人 工 知 能 87−11
（1993. 3. 4）

命題の抽象化に基づく発見的類推

石川 孝[1,2]、 寺野 隆雄[1]

1 筑波大学経営システム科学専攻
2 ぺんてる（株）電子研究所

本論文では、事例を利用した発見的類推の一形式として命題の抽象化に基づく類推（Abstraction-Based Analogy）を提案し、その述語論理による定式化とアルゴリズムのHorn論理による実現について述べる。抽象化に基づく類推ABAは、推論過程で事例を利用するための計算論的手法であって、(1) 命題の抽象化に基づいて類似性のある既知の命題を選択し、(2) 類比な命題を一般化して一般化された類比を生成し、(3) 一般化された類比を特殊化して目的とする命題（仮説）を得る、という abstract - generalize - specialize の３ステップからなる。

Heuristic Discovery by Abstraction-Based Analogy

Takashi ISHIKAWA[1,2] Takao TERANO[1]

1 Graduate School of Systems Management, The University of Tsukuba
3-29-1 Otsuka, Bunkyo-ku, Tokyo, Japan
2 Electronics Laboratory, Pentel Co., Ltd.

We propose, in this paper, Abstraction-Based Analogy which is a heuristic analogical reasoning to use cases as search control knowledge. We formalize ABA using first order predicate logic and implement its algorithm with a Horn logic programing language. ABA has abstract-generalize-specialize steps, that 1) selects analogies using problem abstraction, 2) generalize solution associated with the analogies, and 3) specialize the generalized solutions to get an adapted solution.

1．はじめに

類推的推論(analogical reasoning)（以下、類推と呼ぶ）は、問題解決の過程で事例を利用するための推論方式である。知識ベースシステムにおいて類推を実現するには、以下の課題を解決することが必要とされる[原口, 1986]。

(1) 類比の明確な定義
(2) 対象間の類比を検出する方法
(3) 類比に基づいて類推する推論システムの開発

本研究の目的は、これらの課題に対して知識ベースシステムで実行可能な類推の実現手段を開発することである。この目的に対して本研究では、知識変換の基本プロセスである抽象化、一般化、特殊化を基礎として類推を定式化し、一階述語論理のサブセットであるHorn論理によってそのアルゴリズムを実現した。本論文では特に、既知の事実から類推によって未知の事実を発見する発見的類推の定式化について述べる。

知識変換としての類推は、大きく分けて置換に基づくものと、一般化に基づくものがあるが[岩山 et al, 1991]、本研究で提案する定式化は類推の基礎を一般化に置き、一般化の記号操作を置換によって行うという意味で両者の要素を持つ。類推の定式化における基本的な課題は、効率的な類比の発見と事例変換の正当性を保証するアルゴリズムの確立であるが[有馬, 1992]、本研究では抽象化によって前者を、また一般化と特殊化によって後者の課題を解決した。本論文では、第2章で発見的類推の概念構造について分析し、第3章で抽象化に基づく類比、第4章で発見的類推の定式化について述べる。つぎにこの定式化に基づいて第5章で発見的類推のHorn論理による実現と、第6章で例題に対する記述と実行について述べ、最後に第7章で考察、第8章で結論を述べる。

2．発見的類推の概念構造

2．1　類推の定義

類推は、いくつかの与えられた対象間に類似性（以後、これを類比という）を検出し、その類比を用いて一方の対象で成立する事実や知識をもう一方の対象に変換することにより、問題解決の手掛りを得たり（問題解決型）、未知の事実などを予測推定する（予測推定型）推論方式である[有川, 1986]。問題解決型は、一応問題が解決できるときに、効率化のために適用され、予測推定型は、二つの対象間に類似性が認められるとき、一方の対象で成立する知識を他方の対象の知識として変換し、仮説を予測推定する。類推の定義には、類推の構成要素間の明白な関係付けを前提とするPolyaの定義[Polya, 1959]や、「類似性は因果関係を保持する」という原則に基づいて「既知の状況を新しい未知の状況へ関係を写像すること」というWinstonの定義[Winston, 1980]などがある。ここではこれらの定義を参考にして、類推を「類似な対象で成り立つ事実や関係を変換して別の対象での事実や関係を見いだすこと」と定義する。

2．2　目的の導入

類推における類似な対象すなわち類比の発見は、類似性の観点に依存している。このことは類推が目的の導入を必要とするとして指摘されており、目的指向類比(purpose-directed analogy)が提案されている[Kedar-Cabelli, 1985]。類推の人工知能モデルは、類似した状況の間の関係の基礎にある何らかの因果ネットワークを写像することに基づく。しかし、一つの類比の目的に適切な因果関係は、他の目的には不適切であるかも知れない。目的指向類比では類比の目的の陽な記述を用いて適切な因果ネットワークを自動的に生成する。類推の目的(は)別の見方をすれば類推によって得られる仮説の有用性であり、それは類推の機能が仮説の生成であるとの認識である[Greiner, 1988]。

