を行う．

- **完全因果性**

以下に完全因果性の定義を示す．

いま，OPi … OPj ($1 \leq i < j \leq n$: OPn は終了条件ルール）が解法例におけるオペレータのシーケンスとする．そして，OPj のそれぞれの定数化された条件部（拘束部も含む），結論部を PCj, Aj, また OPj 以前に OPi と完全因果性のあるオペレータの結論部を Ap … Aq ($i<l \leq m<j$) とすると，

$$PCj \subseteq Ai \cup Ap \cup \cdots \cup Aq$$

が成り立つなら，OPi と OPj には完全因果性がある（OPi $ OPj）という．具体例を Fig. 8 に示す．この

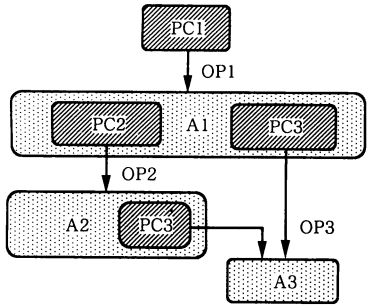


Fig. 8 Perfect causality.

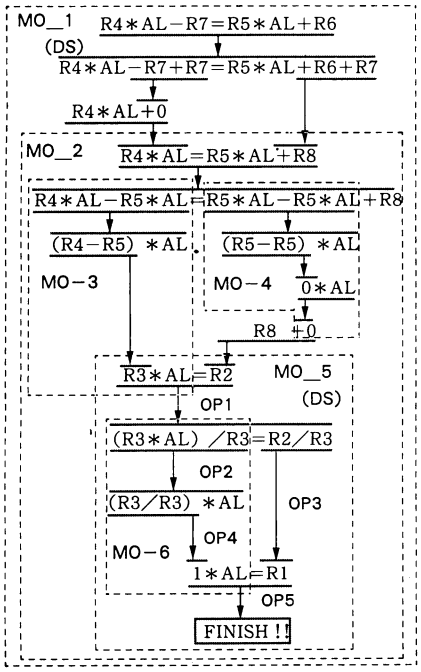


Fig. 9 Extraction of macro operator.

図では，PC2⊆A1 から OP1 $ OP2 となり，PC3⊆A1 ∪A2 から OP1 $ OP3 となる．

OPi, OPj 間に完全因果性があるということは，直感的には OPi が適用された後には必ず OPj が適用可能であるということを意味し，閉じた強い因果関係があることを表している．PiL は，解法例におけるオペレータのシーケンス OP1 … OPn の各行 OPi (1≦i< n) から終了状態に向かって (Fig. 9 では下向きに) 任意の 2 つのオペレータ間の完全因果性を調べていく．そして，完全因果性が連続する場合は，その最大のシーケンスを 1 つのマクロオペレータとして抽出する．

以上の処理により Fig. 7 の解法例から求められたマクロオペレータの例が，Fig. 9 に示された MO_1〜6 である．Fig. 9 では Fig. 7 のオペレータ間の因果関係がわかりやすいネットワーク表現がしてある．例として，MO_5, MO_6 の抽出過程について説明する．まず，OP1 から下に向かって順に完全因果性のあるオペレータを前述の定義に基づいて調べていく．すると，OP1 は残りの OP2〜5 のすべてと完全因果性をもっている．つまり，これらは完全因果性が連続しているので，OP1〜5 を 1 つのマクロオペレータ (MO_5) として抽出する．次に，OP2 から下に完全因果性を調べると，OP4 だけが検出され，OP2, OP4 をマクロオペレータ (MO_6) として抽出する．残りの OP3, OP4, OP5 から下向きに調べても，完全因果性をもつオペレータはないので，マクロオペレータは抽出されない．

さらに，終了条件まで含んでいるマクロオペレータ，つまりそのマクロオペレータが適用されれば直接的に問題が解決されてしまうものには，直接解決可能 (DS: Directly Solvable) というラベルを付けておく．この直接解決可能なマクロオペレータの条件部の論理和は，Mitchell [6][7] の解決可能 (solvable) 概念の操作可能化 (operationalization) [9] につながると考えられるが，詳細な議論は次の機会に譲る [10]．

