1A4-2

原始的な推論を使い数学の問題の解き方を獲得するプログラム

A Program to Acquire How to Solve Mathematical Problems by Using Elementary Reasoning

三宅 智由紀[*1] 田中 雅次[*1] 岩間 憲三[*2] 富士原 真知子[*2]
Tomoyuki Miyake Masaji Tanaka Kenzo Iwama Machiko Fujiwara

[*1] 岡山理科大学
Okayama University of Science

[*2] エンジコム株式会社
Engicom Corporation

In mathematics, a sequence of sentences describes how to solve a problem. For instance, an example of problems and answers is as follows. “When you buy five cakes of 100 yen, how much money is needed? Because 5 times 100 is 500, 500 yen is needed.” When a few examples whose values are different from each other are given to our program, the program changes the values into internal variables. And the examples can be generalized and learned to generate a procedure to solve the problem by using elementary reasoning. When a new example of the problem is given to the program, correct answer can be generated automatically. This paper shows our program learns how to solve a linear equation, and then learns how to solve a system of linear equations by applying procedures previously learned to solve the linear equation. Also, an application of the program to learn geometry is studied in this paper.

1. はじめに

古くから言語哲学者達は,「自然言語の意味は，その言葉が指し示すものだと言える.」などと議論してきた. [Frege 1892] そしてこの考え方は,「文の意味は，その文の真偽を計算する手続きである.」という方向に発達した. [Gierasimczuk 2007]

また，心理学や言語学，および認知科学では，言葉の意味を帰納的に獲得する研究・開発が多くなされて来た. [Margolis 2008] 特に, 機械学習の分野においては, 機械に対象物の概念を獲得させる, 多様なフレームワークが, これまでに多く開発された. 例えば, 与えられたデータベースを繰り返し分割することで, 決定木(decision tree)を自動生成させるものや [Breiman 1984] , 行動クローニング(behavioral cloning)において, 例えば航空機を操縦している熟練者の多様なデータを学習させることで, その自動操縦を試みるもの[Bain 1995], さらに, 熟練者の観察を通して手続的知識(procedural knowledge)を自動獲得させるものなどである. [Lent 2001]

一方，[Muggleton 1994]では，帰納論理プログラミング (Inductive Logic Programming: ILP) のシステムを開発し, [小林 1999] においては, ILP が幼児の言葉の帰納的学習に適用された. また, 多くの入力データと出力データの対を, 例として与えると, 入力から出力を生成するプログラムを合成するようなプログラムの開発も行われている. [Kitzelmann 2006] これらのプログラムは，一つの入力文に，論理を見出して一般化している. Programming By Example と呼ばれる分野では, 複数の文が繰り返す場合, あるいは分岐をする場合を見出している. [Cypher 1993]

本論文は, [Iwama 2006, Fujiwara 2009] で述べられた学習手法を基本とする．この手法では，小中学程度の数学における問題の解き方の例を与えるが，解き方の手順における数値の間の関係，繰り返し，および分岐を見出し，解き方を一般化するといった原始的な推論を使う．その結果，数値を変えた新しい例題を解くような手続きを自動生成する．

連絡先：〒700-0005 岡山市北区理大町 1-1 岡山理科大学工学部機械システム工学科[*1]，〒114-0001 東京都北区東十条 2-16-6 エンジコム株式会社[*2]

本論文で紹介する学習手法では, 次の 2 点をテーマとする.
・学習された解き方を他の例題の学習に応用させる.
・式や語句といった, 数値以外の文の要素が, 与える問題によって少し異なる場合でも, 同じ問題と理解できる.

具体的には, 一次方程式の解き方を学習させた後, それを連立方程式の解き方に応用させ, また, この際, 入力する問題文が互いに少し異なっていても, 正確に学習できるようにする. 学習した結果, 作成される手順は, すでに学習した手順を使う階層的な形をとる.

2. 原始的な推論の概説

これまでの AI での推論では, 問題に対する解答を論理的に探索するのが基本的であった. この場合, 一般的に探索領域が大きく複雑になると, 推論も非常に時間を要し, 複雑になる. 本論文で述べる手法の基礎となる [Iwama 2006, Fujiwara 2009] では, このような論理的な探索を行わずに, 与えられた問題文や解答文の構造を分析して,「変数」や「繰り返し」,「判断」などを文中から直接抽出した後, それらが他の例でも成立するだろうという原始的な推論を行う.

