



HOKKAIDO UNIVERSITY

Title	書換え系の性質の形式的検証に関する研究
Author(s)	能登, 正人
Degree Grantor	北海道大学
Degree Name	博士(工学)
Dissertation Number	甲第3856号
Issue Date	1996-03-25
DOI	https://doi.org/10.11501/3111981
Doc URL	https://hdl.handle.net/2115/51306
Type	doctoral thesis
File Information	000000297172.pdf



書換え系の性質の形式的検証に関する研究

能 登 正 人

書換え系の性質の形式的検証に関する研究

能 登 正 人

目次

1 序論	1
1.1 背景・目的	1
1.2 論文の構成	3
2 書換え系	5
2.1 抽象書換え系	6
2.1.1 性質	6
2.1.2 諸性質間の相互関係	14
2.1.3 準 Thue 系	19
2.2 項書換え系	20
2.2.1 諸定義	21
2.2.2 代入と文脈	23
2.2.3 正則項書換え系	29
3 項書換え系の停止性	34

3.1	停止性	35
3.1.1	停止性検証の重要性	35
3.1.2	停止性検証法	38
3.2	経路順序	39
3.2.1	辞書式順序	39
3.2.2	多重集合順序	40
3.2.3	再帰経路順序	42
3.2.4	種々の経路順序	43
3.2.5	ステータス付き再帰経路順序	45
4	拡張ステータスによる項書換え系の停止性検証	47
4.1	拡張ステータス	48
4.1.1	拡張ステータス	48
4.1.2	拡張ステータス付き再帰経路順序	49
4.2	例題	55
4.3	まとめ	56
5	抽象書換え系理論による講座配属アルゴリズムの完備性の解析	57
5.1	講座配属アルゴリズム	58
5.1.1	講座配属とは	58
5.1.2	アルゴリズム	59

5.2	抽象書換え系としての定式化	60
5.3	完備性	65
5.3.1	$\rightarrow$ の停止 (SN) 性	65
5.3.2	$\rightarrow$ の合流 (CR) 性	65
5.4	情報処理上の留意点	77
5.5	まとめ	79
6	結論	80
6.1	各章のまとめ	80
6.1.1	第1章	80
6.1.2	第2章	80
6.1.3	第3章	81
6.1.4	第4章	81
6.1.5	第5章	82
6.2	まとめ	82
	謝辞	84
	参考文献	85

目 次

2.1	抽象書換え系 $\mathcal{A}$	7
2.2	合流性	10
2.3	局所合流性	11
2.4	簡約グラフ $\mathcal{A}_1$	13
2.5	諸性質間の関係	15
2.6	Newman の補題の証明	17
3.1	経路順序間の包含関係	44
5.1	ケース I ($c_1 >_1 a_1 >_1 b_1 ; a_1 >_2 c_1 >_2 d_1$)	69
5.2	ケース III ($a_1 >_1 b_1, a_1 >_1 c_1 ; a_1 >_2 c_1 >_2 d_1$)	70
5.3	ケース IV ($a_1 >_1 b_1, a_1 >_1 c_1 ; c_1 >_2 a_1, c_1 >_2 d_1$)	71
5.4	ケース V ($a_1 >_1 b_1 ; a_1 >_2 d_1 >_2 c_1$)	72
5.5	ケース VI ($a_1 >_1 b_1 ; d_1 >_2 c_1, d_1 >_2 a_1$)	73
5.6	ケース VIII ($b_1 >_1 a_1 ; d_1 >_2 c_1$)	74

表 目 次

5.1 命題 5.6 の場合分け	68
5.2 p, q, r, s が 0 か否かの場合分け	76

第 1 章

序論

1.1 背景・目的

近年、大規模化の進むソフトウェアに対して、全体的な正しさを保証することは非常に難しい問題であるとされている。そこで、この問題を解決するために、新しいパラダイムに基づくプログラミング言語が次々と提案されている。これらの言語に共通する特徴は、根底にある論理体系が明確に定まっていることである。プログラミングを数学的な枠組みの中に置くことによって、ソフトウェアのメカニズムや性質、計算の意味を明らかにすることができ、厳密に議論することが可能である。

これらの言語のいくつかは、等式系とみなすことができる。等式は、数学をはじめとする科学のさまざまな場面で用いられる。あるときには、与えられた公理からある恒等式が論理的に導かれるかどうかを定めようとし、またあるときには方程式に対し

てその解を求めることをする。等式を用いたこのような推論はまた、さまざまなコンピュータ応用、たとえば、定理自動証明、代数的仕様記述、関数型言語などの理論的基礎として非常に重要である。

等式系に基づく計算モデルの一つとして、書換え系 [3, 6, 8, 9, 13, 24] (抽象書換え系および項書換え系) がある。書換え系は、書換え規則とよばれる左辺から右辺へ向き付けられた等式の集まりであり、与えられた式に対して、その一部をそれとパターン照合する左辺をもつ規則の右辺で置き換える操作を繰り返して式を最も簡単な形にするものである。たとえば、以下のブール代数を例にして考える。関数記号の集合 $\mathcal{F} = \{0, 1, \times\}$ 、書換え規則の集合 $\mathcal{R} = \{0 \times 0 \rightarrow 0, 1 \times 0 \rightarrow 0, 0 \times 1 \rightarrow 0, 1 \times 1 \rightarrow 1\}$ とすると、書換え系を定義することができる。0 を真理値の F 、1 を真 T 、 $\times$ を論理積 $\wedge$ と解釈することができ、この書換え系では、 $1 \times 0 \times 1 \rightarrow 0 \times 1 \rightarrow 0$ もしくは $1 \times 0 \times 1 \rightarrow 1 \times 0 \rightarrow 0$ の書換えができる。書換え系にはいくつかの性質があるが、特に重要な性質として停止性と合流性がある。停止性は無限に書換えが続かないことを意味し、合流性は計算結果が一意であることを意味する。停止性と合流性を満たす書換え系は完備であるという。停止性と合流性の検証は一般には決定不能な問題であるが、いくつかの十分条件が知られている。それらの条件を拡張する基礎研究やその応用研究はこの分野で最も重要な研究の一つである [5, 14, 16, 19, 21, 27]。

本論文では、書換え系のこれらの性質を形式的に検証する手法の拡張とその応用を述べている。具体的には、項書換え系の停止性に関して、従来の手法では形式的に検証できなかった問題を解くため、拡張ステータスという新しい概念を導入した手法を

提案し、その有効性を示す。また、合流性に関する応用として、講座配属アルゴリズムの解の一意性を、抽象書換え系理論を用いて形式的に検証する。

1.2 論文の構成

本論文の構成は、以下のとおりである。

第2章では、準備として、書換え系（抽象書換え系および項書換え系）の概要と諸性質について述べている。

第3章では、項書換え系の停止性検証法について述べている。項書換え系の停止性の検証は一般には決定不能な問題であるが、Dershowitzによって提案された単純化順序を用いることにより、一部の（しかし、実用上重要な多くの）項書換え系に対して停止性の形式的検証が可能であることを述べる。本論文では特に、経路順序とよばれる単純化順序の部分クラスに着目する。経路順序は、関数記号の集合上に定義される優先順位とよばれる半順序と項の構造に基づき、二項の相対的な重さ（大小関係）を比較するものである。この方法は、機械的な手順で二項の比較ができることから実装も容易で、したがって複雑な項にも十分対応し、また前提となる優先順位は項の構造から逆に推論できるなど、自動化という観点から最も有望な方法である。次に、経路順序のうちで特に代表的な（ステータス付き）再帰経路順序（RPOS）について検討する。

第4章では、ステータスの概念を拡張した“拡張ステータス”を提案し、それを適用

した経路順序について述べている。ステータスは各関数記号ごとにもっていて、比較すべき二つの項の最外（最左）の関数記号が同一のときの引数の比較方法を示している。ステータスの種類としては、多重集合順序、左辞書式順序、右辞書式順序の3種類あり、慣習的にそれぞれ *mult*, *left*, *right* の記号を用いる。従来の *left*, *right* ステータスは、引数の比較する順序を辞書式順序で1引数ずつ比較していたが、拡張ステータスを用いることにより、いくつかの引数をまとめて多重集合順序で比較できるようになっている。本論文では、従来の RPOS よりも検証に成功しやすい拡張ステータス付き再帰経路順序 (RPOES) を提案し、それが単純化順序であることを証明し、その有効性について述べている。

第5章では、書換え系の検証理論を用いた応用として、卒業研究の指導などのために学生を大学の各講座（研究室）に配属させる講座配属問題について述べている。本論文では、学生と講座の双方の優先度を考慮して配属を決定するアルゴリズムを考察する。このアルゴリズムには非決定性があるので、解（すなわち配属結果）が一意であるということは自明ではない。そこで、解の一意性を検証するために、このアルゴリズムの停止性と合流性（すなわち完備性）について抽象書換え系の理論を用いて形式的に証明する。

第6章では、本論文の結論および今後の展望について述べている。

第 2 章

書換え系

書換えという概念が意図するのは、等式の適用に方向性を与え、書換え規則として解釈することである。等式が項の非順序対 $l = r$ として定義されるのに対し、書換え規則は項の順序対として定義され、これは $l \rightarrow r$ のように表記される。書換え規則はその使い方において等式と異なっている。すなわち、左辺 l の例を右辺 r の対応する例で置き換えるという点は等式と同じであるが、しかし、逆に r の例を l の例で置き換えることは許されない。関数記号から作られる項の集合上の書換え規則の集合を項書換え系という。

項書換え系を定義するにあたって、書換の抽象的な概念を与えると都合がよい。これは、抽象書換え系とよばれ、項の概念とは独立した性質を明確に、かつ、一般的にとらえることができる。

本章では、抽象書換え系および項書換え系の概要と性質について述べる。

2.1 抽象書換え系

本節では、抽象書換え系の概要と諸性質について述べる。抽象書換え系を以下に定義する。

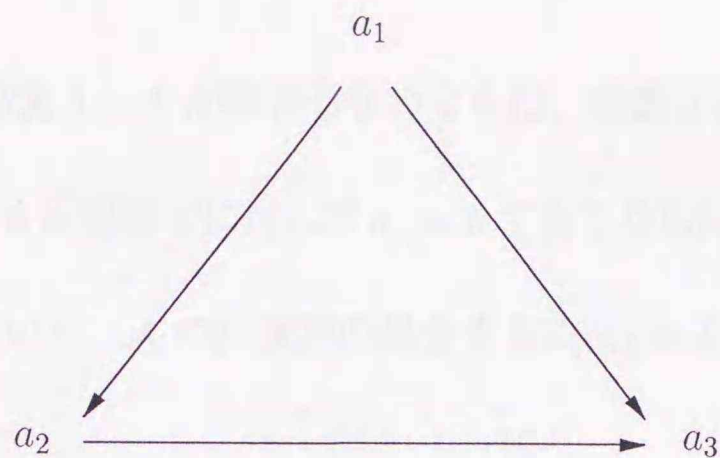
定義 2.1

抽象書換え系は、空でない集合 A と二項関係 $\rightarrow \subseteq A \times A$ から成る対 $\mathcal{A} = \langle A, \rightarrow \rangle$ である。 $(a, b) \in \rightarrow$ の代わりに、 $a \rightarrow b$ と書き、 a は b に1ステップで簡約 (reduction) されるという。

例えば、 $A = \{a_1, a_2, a_3\}$ 、 $\rightarrow = \{(a_1, a_2), (a_1, a_3), (a_2, a_3)\}$ とし、 $\mathcal{R} = \{a_1 \rightarrow a_2, a_1 \rightarrow a_3, a_2 \rightarrow a_3\}$ とすると、 $\mathcal{A} = \langle A, \mathcal{R} \rangle$ は、一つの抽象書換え系を定義する。 $\mathcal{A}$ の簡約グラフを図 2.1 に示す。このような簡約グラフは一般には非連結グラフになる。すべての要素が簡約グラフの節となるように簡約グラフを描くことによって、抽象書換え系を表現することができ、性質を理解しやすい。

2.1.1 性質

抽象書換え系にはいろいろな性質があるが、その諸性質を本節で定義する。もし抽象書換え系 $\mathcal{A} = \langle A, \rightarrow \rangle$ がある性質をもっているならば、関係 $\rightarrow$ もまたその性質をもつという。

図 2.1: 抽象書換え系 A

定義 2.2

- (1) $a, b \in A$ が同一要素であるとき $a \equiv b$ と表す.
- (2) $\rightarrow$ の反射推移閉包 (reflexive-transitive closure) を $\rightarrow^*$ で表す. もし $a \rightarrow^* b$ ならば, a は b に簡約されるという.
- (3) $\rightarrow$ の推移閉包 (transitive closure) を $\rightarrow^+$ で表す.
- (4) $\rightarrow$ の対称閉包 (symmetric closure) を $\leftrightarrow$ で表す. (すなわち, $\leftrightarrow = \rightarrow \cup \leftarrow$.)
- (5) $\leftrightarrow$ の反射推移閉包を $=$ で表す.

すなわち, $a \rightarrow^* b$ ($a \rightarrow^+ b$) は a から b への 0 ステップ (1 ステップ) 以上の簡約の列 $a \rightarrow \dots \rightarrow b$ が存在することを表す. もし, $b \rightarrow^* a$ ならば $a \leftarrow^* b$ と書くことも可能である. $a \leftarrow^* b$ に対しても同様である.

定義 2.3

- (1) もし $a \rightarrow b$ なる要素 $b \in A$ が存在しないならば, 要素 $a \in A$ は正規形 (*normal form*) である. ある正規形 b に対して $a \rightarrow^* b$ であるならば, 抽象書換え系 A は正規形をもつという. A の正規形の集合を $NF(A)$ あるいは $NF(\rightarrow)$ によって表す.
- (2) もし A のすべての要素が正規形をもつならば, 抽象書換え系 A は弱正規化性 (*weakly normalizing* : WN) を満たす.
- (3) もし A の要素の無限簡約列 $a_1 \rightarrow a_2 \rightarrow a_3 \rightarrow \dots$ が存在しないならば, 抽象書換え系 A は強正規化性 (*strongly normalizing* : SN) もしくは停止性 (*termination*) を満たす. 言い換えれば, すべての簡約経路がいつかは正規形で終わるということの意味する.

定義 2.4

- (1) 任意の $a, b, c \in A$ に対し, $b \leftarrow a \rightarrow c$ ならば $\exists d \in A, b \rightarrow^* d \leftarrow^* c$ であるとき, 抽象書換え系 A は合流性 (*confluence*) を満たす, もしくはチャーチ・ロッサ性 (*Church-Rosser* : CR) をもつという.
- (2) 任意の $a, b, c \in A$ に対し, $b \leftarrow a \rightarrow c$ ならば $\exists d \in A, b \rightarrow^* d \leftarrow^* c$ であるとき, 抽象書換え系 A は局所合流性 (*local confluence*) を満たす, もしくは弱チャーチ・ロッサ性 (*weakly Church-Rosser* : WCR) をもつという.

合流性と局所合流性を表現するよく知られた図式を図 2.2, 図 2.3 に示す.

合流性と局所合流性の相違点は, 合流性では a の複数回の書換えにより b, c になるが, 局所合流性では, a の 1 回の書換えにより b, c になるところである. また, 合流性と局所合流性の両方に共通する重要な性質は, a から b, c に一度分岐しても d で再び合流するという点である.

合流性に関して, 以下の命題 [21] がある.

命題 2.1

抽象書換え系 $\mathcal{A} = \langle A, \rightarrow \rangle$ が合流性を満たすとき, かつそのときに限り, $a = b$ を満たす A の要素のすべての対は共通の簡約形をもつ. (すなわち, $\forall a, b \in A, \exists c \in A, a = b \Rightarrow a \rightarrow^* c \leftarrow^* b$.)

証明

$\Rightarrow$ $\mathcal{A}$ は合流性を満たし $a = b$ と仮定し, $a = b$ の長さに関する帰納法によって a と b が共通の簡約形をもつことを証明する. まず, $a \equiv b$ の長さ 0 の場合は明らか. 次に, $a \rightarrow a_1 = b$ のときは, 帰納法の仮定から a_1 と b は共通の簡約形をもつ. これは a と b の共通簡約形になっている. 最後に, $a \leftarrow a_1 = b$ のときは, 帰納法の仮定より a_1 と b は共通の簡約形をもつから $a_1 \rightarrow^* c \leftarrow^* b$ である共通簡約形 c が存在する. このとき $a \leftarrow a_1 \rightarrow^* c$ が成立しており仮定より a と c は共通の簡約形 d をもつことがわかる. したがって, a と b は共通簡約形 d をもつ.