2．4　発見的類推の論理構造

発見的類推の構成要素は、類比と変換である。類比は、目的とする命題（事実や知識の記述）と特定の類似性をもつある対象領域での命題である。この類似性を定義する関係概念は、もう一つの構成要素である変換と密接に関係する。変換は、類

比な命題を問題とする対象領域に対応させて変換することである。変換結果が既知でないとき、発見となる。変換には、置換に基づくもの、一般化に基づくものなどがあるが、命題を形式的に記述する言語に対する形式的操作として定義される。変換を定義する言語操作は、置換に基づくものでは概念の対応付け、また一般化に基づくものでは概念の一般化である。これらの言語操作は、類似性の定義における概念の対応付け（置換）、一般化階層に基づく抽象化（一般化）であり、類似性が変換可能な類比を定義していることになる。

目的とする類推を出発点とすると、類比は変換の逆操作である。つまり、類比によって命題は基本領域に写像され、変換によって目標領域に写像される。類比は目標領域に戻る変換の存在に対応し、類比と変換は互いに逆写像の関係にある。

3．抽象化に基づく類似性

3．1　類比の発見手段

本論文では、類推の第一段階としての類比の発見を上述の類比の定義に従って、「対象間での類似性の発見」と考える。類似性は対象の性質がある抽象化によって同一であると見做せることであり、対象の抽象化を前提としている。抽象化は「集合そのものを集合の要素とすること」であり、同一視の前提である。抽象化による類似性の検出以外の類比の発見手段としては、概念構造を構成する関係の一般化による類似性がある[Winston, 1980]。この類比も別の見方をすると関係の抽象化であり、抽象化による類比に含まれるものと考えられる。

3．2　抽象化に基づく類似性の定義

対象間の類似性を抽象化に基づいて以下のように定義する。二つの対象が抽象化において類似するとは、その抽象化によって二つの対象が同一視されることである。例えば、算術演算の「加算＋」と「減算－」は、これらの抽象化である「算術演算」によって同一視される。この同一視はこれらの二つの対象の類似性を定義する。その類似性は、算術演算の性質（二つの数に別の数を対応させること）である。

3．3　抽象化の役割

類似性を定義するための抽象化の役割には、命題抽象化による単純化、有用性な類比の発見、命題の索引付け、述語の関係付けがある。

抽象化命題抽象化は、抽象化された定理の証明を用いて問題にしている定理の証明を見つけるための抽象化であり、述語の項を落とすことに対応する[Plaisted, 1981]。命題抽象化によって述語論理は命題論理に変換されるので、論理演算は単純化される。この単純化が抽象化の目的である。

抽象化は、類比の有用性と関係する[Greiner, 1988]。有用な類比は問題を解決するのに必要な情報を与えるだけでなく、既知の問題に対して解法を符号化する抽象的な関係であるヒューリスティックスの集合を導く。抽象化を用いることによって、共通の抽象化を見つけるだけでほとんどの有用な類比を発見できる。ここでの抽象化の非公式な記述は、関連する事実の部分集合であり、類比を枝狩りするものである。

類推のための抽象化は、以下のように定式化される[脇園 et al, 1990]。ここでの抽象化の役割は基本領域の索引付けのためであり、確定節で表現された知識の抽象化、命題抽象化すなわち述語記号だけの抽出を考える。

$p(t_1,....,t_n)$、A、B_i $(1 \leqq i \leqq m)$をアトムとするとき、アトム、節、プログラムの抽象化の定義：

$\phi(p(t_1,....,t_n)) = p^n$

$\phi(A<\text{-}B_1,....,B_m) = \phi(A) <\text{-} \phi(B_1),...., \phi(B_m)$

$\phi(P) = \{\phi(C): C \in P\}$

意味論的抽象化は、述語の抽象化において述語の間の意味的関係を用いる[Giordana et al, 1991]。述語の意味的関係とは、述語の記述する関係に存在する抽象的な関係であって、意味論的抽象化はその関係を特定の項に制限することである。意味論的抽象化によって、述語によって表現される関係はより単純化される。

3．4　抽象化の定式化

抽象化とは情報を落とすことである。自動推論における抽象の利用に関する議論では、抽象化は「扱いを単純化するための形式的体系の間の写像」として定義される[Giunchiglia & Walsh, 1990]。情報処理一般では抽象化は、「集合自体を集合の要素と考えること」（「情報処理用語辞典」）である。