ここで, 簡単な例題を挙げて, その推論手法を説明する. 例えば, 次のような例題 1 の問題と解答の文があるとする.

例題 1
問題：100 円の菓子を 5 個買うのに, いくら必要か?
解答: 100 かける 5 は, 500 なので, 500 円必要である.

次に, 例題 1 の数値だけ変えた, 例題 2 を作る.

例題 2
問題：200 円の菓子を 3 個買うのに, いくら必要か?
解答: 200 かける 3 は, 600 なので, 600 円必要である.

ここで, 例題 1 を, 次のように, スペースで区切られた各語句において, 番号付けする(ナンバリングする: numbering).

ナンバリングした例題 1
問題：100(1) 円の菓子を(2) 5(3)個買うのに, いくら必要か?(4)

解答: 100(1) かける(5) 5(3) は, (6) 500(7) なので, (8) 500(7) 円必要である. (9)

なお, 数値の両端には, 前提条件として, 予めスペースを入れてある. また, 同じ語句や数値には, 同じ番号が入る. 続けて, 例題 2 を次のようにナンバリングする.

ナンバリングした例題 2
問題:200(1) 円の菓子を(2) 3(3) 個買うのに, いくら必要か?(4)
解答: 200(1) かける(5) 3(3) は, (6) 600(7) なので, (8) 600(7) 円必要である. (9)

二つの例題の比較において, 変化する数値を内部変数としてトークン(token)と名付け, トークンにもナンバリングを行う(t1, t2, …)と, これらは, 次のように一般化できる.

例題 1, 2 の一般化
問題:t1(1) 円の菓子を(2) t2(3) 個買うのに, いくら必要か?(4)
解答: t1(1) かける(5) t2(3) は,(6) t3(7) なので,(8) t3(7) 円必要である.(9)

結果として, 一般化された文は, 数値の変化に関係なく成り立つ論理式とみなせる. その真偽は, 次のような, 新しい問題を入力し, 解答を推論させることで明らかにできる. ここで前提条件として,「2 かける 6 は, 12」といった掛け算を含む四則演算は, 既に学習されているものとする.

新しい問題:150 円の菓子を 7 個買うのに, いくら必要か?

この問題を, 先の一般化された例題に当てはめると, t1 は, 150 に, t2 は, 7 になる. これらを, 解答に当てはめると, t3 は, 1050 となり, 次のような正しい解答が得られる.

解答: 150 かける 7 は, 1050 なので, 1050 円必要である.

少し複雑な例題として, 素数の判断を例題とすると, 例えば,「127 は素数か?」という質問に対する解答は, 最初に 127 を 2 で割る. 割れない. 2 たす 1 は, 3. 127 を 3 で割る. 割れない. といった文の繰り返しとなり, 最後に, 127 を 127 で割る. としたとき,「割れる」として,「素数」と判断できる. [Fujiwara 2009] では, この「繰り返し」と「判断」を自動的に見出した後, 一般化した手順を自動生成できる.

3. 一次方程式の学習

前述の手法を基本として, 本手法では, 最初に一次方程式を学習させ, その応用として, 連立方程式の学習を行う. ここで, 学習を行う際の前提条件を, 次の二つとする.
・文中の数値の両端には, 予めスペースを入れる.
・四則演算は学習されている. また, 各数値は, 小数点以下 5 桁を四捨五入する.

一次方程式を解く際には,「移項」操作を予め学習すると効果的である. 次に, その例題と一般化を示す. 以下, 解答の文は, 問題が難しくなると長くなるので, 各文を,「」で区切って記す. 実際の処理においても, 各文は, 独立して扱う.

「移項」の例題 1
問題: 10 x + 3 = 7 より, 3 を移項する.
解答:「3 に -1 をかける.」「-3」「10 x = 7 -3」「7 と -3 をたす.」「4」「10 x = 4 」

「移項」の例題 2
問題: 3 x + 6 = -2 より, 6 を移項する.
解答:「6 に -1 をかける.」「-6」「3 x = -2 -6」「-2 と -6 をたす.」「-8」「3 x = -8 」

「移項」の一般化
問題: t1 x + t2 = t3 より, t2 を移項する.
解答:「t2 に -1 をかける.」「t4」「t1 x = t3 + t4」「t3 と t4 をたす.」「t5」「t1 x = t5 」

一般化された「移項」は, 学習データとして, コンピュータに保存する. これを用いることで, 一次方程式の学習は, 次のように実現できる.

