知識工学と人工知能 46－6
（1986. 5. 9）

学習機能をもつ算術問題回答システム

桜井 成一朗　　志村 正道
東京工業大学 工学部

自然言語で与えられた算数の文章題を解くシステムにおける、学習機能について述べる。本システムは問題を解くための知識をルールの型式でもち、知識が不足しているために問題が解けない場合には、ユーザに質問を行って不足した知識を補う。獲得した知識は一般化したルールの形式で記憶され、以後の問題解決で利用される。また、本システムは問題解決で利用したルールを合成し、新しいルールを生成することにより問題解決能力が改善される。しかしながら、ルールの一般化を行なうために、必ずしも正しいルールが生成されるとは限らないが、誤ったルールが生成された場合には、ユーザと対話しながらルールの修正を行うことができる。

Learning Arithmetic Problem Solver

Seiichiro SAKURAI and Masamichi SHIMURA
Faculty of Engineering, Tokyo Institute of Technology

The present paper describes a problem solving system with a learning mechanism (Learning Arithmetic Problem Solver, LAPS), which can solve arithmetic problems given in natural language. Since LAPS has knowledge about arithemtic problems in the form of rules, it can solve many different problems without alteration of the program. When LAPS cannot solve a given problem because of a shortage of knowledge, it asks the user how to solve the problem. According to the user's advices LAPS acquires knowledge and rules to solve it. Furthermore, LAPS can improve its performance at problem solving by synthesizing rules that are applied.

1　はじめに

近年、人工知能や知識工学が活発に研究され、人間の専門家と同程度の能力をもつような実用をめざしたシステムがいくつか構築されてきた。しかしながら、現在実用化されているシステムは適用分野がごく狭い範囲に制限されたものであり、人間の知的活動を代行するようなシステムに知識をどのように与えるかという大きな問題は依然として残されたままである。小規模な実験的システムであれば、必要とされるすべての知識を予測して十分な知識を与えることができるが、汎用的なシステムでは知識が膨大な量になり、必要な知識を予測することが困難になる。それゆえ、知識を系統的に獲得し、獲得した知識を容易に利用できる機能をもった知的システムを構成することが望まれる。

本論文では、学習機能をもつ算数の問題解決システムを構築するために、必要とされるデータ構造や学習機構について考察し、実際に作成したシステムＬＡＰＳ(Learning Arithmetic Problem Solver)について述べる。一般に問題解決を行なうための知識は、さまざまなレベルの知識に分類することができる。問題領域の対象に関する知識、問題解決で利用される基本的な知識、それらの知識の利用に関する制御知識などがあるが、ここでは特に問題解決で利用される基本的な知識についてその知識獲得の方法と知識ベースの保守の方法について述べる。

ＬＡＰＳは自然言語で与えられた算数の文章題を解く問題解決システムであり、文章題を解くための基本的な公式をルールの形式でもち、ルールを組み合わせて問題を解いていく。問題解決のための公式があらかじめ与えられていない時には、ＬＡＰＳはユーザと対話してその不足した知識を補いながら問題を解く能力をもち、さらに問題解決過程からルールを合成することによって問題解決能力を改善することができる。

ユーザと対話しながら知識を獲得するシステムとしては、TEIRESIAS[2]がよく知られている。このシステムは学習システムというよりは、むしろエキスパート・システム構築のための知識獲得支援システムということができる。その他に例題から学習を行なうシステムがいくつか実現され、それらの比較が[5][6]で行なわれている。ＬＡＰＳは問題解決を行ないながら、ユーザと自然言語による対話を通して学習を行なうところに特色がある。

