区間変数に関する包含制約の等価変換

Equivalent Transformation of Member Constraints on Interval-Variable Domain

馬淵 浩司
Hiroshi Mabuchi

岩手県立大学ソフトウェア情報学部
Dept. of Software and Information Science, Iwate Prefectural University
mabu@soft.iwate-pu.ac.jp

赤間 清
Kiyoshi Akama

北海道大学情報メディア教育研究総合センター
Center for Information and Multimedia Studies, Hokkaido University
akama@cims.hokudai.ac.jp

繁田 良則
Yoshinori Shigeta

東芝システム LSI 技術研究所
System ULSI Engineering Laboratory, Toshiba Corporation
yoshinori.shigeta@toshiba.co.jp

小池 英勝
Hidekatsu Koike

北海道大学大学院工学研究科システム情報工学専攻
Division of System and Information Engineering, Hokkaido University
koike@cims.hokudai.ac.jp

keywords: interval variable, member constraint, equivalent transformation, problem solving.

Summary

It is important to increase computation efficiency of a system at a low cost while holding the system correct. For this realization, addition of new and efficient equivalent transformation (ET) rules, whose correctness is assured, is useful. As long as correct ET rules are added to a correct system, the computation result of the system is always correct. Improvement of rules is promoted further by improvement of data structure.

In this paper, we improve data structure by introducing interval variables into the usual term domain and add two correct and efficient ET rules, which are promoted by introduction of interval variables, for member constraints on interval variables. These rules are the candidate elimination rule and the common pattern specialization rule. We show by an experiment that computation efficiency is increased by using these rules.

1. ま え が き

1.1 正当性が保証されたルールの追加

高度な知識処理を扱うシステムは、その記述が大規模で複雑なものになりがちで、最初からすべての機能が完備したシステムを構築するのは困難である。そのため、いったん構築したシステムに新しい機能を追加して、システムを徐々に改善していくことが必要になる。システムを改善するのは、新しい機能の追加以外に、新しい推論方式の導入や、より高速な計算の達成である場合も考えられる。

その際問題となるのは、更新のためのコストである。大幅な記述の変更には大変なコストがかかる。このため、少ない変更で機能や性能の改善が行なえるシステム構築法が重要となる。

本論文では、逐次的なシステムの改善に適した方法を採用している。この方法は、システムを等価変換ルールの集合で構成し、この集合の中からルールを選択し繰り返し適用することで計算を行なう。

等価変換アプローチでは、次のようなことがいえる。

(1) 等価変換ルールは、他の等価変換ルールとは無関係に個別に正当であるか否かが判定できる。

(2) 正当な等価変換ルールを適用する限り、いろいろな等価変換ルールをどのような順番で適用しても、その計算結果が正当であることが保証できる。

(1) と (2) から次のようなことが言える。いくつかの正当な等価変換ルールからなるシステムが存在すると仮定する。(2) より、このシステムは正しく計算結果を返すことができる。このとき、このシステムに新しいルールを追加することを考える。追加する新しいルールは、正当であると判断したものに限る。そのルールを追加した新しいルール集合を持つシステムは正しく動作する。なぜならば、そのシステムの持つルールはすべて正当なので、(2) より計算結果は常に正しくなるからである。以上より、次のことがいえる。

(3) 正当な等価変換ルールを追加する限り、システムは常に正しく動作する。

1.2 ルール追加によるシステムの高速化

正当なルールを追加しても、すべてのルールがシステムの改善に貢献するとは限らない。なぜならば、計算対象を効率的に変換しないルールも存在するからである。例えば、変換する対象のサイズ(節の数)が増えるようなルールは通常効率的なルールではない。そこで、追加する新しいルールは、計算対象を効率的に変換すると判断したものに限る。これによって、正当で効率的なルールのみがシステムに追加される。正当で効率的な等価変換ルールを追加することで、効率的なシステムを低コストで得ることが期待できる。

unfold 変換は等価変換ルールであることが知られているが、unfold 変換以外の等価変換ルールも数多く存在する。従って、unfold 変換と unfold 変換以外の等価変換ルールを組み合わせるシステムを構築しても、システムは正当に動作することが等価変換アプローチでは保証されている。unfold 変換だけからなるシステムは論理アプローチで作られるシステムと同等の性能を持つが、このようなシステムに unfold 変換以外の効率的な等価変換ルールを追加することによって、論理アプローチで作られるシステムよりもはるかに効率的なシステムを低コストで得ることができる。

1.3 表現と操作の増加を基礎とした高速化

適切なデータ構造を用いることは、効率的なシステムを作成する上で、極めて重要な働きをすることが知られている。

等価変換アプローチでは、データ構造に基づくシステムの改善が可能である。計算はデータ構造 + 宣言的記述 + 等価変換ルール + 制御と考えられ、データ構造が計算に影響を与える。

データ構造の改善がシステムの効率化に貢献するのは、次の理由からである。データ構造は、計算の対象表現とそれを変更する操作(特殊化)からなる^{*1}。また等価変換ルールは、計算の対象表現を操作して、別の対象表現に変更する手続きである。従って、対象表現やそれを変更する操作が増えると、その分だけ適用可能なルールが増える。その中にはシステムの改善に貢献する効率的な等価変換ルールも存在する。それらのルールを発見し追加することでシステムを改善することができる。

1.4 本論文の目的

1.1 節から 1.3 節で述べたように、データ構造の改善は、正当で効率的なルールの発見と追加を介して、システムを正当性を保持したまま改善し問題解決の高速化に貢献することができる。

本論文では、データ構造の改善として、通常の項領域に区間変数を導入し、これを利用して、区間変数に関す

る包含制約の等価変換ルール(候補の制限ルールと共通性による特殊化ルール)をシステムに追加し、これらのルールの正当性を証明する。また、追加されたルールを適用することで、システムの正当性を保持したまま問題解決の高速化が達成されることを示す。候補の制限ルールは包含制約の候補リストを縮小し、共通性による特殊化ルールは包含制約の候補の共通一般化を用いて節を特殊化する。

1.5 本研究の位置づけ

等価変換アプローチでは、本研究の等価変換ルールの導入は、正当性を保持したまま低コストでシステムを改善するという重要な意味を持つが、他の枠組ではそのような効果をうまく引き出すことができない。それは次のような理由からである。

