情報処理学会論文誌　プログラミング　Vol. 4　No. 2　31–47 (Mar. 2011)

文脈自由言語と超決定性言語の包含判定問題の決定可能性の型理論を用いた証明

塚田武志[†1,†2]　小林直樹[†1]

言語の包含判定問題とは，与えられた言語 L_1 と L_2 について $L_1 \subseteq L_2$ が成立するか否かを判定する問題であり，理論的な興味の対象であるだけでなく，プログラム検証などへの広い応用を持つ重要な問題である．この問題に関する既知の最も強い結果の 1 つが文脈自由言語と超決定性言語の包含判定の決定可能性である．このオリジナルの証明は，Greibach と Friedman によって与えられている．我々はこの問題に対して，小林らによって提案されている型に基づく言語の包含判定の手法を適用し，決定可能性に対する別証明を与えた．この手法は以下のような利点を持つ．(1) 部分型関係やポンプの補題などのよく知られた概念で理論が展開できる．(2) 型推論を効率的に行う方法は多数提案されており，それらを利用することができる．また，提案する証明は小林らのアイデアを正規言語よりも広いクラスに適用したはじめての例であり，その他の非正規言語クラスへの応用も期待される．

A Type-theoretic Proof of the Decidability of the Language Containment between Context-free Languages and Superdeterministic Languages

Takeshi Tsukada[†1,†2] and Naoki Kobayashi[†1]

The language containment problem, which asks whether $L_1 \subseteq L_2$ for given languages L_1 and L_2, is an important problem in the field of formal language theory and has a variety of applications including program verification. One of the strongest result about this problem is the decidability for the case where L_1 and L_2 are context free and superdeterministic languages respectively, originally proved by Greibach and Friedman. In this paper, we give a new proof of the decidability, inspired by Kobayashi's type-based approach to language containment problems. The new proof has the following advantages. (1) The key notions and lemmas in the proof are well-known ones, such as subtyping and Pumping Lemma. (2) We can apply well-studied techniques for efficient typability checking. Furthermore, our proof is the first application of the type-based approach to the inclusion between non-regular languages and it seems applicable to other cases.

1. はじめに

言語の包含判定問題とは，与えられた 2 つの言語 L_1 と L_2 について $L_1 \subseteq L_2$ であるか否かを判定するという問題である．より厳密には，L_1 および L_2 の動く範囲を明示的にして，

「与えられた $L_1 \in \mathcal{L}_1$ と $L_2 \in \mathcal{L}_2$ に対して $L_1 \subseteq L_2$ であるか求めよ」

という問題となる．ここで $\mathcal{L}_1$ および $\mathcal{L}_2$ は言語の集合，すなわち言語クラスである．この問題は，形式言語学における主要な決定問題であり，多数の研究が行われている[1)–5)]．さらに包含判定問題は，理論的な興味の対象であるばかりでなく，プログラム検証などへの様々な応用を持つ問題でもある．たとえば，PHP によって生成される Web ページが XHTML の仕様に従っているかどうかの検証[4)] などが提案されている．基本的なアイデアは，プログラムの出力の集合を文脈自由言語で表し（L_1 とする），また出力仕様を Regular Hedge Language と呼ばれる言語で表すことで（L_2 とする），プログラムが仕様に従っているか否かの判定を $L_1 \subseteq L_2$ が成立するかの検査に帰着するという手法である．

包含判定問題が決定可能であるかどうかは，言語クラス $\mathcal{L}_1$，$\mathcal{L}_2$ に依存することが知られている．たとえば，$\mathcal{L}_1$ を文脈自由言語のクラスとすると，$\mathcal{L}_2$ が正規言語のクラスならば決定可能であるし，$\mathcal{L}_2$ が文脈自由言語のクラスならば決定不能である．

特に応用的な観点から，包含判定が決定可能な広い言語クラスを得ることは重要である．先ほどのプログラム検証の例では，言語クラス $\mathcal{L}_1$ が広いことは検証対象となるプログラムの範囲や検証の際の近似の精度が上がることを意味し，言語クラス $\mathcal{L}_2$ が広いことは検証可能となる仕様の範囲を広げることを意味するためである．このような動機から，包含判定が決定可能であるようなできる限り広い言語クラスが探し求められてきた．

包含判定の決定可能性に関する最も強い結果の 1 つが，Greibach ら[2)] によって証明されたもので，$\mathcal{L}_1$ を文脈自由言語，$\mathcal{L}_2$ を超決定性言語のクラスとした場合である．超決定性言語は Greibach らによって導入された言語クラスであって，文脈自由言語の真の部分集

†1 東北大学大学院情報科学研究科
Graduate School of Information Science, Tohoku University
†2 日本学術振興会特別研究員 DC
JSPS Research Fellow

合である．この言語クラスは超決定性オートマトンと呼ばれる特殊な形のプッシュダウンオートマトンの受理言語のクラスとして定義され，XML Grammar [6] や Regular Hedge Language [7] などの応用上重要な言語クラスを含むことが知られている．

本論文では，彼らの結果に対する別証明を与える．彼らの結果がオートマトン理論を用いたものであるのに対して，我々の証明は，Kobayashi [8] および Kobayashi ら [9] によって提案されている，型理論を用いた包含判定証明の手法によるものである．

小林らの手法は，高階再帰スキーム（文脈自由言語の拡張）が生成する言語を L_1 とし，正規木言語を L_2 とした場合の包含判定問題の決定可能性の証明に使われた．ここで L_1 は文法 G で，L_2 はそれを受理する有限オートマトン M によって与えられるとする．彼らは，与えられた M に対して型システム $\mathcal{T}_M$ を定義して，$\mathcal{T}_M$ で G が型付け可能であることと $L_G \subseteq L_M$ であることが同値であることを証明した．また $\mathcal{T}_M$ の決定可能性は，M の状態集合が有限であることから従う形になっている．

我々の証明も，この戦略に基づくものである．すなわち，与えられた超決定性プッシュダウンオートマトン M に対して型システム $\mathcal{T}_M$ を定義し，(1) 型付け可能性と包含関係の成立が同値であること，(2) 型付け可能性が決定可能であること，の 2 つを証明した．この 2 つから，包含判定問題の決定可能性が従う．

我々の型システムは（G を文脈自由言語に制限したうえで）M が超決定性プッシュダウンオートマトンの場合に拡張したものである．拡張の基本となるアイデアはシンプルなもので，M を無限状態オートマトンだと見なすというものである．プッシュダウンオートマトンの計算途中の様相は，状態 q とスタック $\tilde{A}$ のペアで表現できることに注意する．すると，プッシュダウンオートマトンは，$\{(q, \tilde{A}) \mid q$ は状態，$\tilde{A}$ はスタック $\}$ という無限の状態を持つオートマトンだと見なすことができる．この無限状態オートマトンに対して小林らの手法を適用することで，型システム $\mathcal{T}_M$ を定義できる．すると，$\mathcal{T}_M$ で G が型付け可能であることと $L_G \subseteq L_M$ が同値であることを容易に示すことができる．

次に $\mathcal{T}_M$ における型付け可能性も決定可能であることを示すことになるが，これは自明ではない．なぜなら，小林らの型システムの決定可能性は M の状態が有限であることに依存しているのに対して，我々の型システムは M を無限状態のオートマトンだと見なすためである．したがって，何らかの方法で M の無限種類の状態を有限で記述できることを示す必要がある．このための鍵となるのが，次の 2 つである．1 つは部分型の導入で，これによって 1 つの型で無限種類の型を代表することができるようになる．もう 1 つが文脈自由言語のポンプの補題であり，この補題から有限種類の型で十分であることを示すことができる．

我々の証明の利点は以下のようにまとめられる．

まずは，理論が広く知られた概念や手法を中心に展開されることである．型付け可能性と包含関係の成立が同値であることの証明は一般的な型理論の方法どおりであり，決定可能性の証明もポンプの補題や部分型などのよく知られた命題や概念を使って導かれる．

次に，証明の手法を拡張する道筋が立てやすいことである．これは 2 つの理由による．1 つ目の理由は，小林らの型による証明の技法に従っていることである．この手法は元々 L_1 が文脈自由言語よりも広いクラスの問題に対して考案されたものであるため，L_1 を文脈自由言語からより広いクラスに拡張する方針が立てやすい．2 つ目の理由は，決定可能性とポンプの補題の対応を明示的にしたことである．文脈自由言語のポンプの補題に似た形の補題は様々な言語で提案されているため，それらに対してもこの証明手法が使える可能性がある．

最後に，型付け可能性を判定するアルゴリズムを高速化する手法は広く研究されているという点があげられる．特に，我々の証明のもととなっている小林らによる型システムに対しては，小林によって実用的な時間で型付け可能性を検査する処理系が構築されている [10]．

本論文の構成は，以下のようになっている．2 章で，文脈自由言語および超決定性言語の定義を行い，同時に本論文が採用している記法を導入する．次に 3 章で型システムを定義し，型付け可能性と包含関係の成立が同値であること（完全性および健全性）を証明する．そして 4 章で決定可能性を証明する．この章が本論文の主要な結果である．その後，5 章で関連研究について述べ，最後に結論および今後の課題について述べる．

2. 準　　備

この章では，文脈自由文法とプッシュダウンオートマトンに関して，本論文で使用する記法の定義を行う．また，超決定性言語 [2] の定義も行い，その性質を紹介する．

2.1 語 と 言 語

Σ を記号の有限集合とし，**アルファベット**と呼ぶ．Σ の要素を a で表す．アルファベットには含まれない特殊な記号 $\$ \notin \Sigma$ を用意し，**末尾記号**と呼ぶ．$\Sigma \cup \{\$\}$ を $\hat{\Sigma}$ と書く．Σ の有限列からなる集合を Σ^* と書き，空列を ε で書く．アルファベットと長さ 1 の列を同一視し，$a \in \Sigma^*$ と見なすことがある．集合 Σ^* の要素を**終端しない語**と呼び，これらを x，y，z，v，w などで表す．Σ^* の要素の終わりに末尾記号 $ を付けたものを**終端する語**と呼ぶ．つまり，終端する語の集合は $\{w\$ \mid w \in \Sigma^*\}$ であり，これを $\Sigma^*\$$ と書く．終端する語の集合 $L \subseteq \Sigma^*\$$ を**言語**と呼ぶ．

このような終端する語と終端しない語の区別は一般的なものではないので，注意が必要である．この区別は，以下のように，アルファベットをランク付きアルファベットであると考えると理解できる．Σ の要素はアリティ1 のシンボルの集合であり，また末尾記号 \$ は唯一のアリティ0 のシンボルである．終端する語 $a_1a_2\ldots a_n\$$ は $a_1(a_2(\ldots(a_n(\$))\ldots))$ という項であり，終端しない語 $a_1a_2\ldots a_n$ は関数 $f(x)=a_1(a_2(\ldots(a_n(x))\ldots))$，言語は項の集合に対応する．語をランク付きアルファベットの上の項や関数だと考えると，型システムの理解が容易になる．しかし本論文では，一般的な文脈自由文法やプッシュダウンオートマトンにおける語の扱い方に合わせるために，語は記号の列であるという定義を採用する．末尾記号の導入の目的は，記号列が項を表すのか関数を表すのかを区別することである．

2.2　文脈自由文法

文脈自由文法は 4 つ組 $G=\langle\Sigma,N,S,R\rangle$ である．ここで

- Σ はアルファベット．
- N は**非終端記号**の集合．N の要素を X，Y などで表す．また α，β で $(\Sigma\cup N)^*$ の要素を表すものとする．
- $S\in N$ は開始記号である．
- R は書き換え規則の集合である．各書き換え規則は $X\to\alpha$ の形をしており，各非終端記号 X について複数の書き換え規則が存在していてもよい．