なお，マクロオペレータの抽出に関する研究は，STRIPS [5] 以降 Minton [11]，Iba [12] などにより行われているが，それらはいずれも領域に依存した評価関数やオペレータシーケンスの使用頻度に基づいてマクロオペレータの抽出を行っており，領域独立であるオペレータの因果関係により，単独の解法シーケンスからでも抽出が可能である完全因果性のほうが汎用性があると思われる．

6・5 知識の整理

各オペレータの抽出が行われると，絶対オペレータとヒューリスティックオペレータにおける冗長な知識を削除する．冗長な知識には，①まったく同一の知識，②他のより一般的な知識に包含される知識などがある．これらは構造パターン間および拘束条件間の包含関係を調べることにより検出される．拘束条件の述語は，4 節で述べたように冗長に書かれているので，概念述語のリストの包含関係を調べることにより，拘束条件の包含関係が容易にわかる．

7. 学習結果

まず，1 次方程式の学習結果について述べる．Fig. 2 で示した例を含む 51 個の基本オペレータ (1 次方程式，不等式両方のもの) を与え，次に，与える順序やニアミスなどの教示戦略をまったく考慮せずに，中学生用の問題集を参考にして 100 問以上の問題を PiL に与えた．そして，PiL が解法例を要求してきたとき

$10x+5=6x+9$	$6\ (3x-5)\ =24$
$3x+10=25$	$7x-8=3x$
$15=5x-5$	$(9x-12)/3+2x=9$

Fig. 10 Typical training examples.

$$\frac{4x+10}{2}-\frac{3\ (x+3)}{5}=3x-\frac{5\ (2-x)}{4}$$

$$2y+\frac{y-1}{3}=9$$

$$(3/5)\ (4x-7)\ =5\ ((7/5)\ x-0.6)$$

Fig. 11 Solvable problems.

にだけ，教示を行った．その結果，Fig. 10 に示す典型的な 6 つのパターンの方程式の解法例を PiL に教示すれば，後は Fig. 11 のような高校入試問題を含む 100 問以上の 1 次方程式を教師の助力なしに解答可能であることがわかった．また，抽出される各オペレータは，Fig. 10 の 6 パターンの解法例で定常状態に達し，それ以上はいくら方程式を解かせようとも抽出される各オペレータの個数は増加しなかった．この定常状態におけるマクロオペレータ (13 個)，絶対オペレータ (6 個) およびヒューリスティックオペレータ (12 個) を Fig. 7 と同様の表記法を用いてわかりやすく記述したものを Fig. 12 に示す．図から明らかなように，マクロ，絶対オペレータともに人間が方程式を解くときのオペレータの適用戦略 (たとえば，変数項をまとめる，実数どうしの計算を先にやる，カッコを外すなど) と非常に類似しており，各オペレータの抽出の基準が妥当であることを裏付けている．また，Fig. 12 のマクロオペレータのルール名の次の [ ] の中の数字は，そのマクロがいくつの基本オペレータで構成されているかを示しており，多いものでは 15 個の基本オペレータから構成されているのがわかる．

同様に，1 次不等式については，12 個のパターンの例題で定常状態に達した．このとき新たに得られたマクロオペレータ，ヒューリスティックオペレータの個数はそれぞれ 37 個，31 個で，新しい絶対オペレータは得られなかった．また，実験してはいないが基本オペレータを追加することにより PiL は 1 次方程式・不等式以外に 1 次式に帰着できる 2 次方程式および指数，対数，分数方程式も学習可能であると考えられる．

最後に，Fig. 13 に定常状態に達した PiL が 1 次方程式を解く過程を示す．下線部は書き換えられる領域

[macro operator]

MO_1 [5]: R1=R2*AL → 1*AL=R3 (DS)
MO_2 [9]: R1=R2*AL+R3 → 1*AL=R4 (DS)
MO_3 [12]: R1*AL+R2=R3*AL → 1*AL=R4 (DS)
MO_4 [8]: R1*AL+R2=R3 → 1*AL=R4 (DS)
MO_5 [15]: R1*AL+R2=R3*AL+R4 → 1*AL=R5 (DS)
MO_6 [11]: R1*AL=R2*AL+R3 → 1*AL=R4 (DS)
MO_7 [4]: R1*AL=R2 → 1*AL=R3 (DS)
MO_8 [4]: (R1*AL+R2) /NR3 → R4*AL+R5
MO_9 [2]: (R1*AL) /NR2 → R3*AL
MO_10 [3]: (R1*AL+R9) *R3 → R4*AL+R5
MO_11 [2]: R1*AL+R2*AL → R3*AL
MO_12 [2]: R1*AL−R1*AL → 0
MO_13 [2]: (NR1*AL) /NR1 → 1*AL

