知識工学と人工知能　41－1
（１９８５．７．４）

算数の文章題を解くシステムにおける問題の内部表現について

安西　祐一郎　（北海道大学）　上里　譲　　田村　淳　（慶応義塾大学）

１．　はじめに

算数の文章題は，人工知能，特に自然言語処理と問題解決を組み合わせたシステムの研究のためによく使われてきた（Bobrow，1968；桜井，志村，1984など）。これらの研究は，もちろんさまざまな成果をあげてきたとはいえるが，一般的にいって，文章題そのものをいかにうまく計算機に解かせるかを目標にしており，問題の構造そのものを分析して，それをプログラムに生かそうとしたものは少ない。

本研究では，小学校１，２年生の算数の文章題について，問題の構造を分析した結果を示すとともに，それを用いて試作した問題解決システムJAWS（Japanese Arithmetic Word problem Solver）について説明する。　研究のもともとの目的は，小学校の児童が自分で問題を作り，システムに解かせようとすることによって問題自体の構造を理解させるという，intelligent computer-assisted problem understanding systemを開発するための基礎研究であり，システムの設計はすべてこの目的の観点に沿って行われていることを付記しておきたい。

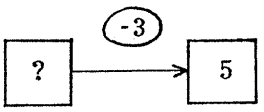
２．　例　題

小学校１年の算数の問題には，たとえばつぎのようなものがある。

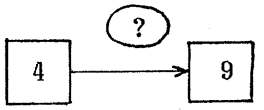
［１］りんごが　あります。　たろうくんが　３こ　たべました。
　　いま　５こ　あります。　はじめ　いくつ　ありましたか。

［２］りんごが　４こ　ありました。　いま　９こ　あります。
　　たろうは　いくつ　いれましたか。

上の２つの問題は，ともに簡単なたし算で解けるが，問題の構造は異なっている。つまり，ある対象の状態を　□　，状態の変化を $\xrightarrow{\bigcirc}$ で表し，未知数を　？　で表すとすれば，［例１］，［例２］は，それぞれつぎのように表現できる。



［1］



［2］

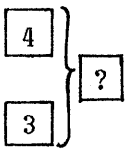
また，つぎのような問題を考えてみよう。

［3］りんごが　5こ　あります。　みかんが　4こ　あります。
　　みんなで　いくつ　ありますか。

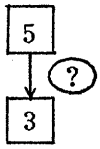
［4］りんごが　5こ　あります。　みかんは　3こ　あります。
　　りんごは　みかんより　いくつ　おおいですか。

［5］たろうは　はこに　りんごを　いれました。
　　はなこは　たろうより　5こ　すくなく　いれました。
　　はなこは　8こ　いれました。　たろうは　いくつ　いれましたか。

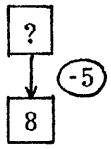
これらの問題の構造もまた，上と同じようにして以下のように表現することができる。　ただし，以下で，↓○ は比較を表し，} は合併を表す。