論理プログラミングの枠組では、等価変換ルールをシステムに導入しても、それを組み込んで直ちに正当な計算として用いることはできない。なぜなら、論理プログラミングの枠組での計算手続き(SLD resolution など)は、個別の推論規則ではなく、それらの集まりである resolution 全体の正当性を示せない限り、正当な計算として使うことができないからである。つまり、個別のルールの正当性を積み重ねるだけでは十分でなく、ルールを追加することに resolution 全体の正当性を証明し直さなければならない[Lloyd 87]。

論理プログラミングのプログラム変換の枠組でも、等価変換ルールを追加すると、正当性が維持できない恐れがある。なぜなら、プログラム変換(unfold/fold 変換)という全体の手続きとしての正当性は、個別のルールの正当性の積み重ねによって保証する形になっていないからである。従って、ルールを追加することに、unfold/fold 変換全体に悪影響がでないことを確かめ、unfold/fold 変換全体の手続きの正当性を証明しなければならない[Pettorossi 94, Tamaki 84]。

等価変換アプローチで、正当性を保持したまま低コストなシステム改善を行ない、問題解決の高速化を達成した研究に[赤間 98a]がある。その研究では、項領域における 2 つの等価変換ルール(候補の制限ルールと共通性による特殊化ルール)によって高速化を達成したが、本研究ではデータ構造を改善して更なる高速化を達成する。

2. 区間変数領域の表現と計算の基礎

2.1 区間変数を含む項

区間変数とは、 $B: [2, 5]$ のように、タグ(B)と区間($[2, 5]$)の 2 項組であり、その区間の範囲の実数 B という直観的意味を持つ。区間変数は、問題解決の高速化に著しく貢献する場合がある。高速化が達成される主な原因は、区間変数によって大まかな制約が伝播されることにより、最終的に不要となる多くの選択肢が効果的に枝

^{*1} 対象表現とそれに関する特殊化は、数学的には特殊化システム(3.1 節)で議論される。

刈りされることである。

プログラムの基本要素として、3種類の項を導入する。それは、4のような定数、 Y のような純変数^{*2}、それに $B : [2, 5]$ のような区間変数である。純変数と区間変数をあわせて変数と呼ぶ。数学的には、 $B : [2, 5]$ のような区間変数は、 B のようなタグ^{*3} と $[2, 5]$ のような区間の2項組として扱われる。これらの基本要素と f や g のような関数^{*4} を用いて再帰的に $f(4, B : [2, 5], g(Y))$ のような複合項をつくる。このようにして、本論文で対象とする「区間変数を含む項」全体の集合が得られる。本論文では、簡単のために、区間変数を含む項を単に項と呼ぶことがある。項のうち変数を含まないものを基礎項と呼ぶ。

2.2 区間変数を含む項に対する基本特殊化

論理プログラミングでは、項は代入によって変化(特殊化)させられる[Lloyd 87]。しかし区間変数を含む項の世界は、代入の概念だけでは不十分である。そこで本論文では、代入の拡張概念である「特殊化」の概念[赤間 98c]を用いる。特殊化は論理プログラムの理論での代入に相当し、項に対して任意の変更を可能にする。

まず、区間変数を含む項を変更する基本的な操作として「基本特殊化」を導入する。(1) 純変数は、変数代入によって特殊化される。例えば、 Z は $(Z, 2)$ という基本特殊化によって2へ特殊化される。(2) 区間変数は、タグを変更することによって特殊化される。例えば、 $B : [3, 8]$ は (B, C) によって $C : [3, 8]$ に特殊化される。(3) 区間変数は、区間をその部分集合(部分区間)に変更することによって特殊化される。 $B : [3, 8]$ は $(B, [4, 5])$ という基本特殊化によって $B : [4, 5]$ に特殊化される。(4) 区間変数は区間に属する定数に特殊化される。 $B : [3, 7]$ は $(B, 5)$ という基本特殊化によって5に特殊化される。

4種類の基本特殊化のうち、(3),(4)の基本特殊化は、場合によっては適用不可能になる。例えば、 $(B, [4, 5])$ という基本特殊化は $B : [6, 8]$ には適用できない。これは $[4, 5]$ が $[6, 8]$ の部分区間ではないからである。同様に、 $(B, 5)$ という基本特殊化は $B : [6, 8]$ には適用できない。これは5が $[6, 8]$ の要素ではないからである。

2.3 区間変数を含む項に対する特殊化

等価変換によるアプローチにおける特殊化の概念は、適用不可能性を部分写像の概念によって正確に扱うことができる。このような概念は、従来の理論では取り扱われていない。例えば、 $(B, 5)$ という基本特殊化の適用を1つの部分写像で表現し、その部分写像は、 $B : [4, 6]$ を5に移すが、 $B : [6, 8]$ では未定義であるとする。

*2 純変数とは、いわゆる変数のことであるが、区間変数と区別するためにこのように呼ぶ。

*3 区間変数で使われる変数を特にタグと呼ぶ。

*4 これは数学上の写像の意味の関数ではなく、単なる記号である。

特殊化は、基本特殊化の列(空列 $[]$ を含む)として定義される。特殊化の適用は、特殊化の中の基本特殊化の繰り返し適用によって定義される。本論文では、特殊化を θ や σ などのギリシャ文字で表記し、特殊化の作用を(代入と同様に)後置記法で表す。例えば、項 t が $f(Z, B : [3, 8])$ で、特殊化 θ が $[(Z, C : [1, 9]), (C, 5)]$ であるとき、

$$\begin{aligned} t\theta &= f(Z, C : [3, 8])[(Z, C : [1, 9]), (C, 5)] \\ &= f(C : [1, 9], C : [3, 8])[(C, 5)] \\ &= f(5, 5)[] \\ &= f(5, 5) \end{aligned}$$

となる。基本特殊化が適用不可能な場合があるので、特殊化についても適用不可能な場合が存在する。特殊化が適用可能なのは、それに属する基本特殊化が順次すべて適用可能なときである。

2.4 区間変数を含むアトムとその特殊化

述語と(区間変数を含む)項の並びからなる $p(t_1, t_2, \dots, t_n)$ の形^{*5}の表現を区間変数を含むアトム(または単にアトム)と呼ぶ。アトムのうち変数を含まないものを基礎アトムと呼ぶ。