$X\to\alpha\in R$ のとき $\beta X\gamma\Rightarrow_G\beta\alpha\gamma$ と書き，関係 $\Rightarrow_G$ の反射推移閉包を $\Rightarrow_G^*$ と書く．$\alpha\Rightarrow_G^*\beta$ のとき，「α から β が導出される」という．文脈から明らかな場合には，G を省略する．与えられた列 α について，終端しない語の集合 $L_G(\alpha)$ を次のように定義する．

$$L_G(\alpha)=\{w\in\Sigma^*\mid\alpha\Rightarrow_G^*w\}$$

また，文脈自由文法から定まる言語 L_G を $L_G=L_G(S)\,\$=\{w\$\mid w\in L_G(S)\}$ と定義する．$L_G(\alpha)$ は終端しない語の集合であるのに対して，L_G は言語（＝終端する語の集合）であることに注意する．与えられた言語 L が**文脈自由言語**であるというのは，ある文脈自由文法 G が存在して，$L=L_G$ となることをいう．

以下では，任意の非終端記号 X について $L_G(X)\neq\emptyset$ であることを仮定する．したがって，任意の $\alpha\in(\Sigma\cup N)^*$ について $L_G(\alpha)\neq\emptyset$ であり，特に $L_G\neq\emptyset$ である．

文脈自由文法 G が**チョムスキー標準形**であるとは，すべての書き換え規則が以下のいずれかの形をしていることをいう．

- $X\to Y_1Y_2$．ただし，$Y_1\neq S$ かつ $Y_2\neq S$
- $X\to a$
- $S\to\varepsilon$

ここで S は開始記号である．任意の文脈自由文法 G に対して，等価なチョムスキー標準形の文脈自由文法が存在することが知られている．つまり，任意の文脈自由文法 G に対して，あるチョムスキー標準形の文脈自由文法 G' が存在し，$L_G=L_{G'}$ とできる．また，この変換は多項式時間で行えることが知られている[⋆1]．

2.3　プッシュダウンオートマトン

プッシュダウンオートマトンとは 7 つ組 $M=\langle Q,\hat{\Sigma},\Gamma,\delta,q_S,\bot,F\rangle$ である．ここで

- Q は**状態**の有限集合．q，p，r，s などを状態を表す変数として用いる．
- $\hat{\Sigma}$ はアルファベット（末尾記号 \$ を含む）．
- Γ は有限の記号の集合で，**スタックシンボル**と呼ばれる．スタックシンボルは A，B，C などで表す．また，スタックシンボルの列は $\tilde{A}$，$\tilde{B}$，$\tilde{C}$ などで表す．
- $\bot\in\Gamma$ は初期スタック．
- $\delta\subseteq(Q\times\Gamma\times(\hat{\Sigma}\cup\{\varepsilon\}))\times(Q\times\Gamma^*)$ は遷移関係．
- $q_S\in Q$ は初期状態．
- $F\subseteq Q$ は受理状態の集合．

プッシュダウンオートマトン M が**決定性**であるとは，次の (1)，(2) の両方が成り立つことをいう．

(1)　任意の $q\in Q$ と $A\in\Gamma$ と $a\in\hat{\Sigma}\cup\{\varepsilon\}$ について $(q,A,a,q',\tilde{A}')\in\delta$ となる q' と $\tilde{A}'$ はたかだか 1 つである．

(2)　任意の $q\in Q$ と $A\in\Gamma$ と $a\in\hat{\Sigma}$ について，以下のうちたかだか 1 つが成立する．

- $(q,A,\varepsilon,q',\tilde{B})\in\delta$ となる q' と $\tilde{B}$ が存在する．
- $(q,A,a,q',\tilde{B})\in\delta$ となる q' と $\tilde{B}$ が存在する．

以下ではつねにプッシュダウンオートマトン M は決定性であるとする．

状態とスタック列の組 $(q,\tilde{A})\in Q\times\Gamma^*$ を**様相**と呼び，c で表す．$q,q'\in Q$，$A\in\Gamma$，$\tilde{B},\tilde{C}\in\Gamma^*$，$a\in\hat{\Sigma}\cup\{\varepsilon\}$ に対して，$(q,A,a,q',\tilde{B})\in\delta$ のとき $(q,\tilde{C}A)\Vdash_M^a(q',\tilde{C}\tilde{B})$ と書き，「$(q,\tilde{C}A)$ からの a による *1* **ステップ遷移**で $(q',\tilde{C}\tilde{B})$ になる」という．特に $a=\varepsilon$ のときこの遷移を ε **遷移**と呼ぶ．M の決定性より，任意の様相 c について，$c\Vdash_M^\varepsilon c'$ となる c' が存在するか，$c\Vdash_M^a c'$ となる c' と $a\neq\varepsilon$ が存在するか，c からの遷移は 1 つもないかのうち，ちょうど 1 つが成立する．はじめの条件が成立するとき c は ε **モード**にあるといい，

⋆1 標準的なアルゴリズム[11] と Harrison らの結果[12] から，多項式時間の変形が得られる．

2 番目の条件が成立するときには**読み込みモード**にあるといい，最後の条件が成立するときには**閉塞モード**にあるという．θ や ϕ などで読み込みモードの様相を表す．θ からの $a \in \hat{\Sigma}$ を読む 1 ステップ遷移ののちにできる限りの ε 遷移をした結果 θ' にたどりつくとき，すなわち，

$$\theta \Vdash^{a}_{M} c_0 \Vdash^{\varepsilon} c_1 \Vdash^{\varepsilon}_{M} \ldots \Vdash^{\varepsilon}_{M} \theta'$$

とできるとき，$\theta \vDash^{a}_{M} \theta'$ と書く．また，$w = a_1 a_2 \ldots a_n \in \hat{\Sigma}^*$ について，

$$\theta \vDash^{a_1}_{M} \phi_1 \vDash^{a_2}_{M} \phi_2 \vDash^{a_3}_{M} \ldots \vDash^{a_n}_{M} \theta'$$

が成り立つときに，$\theta \vDash^{w}_{M} \theta'$ と書く．これを語 $w = a_1 \ldots a_n$ による**遷移**と呼ぶ．また $w = \varepsilon$ については，$\theta \vDash^{\varepsilon}_{M} \theta$ と定義する．文脈から明らかな場合には $\Vdash^{a}_{M}$ や $\vDash^{a}_{M}$ の M を省略することがある．

θ から \$ を読み込むと受理状態にいたる場合，すなわち，ある $q_f \in F$ が存在して，

$$\theta \Vdash^{\$}_{M} c_1 \Vdash^{\varepsilon}_{M} c_2 \Vdash^{\varepsilon}_{M} \ldots \Vdash^{\varepsilon}_{M} (q_f, \varepsilon)$$

とできるとき，$\theta \vDash^{\$}_{M} \Box$ と書く．ここで受理条件が終状態かつ空スタックであることに注意する．プッシュダウンオートマトン M において，（読み込みモードの）様相 θ から受理される言語 $L_M(\theta)$ を

$$L_M(\theta) = \{w\$ \in \Sigma^*\$ \mid \theta \vDash^{w}_{M} \theta' \vDash^{\$}_{M} \Box\}$$

と定義する．θ_S で，様相 $(q_S, \bot)$ からできる限りの ε 遷移をした結果の様相を表し，**初期様相**と呼ぶ．つまり，$(q_S, \bot) \Vdash^{\varepsilon}_{M} c_1 \Vdash^{\varepsilon}_{M} \ldots \Vdash^{\varepsilon}_{M} \theta_S$ である．プッシュダウンオートマトン M の受理言語 L_M は $L_M = L_M(\theta_S)$ と定義される．

スタック $\tilde{A}$ について，その長さを $|\tilde{A}|$ で表し，**スタック長**と呼ぶ．たとえば，$|\bot AB| = 3$，$|\bot| = 1$，$|\varepsilon| = 0$ である．様相 $c = (q, \tilde{A})$ について，$|c| = |\tilde{A}|$ と定義する．また，$\mathrm{state}(c) = q$ と定義する．プッシュダウンオートマトン M の**増分**を，$\max\{|\tilde{B}| - 1 \mid (q_0, A, a, q_1, \tilde{B}) \in \Delta\}$ と定義する．

読み込みモードの様相 θ が**到達可能**であるとは，ある $w \in \hat{\Sigma}^*$ が存在して $\theta_S \models^{w} \theta$ となることをいう．また，θ が**生存している**とは，ある $w\$ \in \hat{\Sigma}^*\$$ が存在して，$\theta \models^{w\$} \Box$ であることをいう．読み込みモードの様相 $\theta = (q, \tilde{A})$ が**有効**であるとは，あるスタック列 $\tilde{B}$ が存在して $(q, \tilde{B}\tilde{A})$ が到達可能かつ生存していることをいう．つまり，有効な様相とは，ある語 $w\$ \in L_M$ の受理のための遷移系列に出現する様相のスタックの底の方を捨てて作られる様相のことである．

2.4　超決定性言語

Greibach ら[2] によって導入された超決定性言語（Superdeterministic Language）の定義を行う．

定義 2.1（遅れ）. プッシュダウンオートマトン M の遅れが d であるとは，任意の到達可能な様相 c_0 からの ε 遷移系列

$$c_0 \Vdash^{\varepsilon} c_1 \Vdash^{\varepsilon} c_2 \Vdash^{\varepsilon} \ldots \Vdash^{\varepsilon} c_n$$

に対して，$n \leq d$ であることをいう．

定義 2.2（超決定性プッシュダウンオートマトン（Superdeterministic PDA））. プッシュダウンオートマトン M が次の 3 つの条件を満たすとき，M は超決定性であるという．

- M は決定性オートマトンである．
- M の遅れは，ある有限の値 d である．
- 状態とスタック長の変化は，状態と入力にのみ依存し，スタックの内容には依存しない：θ_0，θ_1，θ'_0，θ'_1 を読み込みモードの到達可能様相，w を任意の語とし，
  $$\theta_0 \vDash^{w}_{M} \theta'_0 \qquad \theta_1 \vDash^{w}_{M} \theta'_1$$
  とする．このとき，もし $\mathit{state}(\theta_0) = \mathit{state}(\theta_1)$ ならば，$|\theta'_0| - |\theta_0| = |\theta'_1| - |\theta_1|$ かつ $\mathit{state}(\theta'_0) = \mathit{state}(\theta'_1)$.