［3］



［4］



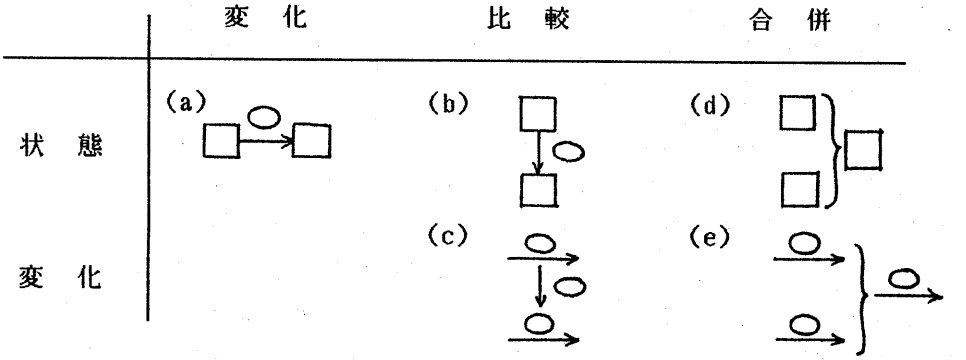
［5］

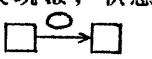
小学校１年の算数の文章題の多くは，上のような構造を持っていると考えられる。実際，われわれのグループで１年生の文章題２１１題について調べた結果，２１１題中１８１題は上の記述で表現できることがわかった。　そこで，以下では，上の表現をもう少し形式化するとともに，それを組み込んだわれわれの問題解決システムJAWSについて説明することにする。　なお，われわれの分析した２１１題の中で，上の表現で記述できない問題には，つぎのような問題がある：　（イ）順序数に関する問題，（ロ）式から問題文を作らせる問題，（ハ）「どちらがどれだけおおい

か」のように，未知数が数字以外のものを含む問題，（ニ）「ちがいはいくつか」のように，比較の方向が指定されない問題。　このうち（ハ）と（ニ）は，表現構造を少し複雑にすれば扱えるが，ここでは省略する。

３．　問題の内部表現

上の例題で示したような構造表現を，ここでは（問題の）**内部表現**と呼ぶことにし，さきにあげた４個の要素記述子を組み合わせて作られるつぎのような５種類の内部表現を定義する：　（ａ）状態の変化，（ｂ）状態の比較，（ｃ）変化の比較，（ｄ）状態の合併，（ｅ）変化の合併。　これらの内部表現は，図式的には，下のように書ける：



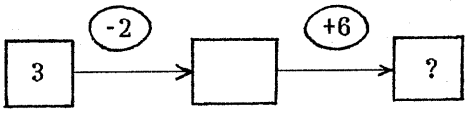
たとえば，さきにあげた例［１］の内部表現は，状態の変化にあたり，状態記述子２個と変化記述子１個の組み合わせで，  のように表される。

さらに，（ａ）ー（ｅ）のそれぞれについて，未知数が表現どこに来るかによって，内部表現を分類することができる。　それぞれに対して未知数のはいりうる場所には３か所の可能性があるが，等価なものを除くと，結局全部で１３種類になる。

これら１３種の内部表現と例題を**図１**に示しておく。

以上にあげた内部表現は，さらに（制限つきの）複合化を許すことによって，生成的にすることができる。　すなわち，図１の１３種類の（要素的）内部表現に対して，状態の重ね合わせによる複合を許した任意の記述を，一般に内部表現と呼ぶことにする。たとえば，下左の表現は２つの状態変化表現が複合された内部表現であり，たとえば下右の問題が対応する。

［６］



りんごが　３こ　あります。
たろうくんが　２こ　たべました。
はなこさんが　６こ　いれました。
いくつ　ありますか。

４．　問題解決システム

われわれのシステムJAWSは，ひらがな文章や式，答の入出力を Techtronix 4052 上のBASICで，自然言語処理と問題解決部分の処理を VAX11/750 上の FRANZ LISP で行い，それらを結合したものである。

自然言語処理サブシステムはごく単純に構成されており，文章は語単位で入力される。　つまり，分かち書きや形態素解析の機能は持っていない。　また，parsing は，大略 Schank の Conceptual Dependency Theory に基づいた one-pass の意味 parser を用いており，文節単位でスタックを積みながら動詞の部分を処理するときに（linguistic な）意味表現を生成する。　したがって，文節の順序はほぼ任意である。　また，別に推論のための知識を持っており，それを用いて意味表現生成のための推論を行う。　たとえば、「りんごが３こあります。。　５こたべたら・・・」で「５こ」が何を指すかといった推論はこれによって簡単に行える。

さらに，システムは（linguistic な）意味表現を問題の内部表現に変換するための前処理を行うようになっている。　たとえば，所有(possesion) は存在（is）に，，所有権の移動（atrans）や edible なものの消化（ingest）は移動（ptrans）に変換される。　たとえば，「たろうはりんごを５こもっています」という文は，まず，

```
(poss (actor (tarou))
      (object (ringo (kosuu (5 (tanni (ko)))))))
```

という（linguisticな）意味表現に変換され，さらに，

```
(is (actor (ringo (kosuu (5 (tanni (ko))))))
    (loc (tarou)))
```

