知能と複雑系 114－3
（1998．10．2）

等価変換ルールの生成方法の理論的基礎

小池 英勝　　赤間 清　　宮本 衛市

北海道大学 大学院 システム情報工学専攻

札幌市北区北 13 条西 8 丁目．Tel. 011-706-6814

{koike,akama,miya}@complex.eng.hokudai.ac.jp

等価変換モデルとは、計算に等価変換ルールを用いて問題を解決する枠組みである。この計算モデルでは、問題解決のために効率の良いルールを準備することが重要である。我々は、等価変換モデルを基礎として確定節集合から等価変換ルールを生成する新しい問題を導入し、そのためのルール生成法をすでに提案した。この方法は、多くの問題記述を 1 つにまとめた形 (メタ問題記述) で表し、それをメタルールで変換していくことによって新しい等価変換ルールを生成する。本論文の目的は、このルール生成方法に理論的基礎を与えることである。

メタルール　　自動生成　　等価変換ルール　　メタ節

A Theoretical Foundation for Generating Equivalent Transformation Rules

Hidekatsu Koike　　Kiyoshi Akama　　Eiichi Miyamoto

Hokkaido University

North 13 West 8 Kita-ku Sapporo. Tel. 011-706-6814

{koike,akama,miya}@complex.eng.hokudai.ac.jp

"Equivalent Transformation model"(ET model) is a framework on which we solve problems using equivalent transformation rules. On this computation model it is important for solving problems to prepare for efficient rules. We have introduced a new problem of generating equivalent transformation rules from a given set of definite clauses based on ET model and proposed a method of generating rules. In this method we generate new equivalent transformation rules by transforming meta problem descriptions each of which represents many problem descriptions. The purpose of this paper is to give a theoretical foundation for the method.

meta rule, automatic generation, equivalent transformation rule, meta clause

1 はじめに

「等価変換パラダイム」は、「問題を等価変換によって簡単化して解く」という考えに基づき、幅広い問題記述に対して適用できる枠組みである [1]。等価変換パラダイムでは、問題記述 (確定節集合) を意味を保存したまま変換するルールを多数準備し、そのルールを用いて計算を行う。さらにこの枠組みでは、ルールを追加することにより、正当性を保ったままシステムの計算効率を改善することが出来る [2]。従って、ルールの自動生成の技術は非常に重要である。本論文では、[3] で提案された、確定節集合から等価変換ルールを生成する方法の正当性を保証する理論的枠組みを与える。

2 等価変換ルールの生成問題

2.1 確定節集合とその意味

確定節の記述には、標準的な Prolog の記法を用いる。確定節集合 (問題記述) P の意味 $\mathcal{M}(P)$ とは、P から定まる 基礎アトムの集合である [4]。本論文では紙面の都合のためその定義は割愛する。

2.2 等価変換ルールの例

等号関係を表す述語 *equal* の定義：

$EQ = \{C_{eq}\}$

$C_{eq} = (equal(X, X) \leftarrow)$

と通常の *append* 述語の定義：

$AP = \{C_{ap1}, C_{ap2}\}$

$C_{ap1} = (append([], Y, Y) \leftarrow)$

$C_{ap2} = (append([A|X], Y, [A|Z]) \leftarrow append(X, Y, Z))$

を仮定する。EQ と AP の和集合を D とする。

$D = EQ \cup AP$

このとき、節：

$C_1 = (p(Y, Z) \leftarrow append([], Y, Z), q(Y, Z))$

を

$C_2 = (p(Y, Z) \leftarrow equal(Y, Z), q(Y, Z))$

に直す変換は 等価変換である。正確には、*equal* 節や *append* 節を含まない任意の確定節集合 Q に対して、

$P_1 = D \cup Q \cup \{C_1\}$

$P_2 = D \cup Q \cup \{C_2\}$

と置けば、P_1 から P_2 への変換は等価変換である。すなわち、

$\mathcal{M}(P_1) = \mathcal{M}(P_2)$

が成り立つ。

この等価変換を引き起こすためのルールとして、たとえば 次のものを考える。

$append([], \&Y, \&Z) \rightarrow equal(\&Y, \&Z).$

これは、$append([], \&Y, \&Z)$ の形のボディアトム（節のボディに出現するアトム）が与えられたときにそれを $equal(\&Y, \&Z)$ の 形のボディアトムに直す変換を意味する。$\&Y$ や $\&Z$ などは 任意の項を代入できる変数であり、& 変数と呼ばれる。上記のルールは、代入 $\{\&Y/Y, \&Z/Z\}$ により、$append([], Y, Z) \rightarrow equal(Y, Z).$ に具体化され、C_1 節の中のボディアトム $append([], Y, Z)$ をアトム $equal(Y, Z)$ に 置き換える。