以下では，M が超決定性であるというときには，その遅れ d が与えられているとする．後の節における計算量の評価の際にも，d は与えられるものとして扱っている．ちなみに，与えられた決定性プッシュダウンオートマトンが超決定性であるかは決定可能であり，さらに遅れ d の値も計算可能である[2),⋆1]．

超決定性言語のクラスの性質については文献 13) を参照されたい．応用上重要な XML Grammar[6] の生成言語や Regular Hedge Language[7] は超決定性言語である．

例 2.3. $M = \langle \{q_1, q_2, q_3\}, \{\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \$\}, \{\bot, \boldsymbol{a}, \boldsymbol{b}\}, \delta, q_1, \bot, \{q_3\} \rangle$ とする．ここで δ は次のように定義される．

$$\delta = \{(q_1, A, a, q_1, Aa) \mid A \in \{\bot, \boldsymbol{a}, \boldsymbol{b}\}, a \in \{\boldsymbol{a}, \boldsymbol{b}\}\ \}$$
$$\cup\ \{(q_1, A, \boldsymbol{c}, q_2, A)\} \cup \{(q_2, a, a, q_2, \varepsilon) \mid a \in \{\boldsymbol{a}, \boldsymbol{b}\}\ \} \cup \{(q_2, \bot, \$, q_3, \varepsilon)\}$$

このオートマトンは，はじめに状態 q_1 で読み込んだ文字をスタックに記憶していき，次

⋆1 遅れを計算する方法の鍵となるアイデアは次のとおり．M に対して，$\{a^n\$ \mid M$ に n 個の連続する ε 遷移がある $\}$ を受理言語とするようなプッシュダウンオートマトン N が比較的簡単に構成できる．M の遅れが有限であるかは N の受理言語が有限であるかに帰着でき，この問題は決定可能である[11]．さらに，$a^n\$ \in L_N$ という問合せを必要なだけ行うことで，遅れの値は容易に計算できる．

に記号 $\boldsymbol{c}$ を読むことで状態 q_2 に移行し，状態 q_2 でスタックに記憶してある文字と読み込む文字を比較していき，ちょうどスタックが空になったところで \$ を読めば受理状態に至る．受理言語は $L_M = \{w\,c\,w^R\,\$ \mid w \in \{\boldsymbol{a}, \boldsymbol{b}\}^*\}$ である．ここで w^R は w を反転した文字列を表す．

このオートマトンが超決定性であることは，次のように確認できる．M が決定性であることは明らか．M は ε 遷移を持たないため，M の遅れは 0 であり，特に有限である．また，状態の変化およびスタック長の変化は，はじめの状態と読み込む文字列のみによって，スタックの内容に依存しない．たとえば，$(q_1, \tilde{A}) \vDash_M^w (p, \tilde{B})$ かつ $w \in \{\boldsymbol{a}, \boldsymbol{b}\}^*$ のとき，$p = q_1$ かつ $|\tilde{A}| + |w| = |\tilde{B}|$ であり，$(q_1, \tilde{A}) \vDash_M^w (p, \tilde{B})$ かつ $w = w_1 \boldsymbol{c} w_2$（$w_1, w_2 \in \{\boldsymbol{a}, \boldsymbol{b}\}^*$）のとき，$p = q_2$ かつ $|\tilde{A}| + |w_1| - |w_2| = |\tilde{B}|$ である．ここで $|w|$ は文字列 w の長さを表す．

3. 型システム

決定性プッシュダウンオートマトン M（超決定性であるとは限らない）に対して，型システム $\mathcal{T}_M$ を定義し，この型システムが包含関係を特徴づけることを証明する．すなわち，与えられた文脈自由文法 G に対して，G が $\mathcal{T}_M$ で型付け可能であることと $L_G \subseteq L_M$ が同値であることを示す．

我々の提案する型システムは，Kobayashi ら[8),9)] の提案した高階再帰スキーム（HORS: Higher-Order Recursion Scheme）と正規木言語の包含判定[⋆1]のための型システムのアイデアに基づく．

この章では，決定性オートマトン M を固定して考える．また，一般性を損なうことなく $L_M \neq \emptyset$ を仮定できる．

3.1 型

型の構文を以下に示す．

$$\begin{array}{ll} \text{基底型} & \theta ::= (q, \tilde{A}) \quad ((q, \tilde{A}) \text{ は読み込みモードの様相}) \\ \text{関数型} & \tau ::= \bigwedge_{i \in I} \theta_i \to \theta \end{array} .$$

ただし，共通型のインデックスの動く範囲 I は任意の集合であり，特に空集合や無限集合であってもよい．M の様相の集合は可算なので，I を可算に限定しても問題は生じない．一点集合 $I_1 = \{1\}$ に対して，$\tau = \bigwedge_{i \in I_1} \theta_i \to \theta$ を単に $\tau = \theta_1 \to \theta$ と書く．2 つのインデックス I と J について，$\{\theta_i \mid i \in I\} = \{\theta_j \mid j \in J\}$ のとき $\bigwedge_{i \in I} \theta_i \to \theta$ と $\bigwedge_{j \in J} \theta'_j \to \theta$ は等しいものと見なす．ここでの等しさは集合的な等しさである．つまり暗黙に，順番の並べ変え，重複する要素の縮約および特定の要素の複製を行うことを許す．

以下ではしばらく，対象 t が型 θ を持つということを $t : \theta$ と書く．これは型の意味を説明するための便宜的な記法であって，厳密なものではない．厳密な型判断および型付け規則の定義は 3.2 節で与える．

型は 2 種類に分けられる．言語のための型である基底型 θ と，言語から言語への関数のための関数型 τ である．基底型 θ は，様相 θ から受理される言語のための型である．つまり，言語 $L \subseteq \Sigma^*\$$ について $L : \theta$ であるのは，$L \subseteq L_M(\theta)$ であるときである．関数型 $\tau = \bigwedge_{i \in I} \theta_i \to \theta$ は言語の上の関数のための型であって，この構文は 2 つの部分に分けられる．1 つは共通型の構成子 $\bigwedge_{i \in I} \theta_i$ であって，$L : \bigwedge_{i \in I} \theta_i$ であるのは，任意の $i \in I$ について $L : \theta_i$ のときである．もう 1 つは関数型の構成子 $\to$ であって，これは通常どおりの意味である．つまり，言語の上の関数 f が型 $\theta \to \theta'$ であるということは，任意の θ 型の言語 $L : \theta$ について $f(L) : \theta'$ であるということを意味する．この 2 つの構成子を組み合わせると，$f : \bigwedge_{i \in I} \theta_i \to \theta$ であるのは，任意の $i \in I$ について $L : \theta_i$ であるような言語 L に対して，つねに $f(L) : \theta$ であるときであると解釈できる．

技術的な利便性から，構文的な制限によって，共通型の構成子は関数型の構成子の左辺にのみ現れるようにしてある．この制限は van Bakel[14)] によって導入されたものであって，これによって十分な型の表現能力と簡潔な型付け規則・証明の両立ができる．

適当な文脈自由文法 G が与えられたとし，$\alpha \in (N \cup \Sigma)^*$ とする．次に導入する型システムでは，終端する記号列 $\alpha\$$ には基底型が付き，終端しない記号列 α には関数型が付くことになる．終端する記号列 $\alpha\$$ について，$\alpha\$: \theta$ は $L_G(\alpha)\$ \subseteq L_M(\theta)$ を意味する．

終端しない記号列に関数型が付くことは，次のように説明できる．終端しない語の集合 $L \subseteq \Sigma^*$ は，終端する語の集合 $L' \subseteq \Sigma^*\$$ を受け取って，$LL' = \{vw\$ \mid v \in L, w\$ \in L'\}$ を返す関数だと見なすことができる．この同一視によって，終端しない記号列 α も自然に言語の上の関数と見なすことができる．

例 3.1. アルファベット $a \in \Sigma$ について，$\theta' \vDash_M^a \theta$ という遷移が存在するならば，$a : \theta \to \theta'$ という型を持つ．なぜなら，任意の $L : \theta$ を満たす言語に対して，$aL = \{aw\$ \mid w\$ \in L\}$ の任意の要素は $\theta' \vDash_M^a \theta \vDash_M^{w\$} \square$ という遷移系列で受理させるためである．

終端しない語の集合 $\{w_1, w_2\}$ について，$\theta \vDash_M^{w_1} \theta_1$ および $\theta \vDash_M^{w_2} \theta_2$ という遷移が存在

⋆1 高階再帰スキームは 1 つの木を生成する文法であるため，正確には包含判定ではなく所属性判定である．

するならば，$\{w_1, w_2\} : \bigwedge_{i\in\{1,2\}} \theta_i \to \theta$ という型を持つことが分かる．試しに，$L : \theta_1$ かつ $L : \theta_2$ を満たす言語 L を任意にとる．このことは，任意の要素 $w\$ \in L$ について，$\theta_1 \vDash_M^{w\$} \square$ および $\theta_2 \vDash_M^{w\$} \square$ という受理に至る遷移系列が存在することを意味する．すると，$\{w_1, w_2\}L = \{w_1 w\$ \mid w\$ \in L\} \cup \{w_2 w\$ \mid w\$ \in L\}$ の任意の要素が，θ から受理されることが分かる．受理に至る遷移系列は，語が $w_1 w\$$ の形のとき $\theta \vDash_M^{w_1} \theta_1 \vDash_M^{w\$} \square$ であり，また $w_2 w\$$ の形のとき $\theta \vDash_M^{w_2} \theta_2 \vDash_M^{w\$} \square$ である．

3.2　判断と型付け規則

型環境（type environment）は 2 つ組 $\Delta_1 \mid \Delta_0$ である．Δ_1 は関数型の型環境で，これは $X : \tau$ の形の非終端記号 X と関数型 τ の組の集合である．Δ_1 の要素数は無限でもよく，また 1 つの非終端記号 X に対して複数の型が対応していてもよい．すなわち，$\{X : \tau_1, X : \tau_2, X : \tau_3\}$ なども許すことにする．言語の型環境 Δ_0 は，変数 x に対する型宣言 $x : \theta$ の集合であって，具体的には $\{x : \theta_1, x : \theta_2, \ldots\}$ という形をしたものである．直感的には x は言語を値域とする変数を意味する．Δ_0 に含まれる変数は x ただ 1 種類であることに注意する．また，1 つの変数に対して複数の型が対応していてもよく，さらに Δ_0 も無限集合であってもよいという点については，Δ_1 と同様である．以降では型環境を書く際には $\{\}$ を省略し，単に $X_1 : \tau_1, X_2 : \tau_2, \ldots \mid x : \theta_1, x : \theta_2, \ldots$ と書く．また，$\{x : \theta_i \mid i \in I\}$ のことを，$x : \bigwedge_{i\in I} \theta_i$ と書く．

型判断は $\Delta_1 \mid \Delta_0 \vdash_M X : \tau$ または $\Delta_1 \mid \Delta_0 \vdash_M a : \tau$ または $\Delta_1 \mid \Delta_0 \vdash_M t : \theta$ の形をしている．ここで，$t = \alpha x$ または $\alpha\$$ である．以下でも，t を αx または $\alpha\$$ を表す変数として持ちいる．**図 1** に型付け規則を示す．(App) ルールにおいて α はつねに長さが 1 の列，つまり $N \cup \Sigma$ の要素であることに注意する．これは，(App) ルールの前提部分で α に関数型が付くことを要求しているが，関数型を結論に持つルールは (Terminal) ルールと (Non-Terminal) ルールしかない，という事実から分かる．

次の 2 つの条件を満すとき，$\Delta_1 \vdash_M G$ と書く．

(1)　任意の書き換え規則 $X \to \alpha \in R$ と，$X : \tau \in \Delta_1$ を満す任意の $\tau = \bigwedge_{i\in I} \theta_i \to \theta$ について，$\Delta_1 \mid x : \bigwedge_{i\in I} \theta_i \vdash_M \alpha x : \theta$ である．

(2)　$\Delta_1 \mid \emptyset \vdash_M S\$: \theta_S$ である．

直観的には，(1) は型が相互再帰的定義に合っていることを要求しており，また (2) は開始記号が開始状態から受理されるという条件を表している．

3.3　型システムの健全性

この節では，型システムの健全性，すなわち $\Delta_1 \vdash_M G$ ならば $L_G \subseteq L_M$ であることを

$$\frac{x : \theta \in \Delta_0}{\Delta_1 \mid \Delta_0 \vdash_M x : \theta} \quad \text{(Lang-Var)}$$

$$\frac{\theta \vDash_M^{\$} \square}{\Delta_1 \mid \Delta_0 \vdash_M \$: \theta} \quad \text{(Lang-Const)}$$

$$\frac{\theta \vDash_M^{a} \theta'}{\Delta_1 \mid \Delta_0 \vdash_M a : \theta' \to \theta} \quad \text{(Terminal)}$$

$$\frac{X : \tau \in \Delta_1}{\Delta_1 \mid \Delta_0 \vdash_M X : \tau} \quad \text{(Non-Terminal)}$$

$$\frac{\Delta_1 \mid \Delta_0 \vdash_M \alpha : \bigwedge_{i\in I} \theta_i \to \theta \qquad \Delta_1 \mid \Delta_0 \vdash_M t : \theta_i \text{ (for all } i \in I)}{\Delta_1 \mid \Delta_0 \vdash_M \alpha t : \theta} \quad \text{(App)}$$

図 1　型付け規則
Fig. 1　Typing rules.