例1．「AはBである」：　集合Aは集合Bの要素である。（包含関係としての抽象化）

注）このとき、集合Aは「名前」で指示される。ここでも抽象化の役割は、「情報を落とすこと」である。

例2．「加算は数の演算である」：　集合「加算」は集合「数の演算」の要素である。

注）抽象化によって、加算の詳細な規則に関する情報が落とされる。

例3．例2において、「加算」を集合と考えるとき。「加算“2＋2＝4”は数の演算である」

注）命題“2+2=4”は、一般化された命題「加算」の要素（個体）である。

命題の抽象化には項の一般化［変数化］が必要である。項の一般化はある特定の個体について成り立つ述語が、その個体が属する集合で成り立つことに一般化することであり、事実から法則を作り出す操作である。いま、個体Aに対して述語p(A)が成り立つとき、この変数化とは、p(A) <- D(A) なる節で表される。ここで、D(A) は個体Aが集合Dに属することを表す。

論理による抽象化の表現では、抽象化操作を命題を項とする述語で表現できる。

例4．（数の演算　“2＋2＝4”）：　例3の論理による表現。

抽象化に対して一般化は、「個体を変数にすること」である。

例5．命題“2＋2＝4”の一般化

→　命題“x＋x＝2x”

例6．加算“2＋2＝4”の一般化

→　加算“数＋数＝数”

注）抽象化命題「数の演算」の要素としての加算では、個々の命題における個体が「数」として一般化（変数化）されている。

命題抽象化は述語の項を落とす操作として定式化でき、命題を要素とする集合に対して「抽象化」を適用する。さらに、複数の命題に対する抽象化操作として連言汎化が定義できる。連言汎化とは、連言を一つの述語に置き換える操作である。この操作は複数の述語を項の並びを保存して一つの述語に変換し、ついで得られた述語の項を落とす操作の複合として考えられる。連言汎化によって、連言を述語に抽象化できるようになる。この連言汎化の導入の効果は、例題としてのピタゴラスの定理の発見（後述）によって例証される。

3．5　抽象化に基づく類推

命題の発見を目的とした類推（発見的類推）を定式化するために、抽象化写像を考える。抽象化写像absは、集合Sを抽象化集合Aの要素sに対応付けることである。

abs：　S → s、

S＝｛a、b、c、．．．｝、

s∈A

もとの集合も要素としての集合もそれぞれS、sという記号（名前）で指示されるが、指示される実体は異なる。すなわち、もとの集合の名前Sは、その要素a、b、c、．．．との間に関係a∈S、．．．をもつのに対して、集合要素としてのsは、写像absによってSと対応付けられているだけである。

抽象化写像absの上記の意味を一階述語論理で表現すると、以下のようになる。

（命題　s）←（命題　（abs　S））

この式は、命題（abs　S）を命題sで置き換えることを表わす。

注）抽象化を論理で表現するには、「命題の命題」を考える必要がある。

抽象化に基づく類推を定式化する目標は、「命題の変換」として類推を定式化することであり、つぎの章において命題を節で表現し、節を上述の変換操作によって発見的類推を定式化する。

4．発見的類推の定式化

4．1　一般化／特殊化変換

類推を知識変換とみなす立場で、知識を節で表現するとき、その知識変換を節に対する変換操作として定義する。まず、知識変換としての帰納と演繹について考える。帰納は事実の集まりから一般的な知識を得ること、すなわち一般化であり、演繹は一般的知識から具体的な事実を推論こと、すなわち具体化（特殊化）である。命題を推論によって変換するという意味でどちらも知識変換である。特殊化は一般化の逆変換と考える。

一般化は、知識の適用範囲を広げることであり、集合の拡大操作と考えられる。集合の拡大には、変数汎化、述語汎化、連言汎化がある。

(1) 変数汎化

ある集合の要素について成り立つ命題を、その個体の属する集合について成り立つことに拡大することで、形式的には命題における個体記号を変数記号に置き換える操作である。

(2) 述語汎化

命題の構成要素である述語において、集合の所属関係を表す述語の集合の範囲を拡大することであって、述語の間の含意関係によって規定される。形式的には、P(X) <- Q(X) であるとき、述語Qを述語Pで置き換える操作である。

(3) 連言汎化

述語の連言で表される命題に対して、連言として含意関係があるときに、より一般的な連言に置き換えることである。連言汎化では、P(X,Y),Q(Y) <- R(X,Y), S(Y) であるとき、右辺の連言を左辺の連言で置き換える。

述語論理における知識変換を、以上の述語における項の置き換え（個体の変更、変数化）、述語記号の置き換え、連言の置き換えと考えるとき、命題の抽象化／一般化／特殊化はこれらの記号操作によって表現される。