[absolute operator]

AO_1 : A*0 → 0
AO_2 : R1−R1 → 0
AO_3 : R1/NR2 → R3
AO_4 : R1*R2 → R3
AO_5 : R1+R2 → R3
AO_6 : NA/NA → 1

[heuristic operator]

HO_1 : (R1*AL+R2) /NR3 → (R1*AL) /NR3+R2/NR3
HO_2 : R1=NR2*AL → NR2*AL=R1
HO_3 : R1=NR2*AL+R3 → NR2*AL+R3=R1
HO_4 : NR1*AL+R2=R3*AL → NR1*AL+R2−R2=R3*AL−R2
HO_5 : R1* (R2*AL+R3) → R1*R2*AL+R1*R3
HO_6 : NR1*AL+R2=R3 → NR1*AL+R2−R2=R3=R2
HO_7 : (A*AL) /A → (A/A) *AL
HO_8 : NR1*AL=R2 → (NR1*AL) /NR1=R2/NR1
HO_9 : B*A+R1*A → (B+R1) *A
HO_10 : R1*AL=R2*AL+R3 → R1*A−R2*AL=R2*AL+R3−R2*AL
HO_11 : R1*AL+R2=R3*AL+R4 → R1*AL+R2−R2=R3*AL+R4−R2
HO_12 : 0+R1*AL → R1*AL

Fig. 12 Obtained operators.

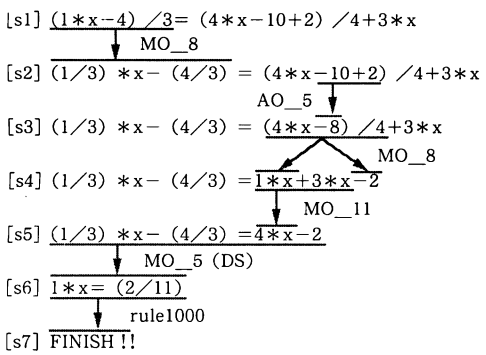


Fig. 13 Problem solving process of PiL.

を示している．これを見ればマクロオペレータ，絶対オペレータにより問題状態が徐々に直接解決可能な状態へ変化していくのがよくわかると思う．また，この問題をマクロオペレータを使わずに解くと 26 ステップかかるところが，マクロにより5ステップに軽減されており，マクロオペレータの効果を顕著に示している．

8．まとめと今後の課題

少数の解法例を与えられ，それを解析することにより問題解決の戦略知識を学習するシステム PiL について報告した．なお，PiL の本報告のバージョンは SUN3 上で K-PROLOG を用いてのインプリメントが完了している．

以下，現状での PiL の問題点と今後の課題について述べる．まず，本文中でも述べたように PiL が十分な能力を発揮するには次のような前提が必要である．

(1) 適用範囲が必要十分に一般的で，かつ問題解決のために十分な基本オペレータが与えられている．

(2) 教師により常に正しい解法例が与えられる．

これらの前提が成り立たないとき，不十分な一般化や誤った解法例の学習を行ってしまい，これが現状での PiL の限界であり，問題点である．次に，これらの前提が成立しない場合への対処についてであるが，まず，(1)については基本オペレータが十分に与えられていない場合は，教師が解法過程をモニタすることにより比較的容易に検出・修正できると思われるが，適用範囲が必要十分に一般的でない基本オペレータの検出・修正は困難である．対策としては，最初に与えられた概念述語を一般化処理の過程で修正していく（半）自動化されたデバッガ[13]の構築が考えられるが，具体的なところは今後の課題である．(2)についても，PiL 自身が解法例の誤りを認識するのは困難であり，「説明に基づく否定例からの学習」[14]として今後の大きな課題である．

また，PiL の発展性としては各種方程式への応用，さらに積木の世界でのロボットハンドの行動計画，導出法における戦略知識の学習などにも応用可能ではないかと考えている．これらの他分野への応用により，1次方程式で有効であった絶対オペレータおよびマクロオペレータの抽出基準の汎用性が検証されるであろう．また，抽出される各オペレータは，問題解決の理想モデルとも考えられるので ICAI への応用も考えられる．

◇ 参 考 文 献 ◇

(1) Mitchell, T. M., Utgoff, P. E., Nudel, B. and Banerji, R. : Learning problem-solving heuristics through practice, Proc. of IJCAI-81, pp. 127-134 (1981).