証明する．証明は一般的な型システムの健全性証明の流れに従っている．

まずは，文脈自由文法による導出は型を保存することを示す．

補題 3.2. $\Delta_1 \mid x : \bigwedge_{i\in I} \theta_i \vdash \alpha x : \theta$ かつ任意の $i \in I$ について $\Delta_1 \mid \Delta_0 \vdash t : \theta_i$ とする．このとき，$\Delta_1 \mid \Delta_0 \vdash \alpha t : \theta$ である．

証明．α の長さに関する簡単な帰納法による．　□

補題 3.3（導出による型の保存性（Preservation））. $\Delta_1 \vdash_M G$ とする．もし $\Delta_1 \mid \emptyset \vdash_M \alpha\$: \theta$ かつ $\alpha \Rightarrow_G^* \beta$ ならば $\Delta_1 \mid \emptyset \vdash_M \beta\$: \theta$.

証明．$\alpha \Rightarrow_G \beta$ の場合さえ示せれば十分．

$\alpha \Rightarrow_G \beta$ を仮定する．このとき，ある非終端記号 X と規則 $X \to \gamma \in R$ があり，$\alpha = \alpha_1 X \alpha_2 \Rightarrow_G \alpha_1\gamma\alpha_2 = \beta$ と書ける．命題を α_1 の長さに関する帰納法で示す．

$\alpha_1 = \varepsilon$ の場合．仮定から $\Delta_1 \mid \emptyset \vdash_M X\alpha_2\$: \theta$ である．項の形から，ある $\tau = \bigwedge_{i\in I} \theta_i \to \theta$ が存在して，$X : \bigwedge_{i\in I} \theta_i \to \theta \in \Delta_1$ かつ任意の $i \in I$ について $\Delta_1 \mid \emptyset \vdash_M \alpha_2\$: \theta_i$ であることが分かる．$\Delta_1 \vdash G$ の定義より，$\Delta_1 \mid x : \bigwedge_{i\in I} \theta_i \vdash_M \gamma x : \theta$ である．補題 3.2 により，$\Delta_1 \mid \emptyset \vdash_M \gamma\alpha_2\$: \theta$ である．

帰納法のステップを示すことは容易．　□

次に，型 θ は言語 $L_M(\theta)$ を意味するという直感の妥当性を見る．以下の定理において，Δ_1 は任意でよいことに注意する．これは，$w\$$ が自由変数を持たないためである．

補題 3.4. $\Delta_1 \mid \emptyset \vdash_M w\$: \theta$ ならば，$w\$ \in L_M(\theta)$ である．

証明．w の長さに関する帰納法による．$w = \varepsilon$ の場合は明らか．$w = aw'$ を仮定する．

$\Delta_1 \mid \emptyset \vdash_M aw'\$: \theta$ の導出木の形から，ある θ' が存在して $\theta \vDash^a_M \theta'$ かつ $\Delta_1 \mid \emptyset \vdash_M w'\$: \theta'$ であることが分かる．帰納法の仮定により $w' \in L_M(\theta')$ であり，つまり $\theta' \vDash^{w'\$}_M \square$ という受理に至る遷移系列がある．したがって $\theta \vDash^a_M \theta' \vDash^{w'\$}_M \square$ という遷移系列で $aw'\$$ は θ から受理される．□

上の補題から，以下の健全性が得られる．

定理 3.5（健全性（Soundness））. ある Δ_1 について $\Delta_1 \vdash_M G$ ならば，$L_G \subseteq L_M$.

証明．$w \in \Sigma^*$ を $S \Rightarrow^*_G w$ を満す任意の終端しない語とする．$w\$ \in L_M$ をいえばよい．

$\Delta_1 \vdash G$ であるので，特に $\Delta_1 \mid \emptyset \vdash_M S\$: \theta_S$ である．導出による型の保存性（補題 3.3）より，$\Delta_1 \mid \emptyset \vdash_M w\$: \theta_S$ である．補題 3.4 により，$w\$ \in L_M(\theta_S)$ である．□

3.4　型システムの完全性

文脈自由文法 G を固定して考える．$L_G \subseteq L_M$ を仮定して，$\Delta_G \vdash_M G$ を満たす Δ_G を構成することで完全性の証明を行う．読み込みモードの様相 θ と終端しない語 w について，θ_w を $\theta \vDash^w_M \theta_w$ を満たす読み込みモードの様相と定義する．M の決定性から，θ_w は存在すれば一意に定まる．そのような様相が存在しない場合には，θ_w は未定義とする．

関数型環境 Δ_G を次のように定義する．

$$\Delta_G = \left\{ X : \bigwedge_{w \in L_G(X)} \theta_w \to \theta \;\middle|\; X \in N, \forall w \in L_G(X).\ \theta_w \text{ が定義される} \right\}$$

Δ_G の定義は，直観的には次のことをいっている．いま語 $v\$ \in \bigcap_{w \in L_G(X)} L_M(\theta_w)$ を任意に選ぶ．このとき，$L_G(Xv)\$ \subseteq L_M(\theta)$ である．なぜならば，任意の語 $u \in L_G(Xv)$ はある $u' \in L_G(X)$ を用いて $u = u'v$ と分解でき，遷移 $\theta \vDash^{u'}_M \theta_{u'} \vDash^{v\$}_M \square$ を持つためである．ここで遷移の前半部分の存在は関数型環境 Δ_G の定義から，遷移の後半部分の存在は $v\$ \in \bigcap_{w \in L_G(X)} L_M(\theta_w) \subseteq L_M(\theta_{u'})$ からいえる．

このとき，任意の $\alpha \in (\Sigma \cup N)^*$ に対しても同様の型判断を導出することができる．

補題 3.6. $\alpha \in (\Sigma \cup N)^*$，$\theta$ を読み込みモードの様相とする．任意の $w \in L_G(\alpha)$ について θ_w が定義されるならば，$\Delta_G \mid x : \bigwedge_{w \in L_G(\alpha)} \theta_w \vdash_M \alpha x : \theta$.

証明．α の長さに関する帰納法による．$\alpha = \varepsilon$ の場合は明らか．$\alpha = X\alpha'$ とする．$\alpha = a\alpha'$ の場合も同様．

語 u を $u \in L_G(X)$ を満す任意の語とする．ここで v を $v \in L_G(\alpha')$ を満す語とすると，仮定より $\theta \vDash^{uv}_M \theta_{uv}$ である．ここから特に $\theta \vDash^u_M \theta_u$ となる読み込みモードの様相 θ_u が存在することが分かる．また M の決定性から，このような θ_u は語 u に対して唯 1 つ存在することが分かる．

このとき，α' と θ_u は補題の仮定を満足する．つまり，任意の $v \in L_G(\alpha')$ について，$(\theta_u)_v = \theta_{uv}$ は定義される．したがって帰納法の仮定によって，$\Delta_G \mid x : \bigwedge_{v \in L_G(\alpha')} \theta_{uv} \vdash_M \alpha' x : \theta_u$ を得る．ここで，$\{uv \mid v \in L_G(\alpha')\} \subseteq \{w \mid w \in L_G(X\alpha')\}$ であることから，簡単な議論によって $\Delta_G \mid x : \bigwedge_{w \in L_G(X\alpha')} \theta_w \vdash_M \alpha' x : \theta_u$ が導出できることが分かる．

ここで，任意の $u \in L_G(X)$ について $\theta \vDash^u_M \theta_u$ であるため，Δ_G の定義から $X : \bigwedge_{u \in L_G(X)} \theta_u \to \theta \in \Delta_G$ であることが分かる．また任意の $u \in L_G(X)$ について $\Delta_G \mid x : \bigwedge_{w \in L_G(X\alpha')} \theta_w \vdash_M \alpha' x : \theta_u$ であることから，(App) ルールを用いることで $\Delta_G \mid x : \bigwedge_{w \in L_G(X\alpha')} \theta_w \vdash_M X\alpha' x : \theta$ を得る．□

補題 3.7. $L_G \subseteq L_M$ とすると，$\Delta_G \vdash_M G$.

証明．まず任意の書き換え規則 $(X \to \alpha) \in R$ と型宣言 $X : \bigwedge_{w \in L_G(X)} \theta_w \to \theta \in \Delta_G$ について，$\Delta_G \mid x : \bigwedge_{\theta_w \in L_G(X)} \theta_w \vdash_M \alpha x : \theta$ を示す．Δ_G の定義より，任意の $w \in L_G(X)$ について，遷移 $\theta \vDash^w_M \theta_w$ が存在する．特に任意の $u \in L_G(\alpha) \subseteq L_G(X)$ についてこれがいえるため，補題 3.6 から $\Delta_G \mid x : \bigwedge_{u \in L_G(\alpha)} \theta_u \vdash_M \alpha x : \theta$ を得る．$\{x : \theta_u \mid u \in L_G(\alpha)\} \subseteq \{x : \theta_w \mid w \in L_G(X)\}$ であることから，$\Delta_G \mid x : \bigwedge_{w \in L_G(X)} \theta_w \vdash_M \alpha x : \theta$ を得る．

次に $\Delta_G \mid \emptyset \vdash S\$: \theta_S$ を示す．ここで θ_S は初期様相である．$L_G \subseteq L_M$ であるので，任意の $w\$ \in L_G$ について $w\$ \in L_M = L_M(\theta_S)$ であり，$\theta_S \vDash^w_M \theta_w \vDash^{\$}_M \square$ となる遷移系列が存在することが分かる．これと Δ_G の定義から，$S : \bigwedge_{w \in L_G(S)} \theta_w \to \theta_S \in \Delta_G$ である．加えて，先ほどの遷移系列から，任意の $w \in L_G(S)$ について $\theta_w \vDash^{\$}_M \square$ であることが分かり，これと (Lang-Const) ルールから $\Delta_G \mid \emptyset \vdash_M \$: \theta_w$ が任意の $w \in L_G(S)$ についていえる．ここから (App) ルールによって $\Delta_G | \emptyset \vdash S\$: \theta_S$ を得る．□

上の補題の直接の帰結として，型システムの完全性が得られる．

定理 3.8 (完全性 (Completeness))． $L_G \subseteq L_M$ ならば関数型環境 Δ_1 が存在して $\Delta_1 \vdash G$.