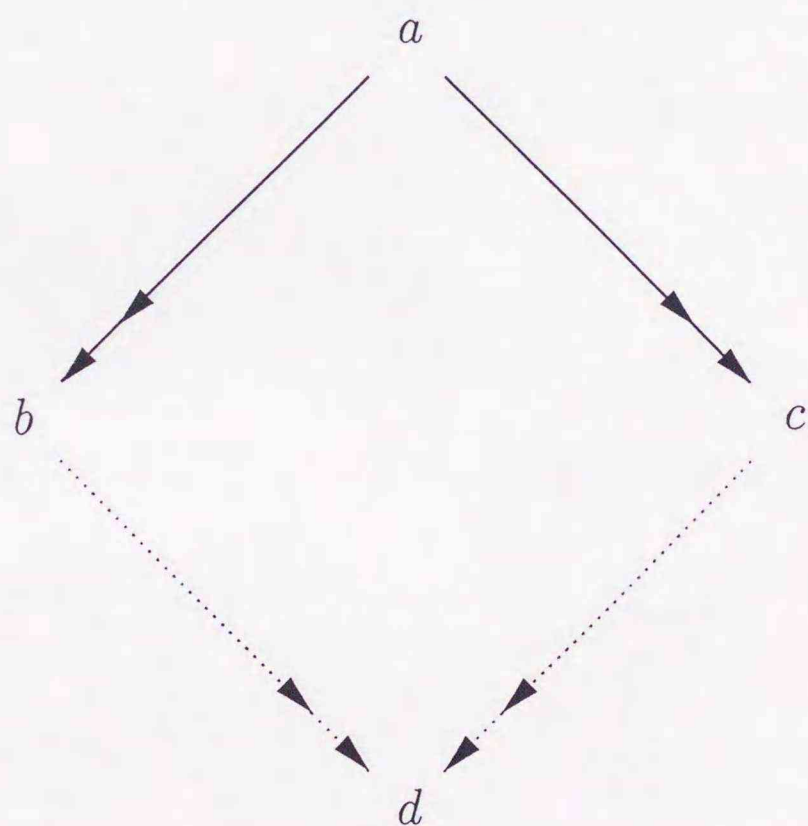


図 2.2: 合流性

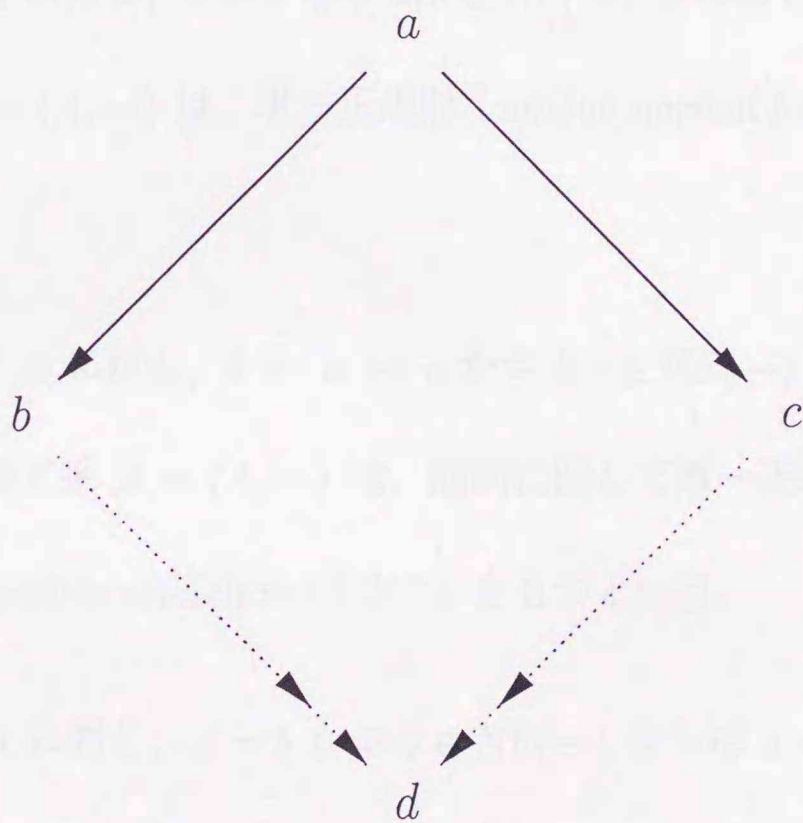


図 2.3: 局所合流性

$\Leftarrow b \Leftarrow a \rightarrow c$ を仮定する. $b = c$ であるから, 仮定より b と c の共通簡約形 d が存在する. ゆえに $\mathcal{A}$ は合流性を満たす.

□

定義 2.5

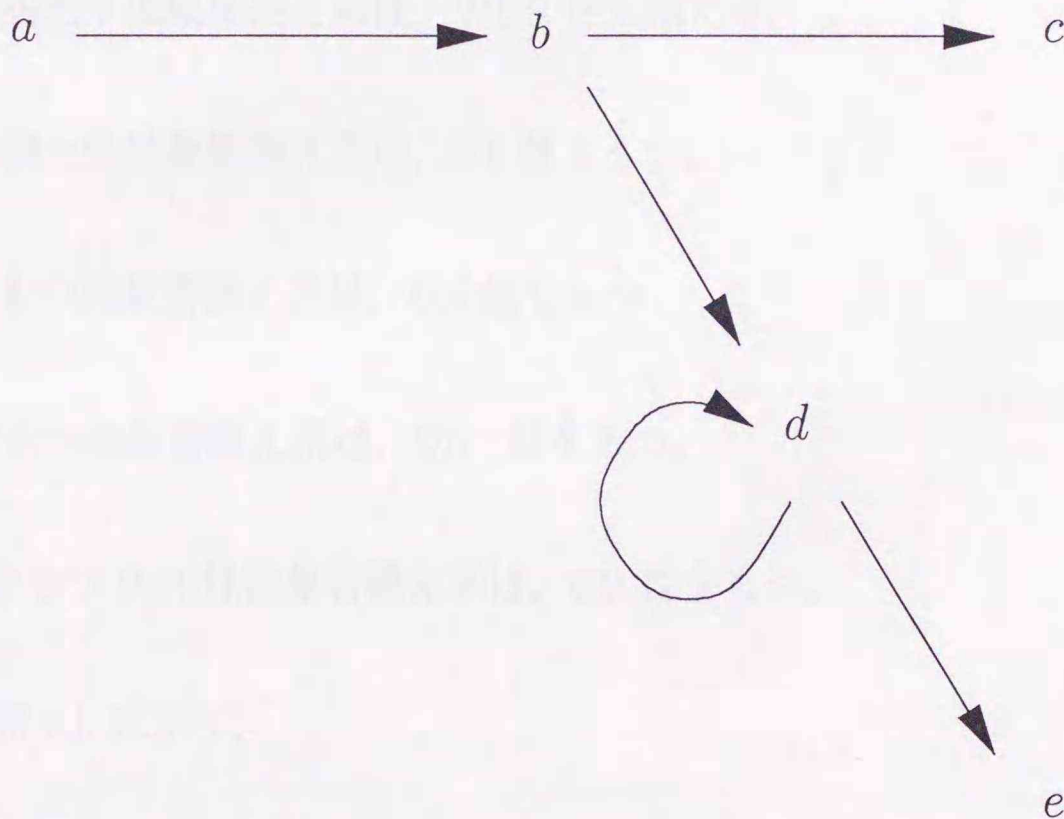
- (1) 任意の $a, b \in A$ に対し, $a = b$ かつ $a, b \in \text{NF}(\rightarrow)$ ならば $a \equiv b$ であるとき, 抽象書換え系 $\mathcal{A} = \langle A, \rightarrow \rangle$ は, 単一正規形 (*unique normal forms* : UN) をもつという.
- (2) 任意の $a, b, c \in A$ に対し, $b \Leftarrow a \rightarrow c$ かつ $b, c \in \text{NF}(\rightarrow)$ ならば $b = c$ であるとき, 抽象書換え系 $\mathcal{A} = \langle A, \rightarrow \rangle$ は, 簡約に関して単一正規形 (*unique normal forms with respect to reduction* : UN $\rightarrow$) をもつという.
- (3) 任意の $a, b \in A$ に対し, $a = b$ かつ $b \in \text{NF}(\rightarrow)$ ならば $a \rightarrow b$ であるとき, 抽象書換え系 $\mathcal{A} = \langle A, \rightarrow \rangle$ は, 正規形性 (*normal form property* : NF) をもつという.

例題 2.1

$\mathcal{A}_1 = \langle A_1, \rightarrow \rangle$ とする. ここで, $A_1 = \{a, b, c, d, e\}$, $\rightarrow = \{a \rightarrow b, b \rightarrow c, b \rightarrow d, d \rightarrow e, d \rightarrow e\}$ である. 図 2.4 は $\mathcal{A}_1$ の簡約グラフである.

c, e は正規形である. 正規形であるか否かは簡約グラフをみれば明らかである. 正規形とは, 簡約グラフで対応する節からいかなる弧もでていないものである. $\mathcal{A}_1$ は WN

であるが SN ではない。なぜならば、 $a \rightarrow b \rightarrow d \rightarrow d \rightarrow \dots$ という無限の簡約経路が存在する。 $\mathcal{A}_1$ は UN ではない。なぜならば、 $c \neq e$ である。 $\mathcal{A}_1$ は $\text{UN}^\rightarrow$ ではない。なぜならば、 a と b は二つの異なる正規形 c, e をもつ。 $\mathcal{A}_1$ は NF ではない。なぜならば、 e は正規形であり、 $c = e$ であるが、 $c \twoheadrightarrow e$ とはならない。

図 2.4: 簡約グラフ $\mathcal{A}_1$

定義 2.6

停止性と合流性を満たす抽象書換え系は完備 (*complete*) であるとよばれ、弱正規化性と合流性を満たす抽象書換え系は部分完備 (*semi-complete*) であるとよばれる。

2.1.2 諸性質間の相互関係

これまで導入した抽象書換え系の諸性質がどのように関連しているかを示すのが次の補題である.

補題 2.1

- (1) SN 性を満たす抽象書換え系は, WN 性も満たす.
- (2) CR 性を満たす抽象書換え系は, WCR 性も満たす.
- (3) CR 性を満たす抽象書換え系は, NF 性をもつ.
- (4) NF 性をもつ抽象書換え系は, UN 性をもつ.
- (5) UN 性をもつ抽象書換え系は, UN^{-} 性をもつ.
- (6) UN^{-} 性をもつ WN 性抽象書換え系は, CR 性をもつ.

補題 2.1 を図 2.5 に示す.

証明

- (1) SN と WN の定義より明らか.
- (2) CR と WCR の定義より明らか.
- (3) 任意の $a, b \in A$ に対し, $a = b$ かつ $b \in NF(A)$ とする. CR 性および命題 2.1 より, $a \rightarrow c \leftarrow b$ なる c が存在する. b は正規形なので, $b \equiv c$. したがって, $a \rightarrow b$ であり, A は NF 性をもつ.

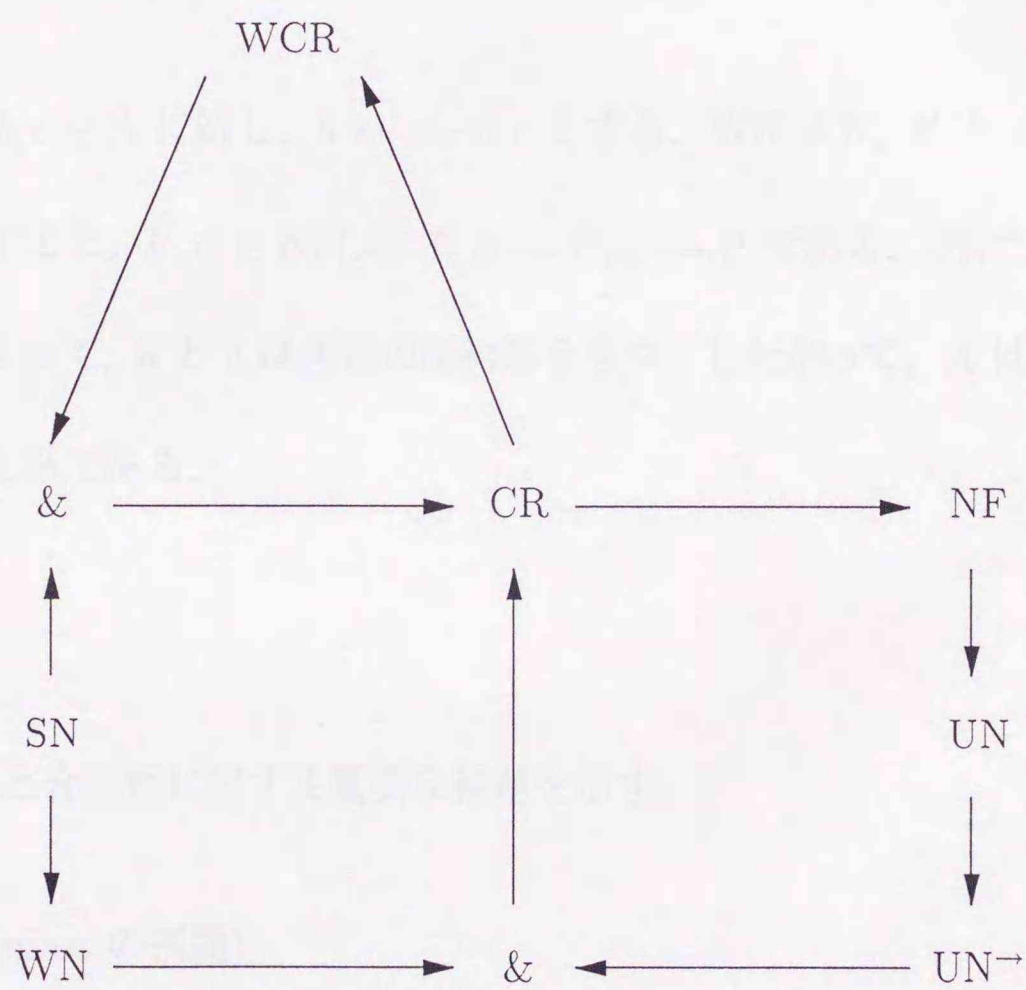


図 2.5: 諸性質間の関係

- (4) 任意の $a, b, c \in A$ に対し, $b \leftarrow a \rightarrow c$ かつ $b, c \in \text{NF}(A)$ とする. $=$ の定義より, $a = b, a = c$ である. $=$ の対称性, 推移性より, $b = c$. NF より, $b \rightarrow c$ (または, $c \rightarrow b$). ところが, b, c はともに正規形であるので, $b \equiv c$ でなければならない. ゆえに, A は UN 性をもつ.
- (5) UN と $\text{UN}^\rightarrow$ の定義より明らか.
- (6) 任意の $a, b, c \in A$ に対し, $b \leftarrow a \rightarrow c$ とする. WN より, b' と c' を b と c の正規形とすると, $b', c' \in \text{NF}(A)$ で $a \rightarrow b', a \rightarrow c'$ である. $\text{UN}^\rightarrow$ より, $b' \equiv c'$ である. よって, b と c は共通の簡約形をもつ. したがって, A は CR 性のある抽象書換え系である.

□

以下に停止性と合流性に関する重要な補題を示す.

補題 2.2 (Newman の補題)

SN 性を満たす抽象書換え系が WCR 性を満たすならば CR 性も満たす.

証明

Noetherian 帰納法に基づく図 2.6 の図式追跡法による.

Noetherian 帰納法とは $\rightarrow$ が SN のときに用いる帰納法である. $P(a)$ を A の要素 a に関する任意の命題とし, $\rightarrow$ を SN とする. Noetherian 帰納法は, 次の原理に基づく.

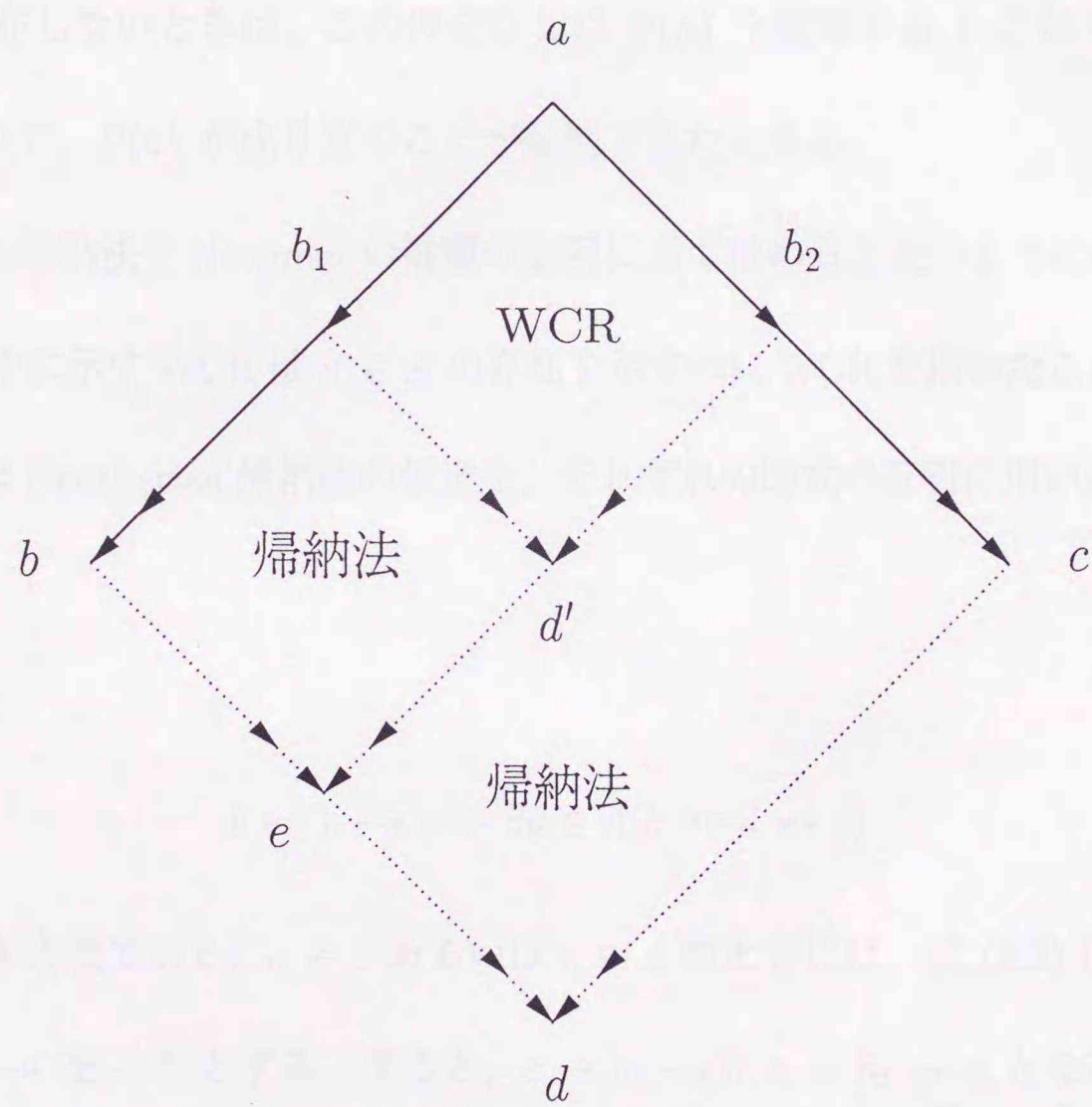


図 2.6: Newman の補題の証明

$$[\forall a \in A [\forall b \in A (a \rightarrow b \Rightarrow P(b))] \Rightarrow P(a)] \Rightarrow \forall a \in A P(a) \quad (2.1)$$

式 (2.1) は次のことを示している. 任意の $a \in A$ をとって考える. $a \rightarrow b$ を満たすすべての b について $P(b)$ が成り立つことを仮定して, $P(a)$ を証明する. (特に, $a \rightarrow b$ なる b が存在しないときは, この仮定なしに $P(a)$ を証明する.) これで, すべての $a \in A$ について, $P(a)$ が成り立つことが証明できたとする.