という表現に変換される。　後者のような表現を一般に，primitive な意味表現と

呼ぶ。

つぎに，問題解決サブシステムは，入力として上のような primitiveな意味表現の列を受け取り，まずそれを，situation部とquestion部に分ける。 situation部は問題の条件部分であり，question部は質問文の部分にあたる。 つぎに，question部に含まれる，状態と変化に関する情報（属性と呼ぶ）が抽出される。 たとえば，actor や loc の情報は属性にあたり，すぐ上の例では，actor に対応する ringo，および loc に対応する tarou が属性である（ただし，この例は question部ではない）。

question 部の状態と変化に関する情報が抽出された後，システムは question部から situation部に向けて後ろ向き，および前向きに，situation部の中の question部と関係のある情報を探しにいく。 たとえば，「りんごが ５こ みかんが ６こ ありました。 みかんを ４こ たべました。 りんごは いくつ のこっていますか。」という問題では，question 部の状態に関する属性は ringo であり，後ろ向きの探索によって，ringo と関係のない mikan に関する情報は除かれ，situation部の情報としては，「りんごが５こ」の部分だけが残る。 （このような類の冗長な問題は，パフォーマンスだけを問題にしたシステムでは考えられないであろうが，本稿のはじめにのべたように小学生に問題を作らせて計算機に解かせるというシステムでは十分に起こりうるので，重要な問題点になる。） ただし，situation 部に異なる属性間の関係が含まれている場合，たとえば，「りんごが ３こ あります。 みかんは りんごより ４こ おおく あります。 みかんは いくつ ありますか。」のように比較の関係が含まれている場合や，question部の属性が特定されていない場合，たとえば，さきの例［３］の「りんごが ５こ あります。 みかんが ４こ あります。 みんなで いくつ ありますか。」のような場合には，前向きの探索を行って situation部の情報を抽出する。

上のような前処理の後，問題解決サブシステムは，さきにのべた問題の内部表現の生成にはいる。 まず，状態，変化，比較の primitiveな意味表現のテンプレートとして，つぎのようなものが用意されている。

```
        状態                変化                比較

(is (actor +++)  (trans (actor +++)  (is_or_trans (actor +++)
    (loc +++))           (object +++)              (object +++)
                         (from +++)                (loc +++)
```

(to +++))

(from +++)
(to +++)
(comp more_or_less)
(than +++))

システムは，状態あるいは変化に関する属性を見つけると，それを核として，上のテンプレートに合うように，他の情報を primitive な意味表現から抽出する。合併については，primitive な意味表現の中の question 部の特殊なパタンを見つけたときに，合併の内部表現を生成するのに必要な情報が集められるようになっている。たとえば，「ぜんぶで　いくつ　ありますか」という文の primitiveな意味表現は，

(is (actor (all (kosuu (howmanyvar (tanni (ko))))))
(loc (any (id (1)))))

となり，この中の all に基づいて situation 部の状態属性を調べにいく。　ただし，上で，howmanyvar は未知数を表し，(loc (any (id (1))))は場所が（任意の）(id (1))であることを表す。

問題解決サブシステムによって生成される上のようなテンプレートのインスタンスが，前にのべた要素記述子の例にあたり，たがいに関係づけられたインスタンスの集合がさきにのべた内部表現の例にあたる。　このさきは省略するが，システムはこうして作られた内部表現から式を生成し，それを表示するとともに，解いて答を出力する。　図２に，前にあげた例［１］についての問題解決プロセスを示しておく。

５．　おわりに

小学校１年生の算数の文章題に対して，問題の内部表現を１３種類のどれか，あるいはその複合表現に帰着し，それを用いて問題を解く計算機システムについて述べた。　本稿で説明したシステムに対しては，その評価のために，非公式にではあるが，実際に小学校の児童に問題を作成してもらい，その場でシステムに解かせるという試みがなされ，いくつかの興味ある結果を得ている。　また，小学校の算数の先生方に本システムの教室での利用に関するアンケート調査も行っている。　さらに，小学校児童を被験者として算数の文章題に関する認知心理学的実験も併せて