(区間変数を含む)項に対する特殊化は自然に(区間変数を含む)アトムに拡張できる。特殊化のアトムへの作用も後置記法で表す。変数を含まないアトムを基礎アトムと呼ぶ。

2.5 区間変数を用いた確定節

区間変数を含むアトムを用いて、確定節を作ることができる。例えば、

$$p(X : [2, 5]) \leftarrow q(X : [3, 8], Y : [2, 5])$$

は、確定節の例である。確定節のボディにさらに制約も書きたい場合がある。例えば、

$$p(Z) \leftarrow Z = X : [2, 5], q(X : [3, 8], Y : [2, 5])$$

は、等式制約 $Z = X : [2, 5]$ をボディの先頭に持つ。

制約を形式的に扱うために、制約対象 a と制約領域 G の組 (a, G) で制約を表す。この制約 (a, G) の直観的意味は、制約対象 a が制約領域 G の元だけにしか具体化されないことである。例えば、上記の等式制約 $Z = X : [2, 5]$ を正式に表現するには、制約対象として

$$a = \text{equal}(Z, X : [2, 5])$$

を用い、制約領域として

$$G = \{\text{equal}(g, g) \mid g \text{ は基礎項}\}$$

を使えばよい。

制約 (a, G) は、 a が基礎アトムであるとき基礎制約と呼ばれる。 $a \in G$ を満たす基礎制約 (a, G) を真な制約と呼ぶ。 $a \notin G$ を満たす基礎制約 (a, G) を偽な制約と呼ぶ。

特殊化 θ が制約 (a, G) に適用可能なのは、 θ が a に適用可能なときであり、その適用結果 $(a, G)\theta$ は制約 $(a\theta, G)$

*5 但し、通常の Prolog の記法にならって、 $n = 0$ のときは括弧は省略し述語だけを書く。

である。このとき、 (a, G) は θ によって $(a\theta, G)$ に特殊化されるという。

ヘッドがアトムであり、ボディがアトムや制約からなる節を制約つき確定節と呼ぶ。本論文では、制約つき確定節をしばしば単に確定節、或いは節と呼ぶことがある。基礎アトムまたは基礎制約だけからなる制約つき確定節を基礎節と呼ぶ。

制約つき確定節

$$C = (H \leftarrow B_1, B_2, \dots, B_n)$$

と特殊化 θ に対して、もし θ が $H, B_1, B_2, \dots, B_n$ のすべてに適用可能であるならば、 θ は C に適用可能で、適用結果 $C\theta$ は

$$C\theta = (H\theta \leftarrow B_1\theta, B_2\theta, \dots, B_n\theta)$$

であると定義する。 $C\theta = C'$ のとき、 C は θ で C' に特殊化される、または、 C' は C の例であるという。基礎節 C' が C の例であるとき、 C' は C の基礎例であるという。

3. 等価変換の正当性のための基礎

3.1 特殊化システム

特殊化システム Γ [赤間 98c] とは、知識表現が扱う対象とその対象を変更する操作である特殊化がなす構造で、アトムの集合 $\mathcal{A}$ 、基礎アトムの集合 $\mathcal{G}$ 、特殊化の集合 $\mathcal{S}$ と、 $\mathcal{S}$ から $\text{partial_map}(\mathcal{A})$ への写像 μ の 4 項組である。 μ は特殊化 s が与えられると、 $\mathcal{A}$ のある元を別の $\mathcal{A}$ の元 (同じこともある) に変更する部分写像を決定する写像である。 $\text{partial_map}(\mathcal{A})$ とは、 $\mathcal{A}$ から $\mathcal{A}$ への部分写像全体の集合である。 $\mathcal{A}$ の元 a に、部分写像 $\mu(s)$ を適用して $\mathcal{A}$ の元 b になるとき、 a は s によって b に特殊化されるといい、 $\mu(s)(a) = b$ と書く。この部分写像によって、すべての節に対する処理のみならず、節ごとの処理が可能になる。

本論文で区間変数を理論的に扱う基盤はこの特殊化システムの構造である。区間変数の存在により特殊化の作用が部分写像となるため、項領域の理論の単純な拡張だけでは不十分である。

3.2 制約つき宣言的プログラムの意味

2.5 節で述べた制約つき確定節の集合が制約つき宣言的プログラムである。制約つき確定節 C のヘッドを $\text{head}(C)$ 、ボディ中のアトムの集合を $\text{atom}(C)$ 、ボディ中の制約の集合を $\text{const}(C)$ で表す。 $\Gamma = \langle \mathcal{A}, \mathcal{G}, \mathcal{S}, \mu \rangle$ とすると、 $T\text{const}(\Gamma)$ は Γ 上の真な制約全体の集合である。 Γ 上の基礎節全体の集合を $G\text{clause}(\Gamma)$ で表す。

まず、節 C の意味 $\mathcal{M}(C)$ を

$$\begin{aligned} \mathcal{M}(C) = \{ & (\text{head}(C\theta), \text{atom}(C\theta)) \\ & | \theta \in \mathcal{S}, \\ & C\theta \in G\text{clause}(\Gamma), \\ & \text{const}(C\theta) \subseteq T\text{const}(\Gamma) \} \end{aligned}$$

で定義する。

上の定義中の条件 $\text{const}(C\theta) \subseteq T\text{const}(\Gamma)$ は、節 C から得られる基礎例 $C\theta$ のうち、 $C\theta$ のボディ中の制約がすべて真であるような基礎例だけが C の意味を求める際に考慮されることを要請している。

これを用いて、 Γ 上のプログラム P に対して、写像 $T_P: 2^{\mathcal{G}} \rightarrow 2^{\mathcal{G}}$ を定義する。

【定義 1】(写像 T_P) 写像 $T_P: 2^{\mathcal{G}} \rightarrow 2^{\mathcal{G}}$ を、任意の $I \subseteq \mathcal{G}$ に対して

$$T_P(I) \stackrel{\text{def}}{=} \{x \mid C \in P, (x, y) \in \mathcal{M}(C), y \subseteq I\}$$

で定義する。

次に写像 T_P を使って、 P の意味を定義する。

【定義 2】(プログラムの意味) P を Γ 上のプログラムとする。プログラム P の意味 $\mathcal{M}(P)$ を

$$\mathcal{M}(P) \stackrel{\text{def}}{=} \bigcup_{n=1}^{\infty} [T_P]^n(\emptyset)$$

で定義する。但し、

$$[T_P]^0(\emptyset) = \emptyset, [T_P]^{n+1}(\emptyset) = T_P([T_P]^n(\emptyset)) \text{ である。}$$

3.3 節の置き換え

命題 1 P を制約つき宣言的プログラム、 C と C' を制約つき確定節とする。もし $\mathcal{M}(C) = \mathcal{M}(C')$ ならば、 $\mathcal{M}(P \cup \{C\}) = \mathcal{M}(P \cup \{C'\})$ が成り立つ。

