知識工学と人工知能 34－3
（１９８４．５．８）

算術問題解答システム

桜井 成一朗　志村 正道
（東京工業大学工学部）

1 まえがき

人間の知的活動は幅広く、このような知的活動を代行する機械を構築するには、知能の中心的問題を解明することが必要である。本論文では、「算数の文章題を計算機に解かせる。」という問題をとりあげ、必要とされる知識の表現法や、問題解決を行う機能の実現方法について考察する。

「算数の文章題を計算機に解かせる。」ためには、まず入力された文章を計算機に理解させなければならない。入力文の解析は、構文解析と意味解析とに分けられる。ここでは構文解析に拡張ＬＩＮＧＯＬを用いた。本システムでは、構文解析の後入力文の表現する事実を計算機の内部表現に変換し、さらに与えられた事実関係から数式を抽出し、その数式を処理する。本論文では、上に述べたようなシステムについて述べ、その実際の動作結果を示す。

このような「質問応答システム」としては、ＢｏｂｒｏｗのＳＴＵＤＥＮＴが、よく知られている。ＳＴＵＤＥＮＴは、問題文を入力するとパターンマッチングにより連立方程式を抽出して問題を解くプログラムである。そのパターンマッチングは簡単なものであり、入力文の制限が強いという欠点があった。また、知識が整理されていないので、知識が増加すると知識の利用が困難になると考えられる。文献（３）のプログラムは、意味処理に重点をおき入力文をできるかぎり一つのフレームに整理していくものである。しかし、問題解決はプログラムを起動することにより行うので、あらかじめ問題のタイプを知る必要がある。

本システムでは、構文解析部をもっているので比較的柔軟な自然言語処理を行うことができ、知識を有向グラフとして実現しているので、知識の参照・変更が容易である特徴をもっている。さらに、公式を知識の形態として扱うことにより、問題のタイプに関係なく公式を使うことができる。公式の抽出は、対象を限定することにより不必要な公式の抽出を抑制している。本システムは、簡単な数式処理プログラムをもっているので、その処理が人間に理解しやすいという利点がある。また本システムは、つるかめ算などの小学生の上級の種々の問題を解く能力をもっている。

2 構文解析

算数の文章題を解くためには、まず入力文を解釈しなければならない。本システムでは、構文解析に拡張ＬＩＮＧＯＬを用いている。ＬＩＮＧＯＬは、Ｐｒａｔｔによって開発された、高速で柔軟な構文解析機構をベースにしたプログラミングシステムであり、拡張ＬＩＮＧＯＬは、電総研・推論機構研究室で、自動分かち書きや文法規則の適用条件記述による例外処理の機能が付け加えられたものである。ＬＩＮＧＯＬでは、入力文をあらかじめ登録された文脈自由文法に基づいた構文規則にしたがって解析し、構文解析木を組み上げていく。

3　問題の内部表現

入力文の構文解析が終了後、問題文の表現する世界を計算機の内部表現に変換しなければならないが、適切な内部表現を記述するためのデータ構造が重要となる。以下、本システムにおけるデータ構造について考える。

3．1　事実グラフ

算数の問題で述べられる事柄は、特定の対象について、あるいは、それらの間で成立する関係であることが多い。人間は定理などを常識としてもっているが、計算機もあらかじめ知識をもつ必要がある。ここでは、知識の表現法として一つの有向グラフを考え、その有向グラフを事実グラフと呼ぶことにする。

事実グラフは、問題文解析以前から計算機中に存在しているが、この部分は、意味解析時に生成される部分とは別の長期記憶領域（ＬＴＭ）と呼ばれる記憶領域に格納されている。ＬＴＭは、人間の常識に相当する知識を記憶する領域であり、意味解析により変更されることはない。

意味解析終了後に生成される部分を記憶する領域は中期記憶領域（ＭＴＭ）と呼ばれる。また、構文的には間違いはないが意味的には間違いである文が入力されたときには、意味解析時に生成された部分は、ＭＴＭに記憶せずに捨てなければならない。そこで、意味解析時に生成された部分を一時的に記憶する短期記憶領域（ＳＴＭ）が必要になる。一つの入力文の意味解析が成功したら、ＳＴＭの内容はＭＴＭに移されて、次の入力文の処理が行われることになる。