行い，興味ある結果を得ている。　この実験は，Riley, Greeno & Heller (1983)の実験とも関連したものであるが，彼らのものとは異なっている。　これらのむしろ心理学的な結果については，別の機会に報告したいと考えている。

	内部表現	例　題
状態の変化	? —(-3)→ 5	たろうがみかんを３こたべました。 かごには５このこっています。 みかんはなんこありましたか。
	10 —(?)→ 6	きのうかごにみかんが１０こありました。 きょう６こになっています。 なんこたべましたか。
	1 —(+4)→ ?	みかんがかごに１こあります。 ４こかいました。 いまかごにみかんはなんこありますか。
状態の比較	? (9) —(-5)→ 9	みかんはりんごより５こすくないです。 みかんは９こあります。 りんごはなんこありますか。
	12 (8) —(?)→ 8	みかんが１２こあります。 りんごは８こあります。 りんごはみかんよりなんこすくないですか。
	7 (?) —(+5)→ ?	りんごが７こあります。 みかんはりんごより５こおおいです。 みかんはなんこありますか。
変化の比較	? (8) —(+6)→ 8	たろうははなこより６こおおくみかんをたべました。 たろうは８こたべました。 はなこはなんこたべましたか。
	10 (15) —(?)→ 15	たろうはりんごを１０こかいました。 はなこは１５こかいました。 はなこはたろうよりなんこおおくかいましたか。
	7 (?) —(-3)→ ?	たろうはあめを７こたべました。 じろうはたろうより３こすくなくたべました。 じろうはなんこたべましたか。
状態の合併	? (17) 9	おとこのこがいます。 おんなのこは９にんいます。 みんなで１７にんいます。 おとこのこはなんにんいますか。
	5 (?) 8	りんごが５こあります。 みかんは８こあります。 みんなでなんこありますか。
変化の合併	? —(10)→ 6	たろうははなこにおはじきをあげました。 あつしは６こあげました。 はなこはあわせて１０こもらいました。 たろうはなんこあげましたか。
	3 —(?)→ 4	みかんをはこから３こだしました。 また４こだしました。 みんなでなんこだしましたか。

```
Input is
(ringo ga arimasu)


Input is
(tarou kun ga 3 ko tabemashita)


Input is
(ima 5 ko arimasu)


Input is
(hajime nanko arimashita ka)


*** CD forms are
  (is (actor (ringo (kosuu (xx (tanni (ko))))))
      (loc (any (id (1)))))

  (ingest (actor (tarou))
          (object (ringo (kosuu (3 (tanni (ko))))))
          (from (any (id (1)))))

  (is (actor (ringo (kosuu (5 (tanni (ko))))))
      (loc (any (id (1)))))

  (is (actor (ringo (kosuu (howmanyvar (tanni (ko))))))
      (loc (any (id (1)))))


*** Primitive CD forms are
  (is (actor (ringo (kosuu (xx (tanni (ko))))))
      (loc (any (id (1)))))

  (ptrans (actor (tarou))
          (object (ringo (kosuu (3 (tanni (ko))))))
          (from (any (id (1))))
          (to (tarou)))

  (is (actor (ringo (kosuu (5 (tanni (ko))))))
      (loc (any (id (1)))))

  (is (actor (ringo (kosuu (howmanyvar (tanni (ko))))))
      (loc (any (id (1)))))

                  •
                  •
                  •
                  •
                  •


***third-cdforms***

(setq *cdforms*
      '((is (actor (ringo (kosuu (xx (tanni (ko)))))) (loc (any (id (1)))))
        (ptrans (actor (tarou))
                (object (ringo (kosuu (3 (tanni (ko))))))
                (from (any (id (1))))
                (to (tarou)))
        (is (actor (ringo (kosuu (5 (tanni (ko)))))) (loc (any (id (1)))))
        (is (actor (ringo (kosuu (howmanyvar (tanni (ko))))))
            (loc (any (id (1)))))))

0



***keisanshiki***

(5 + 3 = 8)

***kotae***

(8 ko)
```

［図２］　例［１］の問題解決プロセスの一部　（日本語入出力部分は省略）