2.3 等価変換ルールの生成問題

多くの問題では、変換対象の問題記述（確定節集合）は、2つの部分に分けて考えることができる。たとえば、前節の例では、問題記述 P_1 は $D = EQ \cup AP$ と $Q \cup \{C_1\}$ という2つの部分に分けられる。しかも D に含まれる述語 (*equal* と *append*) は D の中だけで定義が完結しており、D 以外の部分には依存しない。

一般に、確定節集合 D が与えられたとする。D の節のヘッドにもボディにも含まれない述語を持つアトムをヘッドとする節だけからなる問題記述を考える。そのようなすべての問題記述の集合は D によって定まる。それを $outside(D)$ とする。Q を $outside(D)$ の元とし、D と Q の和集合

$P = D \cup Q$

を考える。このような条件を満たす問題記述 P 全体の集合は、D によって定まる。それを (独立部分) D に支持された問題記述と呼び、$support(D)$ と書く。明らかに、

$support(D) = \{D \cup Q \mid Q \in outside(D)\}$

である。

本論文では、ある確定節集合 D が与えられたとき、$P \in support(D)$ に対して適用できる等価変換ルールを生成するための基礎的な方法を提案する。ここで生成されるルールは、P の中の Q の部分を変更するルールであり、D の部分を変更することはない。

確定節集合 D を固定しても、等価変換ルールが一意に定まるわけではない。そこで、ルール生成問題をさらに具体的に定義する必要がある。ETC[1]で用いられているルールは、ある節のボディのあるアトムに注目して変換を行なうものであり、いろいろな変換対象アトムの形に応じて、なるべく効率の良いルールを備えておくことが高速な計算につながる [2]。そこで、本論文で扱うルール生成問題では、変換対象アトムのパターンを与え

[1] 人が記述したルールと制御のコードから実行ファイルを生成するコンパイラ。詳細は [5] にある。

て、その形のアトムを変換するルールを生成することを考える。

なお、変換対象アトムのパターンは、人間が与えてもよいし、システムが自動的に生成してもよい。その自動化に関しては 別の論文で議論する。

3 ルール生成の手順

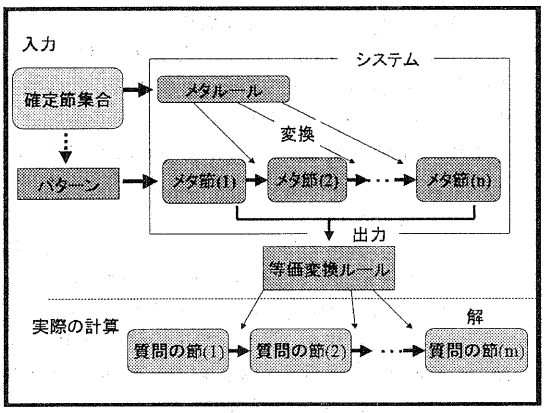

図1 ルール生成システムと入出力の関係

現在、本論文の理論に基づいたルール生成システムが試作されいる。図1はシステムとその入出力の流れを表したものである。図の左上が入力である。図の右上がシステムの内部である。また、図の下の部分は出力された等価変換ルールを用いた実際の計算を表している。本論文で扱う理論は図の右上のシステムの部分に適用されるものである。

3.1 手順の全体像

ルールの生成は以下の手順で行われる。

1. **[メタルールの準備]** 確定節集合からメタルールを生成する。
2. **[パターンの準備]** パターンを与え、変換対象のメタ節を作る。
3. **[変換]** メタ節にメタルールを適用して次々に等価変換して行く。
4. **[ルールの出力]** 得られた等価変換の結果の列を用いて新しい等価変換ルールを得る。