3．2　概念定義ノード

ＬＴＭ中には、個々の概念に対応する節点（ノード）が存在するが、このノードを概念定義ノードと呼び、＜概念名＞＊ で表現する。また、各概念はＳＵＰＥＲＣリンクで結合されて、階層構造を形成している。概念定義ノード間は、属性定義リンクと呼ばれるリンクで結合されている。ここで、属性定義リンクの名前は、一つの関数名に対応する。例えば、概念定義ノードＬｉｑｕｉｄについて考えてみよう。液体には、物質を溶かす性質があり、その性質を示すために属性定義リンクが用いられる。すなわち、図３－１に示すように、Ｌｉｑｕｉｄノードは、ＳＯＬＶＥＮＴ（溶媒）、ＳＯＬＵＴＥ（溶質）、ＣＯＮＣＥＮＴＲＡＴＩＯＮ（濃度）の３つの属性定義リンクをもっている。具体的には、概念定義ノードは連想リストの形で記憶される。ＳＵＰＥＲＣや各属性定義リンク名をｋｅｙｗｏｒｄとして、ＬＩＳＰ関数のＡＳＳＯＣを用いることにより、各リンクの指す概念名を知ることができる。

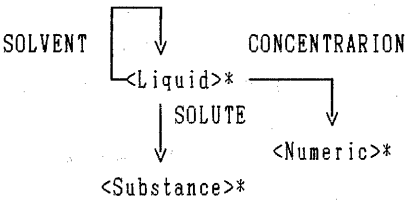

図３－１　属性定義リンクの例

3．3　述語定義ノード

述語定義ノードは、事実関係を記述する述語を表現するためのノードであり、役割定義リンクと呼ばれるリンクによって、概念定義ノードに結合される。この述語定義ノードは、述語が真となるための制約を明示するものである。

３．４　対象ノード

一つの事実を表現するためには、いくつかの関係とその関係によって結びつけられる対象を表現する対象ノードが必要になる。ここでは、対象ノードについて説明する。

３．４．１　ＣＡＴＥＧＯＲＹリンクとＩＳＯＮＯＤＥリンク

すべての対象ノードは、ＣＡＴＥＧＯＲＹリンクを１本もち、このリンクは、概念定義ノードへ結合される。また、ＣＡＴＥＧＯＲＹリンクを結ぶ際に、概念定義ノードからその対象ノードへＩＳＯＮＯＤＥリンクを結ぶことにする。この二つのリンクにより、概念定義ノードと対象ノードは相互に参照が可能となる。ＩＳＯＮＯＤＥリンクは、他のリンクと異なり、概念定義ノードの属性リストの中に記憶される。その属性値は、同一の概念に属するすべての対象ノードのリストである。このＩＳＯＮＯＤＥリンクにより、問題文中で指示された同一名称の対象を容易に知ることができる。

３．４．２　型

各対象ノードは、次の３つの型のいずれかをもっている。

１．個体型　ＣＡＴＥＧＯＲＹリンクの端点の概念の要素であることを示す。

２．集合型　ＣＡＴＥＧＯＲＹリンクの端点の概念の部分集合であることを示す。

３．性質型　ＣＡＴＥＧＯＲＹリンクの端点の概念の性質を示す。この型をもつ対象ノードは、同一の概念に属する個体型・集合型の対象ノードから参照することができる。

例えば、「つるの足の数は２本である。」という知識は、性質型の対象ノードに記憶される。この対象ノードは、同一の概念に属する対象ノードから参照できなければならない。

３．４．３　属性リンク

属性リンクは、関数名に対応するものであり、対象ノードの属する概念を定義域中に含む関数があれば、その対象ノードは、関数名と対応する属性リンクをもつことができる。Ｌｉｑｕｉｄに属する食塩水について考えると、図３－２に示すように、Ｌｉｑｕｉｄ３からＳＯＬＶＥＮＴリンクがＷａｔｅｒ２へ、ＳＯＬＵＴＥリンクがＳａｌｔ１へ結ばれている。このように属性リンクを結ぶことによって、「食塩水」についての事実グラフを生成することができる。