4．2　類推のアルゴリズム

以上のように定式化した発見的類推のアルゴリズムを以下のように構成する。

アルゴリズムABA：

(0) 類比の索引付け

類推に有用な事実や知識は抽象化によって索引付けされているものとする。

(1) 目的の記述

類推の目的を抽象的に与える（具体的に与えられるなら、類推は不要である）。

(2) 仮説基準の設定

類推によって得たい事実や知識の満たすべき基準（性質の限定など）を記述する。基準を厳しくするほど仮説の候補を絞れるが、仮説が得られない可能性が増す。

(3) 類比の探索

類推の目的と含意関係によって索引付けされた類比を探索する。索引付けされていない事実や知識は抽象化したうえで含意関係を調べる。

(4) 類比の一般化

発見した類比を変数汎化、述語汎化、連言汎化によって一般化する。一般化された事実や知識も類比となる。

(5) 類比の特殊化

一般化された類比を一般化の逆変換である特殊化して仮説を生成する。

(6) 仮説の評価

一般化／特殊化によって得られた類比が仮説基準を満たすかどうかを評価する。基準を満たさなければ、ステップ(5)、(4)をバックトラックする。

(7) 仮説の検証

得られた仮説が既知の知識から演繹可能かどうかを検証する。仮説の検証に必要な事実で既知でないものは仮定とし、仮定付きで仮説を得る。

5．Horn論理による実現

5．1　類推の機械化

命題の抽象化に基づく類推の機械化を以下のように行なった。ここでは処理系の都合で述語論理のサブセットであるHorn論理を定式化の基礎とした。

5．2　実験システムの構成

実験システムをPrologで作成した。Prolog処理系としてBABYLON [Christaller et al, 1992]を用い、ABA本体および対象知識はPrologで記述した。BABYLONの節記述は、(Head <- Body) の形式で、また、述語、事実は、(P x y) で表す。

5．3　基本モジュールの動作

ABAは、入力として目的と仮説基準を、出力として仮説と仮定の四つの引数をもつ。目的は、類比の抽象的な仕様を与え、仮説基準は類推によって求める類比の満たすべき基準を与える。また、仮説は得られた類比であり、仮定は仮説が成り立つために必要な事実である。以下にABAの動作をBABYLONプログラムによって示す。

ABAメイン：

```
;;; Abstraction-Based Analogy
((ABA _Purpose _Criteria _Conjecture _Hypothesis) <-
        ；目的   仮説基準   仮説        仮定
 (find-analogy _Purpose _Analogy)；類比の発見
 (write ANALOGY=) (write _Analogy)
 (copy _Analogy _A)
 (generalize _A _G)                    ；類比の一般化
 (not (= _A _G)) (not (= _Purpose _G))
 (specialize _G _Conjecture)；特殊化による仮説生成
 (not (= _A _Conjecture))
 (copy _Criteria _Temp-Criteria)
 _Temp-Criteria              ；仮説に対する基準の検査
 (copy _Conjecture _C)
 (or (and (= _C (_H . (<- . _B)))          ；仮説の検証
          (verify-conjecture (_H . _B) nil _Hypothesis))
     (verify-conjecture _C nil _Hypothesis))
)
```

類比の発見：

```
((find-analogy _P (_H . (<- . _A))) <-
              ；目的   節
 (= _P (_H . (<- . _)))          ；目的が節形式の場合
 (clause (_H . _A))
 (not (= _A nil))
 (not (abstraction (abstraction) _A)))   ；抽象化節を除く
((find-analogy _P _A) <-
 (not (= _P (_H . (<- . _))))   ；目的が述語の場合
 (clause (_P . _A)) (not (= _A nil)))
```

一般化／特殊化：

```
;; Generalization
((generalize (_P1 . _S1) (_P2 . _S2)) <-；連言の一般化
 (generalize-predicate _P1 _P2) (generalize _S1 _S2))
(generalize nil nil)          ；停止条件

((generalize-predicate (_P1 . _A1) (_P3 . _A1)) <-；述語の
 (not (= _P1 AND))                                  一般化
 (specialize-predicate (OBJECT _) (_P2 . _A2))
 (clause ((_P2 . _) . ((_P1 . _))))
 (generalize-predicate (_P2 . _A2) (_P3 . _A3)))
((generalize-predicate _P _P) <-
 (not (= (OBJECT . _) _P)))

;; Specialization
((specialize (_P1 . _S1) (_P2 . _S2)) <-；連言の特殊化
 (specialize-predicate _P1 _P2) (specialize _S1 _S2))
((specialize (_P1 . (_P2 . _S1)) (_P4 . (_P5 . _S2))) <-
 (specialize-predicate _P1 _P3)；AND述語の特殊化 1
 (specialize-predicate (AND _P3 _P2) (AND _P4 _P5))
 (specialize _S1 _S2))
((specialize (_P1 . (_P2 . _S1)) (_P3 . (_P4 . _S2))) <-
 (specialize-predicate (AND _P1 _P2) (AND _P3 _P4))
 (specialize _S1 _S2))        ；AND述語の特殊化 2
(specialize nil nil)          ；停止条件

((specialize-predicate (_P1 . _A) (_P3 . _A)) <- (atom _P1)
 (not (= _P1 AND))            ；述語の特殊化
 (clause ((_P1 . _A1) . ((_P2 . _A2))))
 (not (= _P1 _P2))
 (specialize-predicate (_P2 . _) (_P3 . _)))
((specialize-predicate (AND (_P1 . _A1) (_P2 . _A2))
                       (AND (_P3 . _A1) (_P4 . _A2))) <-
 (clause ((AND (_P1 . _) (_P2 . _)) . ((AND (_P3 . _) (_P4 .
                                                  _))))))
 (specialize-predicate _P _P)
```