さて，$L_G \subseteq L_M$ を仮定して，$\Delta_G \vdash G$ となる関数型環境 Δ_G を定義したわけだが，これよりももう少し（集合の包含関係の意味で）小さな関数型環境でも G を型付けすることができる．つまり Δ_G には不要な型宣言が含まれている．このことを以下に見る．

定義 3.9． $\alpha \in (\Sigma \cup N)^*$ を列とし，$\theta = (q, \tilde{A})$ を読み込みモードの様相とする．θ が α において有効であるとは，$S \Rightarrow_G^* w\alpha u$ かつ $\theta_S \vDash^w (q, \tilde{B}\tilde{A})$ を満たす w，u および $\tilde{B}$ が存在することをいう．

関数型環境 Δ_G^s を次のように定義する．

$$\Delta_G^s = \left\{ X : \bigwedge_{i \in I} \theta_i \to \theta \;\middle|\; X : \bigwedge_{i \in I} \theta_i \to \theta \in \Delta_G \text{ かつ } \theta \text{ は } X \text{ において有効} \right\}$$

$\Delta_G^s \subseteq \Delta_G$ は明らかであり，また $\Delta_G^s \vdash_M G$ である．

補題 3.10． $L_G \subseteq L_M$ を仮定する．このとき，$\Delta_G^s \vdash_M G$ である．

証明．$\Delta_G \vdash_M G$ の証明と同様．まずは補題 3.6 に相当する命題を証明し，次いで補題 3.7 と同様の証明を行うことで $\Delta_G^s \vdash_M G$ を得る．

このとき，以下の事実を利用する．(1) θ が X において有効で $(X \to \alpha) \in R$ ならば，θ は α において有効，(2) θ が $a\alpha$ において有効で $\theta \models_M^a \theta_a$ ならば，θ_a は α において有効，(3) θ が $X\alpha$ において有効で $X : \bigwedge_{i \in I} \theta_i \to \theta \in \Delta_G$ ならば，任意の $i \in I$ について θ_i は α において有効．□

4. 決定可能性

前の章では，決定性プッシュダウンオートマトン M に対して $\mathcal{T}_M$ を定義し，これが完全かつ健全であることを見た（定理 3.5 および定理 3.8）．しかしながら，与えられた文脈自由言語 G に対して，$\Delta_1 \vdash_M G$ がとなる Δ_1 の存在が決定可能であるかは明かではない[⋆1]．

この章の目的は，M を超決定性プッシュダウンオートマトンとした際の型システムの型付け可能性が決定可能であることを示すことである．型システムの完全性と健全性に注意すると，型付け可能性が決定可能であることは，包含判定が決定可能であることを意味する．

⋆1 一般の決定性プッシュダウンオートマトン M と文脈自由文法 G に対しては決定不能である[15]．

4.1 部分型の導入

3 章において，言語の包含判定問題を型付け可能性の問題，つまり $\Delta_1 \vdash_M G$ を満たす Δ_1 の存在を問う問題に帰着した．しかし一般には，条件を満たす Δ_1 は無限集合しかない場合もあり，このことが型付け可能性問題を難しくしている．

この問題を解決するため，型システムを拡張して部分型関係を導入する．部分型を導入し，1 つの型で無限の型を代表できるようにすることで，多くの場合に有限の関数型環境で型付けできるようにする．後に 4.2 節で見るように，部分型関係の導入の結果，M が超決定性プッシュダウンオートマトンであるならば，無限の大きさの関数型環境は不要となることが分かる．

部分型を導入する際の鍵になる観察は，次のものである：プッシュダウンオートマトンの動作はスタックの上の方にのみ依存し，一定よりも下のスタックの内容には依存しない．たとえば，$(q, A, a, q'\tilde{A}') \in \delta$ という遷移規則があったとする．この遷移規則を使えば，$(q, \tilde{B}A) \Vdash_M^a (q', \tilde{B}\tilde{A}')$ という 1 ステップの遷移や，$(q, \tilde{C}\tilde{B}A) \Vdash_M^a (q', \tilde{C}\tilde{B}\tilde{A}')$ という 1 ステップ遷移や，$(q, A) \Vdash_M^a (q', \tilde{A}')$ という 1 ステップ遷移が可能である．このように様々な遷移が可能であるが，これらの遷移の可否に影響を与えているのは状態 q とスタックトップのシンボル A だけであって，$\tilde{B}$ や $\tilde{C}$ はこの遷移の可否には影響していない．つまり，$(q, A) \Vdash_M^a (q', \tilde{A})$ という遷移がより本質的であって，他の遷移はこの遷移から派生したものであると考えることができる．つまり，最も本質的な遷移 $(q, A) \Vdash_M^a (q', \tilde{A}')$ のスタックの底に余分なスタックシンボル列 $\tilde{B}$ を加えたものが $(q, \tilde{B}A) \Vdash_M^a (q', \tilde{B}\tilde{A}')$ であり，さらに余分なスタックシンボル列 $\tilde{C}$ を加えたものが $(q, \tilde{C}\tilde{B}A) \Vdash_M^a (q', \tilde{C}\tilde{B}\tilde{A}')$ であると考えることができる．このように，与えられた 1 ステップ遷移のスタックの底に余分なスタック列を作ることで，派生的な別の 1 ステップ遷移を構成することが可能である．

同様の操作は，複数ステップの遷移 $(q, \tilde{A}) \vDash_M^w (q', \tilde{A}')$ に対しても可能である．つまり，そのスタックの底に任意のスタック列 $\tilde{B}$ を加えることによって，新たな遷移系列 $(q, \tilde{B}\tilde{A}) \vDash_M^w (q', \tilde{B}\tilde{A}')$ を得ることができる．

この直観から，部分型関係 $\preceq$ を定義する．つまり，$\tau_1 \preceq \tau_2$ であるのは，τ_1 のスタックの底に余分なスタック記号を詰めた結果，τ_2 が得られるときであるようにする．形式的な定義は次のとおりである．

定義 4.1（部分型関係）． 部分型関係 $\preceq$ を次の条件を満たす最小の関係と定義する：

任意のスタック列 $\tilde{B}$ について，

$$\textstyle\bigwedge_{i \in I}(q_i, \tilde{A}_i) \to (q, \tilde{A}) \preceq \bigwedge_{i \in I}(q_i, \tilde{B}\tilde{A}_i) \to (q, \tilde{B}\tilde{A}).$$

$\tau \preceq \tau'$ かつ $\tau \neq \tau'$ であることを，$\tau \precneqq \tau'$ と書く．

$\preceq$ の反射性，推移性，反対称性は容易に分かる．したがって，$\preceq$ は半順序関係である．

次の部分型関係に関する型規則を導入する．

$$\frac{\Delta_1 \mid \Delta_0 \vdash \alpha : \tau_1 \qquad \tau_1 \preceq \tau_2}{\Delta_1 \mid \Delta_0 \vdash \alpha : \tau_2} \quad (\textsc{Sub})$$

注意 4.2.（1）α は関数型を持つ任意の列でよいのだが，型付け規則の形から関数型を持つことができるのは $\alpha \in N \cup \Sigma$ である場合に限る．（2）$\alpha = a \in \Sigma$ についての部分型規則は冗長である．つまり，$\alpha = a$ については部分型規則の有無にかかわらず付く型は同じである．したがって，上の規則は $\alpha = X \in N$ に制限しても問題ない．（3）α には x は出現しないため，上の型規則において Δ_0 は重要ではない．したがって，上の規則は，次の形の部分型規則 (Sub') と，Δ_0 への弱化規則の組合せだと理解してもよい．

$$\frac{\Delta_1 \mid \emptyset \vdash \alpha : \tau_1 \qquad \tau_1 \preceq \tau_2}{\Delta_1 \mid \emptyset \vdash \alpha : \tau_2} \quad (\textsc{Sub'})$$

注意 4.3. 部分型の規則は 3.1 節で述べた意味論に対して健全ではない．前に述べた意味論は，$\bigwedge_{i\in I} \theta_i \to \theta$ の意味を，$\forall L.(\forall i \in I.L : \theta_i)$ ならば $f(L) : \theta$）としていた．仮に関数 f を $f(L) = L_1 L$（$L_1 \subseteq \Sigma^*$）という形で定義可能なものに限ったとしても，一般には部分型規則は意味論に対して健全ではない．

部分型の規則は，次のような意味論に基づいている．まず，関数は $f(L) = L_1 L$（$L_1 \subseteq \Sigma^*$）と定義可能な形に限る．そのうえで，関数 $f(L) = L_1 L$ が $\bigwedge_{i\in I} \theta_i \to \theta$ 型であるのは，$\forall w \in L_1.\exists i \in I.\theta \vDash^w_M \theta_i$ である場合と定義する．この条件は，上で述べた条件よりも強いものとなっている．(Sub) ルールを含むすべての型付け規則は，この意味論に対して健全である．

部分型規則に類似の規則を，基底型の型判断に対しても適用することができる．

補題 4.4. Δ_1 を関数型環境，τ および σ を関数型とする．$\tau \preceq \sigma$ を仮定し，$\tau = \bigwedge_{i\in I} \theta_i \to \theta$，$\sigma = \bigwedge_{i\in I} \phi_i \to \phi$ とする．このとき，$\Delta_1 \mid x : \bigwedge_{i\in I} \theta_i \vdash_M \alpha x : \theta$ ならば $\Delta_1 \mid x : \bigwedge_{i\in I} \phi_i \vdash_M \alpha x : \phi$ である．

証明. $\Delta_1 \mid \{x : \theta_i \mid i \in I\} \vdash_M \alpha x : \theta$ の導出に関する簡単な帰納法による．□

部分型を導入した型システムと以前に定義した型システムの関係を見る．そのために，関数型環境の部分型閉包という概念を定義する．

定義 4.5（部分型閉包）. 関数型環境 Δ_1 に対して部分型閉包 $\overline{\Delta}_1$ を次のように定義する．

$$\overline{\Delta}_1 = \{X : \tau' \mid \exists \tau.X : \tau \in \Delta_1 \text{ かつ } \tau \preceq \tau'\}$$

次の意味で，部分型を導入した型システムと部分型を持たない型システムは同値となる．

補題 4.6. M を決定性プッシュダウンオートマトン，Δ_1 を関数型環境とする．このとき，$\Delta_1 \mid \Delta_0 \vdash_M \alpha x : \theta$ が導出できることと，$\overline{\Delta}_1 \mid \Delta_0 \vdash_M \alpha x : \theta$ が (Sub) ルールを用いずに導出できることは同値である．

証明. (Sub) ルールは関数型にのみ用いられることに注意すると，次の命題を示せば十分：任意の $\alpha \in N \cup \Sigma$ について，$\Delta_1 \mid \Delta_0 \vdash_M \alpha : \tau$ が導出できることと $\overline{\Delta}_1 \mid \Delta_0 \vdash_M \alpha : \tau$ が (Sub) ルールを用いずに導出できることは同値である．

まずは左から右を示す．つまり，(Sub) ルールを含んだ導出を含まないものに変換する．型付け規則の形から，(Sub) ルールが現れるときの形は，(Terminal) ルールに続いて (Sub) ルールが何度か現れる形か，(Non-Terminal) ルールに続いて (Sub) ルールが何度か現れる形に限られる．はじめのケースは 1 つの (Terminal) ルールに置き換えられることが容易に分かる．(Non-Terminal) ルールに続いて (Sub) ルールが続く形を考える．つまり，$X : \sigma \in \Delta_1$ かつ $\sigma \preceq \tau$ より $\Delta_1 \mid \Delta_0 \vdash_M X : \tau$ を導いているケースである．このとき，$X : \tau \in \overline{\Delta}_1$ であるため，$\overline{\Delta}_1 \mid \Delta_0 \vdash_M X : \tau$ が (Non-Terminal) ルールのみを用いて，(Sub) ルールを用いることなく導出できる．

次に右から左を証明する．つまり，$\alpha \in \Sigma \cup N$ について，$\overline{\Delta}_1 \mid \Delta_0 \vdash_M \alpha : \tau$ が (Sub) ルールを用いずに得られるならば，$\Delta_1 \mid \Delta_0 \vdash_M \alpha : \tau$ が得られることを示す．$\alpha \in \Sigma$ の場合は容易．$\alpha \in N$，つまり α がある非終端記号 X である場合について考える．$\overline{\Delta}_1 \mid \Delta_0 \vdash_M X : \tau$ が (Sub) ルールを用いずに得られることを仮定する．このとき，$X : \tau \in \overline{\Delta}_1$ であることは容易に分かる．$\overline{\Delta}_1$ の定義から，ある関数型 σ が存在して，$X : \sigma \in \Delta_1$ かつ $\sigma \preceq \tau$ である．したがって，まず (Non-Terminal) ルールを用いて，次に (Sub) ルールを用いることで，求める型判断 $\Delta_1 \mid \Delta_0 \vdash_M X : \tau$ を得ることができる．□

この補題から，(Sub) のある型システムとない型システムに関する，次の同値性を得る．

補題 4.7. $\Delta_1 \vdash_M G$ であることと，$\overline{\Delta}_1 \vdash_M G$ が (Sub) ルールを用いずに導出できることは同値である．