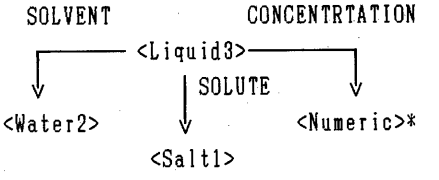

図３－２　属性リンクの例

３．５　関係ノード

一つの事実関係を表現するのが、関係ノードである。ここでは、関係ノードついて説明する。

３．５．１　ＰＲＥＤＩＣＡＴＥリンクとＩＳＯＮＯＤＥリンク

すべての関係ノードはＰＲＥＤＩＣＡＴＥリンクを１本もち、このリンクは述語定義ノードへ結合される。これは、ちょうど対象ノードのＣＡＴＥＧＯＲＹリンクに対応するものであり、対象ノードと同様に、関係ノードは述語定義ノードとＩＳＯＮＯＤＥリンクで結ばれる。

３．５．２　役割リンク

役割リンクも、属性リンクと同様に対応する述語定義ノードが役割定義リンクをもっているときに結ぶことができる。しかし、現在は述語に階層性が意識されていないので、対応する述語定義ノードにない役割リンクをもつことはできない。

４　方程式の抽出

入力文の解析終了後、計算機内には事実グラフが構成されて、次に、対象ノードの値が質問されることになる。この値を求めるために、本システムではその問題に応じたプログラムを起動するのではなく、事実グラフから方程式を抽出しその方程式を順次解いていくという手法をとっている。以下、その方程式の抽出方法について説明する。

４．１　ＲＵＬＥ

数式を得る作業は、与えられた事実と常識的な定理を基に複雑な推論を行なうことによって進められることもある。また、問題全体から適切な特徴を抽出し、これによって求めるべき数式を推定するような場合もある。ここでは、簡単な推論を行なって、方程式を抽出するためのデータ構造について考える。例えば、次の公式が与えられたとする。

（長方形の面積）＝（長方形のたて）×（長方形のよこ）

この公式は、たて・よこの具体的な値から面積の値を計算するだけでなく、たて・よこの値を副目標として設定するために利用されることもある。それゆえ、本システムでは多くの目的で使用されるデータ構造として次のようなものを考える。一つのデータ構造をＲＵＬＥと呼ぶことにし、そのＲＵＬＥは、ＴＡＲＧＥＴ部、ＴＥＭＰＬＡＴＥ部、ＥＱＵＡＴＩＯＮ部、ＶＡＲＩＡＢＬＥ部、ＲＡＮＧＥ部、の５つの部分からなっている。

ＶＡＲＩＡＢＬＥ部の値は、マッチングの際に束縛すべき変数のリストである。公式の抽出は基本的にパターンマッチングにより行われるので、効率を向上させるために各変数を束縛しながらマッチングを行う。また、各変数の値はすべて異なるものとしており、この仮定により無意味な照合を防止することができる。二つの変数の値を等しくしたいときは、同一の変数を用いる。同一の変数名を使うことにより試されるノードの数が減りマッチングのスピードは向上する。

ＴＡＲＧＥＴ部は、変数名（ＶＡＲＩＡＢＬＥ部の変数名）とその変数を引数とするべき関数名からなる連想リストを値とする。このＴＡＲＧＥＴ部は、ＲＵＬＥの適用できる環境を示し、変数を束縛する第一目標となる。ある関数値を知りたいときには、その関数名がＴＡＲＧＥＴ部に含まれるＲＵＬＥだけが適用される。次の段階では、ＴＡＲＧＥＴ部の変数の値を求めたい関数の引数に束縛する。このマッチングは、連想リストの先頭の要素から順に試み、その要素の中に求めたい関数名が含まれるかどうかをＬＩＳＰ関数のＭＥＭＢＥＲで調べる。含まれていれば、ＴＥＭＰＬＡＴＥ部のマッチングを試み、含まれていなければ、次の要素に試みる。ＴＥＭＰＬＡＴＥ部は、マッチングをとるべき事実グラフ中のノードと同様の構造をもつ有向グラフのノードを要素とするリストである。このようにして、マッチングするすべてのパターンが探しだされる。