仮説の検証：

```
((verify-conjecture (_P . _S) _H1 _H2) <- ；'<-' 記号の
 (= _P <-) (verify-conjecture _S _H1 _H2))  ；スキップ
((verify-conjecture (_P . _S) _H1 _H2) <- ；先頭述語の
 (not (= _P <-)) _P (verify-conjecture _S _H1 _H2))；検証
((verify-conjecture (_P . _S) _H1 _H2) <-；仮定の追加
 (not (= _P <-)) (clause (_P . _)) (not _P)；定義述語の失敗
 (verify-conjecture _S (_P . _H1) _H2))
 (verify-conjecture nil _H _H)
```

抽象化（命題の索引付け）：

```
((abstract _S1 _S4) <-
 (abstract-predicates _S1 _S2)；すべての述語の抽象化
 (abstract-conjunction _S2 _S3)        ；連言の抽象化
```

```
(abstract-predicates _S3 _S4))          ；もう一度すべての
                                          述語の抽象化

((abstract-predicates (_P1 . (_P2 . (_P3 . _S1))) _S2) <-
 (abstraction _P4 (AND _P1 _P2 _P3)) ；三つの述語の抽象
 (abstract-predicates (_P4 . _S1) _S2))  ；化
((abstract-predicates (_P1 . (_P2 . _S1)) _S2) <-
 (abstraction _P3 (AND _P1 _P2))；二つの述語の抽象化
 (abstract-predicates (_P3 . _S1) _S2))
((abstract-predicates (_P1 . _S1) _S2) <-
 (abstraction _P2 _P1)          ；一つの述語の抽象化
 (abstract-predicates (_P2 . _S1) _S2))
((abstract-predicates (_P . _S1) (_P . _S2)) <-
 (abstract-predicates _S1 _S2))；連言の抽象化
(abstract-predicates _S _S)    ；何もしない抽象化

((abstraction _R (AND _P1 _P2 _P3)) <-；三つの述語抽象
 (specialize-predicate (abstraction) _R)  ；化の探索
 (clause (_R _P1 _P2 _P3)))
((abstraction _R (AND _P1 _P2)) <-    ；二つの述語抽象化
 (specialize-predicate (abstraction) _R) ；の探索
 (clause (_R _P1 _P2)))
((abstraction _R _P) <-          ；一つの述語抽象化の探索
 (specialize-predicate (abstraction) _R)
 (clause (_R _P))
 (not (= _R (abstraction))))

((abstract-conjunction _S1 _S3) <-      ；連言抽象化
 (conjunctive-abstraction _S1 _S2)
 (abstract-conjunction _S2 _S3))
 (abstract-conjunction _S _S)

((conjunctive-abstraction (_S1 . _S2) _S2) <-
 (member _S1 _S2))
((conjunctive-abstraction (_S1 . _S2) (_S1 . _S3)) <-
 (conjunctive-abstraction _S2 _S3))
```

補助述語：

```
(member _E (_E . _L))
((member _E (_L1 . _L2)) <- (member _E _L2))

((copy _X _Y) <-
 (asserta (copydata _X))
 (retract (copydata _Y))
 (cut))
```

6．例題

(1) ピタゴラスの定理の発見

直角三角形は、その斜辺への垂線によって二つの相似な直角三角形に分割される。これらの三角形はまたもとの直角三角形とも相似であり、直角三角形の直角を挟む2辺の上の、もとの直角三角形と相似な三角形の面積の和は、もとの直角三角形の面積に等しい。この事実からの類推によって、ピタゴラスの定理（三平方の定理）を発見すること[Polya, 1959]をABAでシミュレートしよう。

知識ベースには、既知の直角三角形の性質、その直角三角形の性質を記述する述語の抽象化関係と一般化関係、および仮説検証のための直角三角形の事例データを記述する。なお、抽象化階層の根になる述語を(abstraction)、一般化階層の根になる述語を(OBJECT _X)とする。異なる概念階層を設けることに注意。以下に、BABYLONによる記述を示す。