次節で、具体例を用いてルール生成のそれぞれの段階について説明する。

3.2 ルール生成の具体例

3.2.1 メタルールの準備

まず、システムに確定節集合が入力される。確定節集合の例は次のようなものである。

$append([], X, X) \leftarrow .$

$append([A|X], Y, [A|Z]) \leftarrow append(X, Y, Z).$

次に、確定節集合からメタルールを作る。始めに確定節集合を平坦化[2]と呼ばれる操作で書き換る。平坦化された確定節集合は次のようになる。

$append(X, Y, Z) \rightarrow equal(X, []), equal(Y, Z).$

$append(X, Y, Z) \rightarrow equal(X, [A|M]),$
$\qquad equal(Z, [A|N]),$
$\qquad append(M, Y, N).$

この確定節集合から得られるメタルールは次のようなものである。

$append(*X, *Y, *Z) \rightarrow equal(*X, []), equal(*Y, *Z);$
$\qquad \rightarrow equal(*X, [\%A|\%M]),$
$\qquad equal(*Z, [\%A|\%N]),$
$\qquad append(\%M, *Y, \%N).$

このルールは、ボディに $append(*X, *Y, *Z)$ の形のアトムを含むメタ節が与えられたとき、メタ節の中のその *append* アトムを

$equal(*X, []), equal(*Y, *Z)$

に置き換えたものと

$equal(*X, [\%A|\%M]), equal(*Z, [\%A|\%N]),$
$\qquad append(\%M, *Y, \%N)$

に置き換えたものの2つのメタ節に変換する。

3.2.2 パターンの準備

メタルールと同時にパターンを準備する。パターンから作られるメタ節の例は次のようなものである。

$h \leftarrow append(\&X, \&Y, []).$

パターンとして与えるものはメタ節のボディである。

3.2.3 変換

次に変換では与えられたメタルールを用いてメタ節を以下のように次々と変換していく。

1) $h \leftarrow append(\&X, \&Y, []).$

2) $h \leftarrow equal(\&X, []),$
$\qquad equal(\&Y, []).$
$\quad h \leftarrow equal(\&X, [\#A|\#M]),$
$\qquad equal([], [\#A|\#N]),$
$\qquad append(\#M, \&Y, \#N).$

3) $h \leftarrow equal(\&X, []), equal(\&Y, []).$

[2]ヘッドの引数を全て独立した変数にする。そして、引数のパターンを equal を用いて表しボディに移動する。

equal アトムを変換するためのメタルールはあらかじめシステムに用意しておく[3]。

3.2.4 ルールの出力

最後に 1) と 3) のボディを用いて等価変換ルール

$append(\&X, \&Y, []) \rightarrow equal(\&X, []), equal(\&Y, []).$

を出力する。

4 ルール生成のための理論

4.1 関係としての等価変換ルール

等価変換ルールの厳密な定義を導入する。等価変換ルールは、問題記述を別の問題記述に変換する写像とイメージされることが多い。しかし、一般には、等価変換ルールと問題記述から別の問題記述が一意に決まるわけではない。

たとえば アンフォールド変換では、内部変数 (ボディにのみ出現する変数) を持つ節で展開すれば、変換前の問題記述にはなかった新しい変数が導入される。この新しい変数は、一意的に決まるわけではない。そこで、理論的には アンフォールド変換を 写像ではなく関係 (変換前の問題記述と変換後の問題記述の２項組の集合) として扱うのが自然である。

このことを考慮して、本論文では、等価変換のためのルールを、一般に、問題記述の２項組の集合として扱う。

定義 1 アルファベット Δ 上の問題記述全体の集合を $\mathcal{P}(\Delta)$ とする。Δ 上の等価変換ルールとは、$\mathcal{P}(\Delta)\times\mathcal{P}(\Delta)$ の部分集合 *rule* で、

$(P_1, P_2) \in rule \rightarrow \mathcal{M}(P_1) = \mathcal{M}(P_2)$

を満たすものである。 □

ルール *rule* が問題記述 P_1 に適用されて問題記述 P_2 が得られるとは、

$(P_1, P_2) \in rule$

が成り立つことである。

4.2 メタ節の意味

メタ節とは、節を一般的に表したものである。たとえば「第一引数が空リストの append をボディに持つ節」などを表現できる。上の条件を満たす節は無限に存在する。つまりメタ節は無数の節を一つの表現で代表している。例として、メタ節：