証明．まずは左から右を示す．$\overline{\Delta_1} \mid \emptyset \vdash_M S\$: \theta_S$ が (Sub) ルールを用いずに導出できることは，$\Delta_1 \mid \emptyset \vdash_M S\$: \theta_S$ であることと補題 4.6 より明らか．$X : \tau \in \overline{\Delta_1}$ と $X \to \alpha \in R$ を仮定し，$\tau = \bigwedge_{i\in I} \theta_i \to \theta$ とする．$\overline{\Delta_1} \mid x : \bigwedge_{i\in I} \theta_i \vdash_M \alpha x : \theta$ が (Sub) ルールを用いずに導出できることを示す．

$\overline{\Delta_1}$ の定義から，ある関数型 σ が存在して，$\sigma \preceq \tau$ かつ $X : \sigma \in \Delta_1$ とできる．$\sigma = \bigwedge_{i\in I} \phi_i \to \phi$ とする．$\Delta_1 \vdash_M G$ であるので，$\Delta_1 \mid x : \bigwedge_{i\in I} \phi_i \vdash_M \alpha x : \phi$ である．これと補題 4.4 により $\Delta_1 \mid x : \bigwedge_{i\in I} \theta_i \vdash_M \alpha x : \theta$ を得る．補題 4.6 により $\overline{\Delta_1} \mid x : \bigwedge_{i\in I} \theta_i \vdash_M \alpha x : \theta$ の (Sub) ルールを用いない導出を得る．

次に右から左を示す．$X : \tau \in \Delta_1$ と $X \to \alpha \in R$ を仮定する．$\tau = \bigwedge_{i\in I} \theta_i \to \theta$ とする．$\tau \preceq \tau$ であることから，$X : \tau \in \overline{\Delta_1}$ である．仮定から $\overline{\Delta_1} \vdash_M G$ であるので，$\overline{\Delta_1} \mid x : \bigwedge_{i\in I} \theta_i \vdash_M \alpha x : \theta$ である．また，この導出において (Sub) ルールを必要としない．したがって，補題 4.6 により $\Delta_1 \mid x : \bigwedge_{i\in I} \theta_i \vdash_M \alpha x : \theta$ を得る．$\Delta_1 | \emptyset \vdash_M S\$: \theta_S$ についても同様．□

部分型について拡張した型システムの完全性と健全性は上の補題の帰結である．

定理 4.8（拡張された型システムの完全性および健全性）．任意の文脈自由文法 G について，以下の 2 つは同値である．

(1)　ある関数型環境 Δ_1 が存在し $\Delta_1 \vdash_M G$

(2)　$L_G \subseteq L_M$

ここで，$\Delta_1 \vdash_M G$ は部分型規則を持つ型システムによる導出を意味する．

4.2　決定可能性の証明

この節では，M を超決定性プッシュダウンオートマトンとした場合の型付け可能性問題の決定可能性を示す．証明の戦略は，考慮すべき関数型の種類や関数型環境の種類を有限個におさえるというものである．

関数型 $\tau = \bigwedge_{i\in I} \theta_i \to \theta$ の大きさ $|\tau|$ を，τ に出現する基底型のスタック長の最大値と定義する．つまり，$|\tau| = \sup(\{|\theta_i| \mid i \in I\} \cup \{|\theta|\})$ である．そして，関数型環境 Δ_1 の大きさ $|\Delta_1|$ を，Δ_1 に出現する関数型の大きさの最大値とする：$|\Delta_1| = \sup\{|\tau| \mid X : \tau \in \Delta_1\}$．一般には $|\tau|$ や $|\Delta_1|$ が有限の値を持つとは限らないことに注意されたい．

決定可能性の鍵となるのは，次の補題である．

補題 4.9. G および M から計算可能な定数 K が存在して，次が成立する．$L_G \subseteq L_M$ ならば，$|\Delta_1| < K$ および $\Delta_1 \vdash_M G$ の両方を満たす関数型環境が存在する．□

決定可能性はこの補題からただちに導かれる．$|\Delta_1|$ が有限であるならば関係 $\Delta_1 \vdash_M G$ は決定可能であり，また $|\Delta_1| < K$ を満たす Δ_1 は有限個であるので，そのような Δ_1 を列挙し，$\Delta_1 \vdash_M G$ が成立するかを否かを検査すればよい．不動点計算を用いたより効率の良いアルゴリズムとその計算量については 4.3 節を参照されたい．

それでは補題 4.9 を証明する．以下，この節では，特に断らない限り，超決定性プッシュダウンオートマトン $M = (Q, \Sigma, \Gamma, \delta, q_S, \bot, F)$ と文脈自由文法 $G = (\Sigma, N, S, R)$ を固定し，$L_G \subseteq L_M$ を仮定する．一般性を損なうことなく，M の増分が 1 であることと G がチョムスキー標準形であることを仮定できる．M の遅れを d とする．

はじめに 2 つの補題を示す．1 つ目は超決定性プッシュダウンオートマトンの基礎的な性質であり，Greibach らによって示されたものである．証明は省略する．

補題 4.10（Greibach and Friedman [2]）．M を超決定性プッシュダウンオートマトンとする．θ を M の有効な読み込みモードの様相とする．さらに，終端しない語 x，y，z，u，v が，任意の $n \geq 0$ について

$$xy^n zu^n v\$ \in L_M(\theta)$$

を満たすと仮定し，$n = 1$ の場合の遷移を

$$\theta \vDash_M^x \theta_x \vDash_M^y \theta_y \vDash_M^z \theta_z \vDash_M^u \theta_u \vDash_M^{v\$} \square$$

とする．もし

$$state(\theta_x) = state(\theta_y) \text{ かつ } state(\theta_z) = state(\theta_u)$$

ならば，

$$|\theta_y| - |\theta_x| = |\theta_z| - |\theta_u| \geq 0$$

□

次の補題は，文脈自由言語のポンプの補題の拡張である．まず，補題のための予備的な定義を行う．$\mathcal{I}$ を有限集合とし，その要素をマークと呼ぶ．語 $w = a_1 a_2 \dots a_n$ に対して，関数 $\varrho : \{0, 1, \dots, n\} \to \mathcal{I}$ を語 w に対するマーク付けと呼ぶ．直感的には，語 $a_1 a_2 \dots a_n$ に，$\varrho(0)a_1\varrho(1)a_2\varrho(2)\dots\varrho(n-1)a_n\varrho(n)$ と，マークを挿入すると理解できる．ϱ でマークを付けられた語 $w = a_1 \dots a_n$ とその分解 $w = xvy$ が与えられたとき，v には w から誘導される自然なマーク付けが存在する．先ほどの記法で書くと，w のマーク付けを $\varrho(0)a_1\varrho(1)\dots\varrho(n-1)a_n\varrho(n)$ とすると，その部分列 $v = a_{i+1} \dots a_j$ のマーク付けは $\varrho(i)a_{i+1}\varrho(i+1)\dots\varrho(j-1)a_j\varrho(j)$ とできる．このとき，$\varrho(i)$ を v の左端のマークといい，$\varrho(j)$ を v の右端のマークという．$v = \varepsilon$ で，$x = a_1 \dots a_i$，$y = a_{i+1} \dots a_n$ のときには，v

の左端のマークも右端のマークも $\varrho(i)$ であると定義する．

さて，よく知られている文脈自由文法のポンプの補題は，次のような形をしている．

$w \in L_G$ とし，その長さがある値 k よりも長いと仮定する．このとき $w = w_1w_2w_3w_4w_5$ という分割が存在し（$w_2 \neq \varepsilon$ または $w_4 \neq \varepsilon$），任意の $n \geq 0$ について $w_1w_2^nw_3w_4^nw_5 \in L_G$ である．

この節で述べるポンプの補題の拡張は，w_2 と w_4 を切り出す位置をより細かく指定できるというものである．w はマーク $\mathcal{I}$ でマーク付けされているものとする．このとき，w_2 の左端のマークと右端のマークは一致し，w_4 の右端のマークと左端のマークは一致しているという条件を加えても，ポンプの補題は依然として成立する．形式的には次のように述べられる．

補題 4.11. $G = (\Sigma, N, S, R)$ を文脈自由文法，$\mathcal{I}$ をマークの有限集合とする．このとき，ある定数 k が存在して次の条件を満たす：$w\$ \in L_G$ かつ w の長さは k 以上とし，ϱ を w への I による任意のマーク付けだとする．このとき，$w = w_1w_2w_3w_4w_5$ という分割が存在し，

(1)　$w_2 \neq \varepsilon$ または $w_4 \neq \varepsilon$

(2)　任意の自然数 $n \geq 0$ について，$w_1w_2^nw_3w_4^nw_5\$ \in L_G$

(3)　w_2 の左端のマークと右端のマークは等しく，w_4 の左端のマークと右端のマークも等しい

のすべてを満足する．また，G がチョムスキー標準形であれば，$k = 2^{|N|\cdot|\mathcal{I}|^2} + 1$ が上の条件を満たす．ここで $|N|$ および $|\mathcal{I}|$ はそれぞれ N と $\mathcal{I}$ の要素数である．

証明．G はチョムスキー標準形であるとする．w の生成木を 1 つ選び，以降ではこれを固定して考える．G はチョムスキー標準形なので，生成木は 2 分木であることに注意する．w の長さが $2^{|N|\cdot|\mathcal{I}|^2+1}$ 以上であるので，生成木には長さ $|N| \cdot |\mathcal{I}|^2$ を超える経路が存在する．たとえば，**図 2** のような形であるとする．そのような経路の長さを j とする．

w の部分列 u_i^l と u_i^r を以下のように定義する．

- $1 \leq i < j$ について，X_{i+1} が X_i の左の子供ならば，$u_i^l = \varepsilon$ であり，u_i^r は X_i の右部分木が生成する語とする．
- $1 \leq i < j$ について，X_{i+1} が X_i の右の子供ならば，u_i^l は X_i の左部分木が生成する語であり，$u_i^r = \varepsilon$ とする．
- $i = j$ ならば，$u_j^l = a$ かつ $u_j^r = \varepsilon$ であるとする．ここで a は X_j が生成する語である．

図 2 においては，u_1^r は図で示された部分木の生成する語であり，$u_1^l = \varepsilon$ である．このとき，X_i を根とする部分木の生成する語は $u_i^lu_{i+1}^l \ldots u_j^lu_j^r \ldots u_{i+1}^ru_i^r$ と書ける．これを v_i と書くことにする．

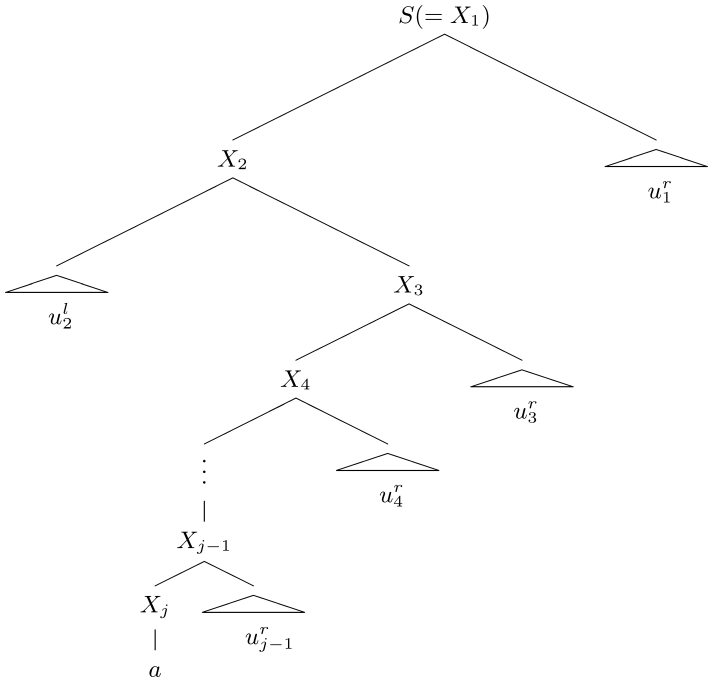

図 2　長さが $|N| \cdot |\mathcal{I}|^2$ を超える経路 $X_1X_2 \ldots X_j$
Fig. 2　A path $X_1X_2 \ldots X_j$ with the length $> |N| \cdot |\mathcal{I}|^2$.