知識記述：

```
;;; Discovery of Pythagoras's theorem
;; properties of right-triangle；既知の直角三角形の性質
((property-of-right-triangle) <-
 (right-triangle (_A _B _C))  ；直角三角形(_A _B _C)
 (analogous-right-triangle-on-edge (edge (_A _B)) _T1)
 (area _T1 _A1)    ；辺上の相似な直角三角形の面積
 (analogous-right-triangle-on-edge (edge (_C _A)) _T2)
 (area _T2 _A2)
 (analogous-right-triangle-on-edge (edge (_B _C)) _T3)
 (area _T3 _A3)
 (+ _A1 _A2 _A1+A2)        ；面積の和 _A1+A2
 (== _A1+A2 _A3))；もとの三角形の面積との関係

;; abstractions for Pythagoras；述語の抽象化関係
((abstraction) <- (property-of-right-triangle))；直角三角形
                                                の性質
((property-of-right-triangle) <-
 (right-triangle) (property-of-edge) (relation))
((abstraction) <- (right-triangle))          ；直角三角形
((right-triangle) <- (right-triangle _A))
((abstraction) <- (relation))                ；関係述語
((relation) <- (== _A _B))
((relation) <- (+ _A _B _C))
((abstraction) <- (property-of-edge))    ；辺の性質
((property-of-edge) <-
```

```
 (analogous-right-triangle-on-edge (edge _E) _D)
 (area _D _A))

;; generalizations for Pythagoras；述語の一般化関係
((OBJECT _X) <- (analogous-polygon-on-edge _X _Y))
                              ；辺の上の直角三角形
((OBJECT _X) <-
  (and (square-on-edge (edge _E) _S) (area _S _A)))
                              ；辺の上の正方形の面積
((analogous-triangle-on-edge _X _Y) <-；辺の上の三角形
 (analogous-right-triangle-on-edge _X _Y))
((analogous-polygon-on-edge _X _Y) <-；辺の上の多角形
 (analogous-triangle-on-edge _X _Y))
((analogous-polygon-on-edge _X _Y) <-；辺の上の正方形
 (square-on-edge _X _Y))
((and (square-on-edge (edge _E) _S) (area _S _A)) <-
 (and (length-of-edge (edge _E) _L) (square _L _A)))
          ；辺の上の正方形の面積（長さによる表現）

;; FACTS；直角三角形の事例データ
(right-triangle (A B C))          ；直角三角形(A B C)
(right-angle A)
(edge (A B))                      ；辺(A B)
(edge (C A))
(edge (B C))
(analogous-right-triangle-on-edge (edge (A B)) T1)
(analogous-right-triangle-on-edge (edge (C A)) T2)
(analogous-right-triangle-on-edge (edge (B C)) T3)
                    ；辺上の相似な直角三角形（三つ）
(area T1 3.84)                    ；直角三角形の面積
(area T2 2.16)
(area T3 6.0)
(length-of-edge (edge (A B)) 4)；辺の長さ
(length-of-edge (edge (C A)) 3)
(length-of-edge (edge (B C)) 5)
(square 4 16)                     ；辺の長さの平方
(square 3 9)
(square 5 25)
```

実行結果：

類推の目的(_Purpose)として「直角三角形の性質」という抽象述語を与える。また、仮説基準(_Criteria)として面積以外の性質を求めるという制約を与える。

```
Goals = (ABA (PROPERTY-OF-RIGHT-TRIANGLE)
                    ；直角三角形の性質を目的とする
          (not (member (area _ _) _Conjecture))
                    ；面積に関する述語を含まない
          _Conjecture _Assumption)
```

ABAの実行によって、与えた目的を満たす類比(ANALOGY)を発見し、その一般化／特殊化によって仮説基準を満たす仮説(_CONJECTURE)を得る。仮説が成り立つための仮定(_ASSUMPTION)（この場合は、仮説は与えた知識によって検証できるので仮定はnil）を同時に得る。

```
ANALOGY=      ；発見した類比1
((RIGHT-TRIANGLE) (PROPERTY-OF-EDGE)
 (RELATION))
-> fail    ；抽象述語は一般化／特殊化できない
ANALOGY=      ；発見した類比2
((RIGHT-TRIANGLE (_A _B _C)
          ；垂線による相似三角形分割
 (ANALOGOUS-RIGHT-TRIANGLE-ON-EDGE
  (EDGE (_A _B)) _T1) (AREA _T1 _A1)
 (ANALOGOUS-RIGHT-TRIANGLE-ON-EDGE
  (EDGE (_C _A)) _T2) (AREA _T2 _A2)
 (ANALOGOUS-RIGHT-TRIANGLE-ON-EDGE
  (EDGE (_B _C)) _T3) (AREA _T3 _A3)
 (+ _A1 _A2 _A1+A2) (== _A1+A2 _A3))

_CONJECTURE=        ；仮説
((RIGHT-TRIANGLE (_A _B _C))    ；ピタゴラスの定理
 (LENGTH-OF-EDGE (EDGE (_A _B)) _T1)
 (SQUARE _T1 _A1)
 (LENGTH-OF-EDGE (EDGE (_C _A)) _T2)
 (SQUARE _T2 _A2)
 (LENGTH-OF-EDGE (EDGE (_B _C)) _T3)
 (SQUARE _T3 _A3)
 (+ _A1 _A2 _A1+A2) (== _A1+A2 _A3))

_ASSUMPTION = NIL   ；仮定はない（事実である）
```