$\hat{C} = (head \leftarrow p(\&X, \#Y, \#Z))$

を考える。このとき、

$\theta = \{\&X/f(b, X), \cdots\}$

$\sigma = \{\#Y/Y, \#Z/Z, \cdots\}$

の２つの代入は、メタ節のボディの & 変数と # 変数を具体化 (基礎化) [4] するのに使える。また、

$\rho = (h(X), [], [q(X, W)])$

の $h(X)$ を メタ節 $\hat{C}$ のヘッドを置き換えるべきアトム、[] と $[q(X, W)]$ をメタ節 $\hat{C}$ のボディの前後に追加するアトム列とすることができる。この結果、節

$h(X) \leftarrow p(f(b, X), Y, Z), q(X, W)$

が得られる。この変形が メタ節から 通常の確定節を生み出すための条件を満たすことは、変数 Y、Z が他の部分に出現しないことからわかる。

理論的な議論を行なうために、記号を定義しておく。メタ節 $\hat{C}$ から上記の方法で通常の節 C を得る操作を、

$C = ext(\hat{C}\theta\sigma, \rho)$

と書く。ただし、θ は & 変数を基礎化する代入であり、σ は # 変数を基礎化する代入である。ρ は、アトムと２つのアトム列の３項組であり、拡張と呼ぶ。アトムはヘッドを作るのに用い、アトム列は ボディの前後に追加するためのものである。写像 *ext* は 節のヘッドの書き換えとボディの追加の操作を表わす。

θ や σ や ρ が満たすべき制約を厳密に表現するために、記号を定義する。$Vars(X)$ は 文字 X を先頭にした変数全体の集合を表す。たとえば、$Vars(\&)$ は & 変数全体の集合であり、$Vars(\#)$ は # 変数全体の集合である。$Term(X)$ は 変数集合 X 中の変数を用いて作られる項全体の集合を表す[5] 。$map(X, Y)$ は X から Y への写像全体の集合を表す。$rename(X, Y)$ は、変数集合 X から 変数集合 Y への単射全体の集合を表す。$Atom(D, X)$ は 節集合 D に出現する述語を使わず、変数集合 X の変数を使って構成されるアトム全体の集合、$AtomSeq(X)$ は 変数集合 X の変数を使って構成されるアトム列全体の集合とする。

θ や σ や ρ が満たすべき制約とは、以下の条件を満たす V_1、V_2 が存在することである

(1) $V = V_1 \cup V_2$、

(2) $V_1 \cap V_2 = \emptyset$、

(3) $\theta \in map(Vars(\&), Term(V_1))$、

(4) $\sigma \in rename(Vars(\#), V_2)$、

(5) $\rho \in Atom(D, V_1) \times (AtomSeq(V_1))^2$

D が与えられたとき、この制約を満たす すべての３項組 (θ, σ, ρ) 全体の集合が定まる。それを $Dom(D)$ で表す。

ここでは、θ は $Vars(\&)$ から $Term(V_1)$ への写像として与えられている。メタアトムに対して θ を適用す

---

[3] *equal* を変換するメタルールを等式制約のためのメタルールという。詳細は [3] にある。

[4] ここでの基礎化とは、代入によって & 変数と # 変数を消すことであり、& 変数に項 (変数を含んでもよい) を代入し、# 変数には変数を代入する。その結果、通常の節が得られる。

[5] Δ の 定数や関数も使うが、ここでは それらを暗黙に仮定し、表には出さないで記述する。

るには、& 変数が出現するたびに θ で決まる項に置き換える。σ の適用も同様である。

この制約によって、次のことが達成されている。

1. # 変数に代入される変数は、節の他の部分（# 変数への代入で得られる以外の部分で、& 変数や ρ が決める部分）には出現しない。これは、V_1 と V_2 が互いに素だからである。