実際にこの帰納法を Newman の補題の証明に当てはめると次のようになる. (図 2.6 参照.) 図式中に示す WCR は $d' \in A$ の存在を示すのに WCR を用いたことを示し, 帰納法の仮定は Noetherian 帰納法の仮定を, それぞれの図式の証明に用いたことを示している.

$$b \leftarrow a \rightarrow c \Rightarrow \exists d \in A (b \rightarrow d \leftarrow c) \quad (2.2)$$

が証明すべき命題である. $a \equiv b$ あるいは $a \equiv c$ のときには, 式 (2.2) は明らかに成り立つので $\rightarrow$ を $\rightarrow^+$ とする. すると, $a \rightarrow b_1 \rightarrow b$, $a \rightarrow b_2 \rightarrow c$ となる b_1, b_2 が存在する. $b_1 \leftarrow a \rightarrow b_2$ を前提とすると, WCR よりある d' が存在して, $b_1 \rightarrow d' \leftarrow b_2$ となる. 次に, $b \leftarrow b_1 \rightarrow d'$ が成り立つので, b_1 について帰納法の仮定を用いて, $b \rightarrow e \leftarrow d'$ なる e が存在する. すると, $e \leftarrow b_2 \rightarrow c$ であるから b_2 について, 帰納法の仮定を用いて, $e \rightarrow d \leftarrow c$ なる d が存在することが証明される. $\square$

2.1.3 準 Thue 系

準 Thue 系 (*semi-Thue system* : STS) は文字列の書換え系である。表現の構成法が簡単で、容易に抽象書換え系へと変換できるので、ここで抽象書換え系と第 2.2 節で述べる項書換え系との橋渡しをする意味でこの書換え系について説明する。

X を任意のアルファベット、 X^* を X の要素の列からなる集合とする。 X^* は空列 ε を含む。 X 上の準 Thue 系は X と書換え規則の集合 $S = \{(u_i, v_i) | u_i, v_i \in X^*, i \in I\}$ の組 $\langle X, S \rangle$ で定義される。例えば、 $X = \{a, b\}$, $S = \{(ba, ab), (bb, c)\}$ とすると、 $\langle X, S \rangle$ は準 Thue 系である。準 Thue 系によって、例えば文字列 bab は、以下のように文字列 ac に書換えることができる。

$$bab \rightarrow abb \rightarrow ac$$

準 Thue 系から、次のように抽象書換え系を作ることができる。 $A = X^*$ とする。 A 上の二項関係 $\rightarrow$ を次のように定義する。

$$\forall x, y \in X^* \text{ に対して, } xuy \rightarrow xvy \Leftrightarrow (u, v) \in S$$

$A = \langle A, \rightarrow \rangle$ は抽象書換え系である。 A に関する諸性質はそのまま準 Thue 系 $\langle X, S \rangle$ の諸性質として引き継がれる。例えば、 $\langle X, S \rangle$ が CR であるとは A が CR であることである。準 Thue 系 $\langle \{a, b\}, \{(aba, \varepsilon)\} \rangle$ は $UN\rightarrow$ ではない。例えば、

$$\underline{ab}aba \rightarrow ba$$

$$a\underline{b}aba \rightarrow ab$$

である。

ここで、下線を引いた部分が書換えられる。一方、この準 Thue 系は SN 性をもつこともわかる。

準 Thue 系は項書換え系とともに、理論的な性質が深く研究されており、計算可能性の理論や言語理論へと応用されている。

2.2 項書換え系

書換え系の中でも、応用範囲が広く、しかも性質がよく知られているものの一つに、項書換え系 (*term rewriting system* : TRS) がある。項書換え系は、関数型プログラミング、自動推論、代数的仕様記述などと深くかかわっている。項書換え系の理論と応用は、ラムダ計算とコンビネータ論理に起源をもつ。これらの系は計算の概念を形式化するために半世紀前に開発された。項書換え系の多くの性質と証明テクニックは、これらの系に関する研究から始まっている。例えば、Church-Rosser 性は、Church と Rosser がラムダ計算と組み合わせ論理の一貫性を確立するのに、この性質を使用した後、名付けられた。また、完備化手続き (*completion procedure*) は、等式論理における語問題 (*word problem*) を解くために、Knuth と Bendix により考案された手続きである。語問題とは項 (語ともよばれる) 間の関係が等式の集合 $E = \{s_i = t_i | i \in I\}$ で与えられたときに、等式 $s = t$ が E から導けるかどうかを決定する問題である。

本節では、第 2.1 節で説明した抽象書換え系を項書換え系に適用し、その性質につ

いて述べる.

2.2.1 諸定義

定義 2.7

指標 (*signature*) もしくはアルファベット (*alphabet*) は関数記号 (*function symbol*) の集合 $\mathcal{F}$ である. すべての $f \in \mathcal{F}$ には, 引数の数を示す項数 (*arity*) とよばれる自然数が割り当ててある. 項数が0の関数記号は定数 (*constant*) とよばれる.

定義 2.8

関数記号の集合 $\mathcal{F}$ と $\mathcal{F} \cap \mathcal{V} = \emptyset$ である変数 (*variable*) $\mathcal{V}$ の可算無限集合から作られる項 (*term*) $T(\mathcal{F}, \mathcal{V})$ の集合は, 以下を満たす最小の集合である.

- (1) $\mathcal{V} \subset T(\mathcal{F}, \mathcal{V})$
- (2) もし $f \in \mathcal{F}$ が項数 n の関数記号で $t_1, \dots, t_n \in T(\mathcal{F}, \mathcal{V})$ ならば, $f(t_1, \dots, t_n) \in T(\mathcal{F}, \mathcal{V})$ である.

関数記号 C が定数のときはいつでも $C()$ の代わりに C と書く. 項 $t \in T(\mathcal{F}, \mathcal{V})$ の中にある変数の集合は $\mathcal{V}(t)$ によって示される. 変数を含まない項は基礎項 (*ground term*) もしくは閉じた項 (*closed term*) とよばれる. すべて基礎項である $T(\mathcal{F}, \mathcal{V})$ の部分集合は $T(\mathcal{F})$ もしくは $T(\mathcal{F}, \emptyset)$ と書ける.

定義 2.9

項書換え系は関数記号の集合 $\mathcal{F}$ と書換え規則の集合 $\mathcal{R} \subset T(\mathcal{F}, \mathcal{V}) \times T(\mathcal{F}, \mathcal{V})$ から成る対 $\langle \mathcal{F}, \mathcal{R} \rangle$ である。すべての書換え規則 (ℓ, r) は以下の二つの条件を満たす。

- (1) 左辺 ℓ は変数ではない。
- (2) 右辺 r で現れる変数は左辺 ℓ にも現れる。

今後は、書換え規則 (ℓ, r) は $\ell \rightarrow r$ と書くものとする。また、関数記号の集合であるアルファベットを宣言することなしに書換え規則の集合として項書換え系を示す。

例題 2.2

関数記号の集合を $\mathcal{F} = \{plus, mult, s, 0\}$ ，変数の集合を $\mathcal{V} = \{x, y\}$ とし、書換え規則の集合を $\mathcal{R}$ とする。なお、関数記号の項数はそれぞれ 2, 2, 1, 0 である。

$$\mathcal{R} = \left\{ \begin{array}{ll} r_1 : plus(x, 0) & \rightarrow x \\ r_2 : plus(x, s(y)) & \rightarrow s(plus(x, y)) \\ r_3 : mult(x, 0) & \rightarrow 0 \\ r_4 : mult(x, s(y)) & \rightarrow plus(mult(x, y), x) \end{array} \right.$$

ここで書換え規則の前につけた r_i ($i = 1, \dots, 4$) は書換え規則を参照するために、書換え規則につけた名前で書換え規則の一部ではない。このように定義された $\langle \mathcal{F}, \mathcal{R} \rangle$ は項書換え系である。

2.2.2 代入と文脈

項書換え系は、第 2.1 節の抽象書換え系に翻訳して理解する。そのために、代入、文脈という概念が必要となる。そこで、与えられた項書換え系にかかわる書換え関係を定義するために、最初に代入と文脈を紹介する。

定義 2.10 (代入)

代入 (*substitution*) σ は変数の集合 $\mathcal{V}$ から項の集合 $\mathcal{T}(\mathcal{F}, \mathcal{V})$ への関数 $\sigma : \mathcal{V} \mapsto \mathcal{T}(\mathcal{F}, \mathcal{V})$ である。このような関数を次のように変数と項の順序対 $x_i \mapsto s_i$ の集合で表す。¹

$$\sigma = \{x_1 \mapsto s_1, x_2 \mapsto s_2, \dots, x_n \mapsto s_n\}$$

特に、 $n = 0$ のとき σ を空代入 (*empty substitution*) という。対 $x_i \mapsto s_i$ は、変数 x_i は項 s_i に写像される (x_i maps to s_i) ということを表現している。すなわち、 $\sigma(x_i) = s_i$ である。また、 $x \notin \{x_1, x_2, \dots, x_n\}$ ならば $\sigma(x) = x$ とする。二つの代入 σ, τ は集合としては異なっているとしても、関数として等しければ $\sigma = \tau$ と書く。例えば、 $\{x \mapsto a, y \mapsto y\} = \{x \mapsto a\}$ 。

代入を次の再帰的な定義により、項の集合 $\mathcal{T}(\mathcal{F}, \mathcal{V})$ から項の集合 $\mathcal{T}(\mathcal{F}, \mathcal{V})$ への関数に拡張して使用する。また、慣習に従い、 $\sigma(t)$ の代わりに $t\sigma$ または t^σ という表記

¹歴史的な理由から、代入を

$$\sigma = \{s_1/x_1, s_2/x_2, \dots, s_n/x_n\}$$

と表す文献も多い。

法を用いる.

- t が変数のとき, $t\sigma$ は上の定義をそのまま使う. すなわち,

$$t\sigma = \begin{cases} s_i, & t = x_i \text{ のとき} \\ t, & x \notin \{x_1, \dots, x_n\} \text{ のとき} \end{cases}$$

- t が定数のとき, $t\sigma = t$.
- $t = f(t_1, \dots, t_n)$ のとき, $t\sigma = f(t_1\sigma, \dots, t_n\sigma)$.

また, $\sigma(t)$ を t のインスタンス (*instance*) とよぶ. 書換え規則の左辺のインスタンスは可簡約項 (*reducible expression* : *redex*) である.

例題 2.3

x, y, z を変数, c を定数, f, g を関数記号とする.

$$t = g(x, f(c, x, y), z), \sigma = \{x \mapsto h(y), y \mapsto a\}$$

のとき

$$x\sigma = h(y), y\sigma = a, z\sigma = z, c\sigma = c$$

であるから

$$\begin{aligned} t\sigma &= g(x, f(c, x, y), z)\sigma \\ &= g(x\sigma, f(c, x, y)\sigma, z\sigma) \\ &= g(x\sigma, f(c\sigma, x\sigma, y\sigma), z\sigma) \\ &= g(h(y), f(c, h(y), a), z) \end{aligned}$$

定義 2.11 (文脈)

文脈 (context) $C[\dots]$ は, いくつかの出現位置が空になっている項であり, この特別な空である項を $\square$ で表す.

もし $C[\dots]$ において $\square$ が n 個の出現位置をもつ文脈で $t_1, \dots, t_n$ が項であるならば, そのとき $C[t_1, \dots, t_n]$ は $t_1, \dots, t_n$ によって $\square$ の出現位置を左から右へ置き換えた結果である. $\square$ が一つである文脈は $C[\]$ によって記述される. もし文脈 $C[\]$ が存在してそのとき $t \equiv C[s]$ であるならば, 項 s は項 t の部分項 (subterm) である. ここで, $C[\]$ は項のメタ表記であり, 項そのものではない. また, $\mathcal{T}(\mathcal{F} \cup \{\square\}, \mathcal{V})$ は $\mathcal{C}(\mathcal{F}, \mathcal{V})$ と省略した表記法で書くことにする.

定義 2.12

項書換え系 $\langle \mathcal{F}, \mathcal{R} \rangle$ の書換え規則は以下のような $\mathcal{T}(\mathcal{F}, \mathcal{V})$ に関する書換え関係 $\rightarrow_{\mathcal{R}}$ を定義する. すなわち, もし, ある文脈 $C[\]$, 代入 σ , および書換え規則 $\ell \rightarrow r \in \mathcal{R}$ が存在し, $s \equiv C[\ell^\sigma], t \equiv C[r^\sigma]$ であるならば, $s \rightarrow_{\mathcal{R}} t$ である. s は可簡約項 ℓ^σ を簡約すること (contracting) によって t に簡約するという. $s \rightarrow_{\mathcal{R}} t$ を書換えステップもしくは簡約ステップとよぶ.

例題 2.4

関数記号の集合を $\mathcal{F} = \{0, S, A, M\}$, 変数の集合を $\mathcal{V} = \{x, y\}$ とし, 以下の書換え規則 $\mathcal{R}$ をもつ項書換え系を考える. なお, 関数記号の項数はそれぞれ 0, 1, 2, 2 である.

$$\mathcal{R} = \begin{cases} r_1 : A(0, x) & \rightarrow x \\ r_2 : A(S(x), y) & \rightarrow S(A(x, y)) \\ r_3 : M(0, x) & \rightarrow 0 \\ r_4 : M(S(x), y) & \rightarrow A(M(x, y), y) \end{cases}$$

$\mathcal{R}$ は後続関数 $S(x) = x + 1$, 加算 $A(x, y) = x + y$, および乗算 $M(x, y) = xy$ の再帰的な定義となっている. $M(S(S(0)), S(S(0)))$ は, 書換え規則 $r_1 \sim r_4$ を適用することにより以下の書換えステップの列を得る.

$$\begin{aligned} \underline{M(S(S(0)), S(S(0)))} &\rightarrow_{\mathcal{R}} \underline{A(M(S(0), S(S(0))), S(S(0)))} && (r_4) \\ &\rightarrow_{\mathcal{R}} A(A(\underline{M(0, S(S(0)))}, S(S(0))), S(S(0))) && (r_4) \\ &\rightarrow_{\mathcal{R}} A(A(\underline{0, S(S(0))}, S(S(0))) && (r_3) \\ &\rightarrow_{\mathcal{R}} \underline{A(S(S(0)), S(S(0)))} && (r_1) \\ &\rightarrow_{\mathcal{R}} S(\underline{A(S(0), S(S(0)))}) && (r_2) \\ &\rightarrow_{\mathcal{R}} S(S(\underline{A(0, S(S(0))})) && (r_2) \\ &\rightarrow_{\mathcal{R}} S(S(S(S(0)))) && (r_1) \end{aligned}$$

ここで, 下線を引いた項が可簡約項であり, r_i ($i = 1, \dots, 4$) の書換え規則により左辺を右辺に書換えている. このことにより最終的に正規形 $S(S(S(S(0))))$ を得る.

第 2.1 節で抽象書換え系に対して定義したすべての表記法は項書換え系に持ち越すことができる. それは, 抽象書換え系 $\langle T(\mathcal{F}, \mathcal{V}), \rightarrow_{\mathcal{R}} \rangle$ をすべての項書換え系 $\langle \mathcal{F}, \mathcal{R} \rangle$ と関連づけることによる.

例題 2.5

関数記号の集合を $\mathcal{F} = \{f, g\}$, 変数の集合を $\mathcal{V} = \{x\}$ とし, 以下の書換え規則 $\mathcal{R}$ をもつ項書換え系を考える. なお, 関数記号の項数は f, g とも 1 である.

$$\mathcal{R} = f(g(x)) \rightarrow g(g(f(f(x))))$$

この項書換え系は WN 性がない. 次の書換えステップの列は WN 性がないことを示す.

$$\begin{aligned} \underline{f(g(g(x)))} &\rightarrow_{\mathcal{R}} g(g(\underline{f(f(g(x)))})) \\ &\rightarrow_{\mathcal{R}} g(g(\underline{f(g(g(f(f(x))))})) \\ &\rightarrow_{\mathcal{R}} \dots \end{aligned}$$

この書換えにおいて, $f(g(g(t)))$ の形をした項が $g(g(\underline{f(g(g(f(f(x))))}))$ の中に真部分項 (下線の部分) として現れ, この書換えは無限に続いてしまうので $f(g(g(x)))$ は正規形をもたない. しかもこれ以外の書換えはない.

項書換え系は, 書換えの対象を基礎項に限定すると, 異なった性質をもつことがある. 次の例でこのことをみてもみる. 項書換え系をプログラムと考えたときに, 書換えの対象となる項を基礎項に限定するというのはごく自然なことである. データと普通によばれる計算の対象は基礎項である.

例題 2.6

関数記号の集合を $\mathcal{F} = \{0, s, f\}$, 変数の集合を $\mathcal{V} = \{x, y\}$ とし, 以下の書換え規則 $\mathcal{R}$ をもつ項書換え系を考える. なお, 関数記号の項数はそれぞれ 0, 1, 2 である.

$$\mathcal{R} = \begin{cases} r_1 : f(x, y) & \rightarrow f(y, x) \\ r_2 : f(s(x), y) & \rightarrow f(x, y) \\ r_3 : f(0, y) & \rightarrow 0 \end{cases}$$

書換え規則 r_1 を用いて $f(z, z)$ の書換えを試みると,

$$f(z, z) \rightarrow_{r_1} f(z, z) \rightarrow_{r_1} \cdots$$

となる。しかも $f(z, z)$ はこれ以外の書換えはできないので、 $f(z, z)$ は正規形をもたない。したがって、 $\mathcal{R}$ は WN 性がない。しかし、すべての基礎項は必ず正規形をもつ。 $f(s(0), s(0))$ を考えると,

$$f(s(0), s(0)) \rightarrow_{r_2} f(0, s(0)) \rightarrow_{r_3} 0$$

である。もちろん、 $f(s(0), s(0)) \rightarrow_{r_1} f(s(0), s(0)) \rightarrow_{r_1} \cdots$ であるから、すべての書換え列が有限というわけではない。

上の例が示すように、書換えの対象を基礎項とする項書換え系は、一般の項書換え系とは別に、その性質を論じる価値がある。なぜならば、書換えの対象である項を制限しなければこの例の項書換え系は WN 性はないが、書換えの対象を基礎項とするとこの項書換え系は WN 性があるといえるからである。

2.2.3 正則項書換え系

ここでは、項書換え系の中でも書換え規則の左辺に書ける項を限定した、正則項書換え系 (*regular term rewriting system*) とよばれるクラスの項書換え系を扱う。正則項書換え系は多くの応用に現れる項書換え系であり、しかも CR という (関数型プログラムと考えたときには) よい性質をもつ項書換え系である。一般の項書換え系について、それが CR であるかそうでないかを知ることは決定不能な問題であるので、最初から CR であることがわかっている項書換え系はいろいろな点で有用性が高い。

定義 2.13

線形 (*linear*) な項は同一変数を二つ以上含まない。

書換え規則 $\ell \rightarrow r$ は、もし (左辺) ℓ が線形であるならば左線形 (*left-linear*) であり、もし (右辺) r が線形ならば右線形 (*right-linear*) である。もしすべてのその書換え規則が左線形であるならば、項書換え系は左線形である。また、右線形項書換え系は右線形の書換え規則しかもたない。

単一化 (*unification*) はパターン照合を一般化したものであり、その結果として適切な代入を作り出す。以下では、項どうしの単一化を定義する。

定義 2.14 (単一化)

適当な代入 θ によって二つの項 s, t が同一、すなわち $s\theta \equiv t\theta$ となるとき、 s と t は単一化可能 (*unifiable*) であるといい、このとき用いた代入 θ を単一化代入 (*unifier*) という。

定義 2.15

σ はある二つの項 s, t の単一化代入であるとする. s, t の任意の単一化代入 θ に対して $\sigma \leq \theta$ であるとき, σ は s, t の最も一般的な単一化代入 (*most general unifier* : mgu) であるという.