(2) 引力に関する発見的類推[有馬, 1992]

含意を表わす節に対する類推の例題である。人間はひもでつながった物を回すことができるが、そのとき物は人間に引っぱられている。この事実をもとにして、太陽と惑星の間の引力を発見する。

知識記述：

```
;;; Heuristic discovery of a new fact about the Sun
;; known facts about the Sun and a Planet
          ；太陽系に関して知られている事実
(Revolves Planet Sun)；惑星は太陽の回りを回っている
(ApartFrom Sun Planet)；太陽と惑星は離れている
```

```
;; known facts about a Stone and Mr. N
        ；石とN氏についての事実
(Revolves Stone N)；石はN氏の回りを回っている
(ApartFrom N Stone)；石とN氏は離れている
(Connects N String)；N氏はひもで結ばれている
(Connects Stone String)；石はひもで結ばれている
(Inextensible String)；ひもは伸びない
(Draws N String)    ；N氏はひもを引っぱる
(Mineral Stone)     ；石は鉱物である
(Human N)           ；N氏は人間である

;; Domain theories to explain (Revolves Stone N)
        ；石の回転を説明する理論
((Revolves _x _y) <-；_xが_yの回りを回転するのは、
 (Human _y)         ；_yが人間で、
 (ApartFrom _y _x)；_xと_yが離れていて、
 (Attracts _y _x))  ；_yが_xを引っぱっているときである

((Attracts _x _y) <-；引っぱることの説明（略）
 (Inextensible _z) (Connects _x _z) (Connects _y _z)
 (Draws _x _z))

;; generalizations (concept hierarchy)   ；一般化関係
((OBJECT _X) <- (Physical-object _X))
                ；物理的物体はものである
((Physical-object _X) <- (Mineral _X))
                ；鉱物は物理的物体である
((Physical-object _X) <- (Life _X))
                ；生物は物理的物体である
((Physical-object _X) <- (Celestial-body _X))
                ；天体は物理的物体である
((Life _X) <- (Human _X))   ；人間は生物である
(Celestial-body Planet)     ；惑星は天体である
(Celestial-body Sun)        ；太陽は天体である
```

実行結果：

類推の目的(_Purpose)として「物体が回転することの説明」という抽象述語を与える。また、仮説基準(_Criteria)として、仮説が「太陽と惑星の関係」および「太陽は天体であること」を含むという制約を与える。

```
Goals = (ABA ((Revolves _ _) <- _)
                ；回転することの説明を目的とする
        (and (member (Revolves Planet Sun) _Conjecture)
                ；太陽と惑星の関係と
             (member (Celestial-body Sun) _Conjecture))
                ；太陽は天体であることを使う
        _Conjecture _Assumption)
```

ABAの実行によって、与えた目的を満たす類比(ANALOGY)を発見し、その一般化／特殊化によって仮説基準を満たす仮説(_CONJECTURE)を得る。仮説が成り立つための仮定(_ASSUMPTION)（この場合は「太陽は惑星を引っぱている」という仮定）を同時に得る。

```
ANALOGY=        ；発見した類比
((REVOLVES _X _Y) <-
 (HUMAN _Y) (APARTFROM _Y _X) (ATTRACTS _Y _X))
                ；人間がものを回すことの説明理論

_CONJECTURE=            ；仮説
((REVOLVES _X _Y) <-    ；天体はものを回す
 (CELESTIAL-BODY _Y) (APARTFROM _Y _X)
 (ATTRACTS _Y _X))

_ASSUMPTION = ((ATTRACTS SUN PLANET))
        ；仮定＝太陽は惑星を引っぱている
```

7．考察と結論

(1) 効率と妥当性の評価

既知の知識から類推によって得ることができる正しい知識は、ベースとなる知識を一般化／特殊化したものに限られる。なぜなら、得られた知識と既知の知識との間に一般化／特殊化の関係がないとすると、上述の類似性の定義によってこの二つの知識の間に類似性はないので、類推によって新しい知識を得たということはできないことになる。このことからも明らかなように、類推の妥当性は類似性の定義に大きく依存している。