2　ＬＡＰＳの構成

ＬＡＰＳプログラムは、図１に示すような自然言語処理部、問題解決部、知識獲得部、知識修正部から構成され、知識ベースを備えている。

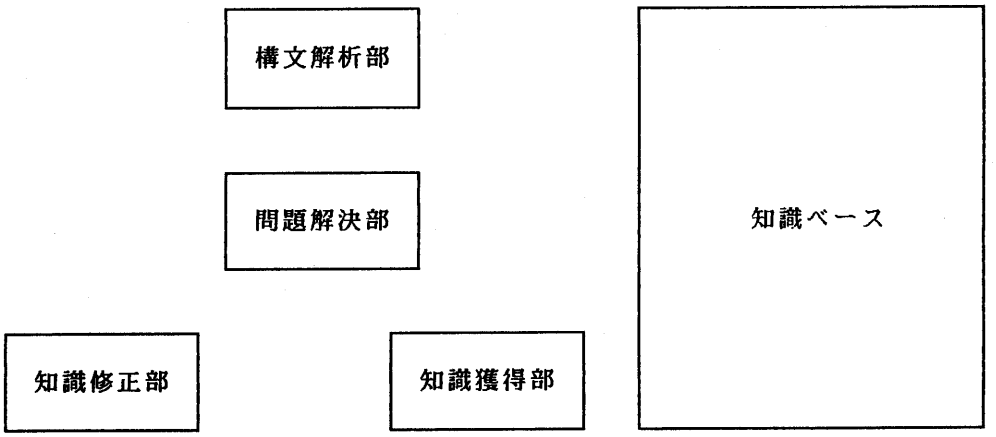

図１　システムの構成

自然言語処理部はユーザの入力文をシステムの内部表現へと変換する。ＬＡＰＳでは内部表現として意味ネットワークの一変形である事実グラフと呼ばれる有効グラフが用いられている。事実グラフでは問題文中に現れた対象は節点で表現され、対象間の関係は節点間をリンクで結ぶことにより表現される。知識ベース中には概念間の階層関係などの常識知識が格納されている。自然言語処理部により事実グラフが構成されると、ＬＡＰＳは問題解決部を呼び出し、問題解決を行なう。問題を解くための知識が不足している場合には、知識獲得部を呼び出し、ユーザと対話を行ない不足した知識を補う。また、問題解決に成功した場合には、知識獲得部を呼び出して、新しいルールを合成する。問題解決の際に知識の修正の必要なことがわかると、知識修正部を呼び出して知識を修正する。

計算機に算数の問題を解かせるためには、基本的な知識として算数の公式をあらかじめ与えておく必要があり、ＬＡＰＳでは図２に示すような条件(condition)部と実行(action)部をもつルールの形式で公式が表現される。図２は*yと*zを合わせて*xになるときには、*x=*y+*zを抽出するというルールを示している。なお、"condition-add"、"action-equation"、"action-term"、"action-expression"はユーザが定義した述語であり、ルールの各部分には、任意のlisp関数や、ユーザが任意に定義した述語を記述することができる。

```
(rule-1 eq-rule
   (condition = (condition-add *x *y *z))
   (action =
       (action-equation
            (action-term *x *property)
            =
            (action-expression
              (action-term *y *property) + (action-term *z *property)))))
```

図２　方程式抽出用ルールの例

方程式を抽出するためにＬＡＰＳは条件部と事実グラフとの照合を行ない、条件部の変数と事実グラフ中の節点やリンクとの対応を取り、その対応により実行部中の変数を置き換えて実行部を評価し、方程式を抽出する。ＬＡＰＳは問われている対象の値を目標変数に設定して、方程式抽出の作業と方程式を解く作業を組み合わせて問題を解いていく。この作業に応じてＡＮＤ木が生成され、その各節点には現在の目標変数や得られた方程式などの情報が格納される。

３　知識獲得

獲得した知識を多くの問題に利用できるように知識の一般化を行なうことが重要である。ＬＡＰＳは知識の一般化を行なう機能を有しており、選択的一般化規則として、条件を落とす規則、概念の抽象化を行なう規則、定数を変数に置換する規則を用いている。また、選言による一般化のために、条件を付加する規則、連言を選言に変換する規則を用いている。また、これらの一般化規則は、逆向きに用いることにより特殊化規則として用いることができる。