次の例題において, 非左線形な項書換え系は一般的に CR 性を満たさないことを示す.

例題 2.7

以下の項書換え系を考える.

$$\mathcal{R}_1 = \begin{cases} r_1 : F(x, x) & \rightarrow A \\ r_2 : F(x, G(x)) & \rightarrow B \\ r_3 : C & \rightarrow G(C) \end{cases}$$

項 $F(C, C)$ は二つの正規形 A, B をもつ. すなわち, 書換え規則を適用することにより以下の 2 通りの書換えが可能となる.

$$A \leftarrow_{r_1} F(C, C) \rightarrow_{r_3} F(C, G(C)) \rightarrow_{r_2} B$$

よって, $\mathcal{R}_1$ は UN 性がない. ゆえに, それは CR 性がない.

次に, 以下の項書換え系は UN 性をもつ.

$$\mathcal{R}_2 = \begin{cases} r_1 : F(x, x) & \rightarrow A \\ r_2 : G(x) & \rightarrow F(x, G(x)) \\ r_3 : B & \rightarrow G(B) \end{cases}$$

この項書換え系により項 $G(B)$ は, $r_1 \sim r_3$ の書換え規則によりまず以下のように A に書換えることができる.

$$G(B) \rightarrow_{\mathcal{R}_2} F(B, G(B)) \quad (r_2)$$

$$\rightarrow_{\mathcal{R}_2} F(G(B), G(B)) \quad (r_3)$$

$$\rightarrow_{\mathcal{R}_2} A \quad (r_1)$$

ここで更に項 $G(B)$ は, 以下のように書換えることができる.

$$G(B) \rightarrow_{\mathcal{R}_2} G(G(B)) \quad (r_3)$$

$$\rightarrow_{\mathcal{R}_2} G(F(B, G(B))) \quad (r_2)$$

$$\rightarrow_{\mathcal{R}_2} G(F(G(B), G(B))) \quad (r_3)$$

$$\rightarrow_{\mathcal{R}_2} G(A) \quad (r_1)$$

$$\rightarrow_{\mathcal{R}_2} \dots$$

簡単に, A と $G(A)$ が CR 性がないということが示された. このように, $\mathcal{R}_2$ は CR 性がないといえる.

二つの書換え規則 $r_1: \ell \rightarrow r$ と $r_2: \ell' \rightarrow r'$ において, ℓ の部分項 ℓ_1 と ℓ' が単一化可能なとき, r_2 は r_1 に重なる (*overlapping*) という. もし, 二つの書換え規則が同じである場合には, 部分項として ℓ の真部分項のみを考えるものとする.

r_2 が r_1 に重なるのと, r_1 が r_2 に重なるというのでは意味が異なる. すなわち, 「に重なる」というのは非対称的な関係である. しかし, 日常生活では, このような非対称性を避けて, r_1 と r_2 は重なるといった表現を用いる. これは, r_2 が r_1 に, あるいは r_1 が r_2 に重なることを意味する. これからは, この「 r_1 と r_2 は重なる」という

表現を使うことにする。項書換え系は、任意の書換え規則どうしが重ならないときに、重なりがない (*non-overlapping*) という。

定義 2.16 (正則項書換え系)

左線形かつ重なりのない項書換え系を正則項書換え系という。

正則項書換え系の例を以下に示す。

例題 2.8

関数記号の集合を $\mathcal{F} = \{0, 1, s, f, g\}$ 、変数の集合を $\mathcal{V} = \{x\}$ とし、以下の書換え規則 $\mathcal{R}$ をもつ項書換え系は正則項書換え系である。なお、関数記号の項数はそれぞれ 0, 0, 1, 1, 2 である。

$$\mathcal{R} = \begin{cases} r_1 : f(0) \rightarrow 1 \\ r_2 : f(s(x)) \rightarrow g(s(x), f(x)) \end{cases}$$

重なりがあるために正則でない項書換え系を次の例で示す。

例題 2.9

関数記号の集合を $\mathcal{F} = \{0, f\}$ 、変数の集合を $\mathcal{V} = \{x, y\}$ とし、以下の書換え規則 $\mathcal{R}$ をもつ項書換え系を考える。なお、関数記号の項数はそれぞれ 0, 0, 3 である。

$$\mathcal{R} = \begin{cases} r_1 : f(0, x, y) \rightarrow x \\ r_2 : f(x, 1, y) \rightarrow y \end{cases}$$

$\mathcal{R}$ は左線形である。しかし、重なりをもつので正則でない。重なりを見るには、定義 2.14 を満足することを見ればよい。 r_1 と r_2 に現れる変数 x, y は、同じ変数であっ

でも、お互いどうしは無関係である。したがって、二つの書換え規則に現れる変数とともに変更して、ここでは $r_1 : f(0, x_1, y_1) \rightarrow x_1$, $r_2 : f(x_2, 1, y_2) \rightarrow y_2$ とする。代入 $\sigma = \{0/x_2, 1/x_1, y_2/y_1\}$ によって、 r_1 は r_2 と重なる。

正則な項書換え系は次の重要な性質をもつ [7].

定理 2.1

正則項書換え系は CR である。

第 3 章

項書換え系の停止性

項書換え系の停止性の検証は一般には決定不能な問題であるが、Dershowitz によって提案された単純化順序を用いることにより、一部の（しかし、実用上重要な多くの）項書換え系に対して停止性の形式的検証が可能である。単純化順序として経路順序とよばれるクラスのものはいくつか提案されている。例えば、（ステータス付き）再帰経路順序（RPOS）が代表的なものであり、それらは、関数記号の集合上に導入した優先順位とよばれる半順序と各関数記号ごとに割り当てた *mult*, *left*, *right* のいずれかのステータスに基づいて項の集合上の単純化順序を定義する。

本章では、項書換え系の停止性検証法（経路順序とよばれるクラスの単純化順序）について述べる。

3.1 停止性

本節では、項書換え系の停止性検証がなぜ困難であるかを例を用いて述べたあと、単純化順序を用いた停止性検証法について述べる。

3.1.1 停止性検証の重要性

一般にプログラムの正当性 (*correctness*) を検証するためには、次の二つの性質を示さなければならない。

- (1) 停止性 (*termination*): プログラムは必ず停止する。
- (2) 部分正当性 (*partial correctness*): プログラムが停止したならば、得られた最終結果は正しいものである。

項書換え系では部分正当性を示すことはしばしば容易である [3]。これは項書換え系では得られた最終結果が望まれた結果でない場合、さらに書換えが続行することを容易に示すことができるためである。したがって、項書換え系の正当性を示すためには停止性を示すことが重要なポイントとなる。

例題 3.1

次の3本の書換え規則からなる簡単な書換え系を考える.

$$\mathcal{R} = \begin{cases} r_1 : \text{white, red} \rightarrow \text{red, white} \\ r_2 : \text{blue, red} \rightarrow \text{red, blue} \\ r_3 : \text{blue, white} \rightarrow \text{white, blue} \end{cases}$$

この書換え系は「オランダ国旗」ゲームを行う. このゲームは赤, 青, 白のマール (基石) を不特定に並べた列が与えられたとき, 赤の基石群を左に寄せ, 青の基石群を右に寄せて, そして白の基石を中央に並べ換えるものである.

例えば, 最初の書換え規則は, 与えられた基石列の任意の隣接する二つの基石の左が白で右が赤ならば, 赤を左に, 白を右に置換えることができるということを示している.

与えられた初期列に関わらず, 上記の書換え規則を任意の順序で適用した結果が常に正しく並べ換えられた基石の列となることを示すことは難しくはない.

ある停止性の検証は青は白より大きく, 白は赤より大きいという順序に基づいて行なわれる. 各書換え規則は二つの基石を並べ換える. つまり, “大きい” 色をもつ左にある基石が “小さい” 色をもつ右の基石と置換えられる.

例題 3.2

次の選言標準型への項書換え系を考える. この例は項書換え系が停止するかどうか, そしてなぜ停止するかを決定することの難しさを例示する.

$$\mathcal{R} = \left\{ \begin{array}{ll} r_1 : --\alpha & \rightarrow \alpha \\ r_2 : -(\alpha + \beta) & \rightarrow -\alpha \times -\beta \\ r_3 : -(\alpha \times \beta) & \rightarrow -\alpha + -\beta \\ r_4 : \alpha \times (\beta + \gamma) & \rightarrow (\alpha \times \beta) + (\alpha \times \gamma) \\ r_5 : (\beta + \gamma) \times \alpha & \rightarrow (\beta \times \alpha) + (\gamma \times \alpha) \end{array} \right.$$

最初の変換規則 r_1 は二重否定を削除し、2, 3 本目の変換規則 r_2, r_3 は否定を論理和と論理積に変換するためにド・モルガンの法則を適用し、最後の 2 本の変換規則 r_4, r_5 は分配則を適用する。

この系が停止することは明らかではない。つまり、ある変換は項の長さを減じ、例えば、

$$-(0 \times (-1 + -1)) \rightarrow \cdots \rightarrow -0 + (1 \times 1)$$

一方、他の変換では項の長さが増える。例えば、

$$-(0 \times (1 + 1)) \rightarrow \cdots \rightarrow (((-0 \times 0) + (-0 \times -1)) + ((-1 \times -0) + (-1 \times -1)))$$

さらに、ある部分項への変換規則の適用はその部分項の構造に影響を与えるばかりでなく、同様にその部分項を含む項の構造にも影響を与える。

停止性の検証は、変換規則と部分項の非決定的な選択によって作られる多くの異なる変換列を考慮に入れなければならない。このため項変換系の停止性検証は非常に難しい問題である。

3.1.2 停止性検証法

項書換え系の停止性を検証する一般的な方法は存在しないが、いくつかの十分条件が知られている。本論文では、Dershowitz によって提案された単純化順序 (*simplification ordering*) を用いた停止性検証法を考える。

定義 3.1 (単純化順序)

項を $s = f(s_1, \dots, s_m)$, $t = g(t_1, \dots, t_n)$ とし、関数記号の集合 $\mathcal{F}$ のみから構成される項の集合を $\mathcal{T}(\mathcal{F})$ とする。 $\mathcal{T}(\mathcal{F})$ 上の半順序 $\succ$ (推移的かつ非反射的な二項関係) は、次の三つの性質を満たすならば単純化順序である。

$$(1) \ s \succ t \Rightarrow f(\dots, s, \dots) \succ f(\dots, t, \dots) \quad \text{置換}$$

$$(2) \ f(\dots, t, \dots) \succ t \quad \text{部分項}$$

$$(3) \ f(\dots, t, \dots) \succ f(\dots, \dots) \quad \text{削除}$$

なお、(3) の性質は関数記号が固定個の引数をもつとき必要がない。

単純化順序と以下の定理 [1, 2] を用いて項書換え系の停止性を検証できる。

定理 3.1 (停止定理)

項書換え系 $\mathcal{R} = \{l_i \rightarrow r_i \mid 1 \leq i \leq n\}$ は任意の代入 θ に対して、

$$l_i\theta \succ r_i\theta, \quad \text{for all } i \in \{1, \dots, n\}$$

となる単純化順序 $\succ$ が $\mathcal{T}(\mathcal{F})$ 上に存在するならば停止する。

単純化順序の最もよく知られた例は、Dershowitz [1] の再帰経路順序である。第 3.2 節において、再帰経路順序を含めた単純化順序について説明する。

3.2 経路順序

項書換え系の停止性検証を行うには、単純化順序を用いた方法がある。単純化順序として経路順序 (*path ordering*) とよばれるクラスの順序 [10, 12, 20] があり、二項の大小関係を比較する際にさまざまな方法がある。

単純化順序は部分項順序を含むので、もし、 s の真部分項が項 t より大きいとか等価ならば、 s は t よりも真に大きい。したがって、二つの項 s と t を比較するには、まず最初にそれぞれの根の記号を優先順位 (*precedence*) によって比較すればよい。 s の根の方が小さいとき、 s が t より大きいとすれば、それは唯一 s の部分項が t よりも大きい場合である。一方、 t の根の方が小さければ、 s が t より大きいためには s は t のすべての部分項より大きくなければならない。両者の根の記号が同じ優先順位をもつときには、何らかの再帰的な方法でその直接の部分項をそれぞれ比較する。このときの部分項を比較する方法により、異なる経路順序のクラスが得られる。

3.2.1 辞書式順序

S の要素 $a_i \in S$ ($i = 1, 2, \dots, m$) の列を $(a_1, a_2, \dots, a_m)$ と書く。 S の要素の有限列の集合を S^* とする。半順序 $(S, \succ)$ が与えられたとき、これを S^* 上の辞書式順序

$(S^*, \succ_{lex})$ に以下のように拡張できる.

定義 3.2 (辞書式順序)

$(a_1, a_2, \dots, a_m) \succ_{lex} (b_1, b_2, \dots, b_n)$ であることは以下の条件のいずれかを満たすときに限る.

$$(1) \quad m > 0 \quad \wedge \quad n = 0$$

$$(2) \quad a_1 \succ b_1$$

$$(3) \quad a_1 = b_1 \quad \wedge \quad (a_2, \dots, a_m) \succ_{lex} (b_2, \dots, b_n)$$

辞書式順序を用いた例を以下に示す. ただし, $\succ_{lex}$ は自然数上の辞書式順序とする.

例題 3.3

$$(3, 0, 1) \succ_{lex} (2, 3, 4)$$

例題 3.4

$$(2, 1, 1, 3) \succ_{lex} (2, 1, 1, 0)$$

3.2.2 多重集合順序

S 上での多重集合 (multiset) は, S の各要素の出現回数が意味をもつような, 順序のない有限の集まりである. 形式的には多重集合とは要素の全体 S から自然数への写

像であり、これはそれぞれの要素が現れる回数を与えている。 S 上のすべての有限な多重集合の集合を $\mathcal{M}(S)$ とする。任意の（厳格）半順序集合 $(S, \succ)$ が与えられたとき、 $\mathcal{M}(S)$ 上の多重集合順序 $(\mathcal{M}(S), \succcurlyeq)$ は以下に定義される。

定義 3.3 (多重集合順序)

集合 S 上の多重集合を $M, N \in \mathcal{M}(S)$ とする。以下のすべての条件を満たす有限な多重集合 $X, Y \in \mathcal{M}(S)$ が存在するとき（またそのときに限り） $M \succcurlyeq N$ である。

- $\{\} \neq X \subseteq M$
- $N = (M - X) \cup Y$
- $(\forall y \in Y)(\exists x \in X) x \succ y$

S の要素の多重集合間の順序 $\succcurlyeq$ は、多重集合の一つの要素を $\succ$ に関してより小さい有限個（0 を含む）の要素で置き換える、という操作の推移的閉包として定義される。

多重集合順序を用いた例を以下に示す。ただし、 $\gg$ は自然数上の多重集合順序とする。

例題 3.5

$$\{3\} \gg \{2, 2, 2\}$$

例題 3.6

$$\{3, 3, 4, 0\} \gg \{3, 2, 1, 2, 0, 4\}$$

Dershowitz と Manna による以下の重要な定理 [4] がある.

定理 3.2 (Dershowitz & Manna)

$(S, \succ)$ が半順序ならば $(\mathcal{M}(S), \succ)$ も半順序である.

3.2.3 再帰経路順序

経路順序の中でも定義の簡潔さや効率の面から再帰経路順序 (*recursive path ordering* : RPO) [1, 3, 9] がよく使用される. これは, 優先順位とよばれる $\mathcal{F}$ 上の半順序 $\succ$ を用いて, $\mathcal{T}(\mathcal{F}, \mathcal{V})$ 上の単純化順序 $\succ$ を定義するものである. その定義を以下に示す.

定義 3.4 (再帰経路順序)

$\succ$ を固定個の引数をもつ関数記号の集合 $\mathcal{F}$ 上の半順序とする. このとき項の集合 $\mathcal{T}(\mathcal{F})$ 上の再帰経路順序 $\succ_{RPO}$ は, 以下のように再帰的に定義される.

項を $s = f(s_1, \dots, s_m), t = g(t_1, \dots, t_n)$ とする. このとき $s \succ_{RPO} t$ であることは以下の条件を満たすときに限る.

- (1) $s_i \succeq_{RPO} t$ for some $i \in \{1, \dots, m\}$
- (2) $f \succ g \wedge s \succ_{RPO} t_j$ for all $j \in \{1, \dots, n\}$
- (3) $f = g \wedge \{s_1, \dots, s_m\} \succ_{RPO} \{t_1, \dots, t_n\}$

定理 3.3 (Dershowitz [1])

再帰経路順序は単純化順序である.

例題 3.7

25 ページの例題 2.4 の項書換え系 $\mathcal{R}$ について考える. 今, 優先順位を $M \succ A \succ S$ とすると以下を得る.

$$(1) x \succeq_{RPO} x \text{ より } A(0, x) \succ_{RPO} x$$

$$(2) A \succ S, A(S(x), y) \succ_{RPO} A(x, y) \text{ より } A(S(x), y) \succ_{RPO} S(A(x, y))$$

$$(3) 0 \succeq_{RPO} 0 \text{ より } M(0, x) \succ_{RPO} 0$$

$$(4) M \succ A, \{M(S(x), y)\} \succ_{RPO} \{M(x, y), y\} \text{ より } M(S(x), y) \succ_{RPO} A(M(x, y), y)$$

3.2.4 種々の経路順序

種々の経路順序を用いることで, 項書換え系の停止性検証が可能である. それらの経路順序は, Steinbach により各々の経路順序間の包含関係 (図 3.1) が示されている [25, 26].