本論文の命題や定理などの証明は付録で与える。

3.4 節の特殊化

命題 2 P を制約つき宣言的プログラム、 C を制約つき確定節、 (A, G) を C 中の制約であるとする。もし特殊化 θ が、

「 $C\sigma \in G\text{clause}(\Gamma)$ かつ $A\sigma \in G$ を満たす任意の $\sigma \in \mathcal{S}$ に対して、 $C\sigma = C\theta\rho$ を満たす $\rho \in \mathcal{S}$ が存在する」

を満たすならば、 $\mathcal{M}(P \cup \{C\}) = \mathcal{M}(P \cup \{C\theta\})$ が成り立つ。

4. 区間変数領域の基礎的命題

前章までと異なり、本章以後は、 $\Gamma = \langle \mathcal{A}, \mathcal{G}, \mathcal{S}, \mu \rangle$ は区間変数領域の特殊化システムに限定される。区間変数を含む項全体の集合を $\mathcal{A}_T$ 、区間変数を含む基礎項全体の集合を $\mathcal{G}_T$ で表す。

4.1 正則な確定節

$u: [2, 6]$ と $u: [5, 7]$ のように、2 つの区間変数が同一のタグを持ち、しかも異なる区間を持つとき、それらの区間変数は互いに「食い違っている」という。節が正則であるとは、互いに食い違っている区間変数を持たないことである。プログラムは、正則な節だけからなるとき、正則であるといわれる。

4.2 区間変数領域の等式制約

等号関係を記述する集合 $Equal$ を

$$Equal = \{equal(g, g) \mid g \in \mathcal{G}_T\}$$

とすると、任意の $x, y \in \mathcal{A}_T$ に対する

$$(equal(x, y), Equal)$$

の形の制約を $equal$ 制約または等式制約と呼ぶ。簡単化のため、 $(equal(x, y), Equal)$ を $[x = y]$ と表記することがある。

4.3 区間変数領域の包含制約

Γ 上のプログラム Q を

$$Q = \{ \text{member}(X, [X|R]) \leftarrow . \\ \text{member}(X, [Y|R]) \leftarrow \text{member}(X, R). \}$$

とする*6。包含関係を記述する集合 $Member$ を

$$Member = \mathcal{M}(Q)$$

とすると、任意の $x, y \in \mathcal{A}_T$ に対する

$$(\text{member}(x, y), Member)$$

の形の制約を member 制約または包含制約と呼ぶ。簡単化のため、 $(\text{member}(x, y), Member)$ を $[x \in y]$ と表記することがある。

4.4 純変数と任意の項の等式制約の処理

命題 3 P を制約つき宣言的プログラム、 C を正則な制約つき確定節

$$H \leftarrow B_1, \dots, B_{i-1}, B_i, B_{i+1}, \dots, B_n$$

C 中の B_i を、純変数 X と項 β から作られた等式制約 $[X = \beta]$ であるとする。そのとき、もし X が項 β に出現しないならば、 C から B_i を取り除いた制約つき確定節

$$C' = (H \leftarrow B_1, B_2, \dots, B_{i-1}, B_{i+1}, \dots, B_n)$$

と、 $\theta = [(X, \beta)]$ に対して、

$$\mathcal{M}(P \cup \{C\}) = \mathcal{M}(P \cup \{C'\theta\})$$

が成り立つ。

4.5 単一化と等式制約解消特殊化

【定義 3】(単一化子と基礎単一化子) α, β を項とする。 $\alpha\theta = \beta\theta$ を満たす特殊化 θ を α と β の単一化子という。 α と β の単一化子が存在するとき、 α と β は単一化可能である。

$\alpha\theta = \beta\theta \in \mathcal{G}_T$ を満たす特殊化 θ を α と β の基礎単一化子という。 α と β の基礎単一化子が存在するとき、 α と β は基礎単一化可能である。

【定義 4】(等式の制約解消特殊化) α, β を項とする。次の条件を満たす特殊化 θ を $[\alpha = \beta]$ の等式制約解消特殊化という。

P は制約つき宣言的プログラム、 C は制約 $[\alpha = \beta]$ をボディに持つ正則な制約つき確定節、 C' は C から制

約 $[\alpha = \beta]$ を除いてできる制約つき確定節とする。そのような任意の P, C, C' に対して、

$$\mathcal{M}(P \cup \{C\}) = \mathcal{M}(P \cup \{C'\theta\})$$

が成り立つ。

この定義は、等式制約 $[\alpha = \beta]$ から決まる等式制約解消特殊化 θ を用いれば、どんな「文脈」(P, C, C')のもとでも等式制約 $[\alpha = \beta]$ を解消できることを要請している。

区間変数の領域では、等式制約解消特殊化は必ず存在することが証明されている[赤間 98d]。項集合 ($\mathcal{A}_T$ の部分集合) は、互いに食い違っている区間変数を含まないとき、正則であると言われる。

【定理 1】 $\{\alpha, \beta\}$ が正則な項集合ならば、等式制約 $[\alpha = \beta]$ の等式制約解消特殊化 θ が存在する。このとき、 θ は α と β の単一化子となる。

5. 候補の制限による変換

5.1 候補の制限による変換の例

候補の制限による変換を例で説明する。例えば、確定節のボディに包含制約

$$[X : [2, 4] \in [3, 5, 2]]$$

が出現しているとき、区間変数 $X : [2, 4]$ は 2 以上 4 以下の値しかとらないので、5 にはなりえない。そこで、候補のリスト $[3, 5, 2]$ から候補 5 を削除することができる。

このような変換は、候補が数以外でも可能である。例えば、確定節のボディに包含制約

$$[X : [2, 4] \in [A : [3, 4], B : [5, 8], [a, b, c, d]]]$$

が出現しているとき、 $B : [5, 8]$ と $[a, b, c, d]$ は削除できる。それは、区間変数 $X : [2, 4]$ を具体化しても 5 以上 8 以下の数にはなりえないし、リストにもなりえないからである。

確定節のボディの包含制約 $[t \in L]$ において、リスト L からその要素 s を削除できる十分条件の 1 つは、 t と s の基礎単一化が不可能なこと、すなわち、 $t\theta = s\theta \in \mathcal{G}_T$ を満たす特殊化 θ が存在しないことである。例えば、 $X : [2, 4]$ と $A : [3, 4]$ は基礎単一化可能だが、 $X : [2, 4]$ と $B : [5, 8]$ はそうではない。