ＥＱＵＡＴＩＯＮ部にはＬＩＳＰのＳ式が記述され、ＴＥＭＰＬＡＴＥ部が、事実グラフの部分グラフとマッチした時点の環境で評価される。すなわち、Ｓ式中の変数は束縛された値に置き換えられて評価される。しかし、ＴＥＭＰＬＡＴＥ部にマッチしただけでは、その公式を適用するために必要なすべての条件を満足しているとは限らない。いいかえれば、ＴＥＭＰＬＡＴＥ部だけでは、細かい情報の検査ができない場合がある。また、それ以外の場合でも公式の汎用性を増すために、故意に不十分なマッチングをする場合もある。例えば、長方形と平行四辺形の面積は同様の公式を用いることができるが、ＴＥＭＰＬＡＴＥ部のマッチング時には検査することができない。このような検査は、ＥＱＵＡＴＩＯＮ部に条件部を加えることにより可能となる。すなわち、方程式の抽出は３段階になっていて、ＥＱＵＡＴＩＯＮ部の条件部が真でなければ、方程式は抽出されない。

４．２　ＲＵＬＥの適用

ＲＵＬＥを適用する際に、事実グラフ中のすべてのノードに対してマッチングを試みるのは、極めて非効率的であり、事実グラフ中のノードから必要なものだけにマッチングを試みる必要がある。そのために、目標とするノードからＳＵＰＥＲＣとＩＳＯＮＯＤＥ以外のリンクをたどることにより到達できるノードを選びだすことにし、この選びだされたノードのみにマッチングを試みることで、すべてのノードに試みるより効率はよくなる。

本システムの最大の特徴は、問題文に応じてプログラムを起動するのではなく、与えられた事実から方程式を抽出することにある。そこで、各ＲＵＬＥには基本的な公式だけを与えて、問題に応じてその公式を組み合わせて問題を解いていくことになる。したがって、ある目標に対してすべてのＲＵＬＥの適用を試みて、その目標から抽出できるすべての式を求める。なお、一度抽出された方程式は、計算機内のＢＵＦＦＥＲに貯えられるので、必要な式がＢＵＦＦＥＲの中にあるときには、方程式を探しにいく必要がない。

```
(R6 RULE
    (TARGET = ((?S AREA BASE HEIGHT)))
    (TEMPLATE = ((?S object (CATEGORY = ?X) (TYPE = Individual)))
    (EQUATION = (AND (MEMQ ?X '(Parallerogram Rectangle))
                     `((,?S . AREA) = ((,?S . BASE) * (,?S . HEIGHT)))))
    (%VARS = (?S ?X))
    (%RANGE = 0))
```

図４－１　ＲＵＬＥの具体例

具体的なＲＵＬＥの構成について述べておく。図４－１にその例として長方形の面積に関する公式を示す。ＴＡＲＧＥＴ部は、この公式がＢＡＳＥ・ＨＥＩＧＨＴ・ＡＲＥＡを求めたいときにだけ利用できることを示している。ＴＥＭＰＬＡＴＥ部は、このＲＵＬＥとマッチする部分グラフがただ一つのノードであることを示している。ＥＱＵＡＴＩＯＮ部では、この公式を利用するための最終検査をしている。この検査を通過すれば、この部分のＳ式が公式として利用される。

5　方程式の解法

算数の問題が与えられたときに、どのようにして問題を解いていくかについて説明しよう。一つの解法として考えられるのは、事実グラフ中のすべてのノードについて方程式の抽出を行い、その方程式を連立させて問題を解いていく方法である。この方法では不必要な式の抽出を行うことになり、極めて効率が悪い。方程式の抽出は、ある目標を定めてその目標に沿って進められるべきである。それゆえ、意味解析部はそのノードを目標に設定して方程式を抽出し問題を解いていく。このとき、新しい変数に関する方程式が必要になれば、その新しい変数を新しい目標に設定して方程式抽出部を呼びだすことになる。すなわち、方程式解答部と方程式抽出部は相互に制御しあいながら解を求めていくことになる。本システムでは、方程式抽出部と方程式解答部とを別々のモジュールとして、その二つが相互に制御しあう方式を採用している。