「一次方程式」の例題 1
問題: 一次方程式 4 x + 3 = 5 を解く.
解答:「4 x + 3 = 5 より, 3 を移項する.」「4 x = 2」「2 を 4 で割る.」「0.5」「x = 0.5」

「一次方程式」の例題 2
問題: 一次方程式 -2 x + 9 = -12 を解く.
解答:「-2 x + 9 = -12 より, 9 を移項する.」「-2 x = -21」「-21 を -2 で割る.」「10.5」「x = 10.5」

「一次方程式」の一般化
問題: 一次方程式 t1 x + t2 = t3 を解く.
解答:「t1 x + t2 = t3 より, t2 を移項する.」「t1 x = t4」「t4 を t1 で割る.」「t5」「x = t5」

この一般化において,「t1 x + t2 = t3 より, t2 を移項する.」という文は, 以前に学習した,「移項」の問題の一般化と一致するので,「一次方程式」の例題においては, 移項後の答を, 直接解答に返せる. つまり, 最初に学習した「移項」は,「一次方程式」の学習で応用できる. なお, この「移項」の「一次方程式」における認識は, 解答における, 一文の長さや使われている語句の同一性から判断できる.

ここで, 次の新しい一次方程式を問題として与えると, 自動的に解が得られる.

新しい一次方程式の問題: 一次方程式 -5 x + 6 = 16 を解く.
自動生成された解答:「-5 x + 6 = 16 より, 6 を移項する.」「-5 x = 10」「10 を -5 で割る.」「-2」「x = -2」

4. 連立方程式の学習

一次方程式の応用として, 連立方程式を学習させる. このとき, 予め,「y の式」を, 次のように学習させておく.

「y の式」の例題 1
問題: 6 x + 2 y = 10 を y の式にする.
解答:「6 x + 2 y = 10 より, 6 x を移項する.」「6 に -1 をかける.」「-6」「2 y = -6 x + 10」「式全体を 2 で割る.」「-6 を 2 で割る.」「-3」「10 を 2 で割る.」「5」「y = -3 x + 5」

「y の式」の例題 2
問題: 21 x + 3 y = 12 を y の式にする.

解答:「21 x + 3 y = 12 より, 21 x を移項する.」「21 に -1 をかける.」「-21」「3 y = -21 x + 10」「式全体を 3 で割る.」「-21 を 3 で割る.」「-7」「12 を 3 で割る.」「4」「y = -7 x + 4」

「y の式」の一般化
問題: t1 x + t2 y = t3 を y の式にする.
解答:「t1 x + t2 y = t3 より, t1 x を移項する.」「t1 に -1 をかける.」「t4」「t2 y = t4 x + t3」「式全体を t2 で割る.」「t4 を t2 で割る.」「t5」「t3 を t2 で割る.」「t6」「y = t5 x + t6」

なお, 前章で述べた「移項」は, ここでは適用できない. これは, 移項の対象となる語句が, 1 個から 2 個(例: 3 x)に増えたためである.「y の式」を用いて, 連立方程式は, 次の二つの例題より学習できる. これらの問題の数値は, 大きく異なるが, これは, 一般化で数値を確実にトークン化するためであり, 実際には, 例題の数を増やせば増やすほど, 数値の与え方は自由になる.

「連立方程式」の例題 1
問題: 連立方程式 6 x + 2 y = 7 , 20 x + 3 y = 23 を解く.
解答:「6 x + 2 y = 7 を y の式にする.」「y = -3 x + 3.5」「20 x + 3 y = 23 を y の式にする.」「y = -6.6667 x + 7.6667」「-3 x + 3.5 = -6.6667 x + 7.6667」「-3 x + 3.5 = -6.6667 x + 7.6667 より, -6.6667 x を移項する.」「-6.6667 に-1 をかける.」「6.6667」「6.6667 x -3 x + 3.5 = 7.6667」「6.6667 と -3 をたす.」「3.6667」「3.6667 x + 3.5 = 7.6667」「3.6667 x + 3.5 = 7.6667 より, 3.5 を移項する.」「3.6667 x = 4.1667」「4.1667 を 3.6667 で割る.」「1.1364」「x = 1.1364」「-3 に 1.1364 をかける」「-3.4092」「-3.4092 と 3.5 をたす.」「0.0908」「y = 0.0908」「x = 1.1364 , y = 0.0908」