5.2 変換の正当性のポイント

候補の制限による変換の正当性のポイントを説明する。

C を、包含制約 $[x \in L]$ をボディに含む節とする。リスト L から、 x と基礎単一化可能でない要素を取り除いたリストを L' とし、 C の包含制約 $[x \in L]$ を包含制約 $[x \in L']$ に置き換えた節を C' とする。

包含制約 $[x \in L]$ がある特殊化で基礎化して真になるのは、 $x\theta$ が $L\theta$ のある要素と一致することである。しかし x と基礎単一化可能でない L の要素は、その真偽に関係しない。従って、節 C がある特殊化 θ の適用で基礎化して $[x \in L]\theta$ も真になるならば、節 C' も同じ特殊化

*6 Prolog の標準的な記法を用いている。 $[X|R]$ は、項 $\text{cons}(X, R)$ を意味する。

で基礎化し $[x \in L']\theta$ も真になる．

逆に，節 C' がある特殊化 θ で基礎化して $[x \in L']\theta$ が真になると仮定する．このとき，もし節 C が θ で基礎化すれば， $[x \in L]\theta$ が真になるのは明らかである．しかし， θ には次の問題がある．それは，(1) θ は $[x \in L]$ に適用できるか，(2) 適用したとき基礎化するか，の 2 つである．

これを解決するために， θ をもとにして，これらの問題を解決する新しい特殊化をつくり出す．その手順は，(1) θ を簡素化して C' に出現しない変数への操作部分を取り除き， σ を得る．(2) σ を C に適用して， $C\sigma$ を作る．(3) $C\sigma$ 中の変数を基礎化する特殊化 ρ を導入する．このとき， $\sigma \circ \rho$ は節 C を基礎化し， $[x \in L]\theta$ が真になることが証明できる [定理 2]．

5.3 包 含 関 係

包含制約の制約領域 *Member* を，より具体的に書き下す準備として，次のように定義する．
任意の自然数 ($k = 1, 2, \dots$) に対して， T_k は述語が *member* である基礎アトムで，第 1 引数が第 2 引数の第 k 番目の要素と等しいものの全体の集合とする．

$$T_k = \{ \text{member}(g_k, [g_1, g_2, \dots, g_k | x]) \mid g_1, g_2, \dots, g_k, x \in \mathcal{G}_T \}$$

以下で $[T_Q]^n(\emptyset)$ を T_i ($i = 1, 2, \dots$) と関係付ける．

命題 4 任意の $n = 1, 2, \dots$ に対して， $[T_Q]^n(\emptyset) = \bigcup_{i=1}^n T_i$ が成り立つ．

命題 5 $\text{Member} = \bigcup_{i=1}^{\infty} T_i$

命題 6 次の 2 つは同値である．

- (1) $\text{member}(g, [g_1, g_2, \dots, g_n]) \in \text{Member}$
- (2) $g, g_1, g_2, \dots, g_n \in \mathcal{G}_T, \bigvee_{i=1}^n (g = g_i)$

5.4 リストの要素の除去

以下では，

$$t_1, t_2, \dots, t_{i-1}, t_i, t_{i+1}, \dots, t_n$$

から t_i を抜いた列

$$t_1, t_2, \dots, t_{i-1}, t_{i+1}, \dots, t_n$$

を

$$t_1, t_2, \dots, \hat{t}_i, \dots, t_n$$

と表記する．同様に，

$$1, 2, \dots, \hat{i}, \dots, n$$

は 1 から n までの列から i だけを除いた列を表す．

[定理 2] P を Γ 上の制約つき宣言的プログラム， B_i を制約

$$(\text{member}(x, [t_1, t_2, \dots, t_i, \dots, t_n]), \text{Member})$$

B'_i を制約

$$(\text{member}(x, [t_1, t_2, \dots, \hat{t}_i, \dots, t_n]), \text{Member})$$

C と C' を Γ 上の制約つき確定節

$$C = (H \leftarrow B_1, B_2, \dots, B_{i-1}, B_i, B_{i+1}, \dots, B_n)$$

$$C' = (H \leftarrow B_1, B_2, \dots, B_{i-1}, B'_i, B_{i+1}, \dots, B_n)$$

とする．もし， C が正則で， x と t_i が基礎単一化可能でなければ， $\mathcal{M}(P \cup \{C\}) = \mathcal{M}(P \cup \{C'\})$ が成り立つ．

この定理を基にして作られる等価変換ルールを「候補の制限ルール」と呼ぶ．

6. 共通性による特殊化変換

6.1 共通性による特殊化変換の例

共通性による特殊化変換を例で説明する．例えば，確定節のボディに包含制約

$$[X \in [3, 5, 2]]$$

が出現しているとき，純変数 X は 2 以上 5 以下の値しかとらないことがわかるので， X を $X : [2, 5]$ に直すことができる．また，包含制約が

$$[Y : [1, 4] \in [3, 5, 2]]$$

の場合には，区間変数 $Y : [1, 4]$ を $Y : [2, 4]$ に直すことができる．

勿論もっと複雑な場合もありうる．例えば，包含制約が

$$[[3, Z] \in [[X, Y : [2, 6]], [3, 7]]]$$

の場合には，純変数 Z は 2 以上 7 以下の値しかとらないことがわかるので， Z を $Z : [2, 7]$ に直すことができる．

とりうる範囲を知るために，共通一般化の概念が使われる．例えば， $W : [2, 5]$ は 3, 5, 2 の共通一般化であり， $W : [2, 7]$ は $Y : [2, 6]$, 7 の共通一般化である．上記の第 1 の例では， X を $W : [2, 5]$ と単一化することによって， X を $X : [2, 5]$ に直すことができる．また，第 2, 第 3 の例でも， $Y : [1, 4]$ や Z を $W : [2, 7]$ と単一化することによって， $Y : [1, 4] \rightarrow Y : [2, 4]$ や $Z \rightarrow Z : [2, 7]$ の変更を知ることができる．

この単一化が失敗する場合には，その包含制約を含む節自体を削除することができる．例えば，節のボディに包含制約 $[X : [6, 8] \in [3, 5, 2]]$ が出現しているとき，純変数 X は 2 以上 5 以下の値しかとらないことがわかるので，この節は削除できる．これは，3, 5, 2 の共通一般化 $W : [2, 5]$ が $X : [6, 8]$ と単一化しないことによって説明できる．

6.2 共通性による特殊化変換の定義

π と $t_1, t_2, \dots, t_n$ を $\mathcal{A}_T$ の元とする． π が $t_1, t_2, \dots, t_n$ の共通一般化であるとは，

$$\pi \sigma_i = t_i \quad (i = 1, 2, \dots, n)$$

を満たす特殊化 $\sigma_i \in \mathcal{S}$ ($i = 1, 2, \dots, n$) が存在することである．

P を Γ 上の制約つき宣言的プログラム， B_i を制約

$$(\text{member}(s, [t_1, t_2, \dots, t_n]), \text{Member})$$

C を Γ 上の制約つき確定節

$$C = (H \leftarrow B_1, B_2, \dots, B_i, \dots, B_n)$$

とする． π を $t_1, t_2, \dots, t_n$ の共通一般化で， C と共有変数を持たない項とする．このとき，

- (1) $[\pi = s]$ の等式制約解消特殊化 θ が存在するならば, $P \cup \{C\}$ を $P \cup \{C\theta\}$ に変換する
 - (2) $[\pi = s]$ の等式制約解消特殊化が存在しないならば, $P \cup \{C\}$ を P に変換する
- のが, 共通性による特殊化変換である.