ＬＡＰＳでは、Mitchellの候補消去アルゴリズム[3][4]と呼ばれる学習アルゴリズムを用いて、学習を進めていく。この候補消去アルゴリズムは規則空間を幅優先探索するものであり、大きな規則空間に対しては、ヒューリスティックスを用いて探索の枝刈りを行ない学習を効率的に行なう必要がある。学習の効率を高めるために、ＬＡＰＳでは初期の規則空間を制限し、以後候補消去アルゴリズムによって獲得した知識の修正を行

なっていく。

３．１　与えられていないルールの獲得

知的システムがあらかじめ与えられていないルールを獲得する方法としては、ユーザが直接あるいは知的エディタを介してルールを入力する方法が考えられるが、この方法ではユーザがシステムに関する詳細な知識をもっていなければ、適切なルールを与えることができず一般性に乏しい。ここでは、ユーザがシステムに関する詳細な知識をもたない場合でも、問題解決を通して、自然言語によるユーザとの対話によって、システムが知識を獲得し、洗練する方法について述べる。

３．１．１　実行部の生成

ユーザから与えられた入力文をルールの実行部へ変換するために、ＬＡＰＳはユーザから得られた情報と方程式抽出用ルールを用いて方程式を抽出する。その方程式を連立させて消去法により解くことで一変数だけを含むような方程式を得ることができ、ＬＡＰＳはその方程式を実行可能な形式に変換しルールの実行部を生成する。この方法ではユーザから得られた情報以外に、システムがあらかじめもつ知識も実行部に埋めこまれることになるが、問題解決の際にＬＡＰＳが生成したルールから得られる方程式と、あらかじめ与えられたルールから得られる方程式を組合せることにより解を求めることができる。

方程式から実行可能形式への変換を行なう際に、ＬＡＰＳは事実グラフ中の対象を表わす定数を変数に置換して一般化を行ない、この変数の値が定まるように条件部を決定する。ユーザに複数回質問して問題を解いた場合には、質問をした目標変数について解いた結果の方程式からルールを生成し、複数個のルールを生成する。

３．１．２　条件部の生成

多くの問題で利用可能なルールを獲得するためには、できるかぎり一般的な形式の条件部をもつルールを生成しなければならない。例えば、『食塩水の濃度は、食塩の重さを食塩水の重さで割ったものに１００をかけたものに等しい。』と教えられた時には、食塩水の場合に使える公式ならば砂糖水の場合でも使えるのではないかということが類推できる。すなわち、食塩水の上位概念が水溶液であるということを知っていれば、この公式が水溶液一般に関しても利用できるという仮説を立てることができ、水溶液の下位概念である砂糖水でも利用できるであろうということが予測できる。ＬＡＰＳの推論機構にこのような予測を行なう機能を持たせた場合には、常に予測の必要性の検査が要求され実行効率が低下するので、予測を行なった結果を反映した一般的なルールを生成する必要がある。

方程式を利用するときの前提条件をユーザが与えるのは容易ではないので、ユーザは実行部だけを与え、ＬＡＰＳが問題文より生成された事実グラフからその条件を抽出する。事実グラフは特定の問題に関するものであるので、獲得した知識を以後の問題解決で利用できるように事実グラフから抽出した条件を一般化する必要がある。しかしながら、一般化し過ぎることなく十分な一般化を行なうことは困難であり、逆に一般化し過ぎに注意し過ぎるとほとんど一般化できず、一般的なルールを獲得するのに時間を要する。それゆえ、一般化し過ぎた条件を以後の問題解決で修正できるように、各ルールにはそのための情報を与えておく必要がある。すなわち、ルール修正過程の履歴を保持するために、獲得したルールには最も一般的であると考えられる条件と最も特殊であると考えられる条件の二つの条件を与える。ＬＡＰＳは問題を解きながら、この二つの条件を修正し一般化を進めていく。