方程式解答部に、変数Ｘｉという変数と式Ｖ(Ｘｉ,Ｘｊ)＝Ｆ(Ｘｉ,Ｘｊ)が与えられたとき、Ｘｉ＝Ｇ(Ｘｊ)の形に変形する。このとき、Ｇ中のすべての未知変数の値を求めようとするが、それには、ＡＮＤ木を生成してすべてのＡＮＤ木中のノードが定数または自分の祖先のノードがもつ変数だけを含む式で表現されるようになるまで、式の変形・代入が行われる。すなわち、連立方程式を変数消去の方法で解いていくことになる。

問題が複雑になり事実グラフ中のノードが増加すると、一つの目標から抽出される方程式の数が増加するが、計算機には複数個の方程式が抽出されたときに、どの方程式を使えばよいのかわからない。それゆえ、不適当な方程式を使ってしまったときには、その方程式を使った時点までバックトラックしなければならない。しかし、実際にはどこまでバックトラックするのかを決定するのは、大変困難であり、本システムでは、解けなくなった直前のところへバックトラックさせている。

6　システムの概要

6．1　言語処理

本システムでは、制限された自然言語風の人工言語で記述された問題文を解析し、その結果に応じて、データベースの更新・変更を行っている。言語処理について例を示そう。次の例では、ノードの名前だけが異なる同様の事実グラフが構成される。

例１　「重さ１００ｇの食塩水の濃度は１０％です。」
例２　「食塩水の重さは１００ｇです。」「食塩水の濃度は１０％です。」

例１に対して、「食塩の重さは？」という質問をすれば、「答は１０ｇです。」という計算機からの返答がある。また、例２に対しても同様の結果を得る。すなわち、同一の名詞句に対しては、直前の名詞句によって指示された対象ノードと同一であると解釈する。ただし、型の検査だけは行われ、同一の型のものがない場合には、新しいノードが生成される。また、修飾語がある場合には、その修飾語に応じてノードが生成される。次の例は、やはり同様の事実グラフが構成される。

例３　「食塩水に重さ１０ｇの食塩を加えた。」
例４　「別の食塩の重さは１０ｇです。」「食塩水に食塩を加えた。」

例４で『別の』をとってしまうと、まったく異なる事実を表現することになる。すでに食塩水のノードが生成されていれば、その食塩水に溶けている食塩を指示したことになる。さらに悪いことは、その食塩の重さを求めることによって矛盾が生じても、その検査を行う機能がないので、この入力文を受理してしまうことである。しかしながら、あきらかに矛盾する主張は退けることができる。

「つるの足」が具体的にどのように構成されるかを述べる。あきらかに、「つる」と「足」に対応する二つの対象ノードが生成されるが、この二つのノードを意味のある形に結びつけなければならない。そこで、Ｂｏｄｙｐａｒｔの対象ノードを生成して、このノードと「つる」のノードを結びつける。さらに、Ｂｏｄｙｐａｒｔと「足」を結ぶことによって、「つるの足」に対応する部分グラフが完成する。このように、中間にＢｏｄｙｐａｒｔの対象ノードを設けることにより、「足の指」・「つるの羽」などの表現を容易に事実グラフに変換することができる。

６．２　意味処理プログラム

意味処理プログラムは、事実グラフ中のノード間をリンクで結合させることにより実行される。例えば、「長方形の面積」という名詞句が適格であるかどうかの検査は、ＬＴＭ中の概念定義ノードを調べることにより行われ、ＡＲＥＡという属性定義リンクが＜Rectangle＞* からでているか、または、その上位概念からＡＲＥＡリンクがでている場合に限りリンクがつながれる。しかし、このときＣＡＲＤＮＵＭリンクは、対象ノードの型を集合型に書き換えて無条件につながれる。