6.3 変換の正当性のポイント

共通性による特殊化変換の正当性のポイントを説明する. 節 C のボディに包含制約 $[Y : [1, 4] \in [3, 5, 2]]$ が出現しているとす. 節 C の中のすべての $Y : [1, 4]$ を $Y : [2, 4]$ に置き換えてよいことを示すためには, 次の道筋で節を変形し, それが等価変換であることが証明すればよい.

まず, 節に出現しない純変数 X を用いて, 節の中のその包含制約を新しい包含制約と等式制約

$$[X \in [3, 5, 2]], [X = Y : [1, 4]]$$

に置き換える. 次に, X を $3, 5, 2$ の共通一般化 $W : [2, 5]$ に特殊化する. これは 2 つの制約を

$$[W : [2, 5] \in [3, 5, 2]], [W : [2, 5] = Y : [1, 4]]$$

に直すことに等しい. そして最後に, 等式制約 $[W : [2, 5] = Y : [1, 4]]$ を解消する. これによって得られる節は, 節 C 中のすべての $Y : [1, 4]$ を $Y : [2, 4]$ に変換して得られる節と一致する.

6.4 共通性による特殊化変換の正当性

上記の Γ, C, B_i, π をもとにして, 次のような変数, 制約, 制約つき節, 特殊化を導入する.

- (1) X は, C や π に出現しない変数
- (2) B'_i は, 制約
 $(member(X, [t_1, t_2, \dots, t_n]), Member)$
- (3) E は, 制約
 $(equal(X, s), Equal)$
- (4) C'' は C の B_i を B'_i にかえて得られる Γ 上の制約つき確定節
 $C'' = (H \leftarrow B_1, B_2, \dots, B'_i, \dots, B_n)$
- (5) C' は C'' に E を付け加えて得られる Γ 上の制約つき確定節
 $C' = (H \leftarrow B_1, B_2, \dots, B'_i, \dots, B_n, E)$
- (6) $\{X/\pi\}$ は X に π を代入する基本特殊化 (X, π) だけからなる特殊化 $[(X, \pi)]$

まず, 共通性による特殊化変換の正当性を証明するための準備を行なう.

[補題 1] $\mathcal{M}(P \cup \{C\}) = \mathcal{M}(P \cup \{C'\})$

[補題 2] $\mathcal{M}(P \cup \{C'\}) = \mathcal{M}(P \cup \{C'\{X/\pi\}\})$

[補題 3] $[\pi = s]$ の等式制約解消特殊化 θ が存在するとき, $\mathcal{M}(P \cup \{C'\{X/\pi\}\}) = \mathcal{M}(P \cup \{C\theta\})$ であり, $[\pi = s]$ の等式制約解消特殊化が存在しないとき, $\mathcal{M}(P \cup \{C'\{X/\pi\}\}) = \mathcal{M}(P)$ が成り立つ.

これらの補題をもとにして, 次の定理が証明できる.

[定理 3] P を制約つき宣言的プログラム, C を制約つき確定節

$$C = (H \leftarrow B_1, B_2, \dots, B_i, \dots, B_n)$$

このボディの B_i を制約

$$(member(s, [t_1, t_2, \dots, t_n]), Member)$$

とする. π を $t_1, t_2, \dots, t_n$ の共通一般化で, C と共有変数を持たない項とする. $[\pi = s]$ の等式制約解消特殊化が存在するとき, それを θ とすれば,

$$\mathcal{M}(P \cup \{C\}) = \mathcal{M}(P \cup \{C\theta\})$$

が成り立つ. また, $[\pi = s]$ の等式制約解消特殊化が存在しないとき, $\mathcal{M}(P \cup \{C\}) = \mathcal{M}(P)$ が成り立つ.

この定理を基にして作られる等価変換ルールを「共通性による特殊化ルール」と呼ぶ.

7. 比較

7.1 計算の対象表現

論理プログラムの世界では, 計算の対象表現は項に限定されており, 項以外の対象や代入以外の対象変更操作 (特殊化) を扱うことはできない. また, 対象表現を追加するための理論的枠組がない. これは, 計算が論理 + 制御と捉えられているため, データ構造を一般的に議論する枠組がないことによる.

これに対して宣言的プログラムの世界では, データ構造を一般的に捉える特殊化システムという枠組が存在し, 計算の対象表現を追加することによって計算の空間を拡張することができる [赤間 98c]. 項領域における包含制約の等価変換 [赤間 98a] では, 計算の対象表現は項に限定され, 項領域での 2 つの等価変換ルールが追加されている. これに対して本論文では, 項の世界に区間変数という新しい対象表現とそれに関する特殊化を追加している. これによって, 区間変数を含む項領域での 2 つの新しい等価変換ルールが得られる. 具体的には, 区間変数を含む候補の制限ルールと区間変数を含む共通性による特殊化ルールである.

区間変数を含む項領域では, 区間変数とそれを変更する特殊化が追加されているため, 通常の項領域に比べて, 等価変換ルールの数が多くなる. このとき, 効率的なルールがシステムに導入される可能性がある.

さらに, 計算の対象表現とそれに関する特殊化の空間の違いから, 共通一般化が違ってくる. 次に示す例は, 区間変数を含む項領域では, 共通性による特殊化ルールによって $W : [1, 4]$ という区間変数の共通一般化が新たに発生することを示している.

例) $[[1, Z] \in [[1, 1], [1, 2], [1, 4]]] \rightarrow$

$$[[1, Z : [1, 4]] \in [[1, 1], [1, 2], [1, 4]]]$$

通常の項領域では, このアトムは何も変化しない. これは共通一般化が $[1, Z]$ になるため, 項領域では共通一般化によってこれ以上特殊化できない.