最も特殊であると考えられる条件は、問題文より生成された事実グラフを直接利用可能な形式に変換することによって抽出される。事実グラフ中の一つの節点とその節点か

ら出ているリンクを、対応する一つのlisp関数の形式に変換し、その連言を最も特殊な条件部としてルールを生成する。

最も一般的であると考えられる条件は、最も特殊な条件に一般化規則を複数回適用することにより得られるが、一般化規則を単に適用しただけでは、方程式抽出のための前提条件として不適当な条件となる。すなわち、一般化規則を無制限に適用した場合には常に成立するような条件を生成してしまい、実行部中のすべての変数の値を定められない。したがって、実行部中の変数の値が定まるように一般化を行なわなければならない。ＬＡＰＳは最も特殊と考えられる条件部の中から変数の値を決定できる条件の連言を選び、その連言の中から冗長な条件を除き、それらの選言を最も一般的であると考えられる条件部としてルールを生成する。

３．１．３　ルールの修正と統合

一つの問題から一般化できる範囲には限界があり、また一般化した知識が必ずしも正しい知識であるとは限らないので、生成したルールを修正する必要がある。生成したルールには、最も特殊と考えられる条件と最も一般的と考えられる条件の二つの条件部がある。ＬＡＰＳはこのようなルールを適用する時に、ユーザと対話してルールの適用可能性を調べてルールの条件部を修正していく。すなわち、ルールの適用が適当である場合には特殊な条件を一般化し、ルールの適用が不適当である場合には一般的な条件を特殊化することによって、条件部の成立する範囲を狭めていく。このように、ＬＡＰＳは問題解決を行ないながらルールの修正を進めていき、一般的な条件と特殊な条件が等しくなった時点で、ルールが獲得できたことになる。

ルールの条件部が厳し過ぎるために、方程式が抽出できずに問題が解けない場合には、そのルールを利用できるように修正しなければならない。そのような不完全なルールの中には、ＬＡＰＳが生成したルールだけでなく、あらかじめ与えられたルールが含まれることもある。ＬＡＰＳは不完全なルールを直接修正するのではなく、新たに生成したルールとその不完全なルールを統合することによりルールを修正する。すなわち、一つのルールを生成するごとに、それがルール・ベース中の各ルールとの間で一般化できないかどうかを検査する。一般化できるルールが存在すれば、二つのルールを一般化して一つのルールとしてルール・ベースに登録し、不完全なルールを削除する。生成したルールだけではなくあらかじめ与えられたルールも修正することができるので、知識が不完全なために問題を解けない場合でも、ユーザと対話して問題を解くことができる。

３．２　求解過程からのルール生成

ＬＡＰＳは問題解決時に適用可能なルールが複数個ある場合には、ヒューリスティックスを用いて最も適切であると考えられるルールを選択して問題を解いていく。ヒューリスティックスの性質のため必ずしも最善のルールが選択されるとは限らないので、ＬＡＰＳは問題解決能力を改善するために、問題解決で利用されたルールを合成し新しいルールを生成する。

３．２．１　実行部の生成

問題解決過程で用いられたルールを合成したルールの実行部は、3.1.1で述べたのと同様な方法で、問題解決で利用されたＡＮＤ木中に貯えられたすべての方程式を合成した結果を実行可能な形式に変換することによって得られる。ＬＡＰＳは新しく生成されたルールを使って問題を解くことにより、いくつも方程式を立てたり、また、その方程式を解くことなく数値計算だけで答えを求めることができる。

しかしながら、問題解決のためにＬＡＰＳが生成したＡＮＤ木中に記憶されている方程式は、特定の問題を解くために利用された、変数が含まれない特殊な方程式である。

このような特殊な方程式を合成することによって得られる方程式は、そのままでは特殊過ぎて利用頻度の少ないルールしか合成できない。ＬＡＰＳは定数を変数に置換する規則によって一般化を行ない、変数を含むような実行可能形式を生成する。