「連立方程式」の例題 2
問題: 連立方程式 0.2 x + 2.5 y = 0.7 , 1.5 x + 0.4 y = 0.3 を解く.
解答:「0.2 x + 2.5 y = 0.7 を y の式にする.」「y = -0.08 x + 0.28」「1.5 x + 0.4 y = 0.3 を y の式にする.」「y = -3.75 x + 0.75」「-0.08 x + 0.28 = -3.75 x + 0.75」「-0.08 x + 0.28 = -3.75 x + 0.75 より, -3.75 x を移項する.」「-3.75 に-1 をかける.」「3.75」「3.75 x - 0.08 x + 0.28 = 0.75」「3.75 と -0.08 をたす.」「3.67」「3.67 x + 0.28 = 0.75」「3.67 x + 0.28 = 0.75 より, 0.28 を移項する.」「3.67 x = 0.47」「0.47 を 3.67 で割る.」「0.1281」「x = 0.1281」「-0.08 に 0.1281 をかける」「-0.0102」「-0.0102 と 0.28 をたす.」「0.2698」「y = 0.2698」「x = 0.1281 , y = 0.2698」

「連立方程式」の一般化
問題: 連立方程式 t1 x + t2 y = t3 , t4 x + t5 y = t6 を解く.
解答:「t1 x + t2 y = t3 を y の式にする.」「y = t7 x + t8」「t4 x + t5 y = t6 を y の式にする.」「y = t9 x + t10」「t7 x + t8 = t9 x + t10」「t7 x + t8 = t9 x + t10 より, t9 x を移項する.」「t9 に-1 をかける.」「t11」「t11 x +t7 x + t8 = t10」「t11 と t7 をたす.」「t12」「t12 x + t8 = t10」「t12 x + t8 = t10 より, t8 を移項する.」「t11 x = t13」「t13 を t11 で割る.」「t14」「x = t14」「t7 に t14 をかける」「t15」「t15 と t8 をたす.」「t16」「y = t16」「x = t14 , y = t16」

5. 類似した問題文の同定

これまで述べた, 一次方程式や連立方程式の学習では, 基本的に式の形が変わると, 新たな学習が必要になるので, いかなる形の一次方程式にも対応させようとすると, 膨大な量の学習が必要となる. この学習量を減少させる初期的な方法として, 本研究では, 以下に述べるように, 式の自動認識と, 同義語の学習によって, 多少異なる問題文でも, 同じ問題文であるとコンピュータに理解させるようにする.

5.1 問題文での式の認識

次の二つの問題文は, 式の長さが違うが, 意味としては,「一次方程式を解く」ことである.

類似問題 1: 一次方程式 2 x – 5 = 3 + 3 x を解く.
類似問題 2: 一次方程式 -10 x = 36 + 27 x を解く.

これら二つの問題を同じ形と認識させるには, 予め問題文において, 数式を認識する必要がある. 数式は, 概して, 数字と記号から成るが, 与えられた文字列が正しい数式か否かを判断するのは難しいので, 本手法では, 2 個以上の数字と記号のみから成る文字列を,「式」と認識させ, 独自のナンバリングを行う(e1, e2, ..). これで, 類似問題 1, 2 は, 次の形にでき, 一般化される.

類似問題 1': 一次方程式(1) 2 x – 5 = 3 + 3 x(e1) を解く. (2)
類似問題 2': 一次方程式(1) -10 x = 36 + 27 x(e1) を解く. (2)
類似問題 1', 2'の一般化: 一次方程式(1) t1(e1) を解く. (2)

新しく与えられた問題が,「一次方程式の問題」と認識できれば, 解き方は, 式の形に関係するので, その段階で, 問題全体を再度, 一文字毎にナンバリングする. その結果, 第 3 章の手法が適用可能となる.