ここで，集合 $\{i \mid 1 \leq i \leq j\}$ に同値関係 $\sim$ を導入する．$i_1 \sim i_2$ であるのは，

- X_{i_1} と X_{i_2} が非終端記号として等しい，
- v_{i_1} の左端のマークと v_{i_2} の左端のマークが等しい，
- v_{i_1} の右端のマークと v_{i_2} の右端のマークが等しい，

のすべてを満たすときであると定義する．同値類の個数はたかだか $|N| \cdot |\mathcal{I}|^2$ であり，$\{i \mid 1 \leq i \leq j\}$ の要素数は $|N| \cdot |\mathcal{I}|^2$ よりも大きいため，鳩の巣原理によって $i_1 \sim i_2$ となる $i_1 \neq i_2$ が存在することが分かる．一般性を損なうことなく，$i_1 < i_2$ と仮定することができる．

さて，$w = u_1^l u_2^l \dots u_j^l u_j^r \dots u_2^r u_1^r$ を，$w = w_1 w_2 w_3 w_4 w_5$ と分割する．w_i（$1 \le i \le 5$）は

$$
\begin{aligned}
w_1 &= u_1^l u_2^l \dots u_{i_1-1}^l \\
w_2 &= u_{i_1}^l u_{i_1+1}^l \dots u_{i_2-1}^l \\
w_3 &= u_{i_2}^l u_{i_2+1}^l \dots u_j^l u_j^r \dots u_{i_2+1}^r u_{i_2}^r \\
w_4 &= u_{i_2-1}^r \dots u_{i_1+1}^r u_{i_1}^r \\
w_5 &= u_{i_1-1}^r \dots u_2^r u_1^r
\end{aligned}
$$

と定義される．この分割が 3 つの条件をすべて満たしていることは容易に確認できる． □

さて，3 章において，型システムの完全性（定理 3.8）を示す際に関数型環境 Δ_G を定義し，さらにそこから余分なものを取り除くことによって Δ_G^s を定義した．以下，Δ_G^s の大きさに関して 2 つの補題を示す．1 つ目は，$X : \bigwedge_{i\in I} \theta_i \to \theta \in \Delta_G^s$ とすると，$|\theta_i| - |\theta|$ がある定数 k_1 でおさえられるという結果である．このことは特に，$|\bigwedge_{i\in I} \theta_i \to \theta|$ の型の大きさが θ と k_1 を使っておさえられることを意味する．2 つ目は，$X : \bigwedge_{i\in I} \theta_i \to \theta \in \Delta_G^s$ かつ θ がある定数 k_2 よりも長いスタック長を持っていれば，$\tau' \supsetneq \bigwedge_{i\in I} \theta_i \to \theta$ かつ $X : \tau' \in \Delta_G^s$ を満たす τ' が存在するという結果である．これは本質的には θ の大きさをおさえることができることを意味している．どちらの補題でも，ポンプの補題（補題 4.11）が重要な役割を果たしている．

補題 4.12. $X : \tau \in \Delta_G^s$ を仮定し，$\tau = \bigwedge_{w\in L_G(X)} \theta_w \to \theta$ とする．定数 k_1 を次のように定義する．

$$k_1 = (d+1)\left(2^{|N||Q|^2} + 1\right)$$

すると，任意の $w \in L_G(X)$ について $|\theta_w| < |\theta| + k_1$．

証明．w の長さに関する帰納法で示す．以下，証明中では，$|w|$ で w の長さを表す．文脈自由文法 G_X を $G_X = (\Sigma, N, X, R)$ で定義する．つまり，G の開始記号を X に変更した文法である．

$|w| < 2^{|N||Q|^2} + 1$ の場合．遅れが d であることから，1 つの記号を読んでから次の記号を読むまで，たかだか $d+1$ 回の遷移しか行われない（記号を読む遷移が 1 回とたかだか d 回の ε 遷移）．したがって，$\theta \vDash_M^w \theta_w$ の遷移系列の長さはたかだか $(d+1)|w|$ ステップである．増分が 1 であるので，この遷移系列で変化できるスタック長の最大値も $d \times |w|$ でおさえられる．したがって，$|\theta_w| - |\theta| < (d+1)(2^{|N||Q|^2} + 1)$ が成立する．

$|w| \ge 2^{|N||Q|^2+1}$ とする．$w = a_1 \dots a_n$（$a_i \in \Sigma$）とし，遷移 $\theta \vDash_M^w \theta_w$ を 1 文字ごとに見ることで，

$$\theta = \phi_0 \vDash_M^{a_1} \phi_1 \vDash_M^{a_2} \dots \vDash_M^{a_i} \phi_i \vDash_M^{a_{i+1}} \dots \vDash_M^{a_n} \phi_n = \theta_w$$

という遷移の系列を得る．

ここでマークの集合 $\mathcal{I}$ を $\mathcal{I} = Q$ と定義し，マーク付け ϱ を $\varrho(i) = \mathrm{state}(\phi_i)$ と定義する．すると，G_X，$\mathcal{I}$，ϱ および w は，補題 4.11 の条件を満たすことが分かる．したがって，$w = xyzuv$ という分割（$y \ne \varepsilon$ または $u \ne \varepsilon$）が存在し，任意の n について $xy^n zu^n v \in L_{G_X} = L_G(X)$ かつ y の左端と右端のマークは等しく，u の左端のマークと右端のマークも等しい．

$$\theta \vDash^x \phi_x \vDash^y \phi_y \vDash^z \phi_z \vDash^u \phi_u \vDash^v \phi_v = \theta_w$$

とする．マーク付け ϱ の定義と，y の左端のマークと右端のマークが等しいことから $\mathrm{state}(\phi_x) = \mathrm{state}(\phi_y)$ であることが分かり，同様に $\mathrm{state}(\phi_z) = \mathrm{state}(\phi_u)$ が分かる．

さらに，$\theta = (q, \tilde{A})$ は X において有効であるので，ある語 w_1，w_2 およびスタック列 $\tilde{B}$ が存在して，$S \to_G^* w_1 X w_2$ かつ $\theta_S \vDash_M^{w_1} (q, \tilde{B}\tilde{A})$ とできる．いま $L_G \subseteq L_M$ であることから $L_G(Xw_2) \subseteq L_M(\ (q, \tilde{B}\tilde{A})\ )$ であり，特に任意の n について $xy^n zu^n vw_2 \in L_M(\ (q, \tilde{B}\tilde{A})\ )$ となる．

以上のことから，この分解 $ww_2 = xyzu(vw_2)$ は補題 4.10 の条件を満たす．したがって，$|\phi_y| - |\phi_x| = |\phi_z| - |\phi_u|$ がいえる．

ここで，θ からの xzv による遷移を考える．これを

$$\theta \vDash^x \phi'_x \vDash^z \phi'_z \vDash^v \phi'_v = \theta_{xzv}$$

と書くことにする．このとき $|\theta_{xzv}| = |\theta_w|$ であることを示す．

まずは状態が一致することを見る．M の決定性から $\phi_x = \phi'_x$ が分かり，特に $\mathrm{state}(\phi_x) = \mathrm{state}(\phi'_x)$ である．次に $\mathrm{state}(\phi_x) = \mathrm{state}(\phi_y)$ から，$\mathrm{state}(\phi_y) = \mathrm{state}(\phi'_x)$ である．M は超決定性なので，ここから $\mathrm{state}(\phi_z) = \mathrm{state}(\phi'_z)$ が分かる．再び $\mathrm{state}(\phi_z) = \mathrm{state}(\phi_u)$ より $\mathrm{state}(\phi_u) = \mathrm{state}(\phi'_z)$ が分かる．すると M の超決定性により $\mathrm{state}(\phi_v) = \mathrm{state}(\phi'_v)$ となる．

状態がそれぞれ等しいことから，M の超決定性によって，$|\phi_z| - |\phi_y| = |\phi'_z| - |\phi'_x|$ および $|\phi_v| - |\phi_u| = |\phi'_v| - |\phi'_z|$ を得る．いま $|\phi_y| - |\phi_x| = |\phi_z| - |\phi_u|$ であることから，簡単な計算によって $|\phi_v| = |\phi'_v|$，すなわち $|\theta_w| = |\theta_{xzv}|$ が得られる．

$xzu \in L_G(X)$ であり，これは w よりも真に短いことから，帰納法の仮定によって命題の成立がいえる．　□

補題 4.13. 定数 k_2 を次のように定義する．

$$k_2 = (d+1)\left(2^{|N|\cdot(2|Q|)^2}+1\right)$$

$X : \tau \in \Delta_G^s$ とし，$\tau = \bigwedge_{w\in L_G(X)} \theta_w \to \theta$ とする．もしも $|\theta| > k_2$ ならば，ある $\tau' \not\preceq \tau$ が存在して，$X : \tau' \in \Delta_G^s$．

証明．$X : \tau \in \Delta_G^s$ を満たす τ をとり，$\tau = \bigwedge_{w\in L_G(X)} \theta_w \to \theta$ とする．$|\theta| > k_2$ を仮定する．鍵となるのは次の主張である．

任意の $w \in L_G(X)$ について，遷移 $\theta \models_M^w \theta_w$ はスタック長が 2 以上の様相しか経由しない．

まず，この主張を仮定し，ここから命題が帰結することを示す．w を $w \in L_G(X)$ を満たす任意の終端しない語とし，$\theta = (q, \tilde{A})$，$\theta_w = (q_w, \tilde{A}_w)$ とする．$\theta \models_M^w \theta_w$ の遷移を ε 遷移を含めて 1 ステップごとに書くと，

$$\theta = (q,\tilde{A}) \Vdash_M^{a_1} (q_1,\tilde{A}_1) \Vdash_M^{a_2} \ldots \Vdash_M^{a_i} (q_i,\tilde{A}_i) \Vdash_M^{a_{i+1}} \ldots \Vdash_M^{a_n} (q_w,\tilde{A}_w)$$

となるとする．ここで $a_i \in \Sigma \cup \{\varepsilon\}$（$1 \le i \le n$）であり，$w = a_1 \ldots a_n$ である（1 ステップの遷移を考えているので，a_i が ε でもよい）．$\tilde{A}$ の一番下のスタック記号を B とすると，$\tilde{A} = B\tilde{A}'$ と書ける．$\tilde{A}_i$ のスタック長は 2 以上であることを考えると，上の遷移は

$$\theta = (q,B\tilde{A}) \Vdash_M^{a_1} (q_1,B\tilde{A}_2') \Vdash_M^{a_2} \ldots \Vdash_M^{a_i} (q_i,B\tilde{A}_i') \Vdash_M^{a_{i+1}} \ldots \Vdash_M^{a_n} (q_w,B\tilde{A}_w')$$

という形をしていることが分かる．ここで，$\tilde{A}_i$ の長さは 1 以上である．したがって，

$$\theta = (q,\tilde{A}) \Vdash_M^{a_1} (q_1,\tilde{A}_2') \Vdash_M^{a_2} \ldots \Vdash_M^{a_i} (q_i,\tilde{A}_i') \Vdash_M^{a_{i+1}} \ldots \Vdash_M^{a_n} (q_w,\tilde{A}_w')$$

も M の遷移となる．これは $X : \bigwedge_{w\in L_G(X)} (q_w,\tilde{A}_w') \to (q,\tilde{A}') \in \Delta_G^s$ を意味する（w が任意であることに注意）．$\bigwedge_{w\in L_G(X)} (q_w,\tilde{A}_w') \to (q,\tilde{A}') \not\preceq \tau$ であることから，命題を得る．