2. # 変数には、互いに相異なる変数が代入される。これは、*rename* の定義により、σ が 単射となるからである。

3. 得られる節のヘッドの述語は D には出現しない。これは、$Atom(D,X)$ の定義により、ρ の第一要素のアトムの述語が D に出現しないからである。

例をあげる。メタ節：

$\hat{C} = (h \leftarrow p(\&X, \#Y, \#Z))$

を考える。このとき、

$V_1 = \{X, W, \cdots\}$

$V_2 = \{Y, Z, \cdots\}$

$\theta = \{\&X/f(b,X), \cdots\}$

$\sigma = \{\#Y/Y, \#Z/Z, \cdots\}$

$\rho = (h(X), [], [q(X,W)])$

とすれば、上記の制約をすべて満たすことができて、

$ext(\hat{C}\theta\sigma, \rho)$

$= (h(X) \leftarrow p(f(b,X), Y, Z), q(X,W))$

が得られる。

4.3 メタ問題記述の意味と等価性

メタ節の集合をメタ問題記述という。メタ問題記述の全体の集合を MP で表す。本節では メタ問題記述の意味と等価性を定義する。

$(\theta, \sigma, \rho) \in Dom(D)$ を 1 つ固定する。メタ問題記述 M が与えられたとき、メタ節 $\hat{C}$ から通常の節 C を得る 前節の操作 $(C = ext(\hat{C}\theta\sigma, \rho))$ をメタ問題記述の要素である個々のメタ節に適用すれば、メタ問題記述 M を通常の問題記述 P_M に変換することができる。すなわち、

$P_M = \{C \mid C = ext(\hat{C}\theta\sigma, \rho) \mid \hat{C} \in M\}$

である。この M と P_M の関係も、メタ節 に対する記法を流用して、

$P_M = ext(M\theta\sigma, \rho)$

と書くことにする。

メタ問題記述が表している意味を 正確に定義するために、$Dom(D)$ の要素と、$outside(D)$ の要素を組にして考える。すなわち、$Dom(D)$ の要素 (θ, σ, ρ) と、$outside(D)$ の要素 Q から作られる 4 項組 $(\theta, \sigma, \rho, Q)$ 全体の集合を $Domain(D)$ で表す。

$$Domain(D) = \{(\theta, \sigma, \rho, Q) \mid (\theta, \sigma, \rho) \in Dom(D), \\ Q \in outside(D)\}.$$

定義 2 D を 節集合とする。M をメタ問題記述とする。D に関する M の意味とは、$Domain(D)$ の任意の要素 $(\theta, \sigma, \rho, Q)$ に対して

$\mathcal{M}(D \cup Q \cup ext(M\theta\sigma, \rho))$

を対応させる写像

$\overline{\mathcal{M}}_D(M) : Domain(D) \to 2^{\mathcal{G}}$

である。 □

メタ問題記述 の等価性を定義する。

定義 3 D を 節集合とする。M_1 と M_2 をメタ問題記述とする。M_1 と M_2 が D に関して等価であるとは、D に関するそれらの意味が等しいこと、すなわち、

$\overline{\mathcal{M}}_D(M_1) = \overline{\mathcal{M}}_D(M_2)$

が成り立つことである。 □

定義より、M_1 と M_2 が D に関して等価であるとは、可能な どの $(\theta, \sigma, \rho, Q) \in Domain(D)$ に対しても、$P_1 = D \cup Q \cup ext(M_1\theta\sigma, \rho)$ と $P_2 = D \cup Q \cup ext(M_2\theta\sigma, \rho)$ の 宣言的意味が等しいことであることがわかる。

4.4 メタ問題記述の２項組が作る集合

メタ問題記述の２項組が作る集合を定義する。

定義 4 D を 節集合とする。M_1 と M_2 をメタ問題記述とする。D に関して M_1 と M_2 が決める集合 $set_D(M_1, M_2)$ とは、$Domain(D)$ の任意の要素 $(\theta, \sigma, \rho, Q)$ から作られる２つの問題記述（確定節集合）：

$P_1 = D \cup Q \cup ext(M_1\theta\sigma, \rho)$,

$P_2 = D \cup Q \cup ext(M_2\theta\sigma, \rho)$

の２項組 (P_1, P_2) 全体の集合である。 □

定理 1 D を 節集合とする。M_1 と M_2 をメタ問題記述とする。M_1 と M_2 が D に関して等価であるならば、D に関して M_1 と M_2 が決める集合 $set_D(M_1, M_2)$ は、等価変換ルールである。

証明. $(P_1, P_2) \in set_D(M_1, M_2)$ と仮定する。定義により、$Domain(D)$ のある要素 $(\theta, \sigma, \rho, Q)$ に対して、