ルールの実行部に変数を新たに導入する場合には、その変数の値を決定するような条件をルールの条件部に組み込まなければならない。例えば、一般化した結果、次の式(1)が得られたとしよう。ただし、＊で始まる名前及び大文字の名前は変数を表わしている。

＊x(X) = ＊y(Y) × ＊z(Z) ‥‥ (1)

式(1)の＊で始まる名前の変数の値を決定する場合に、次のような公式が与えられている時、

距離(X) = 速さ(X) × 時間(X) ‥‥ (2)

(1)と(2)のマッチングをとることにより、＊xは"距離"とのみ対応することがわかる。したがって、システムに与えられた公式を調べて制約条件を付加することにより(1)の式から次のような正しくない方程式の抽出を防ぐことができる。

時間(X) = 速さ(Y) × 距離(Z) ‥‥ (3)

あらかじめ与えられた方程式抽出用のルールには、"速さ"や"距離"などの数値に関連した属性がすべて定数で表わされているわけではなく、変数で表わされる場合もあるが、その場合には必ず条件部中にその値を決定する条件が含まれているので、条件部を調べてその変数値の候補を求めることができる。

３．２．２　条件部の生成

ルールの条件部は、問題解決時に適用されたルールの条件部を変数間のユニフィケーションを行なった条件の連言として得られる。この作業のために、適用されたルールからその実行部と条件部間の変数の対応を調べながら、連言中の変数のユニフィケーションを行なう。また、実行部に導入された変数の値を決定するような条件を得るために、あらかじめＬＡＰＳに与えられた各ルールの条件部から変数束縛のための制約条件を抽出する。

合成されたルールの条件部が成立した場合でも、一般化のために抽出される方程式が正しいことは保証されないので、3.1.2と同様に最も一般的であると考えられる条件と最も特殊であると考えられる条件を合成したルールに与えて、以後の問題解決でルールの修正を進めていく。

３．３　ルール生成の比較

ユーザの入力文からルールを生成しそれを一般化する過程は、データ駆動型で行なわれるために、適切な訓練例を与えなければルールが収束するまでに時間がかかる。それに対して、問題のモデルとして方程式抽出用のルールを用いたモデル駆動型の学習を行なう場合には比較的高速に学習が完了する。

４　ＬＡＰＳとの対話例

ここでは、実際のＬＡＰＳの対話例を用いてＬＡＰＳが学習していく過程について述べる。図３は、ＬＡＰＳに食塩水の濃度に関する公式を与えていない場合のＬＡＰＳとの対話例である。"->"で始まる文はユーザの入力した文である。食塩水の濃度に関する

問題を与えてからＬＡＰＳに質問した場合、ＬＡＰＳは知識が不足しているために問題を解けないので、ユーザに対して質問を行なって不足している知識を補おうとする。ユーザが『食塩の重さは食塩水の重さに食塩水の濃度を掛けたものを１００で割ったものに等しい。』と入力すると、ＬＡＰＳは問題を解決し獲得した知識をルールとして記憶する。

－＞３％の食塩水が１００ｇあります。
－＞食塩の重さは？
食塩の重さの求め方がわからないので食塩の重さの求め方を教えて下さい。
－＞食塩の重さは食塩水の重さに食塩水の濃度を掛けたものを１００で割ったものに等しい。
わかりました。どうもありがとうございました。
⋮
答えは３ｇです。

－＞３％の砂糖水が１００ｇあります。
－＞砂糖の重さは？

砂糖の重さは砂糖水の重さに砂糖水の濃度を掛けたものを１００で割ったものに等しいですか。
Ｙ／Ｎ？ｙ
わかりました。どうもありがとうございました。
⋮
答えは３ｇです。