図 3.1 において 13 種類の経路順序を示したが, KBO と KBOS は他の 11 種類とは包含関係がない. また, 矢印は, 矢印先の経路順序は矢印元の経路順序を包含していることを意味している. 例えば, PSDS は, RPO, RPOS, PSO, PSD のそれぞれ四つの経路順序を包含している.

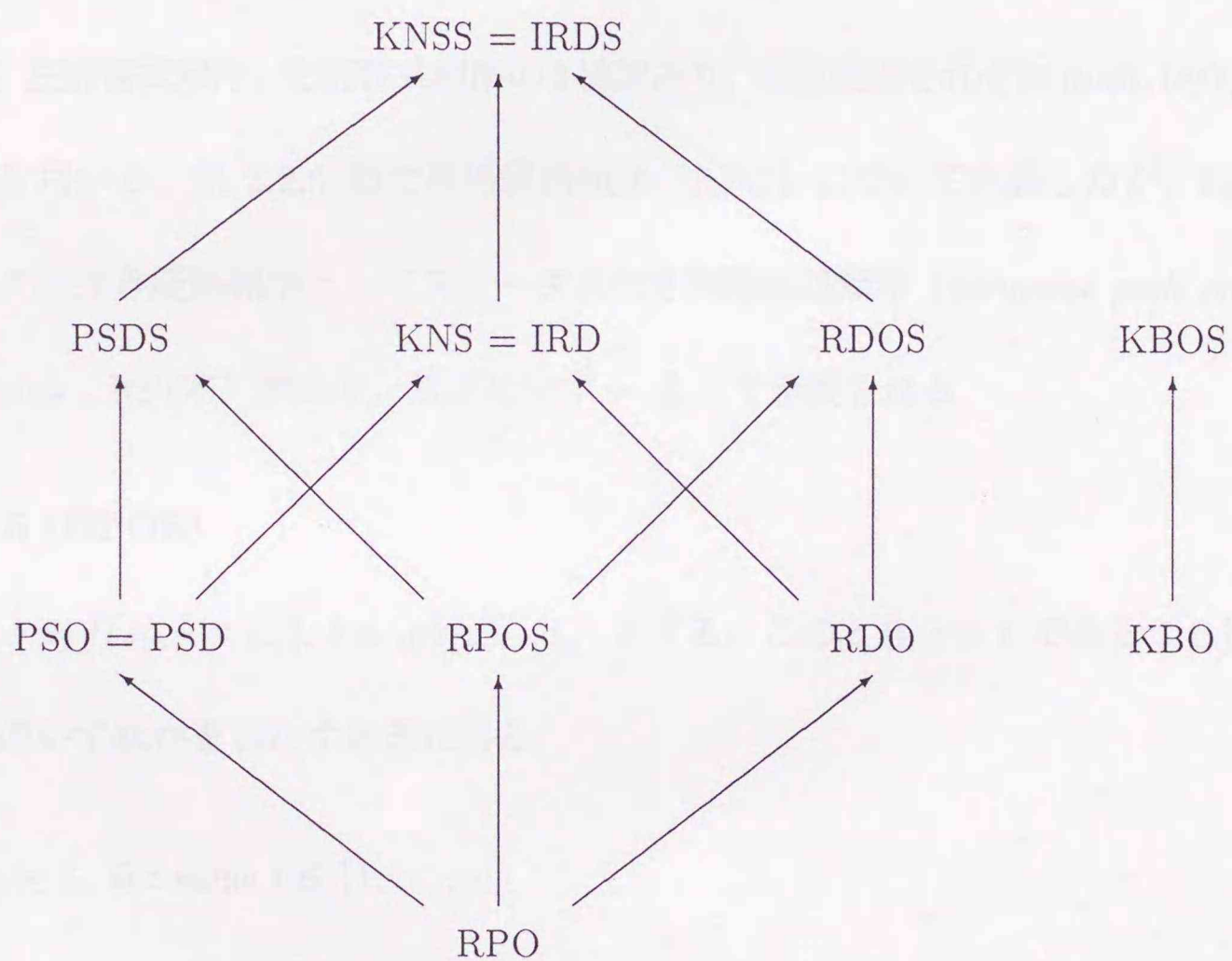


図 3.1: 経路順序間の包含関係

3.2.5 ステータス付き再帰経路順序

図 3.1 で示したようにステータス (*status*) とよばれるものによって、元の経路順序を拡張することができる。

ステータスは各関数記号ごとにもっていて、比較すべき二つの項の最外 (最左) の関数記号が同一のときの引数の比較方法を示している。ステータスの種類としては、多重集合順序、左辞書式順序、右辞書式順序の 3 種類あり、慣習的にそれぞれ *mult*, *left*, *right* の記号を用いる。第 3.2.3 節で再帰経路順序 (RPO) について定義したが、RPO のステータス付き経路順序としてステータス付き再帰経路順序 (*recursive path ordering with status* : RPOS) があり、以下に示す $\succ$ として定義される。

定義 3.5 (RPOS)

項を $s = f(s_1, \dots, s_m)$, $t = g(t_1, \dots, t_n)$ とする。このとき $s \succ t$ であることは以下の条件のいずれかを満たすときに限る。

- (1) $s_i \succeq t$ for some $i \in \{1, \dots, m\}$
- (2) $f \succ g \wedge s \succ t_j$ for all $j \in \{1, \dots, n\}$
- (3) $f = g \wedge f^\sigma = \text{mult} \wedge \{s_1, \dots, s_m\} \succ \{t_1, \dots, t_n\}$
- (4) $f = g \wedge f^\sigma = \text{left} \wedge (s_1, \dots, s_m) \succ_{\text{lex}} (t_1, \dots, t_n) \wedge s \succ t_j$ for all $j \in \{1, \dots, n\}$
- (5) $f = g \wedge f^\sigma = \text{right} \wedge (s_m, \dots, s_1) \succ_{\text{lex}} (t_n, \dots, t_1) \wedge$

$$s \succ t_j \quad \text{for all } j \in \{1, \dots, n\}$$

ただし, $\succ$ は $\mathcal{F}$ 上の優先順位, f^σ は f のステータス, $\succeq$ は $\succ$ と $=$ の和, $\succcurlyeq$ は $\succ$ の多重集合順序への拡張, $\succ_{lex}$ は $\succ$ の辞書式順序への拡張である.

定理 3.4

RPOS は単純化順序である.

第 4 章

拡張ステータスによる項書換え系の停止

性検証

本章では，経路順序のステータスの概念を拡張することにより，従来手法よりも検証に成功しやすい強力な停止性検証法を示す．さらに，この考えを RPOS に適用することにより，拡張ステータス付き再帰経路順序（RPOES）を構成し，それが単純化順序であることを証明し，その有効性について述べる．

4.1 拡張ステータス

4.1.1 拡張ステータス

従来の *left, right* ステータスは、引数の比較する順序を辞書式順序で1引数ずつ比較していたが、本論文では、いくつかの引数をまとめて多重集合順序で比較できるように、ステータスの概念を拡張する [22].

定義 4.1 (拡張ステータス)

- (1) 関数記号 f の拡張ステータス f^σ はある列 $(I_1, I_2, \dots, I_c)$ である。ただし、 f の項数が m のとき、 $I_1, I_2, \dots, I_c$ は集合 $\{1, 2, \dots, m\}$ の被覆 (cover) であるとする。すなわち

- $I_i \subseteq \{1, 2, \dots, m\}$ for all $i \in \{1, 2, \dots, c\}$
- $I_1 \cup I_2 \cup \dots \cup I_c = \{1, 2, \dots, m\}$

- (2) f の引数 $\{s_1, \dots, s_m\}$ に対して、 $f^\sigma(s_1, \dots, s_m)$ は、 f のステータス f^σ 中の整数 i を s_i で置き換えて得られる多重集合の列である。すなわち

$$f^\sigma(s_1, s_2, \dots, s_m) = (S_1, S_2, \dots, S_c)$$

ただし、 $f^\sigma = (I_1, I_2, \dots, I_c)$,

$$S_j = \{s_{i_1}, s_{i_2}, \dots, s_{i_k}\} \text{ if } I_j = \{i_1, i_2, \dots, i_k\} \text{ for all } j \in \{1, 2, \dots, c\}.$$

例題 4.1

項を $s = f(x, x, g(x))$ とし, f のステータスを $f^\sigma = (\{1, 3\}, \{2\})$ とする. このとき,

$$f^\sigma(x, x, g(x)) = (\{x, g(x)\}, \{x\})$$

である.

拡張ステータスを用いると, 従来のステータスもこの表記法により表すことができる. その例を以下に示す.

例題 4.2

5 引数の場合において, 従来の $mult, left, right$ ステータスは, それぞれ次のように書ける.

$$mult : (\{1, 2, 3, 4, 5\})$$

$$left : (\{1\}, \{2\}, \{3\}, \{4\}, \{5\})$$

$$right : (\{5\}, \{4\}, \{3\}, \{2\}, \{1\})$$

4.1.2 拡張ステータス付き再帰経路順序

定義 4.2 (汎関数)

項の集合を T とするとき, 汎関数 $\triangleright$ は, $2^{(T^2)}$ から $2^{(T^2)}$ への写像である.

すなわち, $\triangleright$ は T 上の二項関係 $R \subseteq T^2$ を T 上の二項関係 $R' = \triangleright(R) \subseteq T^2$ に変換する. 特に, 我々が興味があるのは T 上の半順序 $\succ$ であり, $\triangleright(\succ)$ のことを $\triangleright\succ$ と書く.

定義 4.3

項を $s = f(s_1, \dots, s_m)$, $t = g(t_1, \dots, t_n)$ とする. このとき $s \succ t$ であることは以下の条件のいずれかを満たすときに限る.

- (1) $s_i \succeq t$ for some $i \in \{1, \dots, m\}$
- (2) $f \triangleright g \wedge s \succ t_j$ for all $j \in \{1, \dots, n\}$
- (3) $f = g \wedge s \triangleright t \wedge s \succ t_j$ for all $j \in \{1, \dots, n\}$

定義 3.5 の RPOS は定義 4.3 において $\triangleright$ を f のステータスに依存して適切に定義することによって得られる.

補題 4.1 (Kamin & Lévy[11])

汎関数 $\triangleright$ が以下の (a) ~ (c) を満たすならば, $\succ$ は単純化順序である.

- (a) $\succ$ が半順序ならば $\triangleright$ も半順序である.
- (b) $\triangleright$ は連続的である.
- (c) $s \succ t \Rightarrow f(\dots, s, \dots) \triangleright f(\dots, t, \dots)$

なお, $\triangleright$ が「連続的である」とは, $s \triangleright t$ ならば $\succ$ の有限部分集合 $\succ_f$ が存在して $s \triangleright_f t$ であることを意味する. すなわち, $s \triangleright t$ であることをチェックするには, 有限個の対 $s' \succ t'$ のチェックで十分であることを意味する. これは通常, 非常に弱い条件であり, ほぼ自明に成り立つものである.

証明

$\succ = \bigcup_{n=0}^{\infty} \succ^n$ となるような関係の属 $\{\succ^0, \succ^1, \dots\}$ を以下のように定義する.

$s = f(s_1, \dots, s_m) \succ^{n+1} g(t_1, \dots, t_n) = t$ if and only if

(1) $s_i \succeq^{n+1} t$ for some $i \in \{1, \dots, m\}$

(2) $f \succ g \wedge s \succ^{n+1} t_j$ for all $j \in \{1, \dots, n\}$

(3) $f = g \wedge s \triangleright^n t \wedge s \succ^{n+1} t_j$ for all $j \in \{1, \dots, n\}$

$\succ$ が単純化順序であることを証明するには、 $\succ$ の推移性、非反射性、部分項性、単調性を示せばよい.

まず、部分項性について示す. $s \succeq s$ だから定義 4.3(1) より明らかに $f(\dots, s, \dots) \succ s$.

次に、半順序であること（推移性と非反射性）について示す. 証明は、 $\langle n, |s|, |t|, |u| \rangle$ のサイズに関する帰納法による.

推移性の場合: $s = f(s_1, \dots, s_m)$, $t = g(t_1, \dots, t_n)$, $u = h(u_1, \dots, u_p)$ とし、 $s \succ^n t$, $t \succ^n u$ ならば $s \succ^n u$ であることを示せばよい. 九つに場合分けして考えなければならないが、5 通りに集約して示すことができる.

- $s \succ^n t$ には (1) が適用される場合: ある $i \in \{1, \dots, m\}$ に対し、 $s_i \succeq^n t$, $t \succ^n u$.

帰納法の仮定より $s_i \succ^n u$. よって、 $s \succ^n u$.

- $s \succ^n t$ には (2) または (3), $t \succ^n u$ には (1) が適用される場合: 任意の $j \in \{1, \dots, n\}$ に対し、 $s \succ^n t_j$, ある $j \in \{1, \dots, m\}$ に対し、 $t_j \succ^n u$. 帰納法の仮定より $s \succ^n u$. よって、 $s \succ^n u$.

- $s \succ^n t$ には (2) または (3), $t \succ^n u$ には (2) が適用される場合: $f \geq g > h$ より $f > h$. 任意の $k \in \{1, \dots, p\}$ に対し, $t \succ^n u_k$. $s \succ^n t$ および帰納法の仮定より $s \succ^n u_k$. よって, $s \succ^n u$.
- $s \succ^n t$ には (2), $t \succ^n u$ には (3) が適用される場合: $f > g, g = h$ より $f > h$. 任意の $k \in \{1, \dots, p\}$ に対し, $t \succ^n u_k$. $s \succ^n t$ および帰納法の仮定より $s \succ^n u_k$. よって, $s \succ^n u$.
- $s \succ^n t$ には (3), $t \succ^n u$ には (3) が適用される場合: $f = g = h$ より $f = h$. 任意の $k \in \{1, \dots, p\}$ に対し, $t \succ^n u_k$, $s \triangleright \succ^{n-1} t \triangleright \succ^{n-1} u$. 帰納法の仮定より $\succ^{n-1}$ は半順序であるので, 条件 (a) より $\triangleright \succ$ も推移性を満たす. よって, $s \triangleright \succ^{n-1} u$. 仮定より $s \succ^n t \succ^n u_k$. 帰納法の仮定より $s \succ^n u_k$. ゆえに, $s \succ^n u$.

非反射性の場合: $s = f(s_1, \dots, s_m)$ とし, $s \succ^n s$ と仮定する. 以下の三つに場合分けして考えることができる.

- (1) の場合: ある $i \in \{1, \dots, m\}$ に対し, $s_i \succeq^n s$. また, $\succ$ の部分項性より $s \succ^n s_i$. よって, $\succ$ の推移性より $s_i \succ^n s_i$. これは, 帰納法の仮定に矛盾する.
- (2) の場合: $f > f$. これは, $>$ の非反射性に矛盾する.
- (3) の場合: 任意の $i \in \{1, \dots, m\}$ に対し, $s_i \succ^n s_i$ かつ $s \triangleright \succ^{n-1} s$. 帰納法の仮定より $\succ^{n-1}$ は半順序である. よって, 条件 (a) により $\triangleright \succ^{n-1}$ は非反射性を満たす.

最後に、単調性について示す。 $s \succ t$ と仮定すると、条件 (c) より $f(\dots, s, \dots) \triangleright\triangleright f(\dots, t, \dots)$ 。 $\succ$ の部分項性より $f(\dots, s, \dots) \succ s$ 。 $\succ$ の推移性より $f(\dots, s, \dots) \succ t$ 。 よって、定義 4.3(3) より $f(\dots, s, \dots) \succ f(\dots, t, \dots)$

□

定義 4.4 (RPOES)

拡張ステータス付き再帰経路順序 RPOES (*recursive path ordering with extended status*) は、定義 4.3 において $\triangleright$ を以下のように定義したものである。

$$s = f(s_1, \dots, s_m) \triangleright\triangleright g(t_1, \dots, t_n) = t \quad \Leftrightarrow$$

$$f = g \quad \wedge \quad S_1 = T_1, \dots, S_{i-1} = T_{i-1}, S_i \succ T_i \quad \text{for some } i \in \{1, \dots, c\}$$

$$\text{ただし, } f^\sigma(s_1, \dots, s_m) = (S_1, \dots, S_c), f^\sigma(t_1, \dots, t_n) = (T_1, \dots, T_c)$$

定理 4.1

RPOES は単純化順序である。

証明

本論文で定義した $\triangleright$ が補題 4.1 の条件 (a) ~ (c) を満たすことを示せばよい。

(a) $\succ$ が半順序であると仮定し、定理 3.2 より $\succ$ は半順序であることに注意して、

$\triangleright\triangleright$ の推移性と非反射性を示す。

推移性： $s \triangleright\triangleright t, t \triangleright\triangleright u$ ならば $s \triangleright\triangleright u$ であることを示せばよい。

$s = f(s_1, \dots, s_m)$, $t = g(t_1, \dots, t_n)$, $u = h(u_1, \dots, u_p)$ とし, $f^\sigma(s_1, \dots, s_m) = (S_1, \dots, S_c)$, $g^\sigma(t_1, \dots, t_n) = (T_1, \dots, T_d)$, $h^\sigma(u_1, \dots, u_p) = (U_1, \dots, U_e)$ とする.
 $s \triangleright t$ より $f = g$ (よって $m = n$, $f^\sigma = g^\sigma$, $c = d$) であり, ある $i \in \{1, \dots, c\}$ に対し, $S_1 = T_1, \dots, S_{i-1} = T_{i-1}, S_i \succ T_i$. 同様に, $t \triangleright u$ より $g = h$ (よって $n = p$, $g^\sigma = h^\sigma$, $d = e$) で, ある $j \in \{1, \dots, c\}$ に対し, $T_1 = U_1, \dots, T_{j-1} = U_{j-1}, T_j \succ U_j$. よって, $k = \min(i, j)$ とおくと, $\succ$ の推移性より, $f = h$, $S_1 = U_1, \dots, S_{k-1} = U_{k-1}, S_k \succ U_k$. よって, $s \triangleright u$.

非反射性: $s \triangleright s$ と仮定すると, ある i に対し, $S_1 = T_1, \dots, S_{i-1} = T_{i-1}, S_i \succ T_i$.