$P_1 = D \cup Q \cup ext(M_1\theta\sigma, \rho)$

$P_2 = D \cup Q \cup ext(M_2\theta\sigma, \rho)$

が成り立つ。M_1 と M_2 が D に関して等価であるから、

$\mathcal{M}(P_1) = \mathcal{M}(P_2)$

が導かれる。したがって、$set_D(M_1, M_2)$ は等価変換ルールである。 □

4.5 メタルールの定義と正当性

メタルールとは、メタ問題記述を書き換えるルールである。数学的には メタルール mr は、メタ問題記述の2項組の集合として扱われる。

$mr \subset MP \times MP$

メタルール mr によって、メタ問題記述 M_1 がメタ問題記述 M_2 に書き換えられるとは、

$(M_1, M_2) \in mr$

が成り立つことである。

メタルールの正当性を定義する。

定義 5 D を 節集合とする。メタルール mr が D に関して正当であるとは、mr によるメタ問題記述の すべての書き換え $(M_1, M_2) \in mr$ に対して、M_1 と M_2 が D に関して等価であること、すなわち、

$(M_1, M_2) \in mr \rightarrow \overline{\mathcal{M}}_D(M_1) = \overline{\mathcal{M}}_D(M_2)$

である。

4.6 ルール合成の方法と正当性

D を 節集合とする。メタアトムが与えられたとき、その形のアトムを書き換える等価変換ルールの合成方法を与える。

1. D に関して正当なメタルールを要素とする集合 $Rules$ を準備する。
2. メタアトム A が 与えられたとする。
3. メタ節 $h \leftarrow A$ を作る。そのメタ節だけからなるメタ問題記述を M_1 とする。
4. M_1 を $Rules$ の中の メタルールで 繰返し変形し、メタ問題記述 M_n を得る。
5. M_1 と M_n から、ルール $set_D(M_1, M_n)$ を得る。

得られた $set_D(M_1, M_n)$ が 等価変換ルールになることは、変換に使われた個々のメタルールの正当性によって保証できる。すなわち 次の定理が成り立つ。

定理 2 D を 節集合とする。$Rules$ を メタルールの集合とする。$Rules$ の要素であるメタルールがすべて D に関して正当であるならば、$Rules$ を用いて得られるメタ問題記述の書き換え

$M_1 \rightarrow M_2 \rightarrow M_3 \rightarrow \cdots \rightarrow M_n$

から得られる集合 $set_D(M_1, M_n)$ は等価変換ルールである。

証明. $Rules$ の書き換えルールの繰返し適用により、

$M_1 \rightarrow M_2 \rightarrow \cdots \rightarrow M_n$

が得られるとき、隣り合う書き換え

$M_i \rightarrow M_{i+1} \quad (i = 1, 2, \cdots, n-1)$

は D に関する意味を保存する。

$\overline{\mathcal{M}}_D(M_i) = \overline{\mathcal{M}}_D(M_{i+1}) \quad (i = 1, 2, \cdots, n-1)$

したがって、

$\overline{\mathcal{M}}_D(M_1) = \overline{\mathcal{M}}_D(M_n)$

が成り立つ。したがって、定理 1 より、それから得られるルール $set_D(M_1, M_n)$ も等価変換ルールである。 □

5 むすび

これまでに、我々は問題記述 (確定節集合) から等価変換ルールを生成するという新しい問題を導入し、そのための方法を提案した。本論文では、メタな表現を厳密に定義し、ルール生成のための一連の操作の正当性を定義するなど、この生成方法に理論的基礎を与えた。これは、ルール自動生成システム構築のための基礎となる。

今後の課題としては、パターンの自動生成や、生成されるルールから問題解決に有用なものをどのように選択するかなどの理論を作ることがある。

参考文献

[1] 赤間清, 繁田良則, 宮本衛市 : 論理プログラムの等価変換による問題解決の枠組, 人工知能学会誌, Vol.12, No.2, pp.90–99 (1997).

[2] 畑山満美子, 赤間清, 宮本衛市 : 等価変換ルールの追加による知識処理システムの改善, 人工知能学会誌 (1997)

[3] 小池英勝, 赤間清, 宮本衛市 : 仕様からの等価変換ルールの生成法, 電子情報通信学会技術研究報告 KBSE98-8, pp.33-40 (1998)

[4] Lloyd, J.W.: *Foundations of Logic Programming*, Second edition, Springer-Verlag (1987).

[5] 清水伴訓, 赤間清, 宮本衛市 : 等価変換プログラミング言語 ETC, 電子情報通信学会技術研究報告 SS 96-19, pp.9-16 (1996)

[6] 淵　一博, 古川　康一, 溝口　文雄 : プログラム変換, 共立出版 (1987)