図３　システムとの対話例１

ユーザから得られた情報は特定の場合に関する情報であるので、ＬＡＰＳは以後の問題解決で利用できるようにその情報を一般化したルールとして記憶する。一般化したルールからは必ずしも正しい方程式が抽出されるとは限らないので、正しくない方程式が抽出された場合にはルールの修正が必要となる。ＬＡＰＳは正しい方程式が抽出されたかどうかをユーザと対話して確認を行ない、ルールの適用が適切であった場合には特殊な条件を一般化し、そうでない場合には一般化しすぎた条件を特殊化する。図３の例では、食塩水について獲得したルールが砂糖水でも利用できるかどうかをＬＡＰＳは確認しルールの修正を行なっている。

次にＬＡＰＳが算数の文章題を解き、その過程から知識獲得を行なう例を示す。図４は『つるかめ算』の問題をＬＡＰＳに与えた場合に問題を解決した過程である。『つるの数は』という質問を与えるとＬＡＰＳは問題を解きはじめ、方程式抽出と式変形を繰り返し問題を解いていく。

ＬＡＰＳは問題を解いた過程から新しい知識としてルールを獲得し、それを一般化して以後の問題解決で利用する。ユーザから得た情報をルールの形式で記憶したときと同様に、ルールの一般化を行なうと、必ずしも正しい方程式が抽出されるとは限らないので、ユーザに対して確認をしてからルールの適用を行なう。図４の例ではりんごとみかんの問題に対して確認を行ない特殊な条件を一般化する。ＬＡＰＳは、このようにしてユーザと対話を行ない、ＬＡＰＳのもっている知識を洗練しながら問題を解決することができる。

－＞つるとかめがあわせて２０匹です。
－＞つるとかめの足があわせて６０本です。
－＞つるの数は？
一匹のつるの足の数は２本だから
cardnum(crane-1) = cardnum(leg-of-crane-1) * 1/2
つるの足とかめの足があわせて６０本だから
cardnum(leg-of-crane-1) = (60 - cardnum(leg-of-turtle-1))
一匹のかめの足の数は４本だから
cardnum(leg-of-turtle-1) = cardnum(turtle-1) * 4
つるとかめがあわせて２０匹だから
cardnum(turtle-1) = (20 - cardnum(crane-1))
以上より答えは10羽です。

－＞４０円のりんごと２０円のみかんをあわせて２０個買います。
－＞りんごとみかんをあわせたものの値段は６００円です。
－＞りんごの数は？

りんごの数は、くだものの数とみかんの単価との積とくだものの値段との差をみかんの単価とりんごの単価の差で割ったものに等しいですか？
Ｙ／Ｎ？ｙ
りんごの数はくだものの数とみかんの単価との積とくだものの値段との差をみかんの単価とりんごの単価の差で割ったものに等しいので、
cardnum(apple-1) = 10
以上より答えは10個です。

図４　システムとの対話例２

５　まとめ

本論文では、問題解決のための知識をもつ問題解決システムにおいて、自然言語によるユーザとの対話により知識を獲得する方法について述べた。ユーザから得られた知識を以後の問題解決に利用できるように、一般化したルールの形式で表現し、以後の問題解決において不完全なルールを修正することによって完全なルールが獲得できることを示した。

参考文献

[1] 桜井成一朗，志村正道：　算術問題解答システム，情報処理学会第３４回知識工学と人工知能研究会(1984)
[2] Davis, Randall and Buchanan, Bruce.G.: Meta-level Knowledge: Overview and application, Proc. of IJCAI,pp.920-927(1977)
[3] Mitchell, Tom M.: Generaization as Search, Readings in Artificial Intelligence,tioga,pp.517-542,1981
[4] Mitchell, Tom M.: Version Spaces:A Candidate Elimination Approach to Rule Learning, Proc. of IJCAI,pp305-310(1977)
[5] Dietterich, Thomas G. and Michalski, Ryszard S.: Comparative Review Of Selected Methods For Learning From Examples,Machine Learning:An artificial Intelligence Approach,pp41-82(1983)
[6] Michalski, Ryszard S.: A Theory And Methodology for Inductive Learning, Machine Learning: An artificical Intelligence Approach, pp83-134(1983)