これは $\succ$ の非反射性に矛盾する.

(b) $s \triangleright t$ のチェックは高々 s と t の真部分項どうしの有限回の比較で十分である.

すなわち, $\succ_f = \{(s', t') \mid s', t' \text{ はそれぞれ } s, t \text{ の真部分項. } s' \succ t'\} \subseteq \succ$ なる有限の $\succ_f$ が存在する.

(c) $f(\dots, s, \dots)$ および $f(\dots, t, \dots)$ 中の s, t は第 k 番目の引数であるとし, $f^\sigma = (I_1, \dots, I_c)$ において k は I_i ($i \in \{1, \dots, c\}$) ではじめて現れるものとする. ($I_1, \dots, I_c$ の被覆条件よりそのような i の存在が保証される.) すなわち, $k \notin I_1 \cup I_2 \cup \dots \cup I_{i-1}, k \in I_i$. $f^\sigma(\dots, s, \dots) = (S_1, \dots, S_c)$, $f^\sigma(\dots, t, \dots) = (T_1, \dots, T_c)$ とおくと, $s \succ t$ ならば $S_1 = T_1, \dots, S_{i-1} = T_{i-1}, S_i \succ T_i$. (ここで, $S_i \succ T_i$ であるのは, $T_i = (S_i - \{s\}) \cup \{t\}$, $s \succ t$ より.) よって, $f(\dots, s, \dots) \triangleright f(\dots, t, \dots)$.

□

この証明からわかるように、被覆条件が満たされないとき、 $\succ$ は単純化順序にはならない。例えば、 $f^\sigma = (\{1\})$ とすると、 $1 \succ 0$ であっても $f(0,1) \not\succeq f(0,0)$ であり、置換の性質が成り立たない。

4.2 例題

RPOS では検証不可能だが、本論文で提案した RPOES では検証可能である例を以下に示す。

$$\mathcal{R} = \begin{cases} r_1 : f(0,0,2) \rightarrow f(1,0,0) \\ r_2 : f(1,2,0) \rightarrow f(0,1,2) \\ r_3 : f(2,1,0) \rightarrow f(0,2,1) \end{cases}$$

この系においては、優先順位と従来のステータスをどのように選んでも停止性を示すことができない。しかしながら、拡張ステータスを $(\{1,2,3\}, \{1\})$ とし、優先順位を $2 \succ 1 \succ 0$ とすることによって、いずれの書換え規則においても、左辺が右辺よりも大きくなり停止性を示すことができる。

$$\mathcal{R} = \begin{cases} r_1 : f(\text{cons}(x,y), z, w) \rightarrow f(z, y, \text{cons}(x, w)) \\ r_2 : f(\text{nil}, \text{cons}(x,y), z) \rightarrow f(\text{nil}, y, \text{cons}(x, z)) \end{cases}$$

この系は、 f の最初の二つの引数に与えられたリストの要素を交互に第3引数のリストの先頭に移動させるもので、拡張ステータスを $(\{1,2\}, \{3\})$ とすることによって停止性を示すことができる。

4.3 まとめ

本章では、従来のステータスを拡張することによって新しいステータスを提案した。また、拡張ステータスを適用した経路順序として RPOES を提案し、それが単純化順序であることを証明し、その有効性について述べた。明らかに、RPOES は RPOS を包含しており、より強力である。

今後の課題としては、優先順位や拡張ステータスを自動的に決定 [15, 18] すること、RPOS よりも強力な順序 (KNSS, RDOS など) [26] に拡張ステータスを適用し、より強力な拡張ステータス付き経路順序を提案することなどが挙げられる。

第 5 章

抽象書換え系理論による講座配属アルゴリズムの完備性の解析

講座配属とは、卒業研究の指導等のために学生を大学の各研究室に配属させることを指す。講座配属アルゴリズムは、学生と講座の双方の優先度を考慮して配属を決定するための手続きである。このアルゴリズムには非決定性があるので、解（すなわち、配属結果）が一意であるということは自明ではない。

本章では、解の一意性を検証するために、抽象書換え系の理論を用いてこのアルゴリズムの停止性および合流性（すなわち完備性）を形式的に証明する。

5.1 講座配属アルゴリズム

5.1.1 講座配属とは

講座配属とは、卒業研究の指導等のために学生を大学の各研究室に配属させることを指す。講座配属アルゴリズムは、 m 人の学生の各々を n 個の講座のうちの一つに配属するための手続きである [23]。本論文で考察するアルゴリズムは、著者らが実際に用いたアルゴリズムを、理論的考察をしやすいように、単純化かつ抽象化したもので、基本的には、講座と学生の双方の希望に沿って配属を行うものである。この際、配属先の講座には定員があるので、配属希望者数が定員を超過した講座が存在する場合には調整の必要がある。講座配属のための準備（前提条件）を以下に示す。

(1) 希望講座リスト

各学生は、配属先として希望する順番にすべての講座を並べた希望講座リストを提出する。

(2) 講座ごとの学生順位

各講座 i ごとに学生に優先順位を付け、その結果得られた順序を $>_i$ (i は講座) で表す。 $x >_i y$ は、講座 i への配属に関して学生 x は 学生 y より優先順位が低いことを意味する。

(3) 定員配分

各講座ごとに定員は決まっているものとし、定員の合計は学生数以上である。

以上の準備のもとで、講座配属のアルゴリズムを第 5.1.2 節で示す。

5.1.2 アルゴリズム

1. 全学生を第一希望どおりに仮配属させる。
2. 仮配属学生数が定員を越えている講座が存在する間、以下 (2.1 ~ 2.3) を繰り返す。

2.1 そのような講座 i を任意に一つ選択し、その講座に仮配属されている学生の中から、定員超過分の学生を順序 $>_i$ のもとで優先順位が低い順に選ぶ。
(ただし、幸いにも選ばれなかった (定員分の) 学生の配属がここで決定するわけではない。)

2.2 ここで選ばれた学生の希望講座リストから第一希望を削除し、第二希望以下を順次繰り上げて新たに第一希望以下の希望講座リストとする。

2.3 それらの学生を (新たな) 第一希望どおりに仮配属させる。

3. 全員の仮配属を正式な配属と決定する。

ステップ 2.1 ~ 2.3 での調整によりある講座 i の仮配属学生数が一時的に定員と等しくなっても、その後、他講座 j の調整の結果、講座 j から講座 i に仮配属学生が移動してきて、再度、講座 i を調整する必要が有り得ることに注意されたい。なお、ス

テップ 2.1 で選択する講座は任意のものでよいのでこのアルゴリズムには非決定性がある。もし、この選択の仕方によって最終的に解（配属結果）が違うと、このアルゴリズムには不備があると考えられる。また、解が選択に依存する場合、どのような選択をすべきかを適切に設計するのは難しい。したがって、調整の必要のある講座が複数個存在するとき、どの講座からどのような順番で調整しても得られる結果が同じであることが望まれる。しかし、このことが成り立つかどうかは自明ではない。本章では、抽象書換え系の理論を用いてこのアルゴリズムの停止性および解の一意性（すなわち完備性）を証明する。

5.2 抽象書換え系としての定式化

本章の目的は、講座配属アルゴリズムを抽象書換え系としてモデル化し、その SN 性と UN $\rightarrow$ 性を示すことである。次の命題は、UN $\rightarrow$ 性を示すには、CR 性を示すことが十分である [3, 9] ことを示している。

命題 5.1

CR 性をもつ抽象書換え系は、UN $\rightarrow$ 性を満たす。

要するに、抽象書換え系が完備（SN かつ CR）であることを示せば本章の目的が達成される。補題 2.2（Newman の補題）は、SN 性をもつ系の CR 性を示すためのよく知られた十分条件を述べている。

問題を一般的に扱うために次の記号を用いる。

$S = \{S_1, \dots, S_m\}$: 学生の集合

$L = \{L_1, \dots, L_n\}$: 講座の集合

各講座 i ごとに学生に優先順位を付け、その結果得られた順序を $>_i$ (i は講座) で表す. $S_a >_i S_b$ は、講座 i への配属に関して学生 S_a は 学生 S_b より優先順位が低いことを意味する. $>_i$ は厳格全順序 (*strict total ordering*) とする. すなわち、任意の学生 S_a, S_b, S_c に対し、 $>_i$ は以下の条件を満たす.

- $S_a \not>_i S_a$ (非反射性)
- $S_a >_i S_b$ かつ $S_b >_i S_c$ ならば $S_a >_i S_c$ (推移性)
- $S_a \neq S_b$ ならば $S_a >_i S_b$ または $S_b >_i S_a$ (比較可能性)

学生 $s \in S$ の現在の第 i 希望が $\ell_i \in L (i = 1, \dots, k)$ であることを対 $\langle s, (\ell_1, \dots, \ell_k) \rangle$, $k > 0$, で表し、学生 s の希望講座リストとよぶ. k をこのリストの長さとよぶ. アルゴリズムの実行開始の時点では、 $k = n$ であり、 $(\ell_1, \dots, \ell_n)$ は $(L_1, \dots, L_n)$ のある置換 (*permutation*) となっている. 全学生の希望講座リストをちょうど一つずつ含む集合を (講座配属アルゴリズムの) 状態とよぶ. すべての状態の集合を状態空間とよび、 Σ で表す. したがって、任意の状態 $\sigma \in \Sigma$ と任意の学生 $s \in S$ に対し、

- $\exists u, \langle s, u \rangle \in \sigma$ (すなわち、 σ は学生 s の希望講座リストを含む.)
- $\langle s, u \rangle \in \sigma, \langle s, v \rangle \in \sigma$ ならば $u = v$ (すなわち、 σ は学生 s の希望講座リストを二つ以上含まない.)

である． $\langle s, (l_1, \dots, l_k) \rangle \in \sigma$ のとき，状態 σ において学生 s は講座 l_1 に仮配属されている．希望講座リスト $t = \langle s, (l_1, \dots, l_k) \rangle$ からその時点での第一希望講座を削除したリストを $t' = \langle s, (l_2, \dots, l_k) \rangle$ で表す．また， $t'' = (t')' = \langle s, (l_3, \dots, l_k) \rangle$ とする．

ところで，このように希望を削除しつづけて，いつかはリストが空になってしまうことがあるのだろうか？ 次の命題は，そのようなことがないことを示している．

命題 5.2

アルゴリズムのどの時点においても，希望講座リストは空になることはない．

証明

ある学生 s の希望講座リストがある状態 σ で空 $\langle s, () \rangle \in \sigma$ になったとする．この学生はすべての講座に一度は仮配属され，ステップ 2.2 でそこからはずされたことになる．これは，どの講座も状態 σ に至るある状態で定員を超過したことがあることを意味している．いったん，定員を超過した講座の仮配属学生数はその後，定員未満になることはない．したがって，状態 σ ではすべての講座の仮配属学生数は定員以上である．仮定により，全学生数は定員の合計以下であるから，実際には，全学生数と定員の合計は等しい．したがって，すべての講座の仮配属学生数はちょうど定員に等しい．すなわち，全学生の配属はここで決定していることになる．これは学生 s の配属先がないことと矛盾する．□

講座配属アルゴリズムを抽象書換え系 $\langle \Sigma, \Rightarrow \rangle$ としてモデル化する. ただし, $\sigma_1, \sigma_2 \in \Sigma$ のとき $\sigma_1 \Rightarrow \sigma_2$ は, 状態 σ_1 において, 定員を超過しているある講座 L_i を選択し, 定員超過分の学生を順序 $>_i$ のもとで優先順位が低い順に選び, 選んだ学生の希望講座リストから第一希望を削除した状態が σ_2 であることを表す.

ここで, 解析の便宜上, アルゴリズム中のステップ 2.1 の「定員超過分」を「1 人」に書き改めた新たなアルゴリズムを導入し, それに対する抽象書換え系を $\langle \Sigma, \rightarrow \rangle$ とする. すなわち, $\sigma_1, \sigma_2 \in \Sigma$ のとき $\sigma_1 \rightarrow \sigma_2$ は, 状態 σ_1 において, 定員を超過しているある講座 L_i を選択し, 順序 $>_i$ のもとで最も優先順位が低い学生 1 人を選び, 選んだ学生の希望講座リストから第一希望を削除した状態が σ_2 であることを表す. $\Rightarrow, \rightarrow$ いずれの場合も, 特に, 講座 L_i を選択したことを陽に示すために $\sigma_1 \Rightarrow_i \sigma_2, \sigma_1 \rightarrow_i \sigma_2$ などと記すこともある.

命題 5.3

- (1) $\sigma \Rightarrow \sigma'$ ならば $\sigma \rightarrow^+ \sigma'$ である.
- (2) $\sigma \in \text{NF}(\Rightarrow)$ ならば $\sigma \in \text{NF}(\rightarrow)$ である.

証明

- (1) $\sigma \Rightarrow_i \sigma'$ と仮定する. ただし, ここで選択された講座は L_i であり, 定員超過分は k 人 ($k > 0$) だったとする. このとき, 新たなアルゴリズムで L_i を連続して k 回選択すれば k ステップの簡約列 $\sigma \rightarrow_i \cdots \rightarrow_i \sigma'$ を得る. よって, $\sigma \rightarrow^+ \sigma'$ である.

(2) $\sigma \in \text{NF}(\Rightarrow)$ ならば, どの講座も定員を超過していないので, $\sigma \in \text{NF}(\rightarrow)$ である.

□

命題 5.4

$\langle \Sigma, \rightarrow \rangle$ が完備ならば, $\langle \Sigma, \Rightarrow \rangle$ も完備である.

証明

$\rightarrow$ が完備 (SN かつ CR) であると仮定する.

もし, $\Rightarrow$ が SN でなければ, 無限列 $\sigma_1 \Rightarrow \sigma_2 \Rightarrow \dots$ が存在し, 命題 5.3(1) より $\sigma_1 \rightarrow^+ \sigma_2 \rightarrow^+ \dots$ である. これは $\rightarrow$ の SN 性に矛盾する. ゆえに, $\Rightarrow$ は SN である.

$\Rightarrow$ が CR であることを示すために, $\sigma_1 \Leftarrow \sigma \Rightarrow \sigma_2$ と仮定する. ($\Rightarrow$ は $\rightarrow$ の反射推移閉包.) $\Rightarrow$ は SN なので, σ_1, σ_2 の正規形 $\sigma'_1, \sigma'_2 \in \text{NF}(\Rightarrow)$ が存在する. よって,

$$\text{NF}(\Rightarrow) \ni \sigma'_1 \Leftarrow \sigma_1 \Leftarrow \sigma \Rightarrow \sigma_2 \Rightarrow \sigma'_2 \in \text{NF}(\Rightarrow).$$

命題 5.3(1), (2) より, σ'_1 と σ'_2 は σ の $\rightarrow$ に関する正規形でもある. すなわち,

$$\text{NF}(\rightarrow) \ni \sigma'_1 \leftarrow^+ \sigma \rightarrow^+ \sigma'_2 \in \text{NF}(\rightarrow).$$

さらに命題 5.1 より, $\rightarrow$ は UN^+ 性をもつので, $\sigma'_1 = \sigma'_2$ である. よって, $\sigma_1 \Rightarrow \sigma'_1 = \sigma'_2 \Leftarrow \sigma_2$ である. ゆえに, $\Rightarrow$ は CR である. □

以下の節において $\langle \Sigma, \rightarrow \rangle$ が完備であることを示す.

5.3 完備性

5.3.1 $\rightarrow$ の停止 (SN) 性

命題 5.5

$\langle \Sigma, \rightarrow \rangle$ は SN 性を満たす.

証明

$\sigma \rightarrow \sigma'$ のとき, σ' に含まれる全希望講座リストの長さの和は, σ のそれに比べて必ず 1 だけ小さい. リストの長さの和は負になることはない. ゆえに, 無限簡約列 $\sigma_1 \rightarrow \sigma_2 \rightarrow \dots$ は存在しない. $\square$

5.3.2 $\rightarrow$ の合流 (CR) 性

$\langle \Sigma, \rightarrow \rangle$ は SN 性を満たしているので, Newman の補題より, CR 性を満たすことを証明するためには, WCR 性を満たすことを示せばよい. すなわち, $\sigma_1 \leftarrow \sigma \rightarrow \sigma_2$ ならばある σ' が存在し, $\sigma_1 \twoheadrightarrow \sigma' \leftarrow \sigma_2$ を示せばよい.

命題 5.6

$\langle \Sigma, \rightarrow \rangle$ は WCR 性を満たす.

証明

定員を越えている 2 講座 L_i と L_j を任意に選択し, L_i, L_j のみの調整によって合流することを示す. すなわち, $\sigma_1 \leftarrow_i \sigma \rightarrow_j \sigma_2$ ならば $\sigma_1 \twoheadrightarrow_{i,j} \sigma' \leftarrow_{i,j} \sigma_2$ を示す. ただし,

$\rightarrow_{i,j}$ は $\rightarrow_i \cup \rightarrow_j$ の反射推移閉包である。以下では一般性を失うことなく、 $i = 1, j = 2$ と仮定する。

希望講座リストの集合 σ は、 σ 中の各リスト内での L_1, L_2 の順位によって、以下の $A \sim E$ の互いに素な五つの集合に分割することができる。

$$A = \{t \in \sigma \mid t = \langle s, (L_1, L_2, \dots) \rangle\}$$

$$B = \{t \in \sigma \mid t = \langle s, (L_1, \dots) \rangle\} - A$$

$$C = \{t \in \sigma \mid t = \langle s, (L_2, L_1, \dots) \rangle\}$$

$$D = \{t \in \sigma \mid t = \langle s, (L_2, \dots) \rangle\} - C$$

$$E = \sigma - (A \cup B \cup C \cup D)$$

例えば、 A は第一希望が L_1 で第二希望が L_2 、 B は第一希望が L_1 で第二希望が L_2 でない希望講座リストの集合である。

学生間の順序 $>_i$ を希望講座リスト間の順序としても使用できるように、 $t_1 = \langle s_1, \dots \rangle$, $t_2 = \langle s_2, \dots \rangle$ が希望講座リストで、 $s_1 >_i s_2$ のとき $t_1 >_i t_2$ と定義する。このとき、 $t_1 >_i t_2$ ならば $t'_1 >_i t_2, t_1 >_i t'_2$ 、すなわち、第一希望を削除しても順位は不変である。

ここで、一般に

$$A = \{a_1, a_2, \dots, a_p\}$$

$$B = \{b_1, b_2, \dots, b_q\}$$

$$C = \{c_1, c_2, \dots, c_r\}$$

$$D = \{d_1, d_2, \dots, d_s\}$$

$$E = \{e_1, e_2, \dots, e_t\}$$

ただし,

$$a_1 >_1 a_2 >_1 \dots >_1 a_p$$

$$b_1 >_1 b_2 >_1 \dots >_1 b_q$$

$$c_1 >_2 c_2 >_2 \dots >_2 c_r$$

$$d_1 >_2 d_2 >_2 \dots >_2 d_s$$

とおく.