このように、区間変数によって、項の取りうる範囲をより詳細に表現できる。これは計算の高速化に貢献する。

7.2 情報伝播と連鎖的変換

区間変数を含む項領域では、通常の項領域に比べて連鎖的変換や情報の伝播が多く起き、より高速な計算を実現することができる。

区間変数を含む項領域では、区間変数の区間情報が同じ節の他のアトムに項に伝播する。区間情報の伝播の例を次に示す。

(1) 包含制約アトムのリストの要素に伝播する場合:

候補の制限ルールによってリストの要素が絞り込まれる場合がある。これによって、より強い共通性が発見され、共通性による特殊化が再び起こる。

例) $[X : [5, 7] \in [Z, Y : [4, 6], V : [3, 5]]] \rightarrow$
 $[X : [5, 7] \in [Z : [1, 4], Y : [4, 6], V : [3, 5]]] \rightarrow$
 $[X : [5, 7] \in [Y : [4, 6], V : [3, 5]]] \rightarrow$
 $[X : [5, 6] \in [Y : [4, 6], V : [3, 5]]]$

この例の 1 行目から 2 行目の変換は、同じ節の他のアトムで Z が $Z : [1, 4]$ に特殊化されたとき、その区間情報が伝播して起こる。2 行目から 3 行目の変換は、候補の制限ルールによって、 $Z : [1, 4]$ が削除されている。3 行目から 4 行目の変換は、共通性による特殊化によるものである。

通常の項領域では、例えば $[X \in [Z, Y]]$ は共通一般化によってこれ以上特殊化できない。

(2) 包含制約以外のアトムに項に伝播する場合:

例えば、区間変数の情報が算術計算のアトムに伝播すると、求める項の範囲を狭めることができる。

例) 節に $X \times Y = Z$ というアトムが存在するとする。同じ節の他のアトムで、例えば $X \rightarrow X : [-1, 2], Y \rightarrow Y : [1, 3]$ という特殊化が起こると、 $X \times Y = Z$ アトムの X, Y も区間情報伝播によって自動的に $X : [-1, 2], Y : [1, 3]$ になる。これによって、 Z は $Z : [-3, 6]$ になることが分かるので、 Z が取りうる数値範囲を限定することができる。通常の項領域では、 X, Y は変化しないので、このような効果は得られない。

7.3 実験データ

本論文で導入した手法の効果を、将棋の盤面を前にして、駒の動きに関する命令文を理解し実行するシステム [赤間 98a] で確認する。このときシステムは、与えられた文だけでなく、将棋の知識と盤面の情報をうまく使わなければならない。実験結果を表 1 に示す。これはシステム α の計算時間を 1 としたときの時間比である。

用いた例文は、

- (p1) 金を取れる歩を動かせ
- (p2) 7 六の金で取れ
- (p3) 飛車を取れる金で歩を取れ
- (p4) 先手の 7 六の飛車を取れる駒を動かせ

の 4 つである。

システム α は、通常の論理プログラムが用いる計算の対象表現、すなわち項だけを用いている。等価変換ルールとして unfold 変換に基づくルールだけを利用している [Tamaki 84]。システム β は、システム α に項領域での 2 つのルール (「候補の制限ルール」と「共通性による特殊化ルール」) を加え、これら 2 つのルールを優先適用した場合である [赤間 98a]。システム γ が本手法で、区間変数を導入し、区間変数を含む項領域での 2 つのルール (「候補の制限ルール」と「共通性による特殊化ルール」) を加え、これら 2 つのルールを優先適用した場合である。

この結果、システム γ は、システム α に対しては勿論、システム β に対しても計算時間が大幅に減少している。従って、区間変数を含む項領域での 2 つのルールが計算の高速化に貢献しているのは明らかである。

表 1 意味解釈における計算時間比

	system α	system β	system γ
(p1)	1	0.51	0.07
(p2)	1	0.62	0.03
(p3)	1	0.58	0.01
(p4)	1	0.63	0.002

8. む す び

本論文では、通常の項領域に区間変数を導入し、これを利用して、区間変数に関する包含制約の等価変換ルール (候補の制限ルールと共通性による特殊化ルール) をシステムに追加し、これらのルールの正当性を証明した。また、これらのルールを適用することで、システムの正当性を保持したまま問題解決の高速化が達成されることを示した。

通常の項領域に区間変数を導入し、区間変数を変更する特殊化を新しく追加することで、区間変数を含むさまざまな新しい等価変換ルールが得られる。本論文で述べた 2 つの等価変換ルールは、問題解決の高速化のために有用なルールであり、他の数多くの等価変換ルールと自由に組み合わせて計算の効率をあげることができる。

◇ 参 考 文 献 ◇

- [赤間 98a] 赤間清, 川口 雄一, 宮本衛市: 項領域における包含制約の等価変換, 人工知能学会誌, Vol.13, No.2, pp.112-120 (1998).
- [赤間 98b] 赤間清, 繁田良則, 宮本衛市: 区間変数の領域における等式制約の等価変換の正当性, 電子情報通信学会技術研究報告 SS98-33, pp.23-30 (1998).
- [赤間 98c] 赤間清, 繁田良則, 宮本衛市: 特殊化システムの拡張による知識表現系の変更, 人工知能学会誌, Vol 13, No 1, pp.131-138 (1998).
- [赤間 98d] 赤間清: 区間変数領域の等式制約解消特殊化, Hokkaido University Information Engineering Technical Report,

HIER-LI-9803 (1998).

- [Lloyd 87] Lloyd, J. W. : Foundations of Logic Programming, Second edition, Springer-Verlag (1987).
 [Pettorossi 94] Pettorossi, K. and Proietti, M. : Transformation of Logic Programs: Foundations and Techniques, Journal of Logic Programming, 19/20, pp.261–320 (1994).
 [Tamaki 84] Tamaki, H. and Sato, T. : Unfold/fold Transformation of Logic Programs, Proc. of 2nd ILPC, pp.127–138 (1984).