5.2 問題文での同義語の学習

次の二つの問題文は, 語句が異なるが, 同じ意味である.

類似問題 5: 一次方程式 4 x + 3 = 5 を解く.
類似問題 6: 方程式 4 x + 3 = 5 x を解きなさい.

これらの問題文の解答文は同一となるので, 同一の解答文で, 問題文が異なる場合は, 同義語を学習させる. 上記の問題では,「一次方程式」と「方程式」, および,「を解く.」と「を解きなさい.」が各々同義語として学習できる. 与えられたいくつかの問題文の各語句について, 学習された同義語を調べることで, 類似問題 5, 6 のような場合は, 次のようにも, 一般化することが可能である.「類似問題 5, 6 の一般化: 式 4 x + 3 = 5 の解」

6. 考察

本手法の今後の展開としては, 二次方程式･関数などの学習への応用が第一に考えられるが, また, より人間らしい学習を目指して,「一次方程式 a x + b = c が解けるか?」という問題に「解ける.」と解答できるような, 多様な質問に答えられる学習システムの開発も, 大きな課題である.

また, 本論文では, 学習の対象が基本的に解析学にあったが, 幾何学についても, 現在, 学習システムの開発を試みている. 例えば, 図 1 のように x-y 座標系において, 二つの直線(A, B)が描かれていて, これらの両端を, 各々, (a, b), (c, d)とすると, 直線 A, B が「垂直」な関係であることは, 次のように学習できる. ここで, 前提条件として, 次の三つを新たに加える.

・コンピュータ上で, 与えられた 2 点間に直線が描ける.
・作成した各頂点や各直線に名前が付加される.
・2 個の文字同士を比較できる.

なお, 先の二つの条件は, 例えば CAD 上において, DXF 形式のデータを用いることで, 容易にアプリケーションを開発できる.

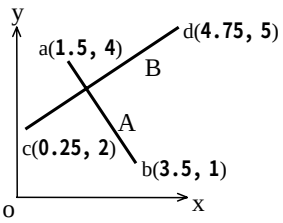

図 1: 垂直の例題 1

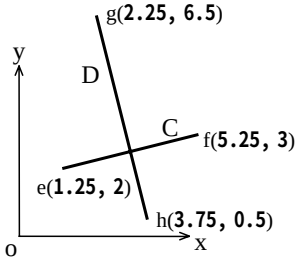

図 2: 垂直の例題 2

「垂直」の例題 1
問題: 直線 A と直線 B の関係は, 垂直か?
解答:「A の両端は, (1.5 , 4) , (3.5 , 1)」「1.5 から 3.5 をひく.」「-2」「4 から 1 をひく.」「3」「A の方向ベクトルは (-2 , 3)」「B の両端は, (0.25 , 2) , (4.75 , 5)」「0.25 から 4.75 をひく.」「-4.5」「2 から 5 をひく.」「-3」「B の方向ベクトルは (-4.5 , -3)」「内積を求める.」「-2 に -4.5 をかける.」「9」「3 に -3 をかける.」「-9」「9 に-9 をたす.」「0」「0 と 0 は同じか?」「同じ.」「垂直である.」

同様にして, 図 2 での, 直線 C(e, f), D(g, h)の関係を例題 2 として学習させると,「垂直」の一般化ができる.

「垂直」の例題 2
問題: 直線 C と直線 D の関係は, 垂直か?
解答:「C の両端は, (1.25 , 2) , (5.25 , 3)」「1.25 から 5.25 をひく.」「-4」「2 から 3 をひく.」「-1」「C の方向ベクトルは (-4 , -1)」「D の両端は, (2.25 , 6.5) , (3.75 , 0.5)」「2.25 から 3.75 をひく.」「-1.5」「6.5 から 0.5 をひく.」「6」「D の方向ベクトルは (-1.5 , 6)」「内積を求める.」「-4 に -1.5 をかける.」「6」「-1 に 6 をかける.」「-6」「6 に-6 をたす.」「0」「0 と 0 は同じか?」「同じ.」「垂直である.」