この状態を表 5.1 の八つに場合分けする. それぞれの場合について WCR 性を示す図の番号もこの表に示している. ただし, II は III において $>_1$ と $>_2$, A と C , B と D の対称性を考慮することによって省略することができる. 同様に, VII は V, VI に対称な場合 (でつくされること) を考慮することによって省略することができる.

表 5.1: 命題 5.6 の場合分け

	条 件	証 明 図
I	$a_1 >_1 b_1, c_1 >_2 d_1, c_1 >_1 a_1, a_1 >_2 c_1$	図 5.1
II	$a_1 >_1 b_1, c_1 >_2 d_1, c_1 >_1 a_1, c_1 >_2 a_1$	図 5.2 と対称
III	$a_1 >_1 b_1, c_1 >_2 d_1, a_1 >_1 c_1, a_1 >_2 c_1$	図 5.2
IV	$a_1 >_1 b_1, c_1 >_2 d_1, a_1 >_1 c_1, c_1 >_2 a_1$	図 5.3
V	$a_1 >_1 b_1, d_1 >_2 c_1, a_1 >_2 d_1$	図 5.4
VI	$a_1 >_1 b_1, d_1 >_2 c_1, d_1 >_2 a_1$	図 5.5
VII	$b_1 >_1 a_1, c_1 >_2 d_1$	図 5.4, 5.5 と対称
VIII	$b_1 >_1 a_1, d_1 >_2 c_1$	図 5.6

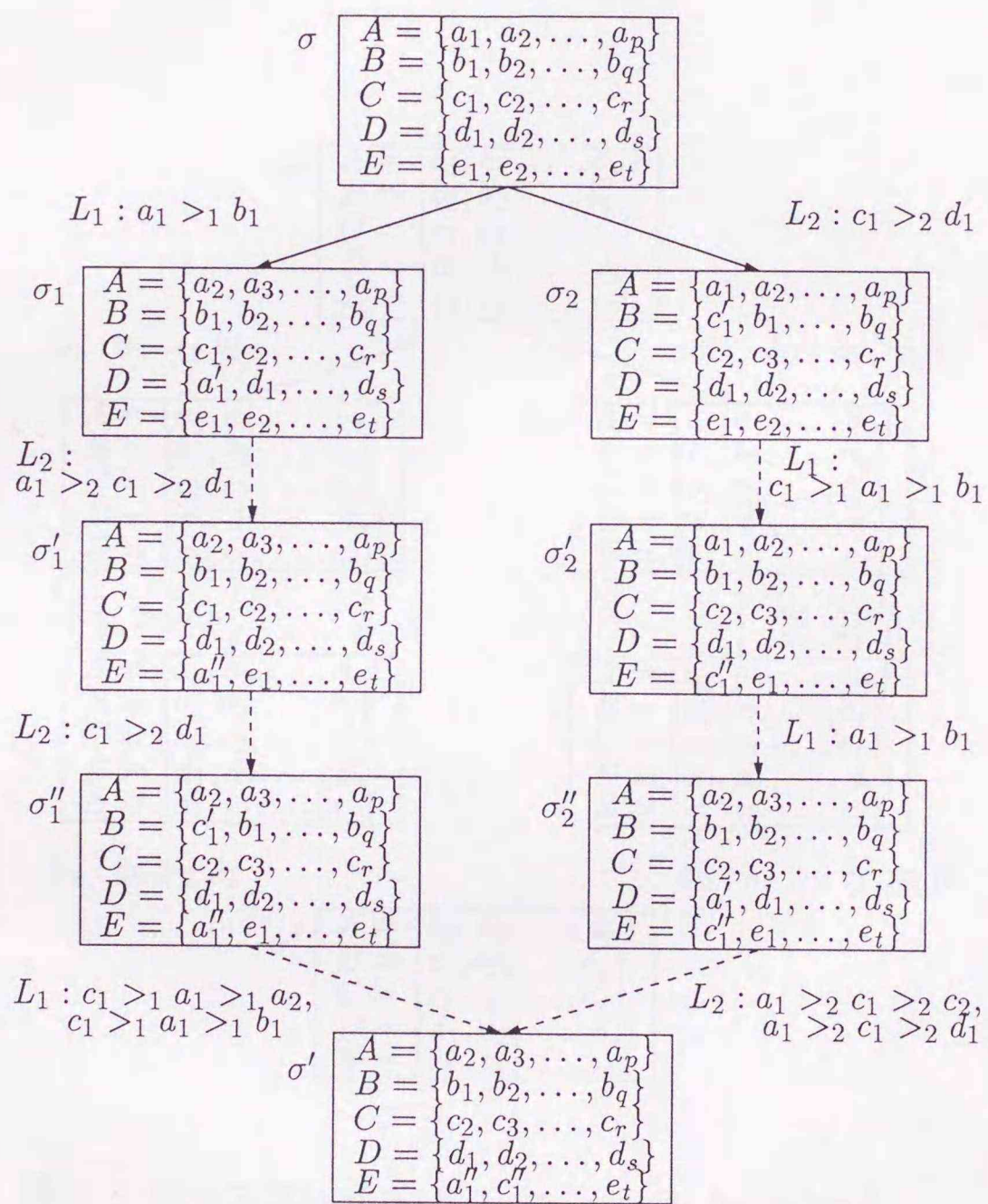


図 5.1: ケース I ($c_1 >_1 a_1 >_1 b_1 ; a_1 >_2 c_1 >_2 d_1$)

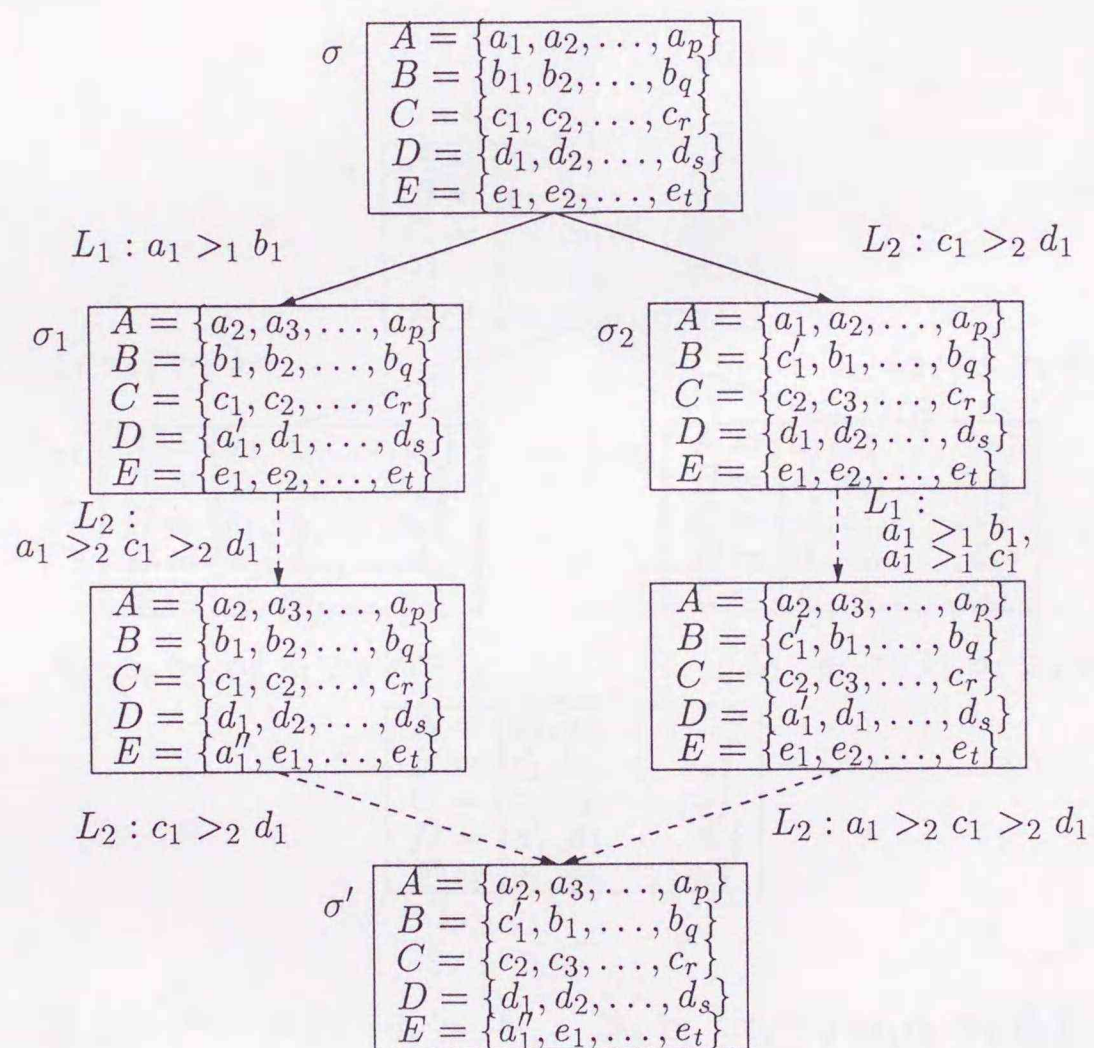


図 5.2: ケース III ($a_1 >_1 b_1, a_1 >_1 c_1$; $a_1 >_2 c_1 >_2 d_1$)

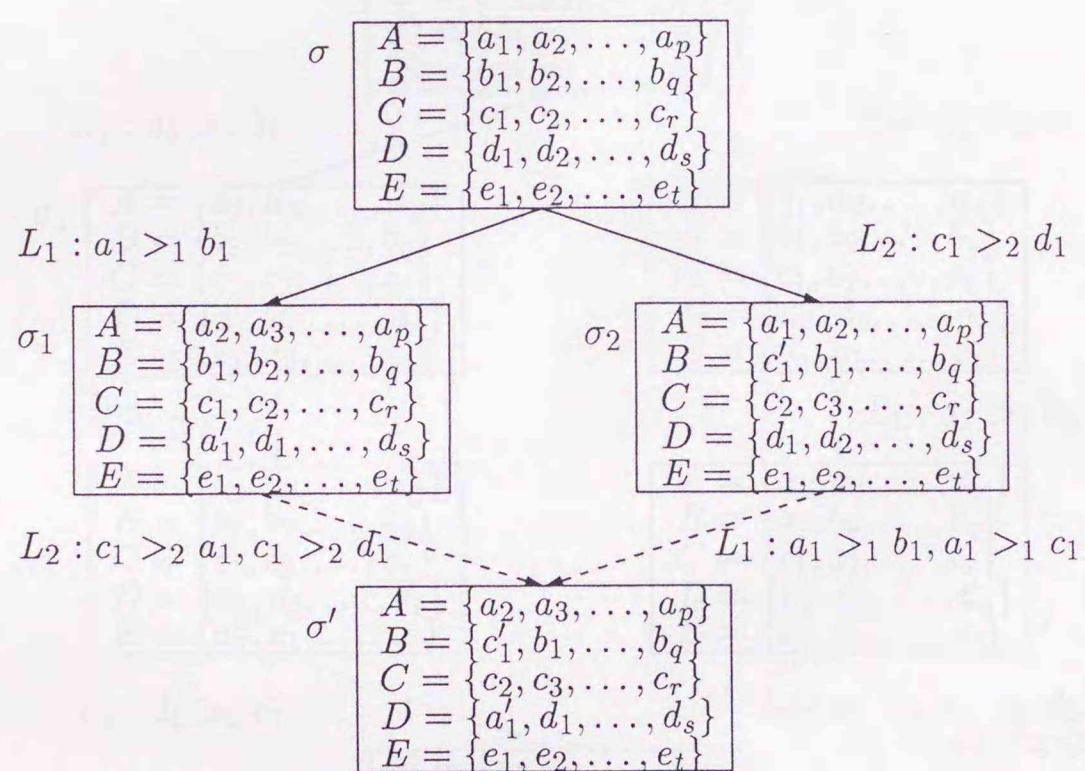


図 5.3: ケース IV ($a_1 >_1 b_1, a_1 >_1 c_1$; $c_1 >_2 a_1, c_1 >_2 d_1$)

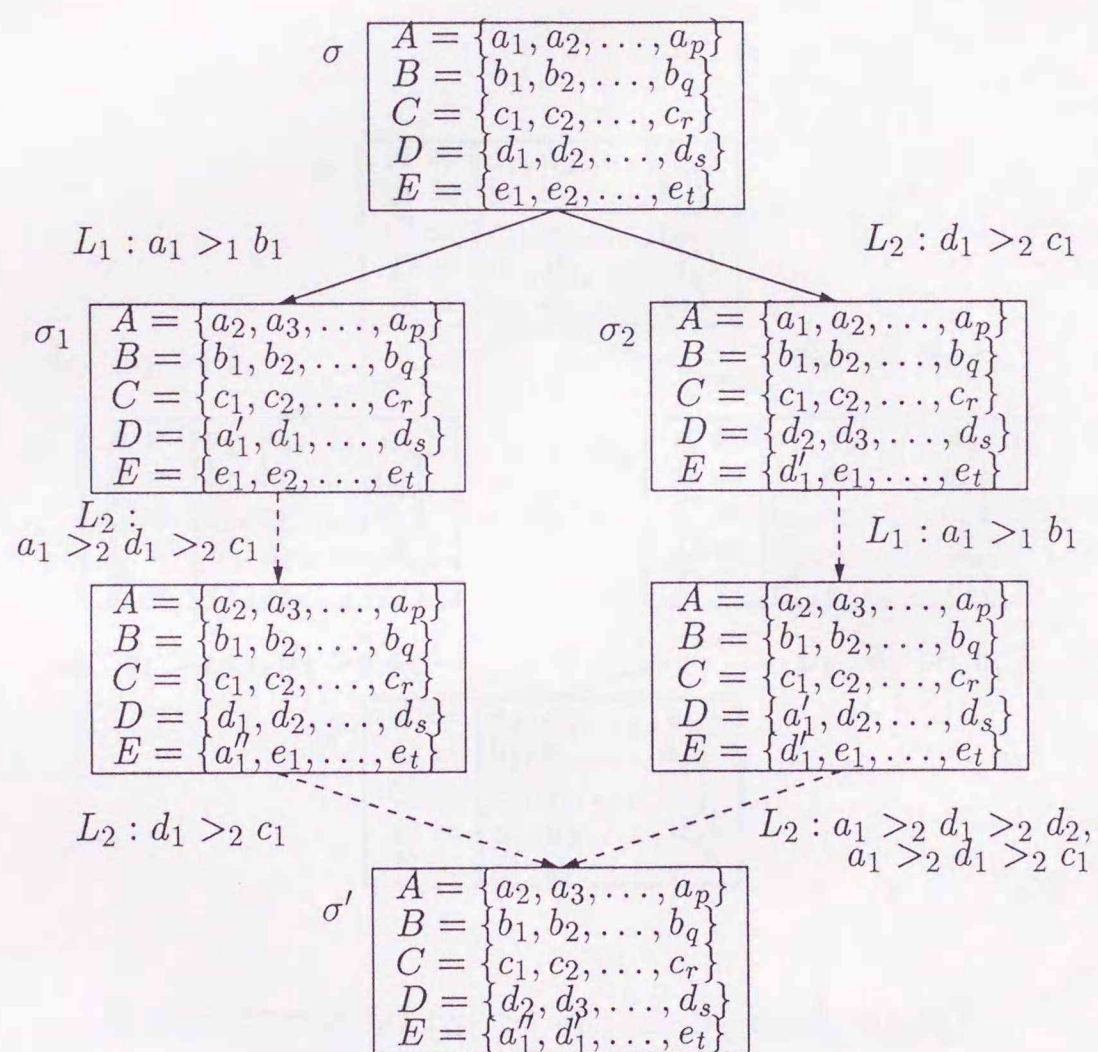


図 5.4: ケース V ($a_1 >_1 b_1$; $a_1 >_2 d_1 >_2 c_1$)

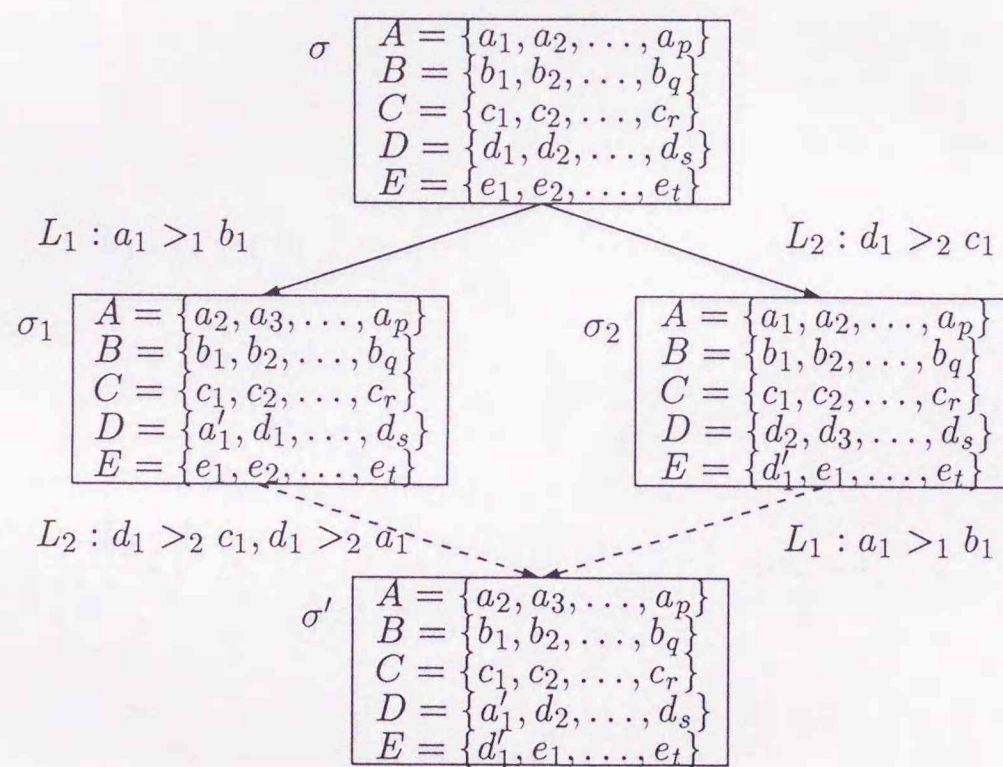


図 5.5: ケース VI ($a_1 >_1 b_1$; $d_1 >_2 c_1, d_1 >_2 a_1$)