(2) 山田，安部，辻：条件連鎖に基づく一般化による学習，第1回人工知能学会大会資料，pp. 81-84 (1987).

(3) 山田，安部，辻：解法例からの問題解決知識の獲得—1次方程式・不等式の場合—，情報処理学会「人工知能と知識工学研究会資料」，87-AI-54 (1987).

(4) バンディ，A.：メタレベル推論と意識，「AI と哲学」，pp. 187-203，産業図書 (1985).

(5) Fikes,R. E.,Hart, P. E. and Nilsson, N. J. : Learning and Executing Generalized Robot Plans, Artif. Intell., Vol. 3, pp. 251-288 (1972).

(6) Mitchell, T. M., Utgoff, P. E. and Banerji, R. : Acquiring and Refining Problem—Solving Heuristic, Machine Learning—An Artificial Intelligence Approach, Vol. 1, pp. 163-190, Tiago Pub. Co., Palo Alto (1983).

(7) Mitchell, T. M. : Learning and Problem Solving, Proc. of IJCAI-83, pp. 1139-1151 (1983).

(8) Kibler, D. and Porter, B. : Perturbation : A means for guiding generalization, Proc. of IJCAI-83, pp. 415-418 (1983).

(9) Mitchell, T. M. and Keller & Kadar-Cabelli : Explanation-Based Generalization : A Unifying View, Machine Learning, Vol. 1, No. 1, pp. 47-80 (1986).

(10) 山田，安部，辻：問題解決における学習システム：PiL「人工知能システムの枠組み」シンポジウム (1987).

(11) Minton, S. : Selectively Generalizing Plans for Problem-Solving, Proc. of IJCAI-85, pp. 596-599 (1985).

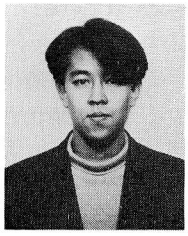
(12) Iba,G.A. : Learning by discovering macros in puzzle-solving, Proc. of IJCAI-85, pp. 640-642 (1985).

(13) Utgoff, P. E. : Sift of Bias for Inductive Concept Learning, in Machine Learning II, pp. 107-147, Morgan Kaufmann (1986).

(14) Mostow,J., Bhatnagar,N. : Failsafe-A Floor Planner that Uses EBG to learn from its Failures, Proc. of IJCAI-87, pp. 249-255 (1987).

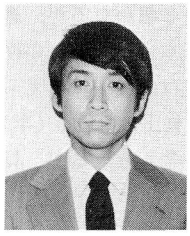
〔担当編集委員：諏訪　基，査読者：仁木　和久〕

著者紹介



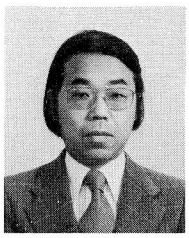
山田　誠二（正会員）

昭和59年大阪大学基礎工学部卒業．昭和61年同大学大学院修士課程修了．現在，同博士課程在学中．人工知能，特に学習，説明に基づく一般化に関する研究に従事．IEEE会員．



安部　憲広（正会員）

昭和44年大阪大学基礎工学部電気工学科卒業．昭和49年同大学大学院博士課程修了．同年同大学基礎工学部制御工学科勤務．現在，同学部助教授．工学博士．人工知能，ロボティックスなどの研究に従事．情報処理学会，電子情報通信学会，計測自動制御学会各会員．



辻　　三郎（正会員）

昭和28年大阪大学工学部卒業．昭和30年同大学大学院修士課程修了．同年，電子技術総合研究所入所．昭和45年より大阪大学基礎工学部制御工学科教授．工学博士．現在，人工知能，ロボティックス，コンピュータビジョンなどの研究に従事．情報処理学会，電子情報通信学会，計測自動制御学会，IEEE各会員．