動詞・形容詞には一つの述語を割りあてておき、関係ノードを生成して対象ノードへ役割リンクをつなぐ。このとき、どの役割リンクをつなぐかは、辞書部に登録された情報により定められる。動詞・形容詞に対する辞書部には、役割名と助詞の dotted pair で情報が記憶されている。

６．３　実験結果

本システムは、東工大計算機センターのＨＩＴＡＣ　Ｍ－２８０Ｈ上のＵＴＩＬＩＳＰで実現されている。システムとの対話例を図６－１に示す。この問題の計算には２９０７ｍ秒、ガベージコレクションには１２６４ｍ秒かかるが、コンパイルしていないので、コンパイルすれば約５倍位の速さになると予想される。本システムは、和差算・消去算、つるかめ算、出会い算、食塩水の問題、図形の面積に関する問題を解く能力をもっている。これらの問題の多くは、声の教育社の「特殊算」の中から選出した。

７　あとがき

計算機に文章題を解かせようとするとき、解決しなければならない主要な問題は、自然言語の理解と日常的知識の表現・利用の問題である。自然言語処理は、拡張ＬＩＮＧＯＬを用いたが、日本語としてはまだ不自然な文章の入力しかできない。より自然な文章の入

力は今後の課題の一つである。知識表現のために本システムで用いた事実グラフは、名詞の指示する対象の同一性や入力文の適格性を調べるのには、役立つ。しかし、述語に関しては不備な点が多く改良していかなければならない。問題文で指示された内容の認識後、必要なのは適当な部分目標を設定して問題を解決していくことであり、ＲＵＬＥの記述もさらに改良が必要であろう。

```
      ワタシハ　サンジュツ　ガ　トクイ　デス。
      シカシ、　ニホンゴ　ハ　ニガテ　デス。
      モンダイ　ヲ　ダシテクダサイ！！
>ツル　ト　カメ　ガ　アワセテ　３０　ヒキ　デス。
      ツズケテクダサイ。
>ツル　ト　カメ　ノ　アシ　ガ　アワセテ　１００　ホン　デス。
      ツズケテクダサイ。
>ツル　ノ　カズ　ハ？
      CARDNUM(Crane0002) = NVALUE(Cardnum0005)
      CARDNUM(Leg0010) = CARDNUM(Crane0002) * CARDNUM(Leg-of-crane)
      CARDNUM(Substance0013) = ( CARDNUM(Leg0010) + CARDNUM(Leg0008) )
      CARDNUM(Substance0013) = 100
      CARDNUM(Leg0008) = CARDNUM(Turtle0001) * CARDNUM(Leg-of-turtle)
      CARDNUM(Substance0004) = ( CARDNUM(Crane0002) + CARDNUM(Turtle0001)  )
      CARDNUM(Substance0004) = 30
      CARDNUM(Turtle0001) = ( -1 * CARDNUM(Crane0002) + 30 )
      CARDNUM(Leg-of-turtle) = 4
      CARDNUM(Leg0008) = ( -4 * CARDNUM(Crane0002) + 120 )
      CARDNUM(Leg0010) = ( 4 * CARDNUM(Crane0002) + -20 )
      CARDNUM(Leg-of-crane) = 2
      CARDNUM(Crane0002) = 10
      NVALUE(Cardnum0005) = 10
        コタエ　ハ　１０　ワ　デス。
      ツズケテクダサイ。
>オワリ。
   サヨウナラ！
```

図６－１　システムとの対話例

参考文献

（１）牧尾善憲：東京工業大学情報工学科修士論文(1982)．

（２）牧尾善憲，志村正道：”簡単な算数の文章題を解くプログラム”，
情報処理学会第２９回知識工学と人工知能研究会(1983)．

（３）高木朗　他：”簡単な算数の問題を解く問題解決プログラム”，
信学技報　Vol.80,No277-AL80-89(1981).

（４）Barr,A. and Feigenbaum,E.A.,
"The Handbook of Artificial Intelligence" Vol.1,pp.284-285(1980).