〔担当委員：木下哲男〕

2001 年 4 月 16 日 受理

◇ 付 録 ◇

- (1) 命題 1, 2 と 4 から 6, 補題 1, 2 の証明.
 本論文では区間変数を含む項領域を扱っているが, 証明は [赤間 98a] と同様にできる. [赤間 98a] 参照.
 (2) 命題 3 の証明. [赤間 98b] 参照.
 (3) 定理 1 の証明. [赤間 98d] 参照.
 (4) 定理 2 の証明.
 $\mathcal{M}(C) = \mathcal{M}(C')$ を証明するために, (I) $\mathcal{M}(C) \subseteq \mathcal{M}(C')$, (II) $\mathcal{M}(C) \supseteq \mathcal{M}(C')$ を証明する.
 (I) $(head, atom) \in \mathcal{M}(C)$ を仮定する.
 $\mathcal{M}(C)$ の定義より $head = head(C\theta)$, $atom = atom(C\theta)$, $const(C\theta) \subseteq Tconst(\Gamma)$, $C\theta \in Gclause(\Gamma)$ を満たす $\theta \in S$ が存在する. $const(C\theta) \subseteq Tconst(\Gamma)$ より, $B_i\theta \in Tconst(\Gamma)$ であり, 仮定より $x\theta \neq t_i\theta$ だから $B'_i\theta \in Tconst(\Gamma)$ が成り立つ. $const(C)$ と $const(C')$ の違いは B_i と B'_i だけであるから, $const(C'\theta) \subseteq Tconst(\Gamma)$ が得られる. C と C' の違いは B_i と B'_i だけであるから, $head = head(C'\theta)$, $atom = atom(C'\theta)$, $C'\theta \in Gclause(\Gamma)$ は明らかであるから $(head, atom) \in \mathcal{M}(C')$ が成り立つ.
 (II) 逆に $(head, atom) \in \mathcal{M}(C')$ を仮定する. 定義より $head = head(C'\theta)$, $atom = atom(C'\theta)$, $const(C'\theta) \subseteq Tconst(\Gamma)$, $C'\theta \in Gclause(\Gamma)$ を満たす $\theta \in S$ が存在する. θ は C' を基礎化するが, それと同じ効果 ($C'\theta = C'\sigma$) を与え, C' に含まれる変数だけを直接基礎項に具体化する基礎特殊化だけからなる特殊化を σ とする. C' に含まれる変数は C に含まれる変数の一部なので, σ は C にも適用可能である.
 σ は C' に含まれる変数だけを基礎化し, それ以外の変数は変更しないから, $C\sigma$ は正則であり, $C\sigma$ を基礎化する特殊化 ρ が存在する. この ρ を用いて $\theta_3 = \sigma \circ \rho$ とすると $C\theta_3 \in Gclause(\Gamma)$ が成り立つ. また, C と C' の違いは B_i と B'_i だけであるから, $head = head(C\theta_3)$ と $atom = atom(C\theta_3)$ が成り立つ. $B'_i\theta$ と $B_i\theta_3$ の違いは $t_i\theta_3$ だけであるから, 命題 6 により, $B_i\theta_3 \in Tconst(\Gamma)$ は $B'_i\theta \in Tconst(\Gamma)$ または $x\theta_3 = t_i\theta_3$ と等価である. 従って, $B'_i\theta \in Tconst(\Gamma)$ から $B_i\theta_3 \in Tconst(\Gamma)$ が導かれる.
 他の制約の部分は同じだから, $const(C\theta_3) \subseteq Tconst(\Gamma)$ が成り立ち, 結局 $(head, atom) \in \mathcal{M}(C)$ が導かれる.
 以上より, $\mathcal{M}(C) = \mathcal{M}(C')$ だから命題 1 より $\mathcal{M}(P \cup \{C\}) = \mathcal{M}(P \cup \{C'\})$ が成り立つ.
 (5) 補題 3 の証明.
 $[\pi = s]$ の等式制約解消特殊化 θ が存在する場合, $C'\{X/\pi\}$ の等式制約 $[\pi = s]$ を θ を適用して取り除くことができるので
 $\mathcal{M}(P \cup \{C'\{X/\pi\}\}) = \mathcal{M}(P \cup \{C''\{X/\pi\}\theta\})$
 が導かれる. また $\pi\theta = s\theta$ より,
 $\mathcal{M}(P \cup \{C''\{X/\pi\}\theta\}) = \mathcal{M}(P \cup \{C\theta\})$
 であるから,
 $\mathcal{M}(P \cup \{C'\{X/\pi\}\}) = \mathcal{M}(P \cup \{C\theta\})$
 が成り立つ.
 $[\pi = s]$ の等式制約解消特殊化が存在しない場合, $C'\{X/\pi\}$ は等式制約 $[\pi = s]$ を含むので, 節全体を消去でき,

$$\mathcal{M}(P \cup \{C'\{X/\pi\}\}) = \mathcal{M}(P)$$

が導かれる.

(6) 定理 3 の証明.

補題 1, 2 より,

$$\mathcal{M}(P \cup \{C\}) = \mathcal{M}(P \cup \{C'\}) = \mathcal{M}(P \cup \{C'\{X/\pi\}\})$$

これは, 補題 3 より, $[\pi = s]$ の等式制約解消特殊化 θ が存在する場合は $\mathcal{M}(P \cup \{C\theta\})$ と等しく, $[\pi = s]$ の等式制約解消特殊化が存在しない場合は $\mathcal{M}(P)$ と等しい.

著 者 紹 介

馬淵 浩司(正会員)

1992 年北海道大学大学院工学研究科情報工学専攻修士課程修了. 1995 年同大学院博士後期課程修了. 同年, 東和大学工学部電気工学科講師. 1998 年岩手県立大学ソフトウェア情報学部講師, 現在に至る. 博士(工学). 人工知能, 特に問題解決と学習の研究に従事.

赤間 清(正会員)

1973 年東京工業大学工学部制御工学科卒業. 1975 年同大学院修士課程修了. 1979 年同大学院博士課程単位修得退学. 同年, 同大学助手. 1981 年北海道大学文学部講師. 1989 年同大学工学部助教授. 1999 年より 同大学情報メディア教育研究総合センター教授として現在に至る. 人工知能, 知識処理, 等価変換に基づく問題解決の研究に従事. 工学博士. ソフトウェア科学会, 情報処理学会, 各会員.

繁田 良則

1995 年北海道大学大学院工学研究科情報工学専攻修士課程修了. 同年, 株式会社東芝に入社. 現在, システム LSI 開発センターにおいて LSI の研究開発に従事.

小池 英勝

1997 年北海学園大学工学部電子情報工学科卒業. 1999 年北海道大学大学院工学研究科システム情報工学専攻修士課程修了. 現在, 同大学院博士課程後期在学中. 電子情報通信学会学生会員. 人工知能, 人工知能用プログラム言語システム設計に興味を持つ.