類推の推論効率に関しては、二つの課題がある。一つは類比の発見の効率であるが、抽象化による類似性の検出は有効な手段と考えられる。一方、知識変換としての一般化／特殊化操作については、現在のところランダム探索であり、対象領域の述語定義が多くなると仮説基準を満たす仮説を得るのに非常に時間が掛かる。したがって、ABAを実用的な問題解決に適用するには、対象領域理論の制限が必要と考えられる。一般化／特殊化操作自体に類推を適用することも有効であると考えられるが、今後の課題である。

(2) 他の類推形式への拡張

本論文での発見的類推は、抽象化／一般化／特殊化を知識変換の基礎とした。この変換操作以外に類推を実現する変換操作はあるのだろうか？

まず、置換に基づく類推との統合が挙げられる。置換に基づく類推では、対象世界での述語の対応付けに基づいて、対応する述語に置換することで類推を行うが、この述語の対応付けは述語の一般化／特殊化に他ならない。すなわち、述語の対応関係は、二つの対象世界を一般化した世界において同一視される場合にその異なる特殊化として与えられる。このことは、置換に基づく類推も、本論文における一般化に基づく類推の変形と見られることを意味する。一般化は置換を包含する。ただし、置換に基づく類推では、特殊化した述語が予め与えられているという点で探索がいらないことがABAと大きく異なる。

それでは、まだ他にも類推のための知識変換の形式はありうるだろうか？　人間の行っている類推のモデルとして抽象化／一般化／特殊化だけで十分であろうか？人間の言語による思考は、このモデルだけでは捉えられないと考えられる。例えば、メタファーの使用は明らかにこれ以外の要素を含む。

(3) 発見的類推の応用

本研究において定式化した発見的類推は、一般化された知識の獲得が困難な領域において、事例を推論に直接利用することの基礎をなす。例えば、発明や発見の支援である。発明は「新しい組み合わせの発見」と考えられているが、この過程において類推が大きな役割を持つと考えられる。類推なしで発明を説明することは、ほとんど困難である。なぜなら、新しい組合せ数はランダム探索に対してはほとんど無限といってよく、何らかの探索のガイド無しに目的とする組み合わせを発見することは不可能である。また、世界についての新しい認識としての発見についても、その認識を構成する語彙自体を既知の言語から構成することが必要であるが、その組み合わせを発見することにおいても発明の場合と同じことが言える。

8．おわりに

本論文では、抽象化／一般化／特殊化を基本操作とする知識変換として発見的類推を定式化した。これらの操作を述語による知識表現上での機械的演算として実現するアルゴリズムABAを構成して、例題に対する実行結果を示した。述語論理による定式化の範囲では、発見的類推の機械化が可能である。

参考文献

[有川, 1986]　有川節夫：帰納推論と類推－理論と応用、淵　一博監修、古川康一・溝口文雄共編、「知識の学習メカニズム」、共立出版(1986)

[有馬, 1992]　有馬　淳：類推要素間の関連性に関する論理的分析、情報処理学会論文誌、Vol.33, No.7, pp.887-896 (1992)

[岩山 et al, 1991]　岩山　登、佐藤　健、有馬　淳：一般化に基づく類推の論理プログラミングによる実現、ICOT研究論文、TR-685 (1991)

[原口, 1986]　原口　誠：類推の機械化について、淵　一博監修、古川康一・溝口文雄共編、「知識の学習メカニズム」、共立出版(1986)

[脇園 et al, 1990]　脇園竜次、有川節夫、原口　誠：類推のための抽象化、人工知能学会研究会資料、SIG-FAI-8904-2、pp.11-20 (1990)

[Christaller et al, 1992]　Christaller, T., DiPrimio, F., and Voss, A.: "The AI Workbench BABYLON", Academic Press (1992)

[Giordana et al, 1991]　Giordana, A., Saitta, L., and Roverso, D.: Abstracting Background Knowledge for Concept Learning, Lecture Notes in Computer Science, Vol.482, pp.1-13 (1991)

[Giunchiglia & Walsh, 1990]　Giunchiglia, F. and Walsh, T.: The Use of Abstraction in Automated Inference, IEE Conf. Publ., No.316, pp.365-370 (1990)

[Greiner, 1988]　Greiner, R.: Learning by Understanding Analogies, Artificial Intelligence, Vol.35, No.1, pp.81-125 (1988)

[Kedar-Cabelli, 1985]　Kedar-Cabelli, S. T.: Purpose-Directed Analogy, Proceedings of the Cognitive Science Society, Irvine, CA, August (1985)

[Plaisted, 1981]　Plaisted, D. A.: Theorem Proving with Abstraction, Artificial Intelligence, Vol.16, pp.47-108 (1981)

[Polya, 1959]　Polya, G.:「帰納と類比」、柴垣和三雄訳、丸善(1959)

[Winston, 1980]　Winston, P. H.: Learning and Reasoning by Analogy, Commun. ACM., Vol.23, No.12, pp.689-703 (1980)