「垂直」の一般化
問題: 直線 t1 と直線 t2 の関係は, 垂直か?
解答:「t1 の両端は, (t3 , t4) , (t5 , t6)」「t3 から t5 をひく.」「t7」「t4 から t6 をひく.」「t8」「t1 の方向ベクトルは (t7 , t8)」「t2 の両端は, (t9 , t10) , (t11 , t12)」「t9 から t11 をひく.」「t13」「t10 から t12 をひく.」「t14」「t2 の方向ベクトルは (t13 , t14)」「内積を求める.」「t7 に t13 をかける.」「t15」「t8 に t14 をかける.」「t16」「t15 に t16 をたす.」「0」「0 と 0 は同じか?」「同じ.」「垂直である.」

垂直でない例題を学習させると, 上述の一般化における, 最後の部分が次のように変わる.「t17」「t17 と 0 は同じか?」「違う.」「垂直でない.」また, 先に「方向ベクトル」や「内積」を学習させると, より効果的である. 同様にして, 二直線の平行や正三角形などの多様な三角形が学習できることを, 本研究では実験用システムによって検証している.

これらの学習システムの実用性としては, 小中学生の教材の開発や, 熟練者の持つ機械設計･製造におけるノウハウの学習などが挙げられる. 特に, 小中学生の算数･数学教育では,「中一ギャップ」と呼ばれる問題がある. これは, 中学一年の始めに数学に活発に取り組んでいた何割かの生徒が, 中学二年になると, うつむき下限になることである. この時期は, 中学一年で変数を導入し, 方程式の解き方を教える期間である.

筆者らは, このギャップを取り除く可能性を追求している. すなわち, 小学生は, 文章題「一つ 100 円のりんご･･･」を解くとき, 頭の中では, 変数に相当する何かを使うはずである. これと中学で教える変数とを, 何らかの形で結びつけることを促すような教材を作ることである.

7. おわりに

本論文では, 小中学程度の数学において, 例題の問題文と解答を求める手順に現れる, 数値の関係の論理性を利用して, 各数値を内部変数化することで一般化し, かつ, すでに学習した手順を使う階層的な手順を生成するシステムを開発した. 現段階では, 一次方程式や連立式が学習でき, また, これらの問題文が多少異なっても対応できるように工夫した. さらに, 幾何学への拡張性についても, 実験用システムにより検証した.

参考文献

[Frege 1892] G. Frege : Über Sinn und Bedeutung, Zeitschrift fur Philosophie und Philosophishe Kritik, Vol. 100, pp. 25-50, 1892.

[Gierasimczuk 2007] N. Gierasimczuk : The problem of learning the semantics of quantifiers, LNAI, Vol. 4363/2007, pp. 117-126. Springer, 2007.

[Margolis 2008] E. Margolis, S. Laurence : How to learn natural numbers: inductive inference and the acquisition of number concepts, Cognition, Vol. 106, Issue 2, pp. 924-939, 2008.

[Breiman 1984] L. Breiman, J. Friedman, R. A. Olshen, C. J. Stone : Classification and Regression Trees. Belmont Wadsworth Int. Group, New York, 1984.

[Bain 1995] M. Bain, C. Sammut : A Framework for Behavioral Cloning. Machine Intelligence Agents. Oxford University Press. 1995.

[Lent 2001] Michael van Lent, J. Laird : Learning procedural knowledge through observation, First Int'l conf. on Knowledge Capture, pp. 179-186, 2001

[Muggleton 1994] S. Muggleton, L. De Raedt : Inductive logic programming: Theory and methods, J. Logic programming, Vol. 19, No. 20, pp. 629-679, 1994.

[小林 1999] 小林郁夫, 古川康一, 尾崎知伸 : 機械学習システムによる幼児の言語獲得のモデル化, 第 13 回人工知能学会全国大会, 1999.

[Kitzelmann 2006] E. Kitzelmann, U. Schmid : Inductive synthesis of functional programs: an explanation based generalization approach, J. Machine learning research, Vol. 7, pp. 429-454, 2006.

[Cypher 1993] A. Cypher : Watch what I do: Programming by demonstration, MIT Press, 1993.

[Iwama 2006] K. Iwama : A robotic program that acquires concepts and begins introspection, NueroQuantology, Vol. 4, No. 4, pp. 321-328, 2006.

[Fujiwara 2009] M. Fujiwara, K. Iwama : A program that acquires how to solve problems in mathematics, WSEAS Trans. on Computers, Vol. 8, Issue 8, pp.1337-1347, 2009.