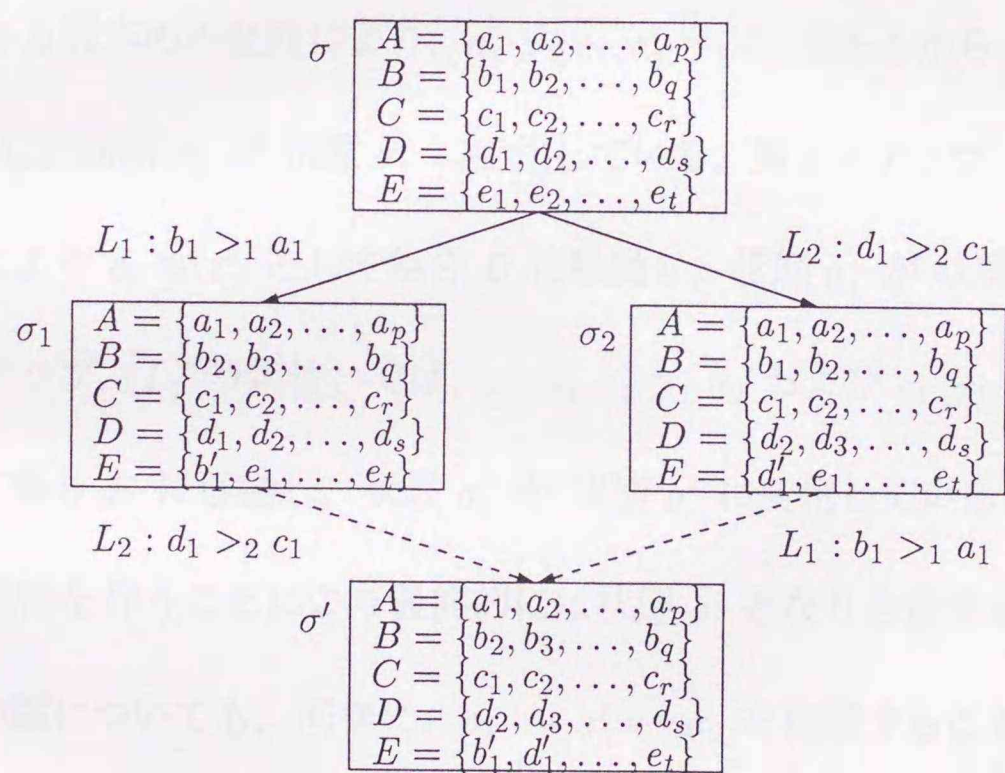


図 5.6: ケース VIII ($b_1 >_1 a_1$; $d_1 >_2 c_1$)

以下では、最も複雑であるケース I (図 5.1) についてのみ説明する。図の矢線の付近にはその簡約に使用した講座と条件を示してある。まず、状態 σ から L_1 講座と L_2 講座についてそれぞれ順序 $>_1, >_2$ に基づき調整していくわけだが、図の左半分の簡約経路について説明していく。 L_1 講座を第一希望にしている集合 A, B において $a_1 >_1 b_1$ により a_1 から第一希望が削除され、 a'_1 として集合 D に移動し、状態 σ が状態 σ_1 に変化している。第 2 ステップ (L_2 の調整) では、 $a_1 >_2 c_1, c_1 >_2 d_1, >_2$ の推移性および希望削除による順序の不変性により、 $a'_1 >_2 c_1, a'_1 >_2 d_1$ であるから a'_1 が a''_1 として集合 E に移動し、状態 σ_1 が状態 σ'_1 に変化している。第 3 ステップ (L_2 の調整) では、 $c_1 >_2 d_1$ により c_1 が c'_1 として集合 B に移動し、状態 σ'_1 が状態 σ''_1 に変化している。第 4 ステップ (L_1 の調整) では、 $c_1 >_1 a_1 >_1 a_2$ および $c_1 >_1 a_1 >_1 b_1$ により c'_1 が c''_1 として集合 E に移動し、状態 σ''_1 が状態 σ' に変化している。図の右側についても同様の操作を行うことにより最終的には状態 σ' となり合流することがわかる。ケース I 以外の図についても、同様に、 $\sigma_1 \rightarrow \sigma' \leftarrow \sigma_2$ を確認することができる。

以上では、暗黙に $p, q, r, s > 0$ と仮定したが、それぞれが 0 の場合も同様である。そのことを以下に補足しておく。 p, q, r, s が 0 か否かの組合せは全部で $2^4 = 16$ 通りある (表 5.2)。表中の 0 は 0 を表し、— は 0 でないことを表す。

調整を行う前提条件として、定員を超過している 2 講座に着目して調整を行っている。ので、 $p+q > 0, r+s > 0$ でなければならない。したがって、表 5.2 の 4, 8, 12 ~ 16 の 7 通りについては除外して考える。また、表 5.2 の 1 は既に証明済である。よって、残りの 8 通りについて考える。

表 5.2: p, q, r, s が 0 か否かの場合分け

	p	q	r	s	備 考
1	—	—	—	—	
2	—	—	—	0	I ~ IV, VII
3	—	—	0	—	V, VI, VIII
4	—	—	0	0	
5	—	0	—	—	I ~ VI
6	—	0	—	0	I ~ IV
7	—	0	0	—	V, VI
8	—	0	0	0	
9	0	—	—	—	VII, VIII
10	0	—	—	0	VII
11	0	—	0	—	VIII
12	0	—	0	0	
13	0	0	—	—	
14	0	0	—	0	
15	0	0	0	—	
16	0	0	0	0	

まず, $p = 0 (q \neq 0)$ のときだが, この場合 L_1 講座の調整によって移動するのは, 集合 B の要素であるので, $b_1 >_1 a_1$ の場合と同じく考えることができる. 同様に, $q = 0$ のときは, $a_1 >_1 b_1$ と同じ, $r = 0$ のときは, $d_1 >_2 c_1$ と同じ, $s = 0$ のときは, $c_1 >_2 d_1$ と同じである. これらが, 表 5.1 の I ~ VIII のどれを参照すればよいかを表 5.2 の備考欄に示してある.

ゆえに, $\langle \Sigma, \rightarrow \rangle$ は WCR 性を満たす. $\square$

定理 5.1

$\langle \Sigma, \rightarrow \rangle$ は完備である.

証明

命題 5.5, 5.6, 補題 2.2 より明らか. $\square$

定理 5.2

$\langle \Sigma, \Rightarrow \rangle$ は完備である.

証明

命題 5.4, 定理 5.1 より明らか. $\square$

5.4 情報処理上の留意点

この種の人員配置問題に関連した研究としては, クラス編成問題 [17] がある. クラス編成問題では, 配属される学生の希望をできるだけ満たすために, 数理計画的アプ

ローチにより解かれている。具体的には、オペレーションズ・リサーチ (OR) の分野で用いられている線形計画法により学生集団の満足度を最大にすることによって、全学生を配属するようになっている。

本論文で考察したアルゴリズムは、「集団の満足度」なるものを仮定しておらず、各学生の希望と各講座ごとの学生順位 $>_i$ のみを仮定して、ある意味では個人主義的に配属を決めている。このアルゴリズムを実際に適用する際の（情報処理上の）留意点を以下に示す。

(1) 希望講座リスト

各学生は、実際には全講座を順位付けする必要はない。例えば、初回のリスト提出時には第三希望まで示し、必要に応じて第四希望以降を漸増的 (*incremental*) に追加していてもよい。

(2) 講座ごとの学生順位

全講座が全学生についてあらかじめ順位付けをする必要はない。とりあえず、調整が必要となった講座 i に対して、調整にかかわる関係学生のみ比較して $>_i$ の一部を決定する。以後は、漸増的に $>_i$ を拡張していく。 $>_i$ を決めるための要因としては

- 講座 i の研究内容に密接に関係する科目の成績
- 講座 i への配属の希望の度合（希望講座リスト中の順位）

- 抽選
- 学生どうしの話し合い

などの一つ、あるいはいくつかの組合せが考えられる。

(3) 定員配分

定員は提出された希望講座リストを見た後に、希望数の多寡に応じて決定してもよい。

5.5 まとめ

本章では、書換え系の検証理論を用いた応用として、卒業研究の指導などのために学生を大学の各講座（研究室）に配属させる講座配属問題について述べた。まず、学生と講座の双方の優先度を考慮して配属を決定するアルゴリズムを考察した。このアルゴリズムには非決定性があるので、解（すなわち配属結果）が一意であるということは自明ではない。そこで、解の一意性を検証するために、このアルゴリズムの停止性と合流性（すなわち完備性）について抽象書換え系の理論を用いて形式的に証明した。

第 6 章

結論

6.1 各章のまとめ

6.1.1 第 1 章

第 1 章では、本論文が扱う問題領域である書換え系について簡単に紹介し、本研究の背景と目的を述べた。

6.1.2 第 2 章

第 2 章では、準備として、書換え系（抽象書換え系および項書換え系）の概要と諸性質について述べた。

6.1.3 第3章

第3章では、項書換え系の停止性検証法について述べた。項書換え系の停止性の検証は一般には決定不能な問題であるが、Dershowitzによって提案された単純化順序を用いることにより、一部の（しかし、実用上重要な多くの）項書換え系に対して停止性の形式的検証が可能であることを述べた。本論文では特に、経路順序とよばれる単純化順序の部分クラスに着目した。経路順序は、関数記号の集合上に定義される優先順位とよばれる半順序と項の構造に基づき、二項の相対的な重さ（大小関係）を比較するものである。この方法は、機械的な手順で二項の比較ができることから実装も容易で、したがって複雑な項にも十分対応し、また前提となる優先順位は項の構造から逆に推論できるなど、自動化という観点から最も有望な方法である。次に、経路順序のうちで特に代表的な（ステータス付き）再帰経路順序（RPOS）について検討した。

6.1.4 第4章

第4章では、ステータスの概念を拡張した“拡張ステータス”を提案し、それを適用した経路順序について述べた。ステータスは各関数記号ごとにもっていて、比較すべき二つの項の最外（最左）の関数記号が同一のときの引数の比較方法を示している。ステータスの種類としては、多重集合順序、左辞書式順序、右辞書式順序の3種類あり、慣習的にそれぞれ *mult*, *left*, *right* の記号を用いる。従来の *left*, *right* ステータスは、引数の比較する順序を辞書式順序で1引数ずつ比較していたが、拡張ステータ

スを用いることにより，いくつかの引数をまとめて多重集合順序で比較できるようになっている．本論文では，従来の RPOS よりも検証に成功しやすい拡張ステータス付き再帰経路順序 (RPOES) を提案し，それが単純化順序であることを証明し，その有効性について論じた．

6.1.5 第5章

第5章では，書換え系の検証理論を用いた応用として，卒業研究の指導などのために学生を大学の各講座（研究室）に配属させる講座配属問題について述べた．本論文では，学生と講座の双方の優先度を考慮して配属を決定するアルゴリズムを考察した．このアルゴリズムには非決定性があるので，解（すなわち配属結果）が一意であるということは自明ではない．そこで，解の一意性を検証するために，このアルゴリズムの停止性と合流性（すなわち完備性）について抽象書換え系の理論を用いて形式的に証明した．

6.2 まとめ

本論文では，書換え系の性質の形式的検証に関する研究として，以下のことを行った．

- (1) 従来の経路順序よりも停止性検証に成功しやすい拡張ステータス付き再帰経路順序 (RPOES) を提案し，それが単純化順序であることを証明し，その有効性を確認した．

- (2) 書換え系の検証理論を用いた応用として，講座配属問題を取り上げ，配属決定アルゴリズムを考察した．このアルゴリズムには非決定性があり，配属結果が一意であるということは自明ではないので，解の一意性を検証するために，抽象書換え系の理論を用いて形式的に証明した．

今後の研究内容としては，以下の検討課題が残されていると考えられる．

- 優先順位や拡張ステータスの自動決定．
- RPOS よりも強力な経路順序に拡張ステータスを適用し，より強力な拡張ステータス付き経路順序の提案．

謝辞

北海道大学工学部システム情報工学専攻大内東教授には、本研究を進めるにあたり、深い議論を通して御指導いただくとともに終始励ましていただいた。深く感謝の意を表します。北海道大学工学部システム情報工学専攻新保勝教授、宮本衛市教授には、本研究に対して詳細な議論や有益な御助言をいただいた。深謝いたします。北海道大学工学部システム情報工学専攻栗原正仁助教授には、本論文の基礎となった書換え系理論をはじめとして、本研究の基礎となる多くの事柄を懇切、丁寧に御教授していただいた。心から感謝いたします。北海学園大学工学部電子情報工学科河口至商教授には、研究の機会を与えていただいた。感謝いたします。また、公私にわたり御世話になった調和系工学分野（および旧システム工学講座）に関係する皆様に感謝します。

最後に、著者の研究姿勢に対して、理解を示してくれた両親に感謝します。

参考文献

- [1] N. Dershowitz. Orderings for term-rewriting systems. *Theoretical Computer Science*, Vol. 17, pp. 279–301, 1982.
- [2] N. Dershowitz. Termination of rewriting. *Journal of Symbolic Computation*, Vol. 3, pp. 69–116, 1987.
- [3] N. Dershowitz and J.-P. Jouannaud. *Rewrite systems*, Vol. B of *Handbook of Theoretical Computer Science*, pp. 243–320. Elsevier Science Publisher B.V., 1990.
- [4] N. Dershowitz and Z. Manna. Proving termination with multiset orderings. *Communications of the ACM*, Vol. 22, No. 8, pp. 465–476, 1979.
- [5] M. C. F. Ferreira. *Termination of Term Rewriting – Well-foundedness, Totality and Transformations*. PhD thesis, University of Utrecht, November 1995.
- [6] 二木厚吉, 外山芳人. 項書き換え型計算モデルとその応用. *情報処理*, Vol. 24, No. 2, pp. 133–146, 1983.

- [7] G. Huet. Confluent reductions : abstract properties and applications to term rewriting systems. *Journal of the ACM*, Vol. 27, No. 4, pp. 797-821, 1980.
- [8] 井田哲雄. 新しいプログラミング・パラダイム, pp. 165-218. 共立出版, 1989.
- [9] 井田哲雄. 計算モデルの基礎理論, pp. 223-296. 岩波書店, 1991.
- [10] J.-P. Jouannaud, P. Lescanne, and F. Reinig. Recursive decomposition ordering. In D. Bjørner, editor, *Working Conference on Formal Description of Programming Concepts II (IFIP)*, pp. 331-353. North-Holland Publishing Company, 1982.
- [11] S. Kamin and J. J. Lévy. Attempts for generalizing the recursive path orderings. Report, Department of Computer Science, University of Illinois, February 1980.
- [12] D. Kapur, P. Narendran, and G. Sivakumar. A path ordering for proving termination of term rewriting systems. In H. Ehrig, editor, *Proceedings of the 10th Colloquium on trees in algebra and programming (CAAP)*, Vol. 185 of Lecture Notes in Computer Science, pp. 173-187. Springer, 1985.
- [13] J. W. Klop. *Term rewriting systems*, Vol. II of *Handbook of Logic in Computer Science*, pp. 1-116. Oxford University Press, 1992.
- [14] 近藤 久. 真偽維持システムを用いた項書換えシステムの推論手続きに関する研究. PhD thesis, 北海道大学, 1994.

-
- [15] 近藤 久, 栗原正仁, 大内 東. Reason Maintenance System による項書き換えシステム停止性検証の効率化. 情報処理学会論文誌, Vol. 32, No. 9, pp. 1046-1056, 1991.
- [16] 近藤 久, 栗原正仁, 大内 東. 複数の簡約順序のもとでの項書き換えシステム完備化手続き. 電子情報通信学会論文誌, Vol. J78-D-I, No. 1, pp. 1-10, 1995.
- [17] 今野 浩. 数理決定法入門 キャンパスの OR, pp. 1-20. 朝倉書店, 1992.
- [18] M. Kurihara, H. Kondo, and A. Ohuchi. Using ATMS to efficiently verify the termination of rewrite rule programs. *International Journal of Software Engineering and Knowledge Engineering*, Vol. 2, No. 4, pp. 547-565, 1992.
- [19] M. Kurihara and A. Ohuchi. Modularity of simple termination of term rewriting systems. *Journal of IPS Japan*, Vol. 31, No. 5, pp. 633-642, 1990.
- [20] P. Lescanne. On the recursive decomposition ordering with lexicographical status and other related orderings. *Journal of Automated Reasoning*, Vol. 6, pp. 39-49, 1990.
- [21] A. Middeldorp. *Modularity Properties of Term Rewriting Systems*. PhD thesis, Vrije Universiteit Amsterdam, November 1990.
- [22] 能登正人, 栗原正仁, 大内 東. 拡張ステータスによる項書き換え系の停止性検証. 情報処理学会論文誌, Vol. 36, No. 12, pp. 2867-2871, 1995.

- [23] 能登正人, 栗原正仁, 大内 東. 抽象書換え系理論による講座配属アルゴリズムの完備性の解析. 情報処理学会論文誌, Vol. 36, No. 10, pp. 2327-2334, 1995.
- [24] D. A. Plaisted. *Equational reasoning and term rewriting systems*, Vol. 1 of *Handbook of Logic in Artificial Intelligence and Logic Programming*, pp. 273-364. Oxford Science Publications, Clarendon Press - Oxford, 1993.
- [25] J. Steinbach. Extensions and comparison of simplification orderings. In N. Dershowitz, editor, *Proceedings of the 3rd Conference on Rewriting Techniques and Applications*, Vol. 355 of Lecture Notes in Computer Science, pp. 434-448. Springer, 1989.
- [26] J. Steinbach. Simplification orderings : Putting them to the test. *Journal of Automated Reasoning*, Vol. 10, pp. 389-397, 1993.
- [27] J. Steinbach. *Termination of Rewriting - Extension, Comparison and Automatic Generation of Simplification Orderings*. PhD thesis, University of Kaiserslautern, January 1994.