それでは，上の主張を背理法で示す．$G_X = (\Sigma, N, X, R)$ と定義する．遷移 $\theta \models_M^w \theta_w$ がスタック長 1 の様相を経由するような $w \in L_G(X)$ が少なくとも 1 つは存在することを仮定する．そのような w の中で，長さが最小であるものを 1 つ選ぶ．以下，w よりも真に短い語 $v \in L_G(X)$ で $\theta \models_M^v \theta_v$ がスタック長 1 の様相を経由するようなものを構成し，そこから矛盾を導く．

$w = a_1 \ldots a_n$（$a_i \in \Sigma$，つまり $a_i \neq \varepsilon$ とする）とし，

$$\theta = \phi_0 \vDash_M^{a_1} \phi_1 \vDash_M^{a_2} \ldots \vDash_M^{a_i} \phi_i \vDash_M^{a_{i+1}} \ldots \vDash_M^{a_n} \phi_n = \theta_w$$

とする．このうち，$\phi_{j-1} \vDash_M^{a_j} \phi_j$ においてスタック長が 1 である様相を経由するとする．M の遅れが d であることを考慮すると，$j \ge 2^{|N|\cdot(2|Q|)^2}+1$ であることが分かる．特に w の長さは $2^{|N|\cdot(2|Q|)^2}+1$ 以上である．

ここでマークの集合 $\mathcal{I}$ を $\mathcal{I} = \{(\mathtt{l}, q) \mid q \in Q\} \cup \{(\mathtt{r}, q) \mid q \in Q\}$ と定義する．ここで，$\mathtt{l}$ および $\mathtt{r}$ は単なる記号である．マーク付け ϱ を

$$\varrho(i) = \begin{cases} (\mathtt{l}, \mathrm{state}(\phi_i)) & (i < j \text{ のとき}) \\ (\mathtt{r}, \mathrm{state}(\phi_i)) & (i \ge j \text{ のとき}) \end{cases}$$

と定義する．$|\mathcal{I}| = 2|Q|$ であることに注意すると，これらは補題 4.11 の条件を満たすので，$w = xyzuv$ という分割（$y \neq \varepsilon$ または $u \neq \varepsilon$）が存在し，任意の n について $xy^nzu^nv \in L_{G_X} = L_G(X)$ であり，さらに y の左端と右端の記号は等しく，u の左端と右端の記号は等しい．θ は X において有効であることに注意すると，補題 4.12 と同様の議論によって，補題 4.10 が適用できる．したがって，

$$\theta \vDash_M^x \phi_x \vDash_M^y \phi_y \vDash_M^z \phi_z \vDash_M^u \phi_u \vDash_M^v \phi_v = \theta_w$$

とすると，$|\phi_y| - |\phi_x| = |\phi_u| - |\phi_z| \ge 0$ である．θ から語 $xzv \in L_G(X)$ を読む遷移を，

$$\theta \vDash_M^x \phi_x' \vDash_M^z \phi_z' \vDash_M^v \phi_v' = \theta_{xzv}$$

とする．以下，この遷移もスタック長が 1 の様相を経由することを示す．

補題 4.12 における議論と同様にして，$\mathrm{state}(\phi_x) = \mathrm{state}(\phi_y) = \mathrm{state}(\phi_x')$ および $\mathrm{state}(\phi_z) = \mathrm{state}(\phi_u) = \mathrm{state}(\phi_z')$ が得られる．a_j を含む語によって場合分けをする．ϱ の定義から，a_j は y にも u にも含まれていないことが分かる．

a_j が x に含まれる場合，$\theta \vDash_M^x \phi_x$ においてスタック長が 1 の様相を経由する．したがって，M の決定性より，$\theta \vDash_M^x \phi_x'$ においてスタック長が 1 の様相を経由する．

a_j が v に含まれる場合．補題 4.12 における議論と同様の議論によって，$|\phi_z'| = |\phi_u|$ が得られる．$\mathrm{state}(\phi_z') = \mathrm{state}(\phi_u)$ であるので，M の超決定性により ϕ_z' から v を読む場合のスタック長の変化と ϕ_u から v を読む場合のスタック長の変化は等しい．したがって，$\phi_z' \vDash_M^v \phi_v'$ の遷移においてスタック長が 1 の様相を経由する．

a_j が z に含まれる場合．$z = a_m \ldots a_{m+l-1}$（l は z の長さ）として，$\phi_y \vDash_M^z \phi_z$ という遷

移列と，$\phi'_x \vDash^z_M \phi'_z$ という遷移列を比べる．それぞれの遷移列が，次のようであったとする．

$$\phi_y \vDash^{a_m}_M \phi_m \vDash^{a_{m+1}}_M \ldots \vDash^{a_{m+l-1}}_M \phi_{m+l-1} = \phi_z$$
$$\phi'_x \vDash^{a_m}_M \phi'_m \vDash^{a_{m+1}}_M \ldots \vDash^{a_{m+l-1}}_M \phi'_{m+l-1} = \phi'_z$$

いま $|\phi_y| - |\phi_x| \geq 0$ であるので，$|\phi_y| \geq |\phi_x| = |\phi'_x|$ である．$\mathrm{state}(\phi'_x) = \mathrm{state}(\phi_y)$ であることと M の超決定性によって，$|\phi_i| \geq |\phi'_i|$ が任意の $m \leq i \leq m+l-1$ についていえる．したがって，$\phi'_x \vDash^z_M \phi'_z$ においてスタック長 1 以下の様相を経由する．

以上によって，$\theta \vDash^{xzv} \theta_{xzv}$ がスタック長 1 の様相を経由することが分かった．xzv は w よりも真に短いため，これは w の長さの最小性に矛盾する．　□

上の 2 つの補題を使うことによって，十分に小さな関数環境で G が型付けできることが分かる．Δ^s_G をもとに，そのような型環境を具体的に構成する．

定義 4.14. 関数型環境 Δ_1 と非終端記号 X について，$\Delta_1(X) = \{\tau \mid X : \tau \in \Delta_1\}$ と定義する．関数型環境 Δ^m_G を次のように定義する．

$$\Delta^m_G = \{X : \tau \mid X : \tau \in \Delta^s_G \text{ かつ } \tau \text{ は } \Delta^s_G(X) \text{ において極小 }\}$$

関数型環境 Δ^m_G によって G は型付け可能で，さらにその大きさは補題 4.12 および補題 4.13 における定数を用いておさえることができる．

定理 4.15. $|\Delta^m_G| < k_1 + k_2$ かつ $\Delta^m_G \vdash_M G$．ここで k_1 および k_2 は，それぞれ補題 4.12 および補題 4.13 における定数である．

証明．まず $|\Delta^m_G| < k_1 + k_2$ を示す．$X : \tau \in \Delta^m_G$ とし，$\tau = \bigwedge_{i\in I} \theta_i \to \theta$ とする．Δ^m_G の定義から，τ は $\Delta^s_G(X)$ において極小であることに注意する．このとき $|\theta| \leq k_2$ である．なぜなら，もし $|\theta| > k_2$ だとすると補題 4.13 によって $\tau' \precneqq \tau$ および $X : \tau' \in \Delta^s_G$ を満たす τ' が存在することになり，これは τ の極小性に反するからである．補題 4.12 によって，任意の $i \in I$ について $|\theta_i| < k_1 + |\theta|$ であるから，$|\theta_i| < k_1 + k_2$ である．明らかに $|\theta| < k_1 + k_2$ であるから，$|\tau| < k_1 + k_2$ である．

次に $\Delta^m_G \vdash_M G$ を示す．これには Δ^m_G の部分型閉包が Δ^s_G を含むことを示せばよい．すると $\Delta^s_G \vdash_M G$ であることと補題 4.7 によって $\Delta^m_G \vdash_M G$ が得られる．

$X : \tau \in \Delta^s_G$ とする．$X : \tau$ が Δ^m_G の部分型閉包に含まれることを示す．集合 $T = \{\tau' \mid X : \tau' \in \Delta^s_G \text{ かつ } \tau' \preceq \tau\}$ を考える．$\tau \preceq \tau$ であるので，T は空集合ではない．また，$\precneqq$ は無限降下列を持たない，つまり整礎であるので，T は必ず極小元を持つ．これを τ' とすると，$X : \tau' \in \Delta^m_G$ である．$\tau' \preceq \tau$ に注意すると，$X : \tau$ は Δ^m_G の部分型閉包に含まれることが分かる．　□

したがって，$K = k_1 + k_2$ ととり，$\Delta_1 = \Delta^m_G$ とすることで，目指していた補題 4.9 を示すことができた．

4.3　判定手続き

この節では，具体的な判定手続きを与え，その計算量について議論を行う．

まずは具体的な判定手続きを与える．以下に述べる手法は，本質的には Kobayashi[8] で提案されたものと同じである．はじめに示すことは，型環境が有限である場合についての型判断の妥当性検査の決定可能性である．

補題 4.16. ある定数 k が存在して $|\Delta_1| < k$ かつ Δ_0 を有限集合とする．このとき，任意の $\alpha \in (\Sigma \cup N)^*$ と θ について，$\Delta_1 \mid \Delta_0 \vdash_M \alpha x : \theta$ および $\Delta_1 \mid \Delta_0 \vdash_M \alpha\$: \theta$ は決定可能である．

証明．α の長さに関する帰納法による．

$\alpha = \varepsilon$ の場合．$\Delta_1 \mid \Delta_0 \vdash_M x : \theta$ は Δ_0 が有限なので決定可能．また $\Delta_1 \mid \Delta_0 \vdash_M \$: \theta$ については，$\theta \vDash^{\$}_M \square$ を満たす θ が有限であることに注意すれば明らか．

α の長さが 0 でない場合．$\alpha = X\alpha'$ のケースの，$\Delta_1 \mid \Delta_0 \vdash_M X\alpha'\$: \theta$ についてのみ示す．他の場合も同様．まず，集合 $\Theta(X, \theta) = \{\bigwedge_{i\in I} \theta_i \to \theta \mid \Delta_1 \mid \Delta_0 \vdash_M X : \bigwedge_{i\in I} \theta_i \to \theta\}$ を求める．以下ではこれを単に Θ と書く．ここで $\Delta_1 \mid \Delta_0 \vdash_M X : \bigwedge_{i\in I} \theta_i \to \theta\}$ を導出するのに (Sub) ルールを用いてもよい（また，ここでしか用いない）．Δ_1 が有限であることから，Θ を求めることが可能であること，さらに」Θ が有限集合であることが分かる．次に，各々の $\bigwedge_{i\in I} \theta_i \to \theta \in \Theta$ について，$\Delta_1 \mid \Delta_0 \vdash_M \alpha'\$: \theta_i$ がすべての $i \in I$ について成り立つかを調べる．$|\Delta_1| < k$ であることから I が有限集合であることに注意すると，帰納法の仮定からこれは可能である．ある要素 $\bigwedge_{i\in I} \theta_i \to \theta \in \Theta$ があって任意の $i \in I$ について $\Delta_1 \mid \Delta_0 \vdash_M \alpha'\$: \theta_i$ が成り立つとき $\Delta_1 \mid \Delta_0 \vdash_M X\alpha'\$: \theta$ であり，逆もまた成り立つ．　□

さて，文脈自由言語 G に対して，次のように関数型環境から関数型環境への関数 $\mathcal{F}_G$ を定義する．

$$\mathcal{F}_G(\Delta_1) = \left\{ X : \bigwedge_{i\in I} \theta_i \to \theta \in \Delta_1 \;\middle|\; \right.$$

1: $\Delta_1^0 = \{X : \tau \mid |\tau| < K\}$
2: $\Delta_1^1 = \mathcal{F}_G(\Delta_1^0)$
3: $n = 1$
4: $\Delta_1^n = \Delta_1^{n-1}$ が成立するまで以下を繰り返す
　4(a): $\Delta_1^{n+1} = \mathcal{F}_G(\Delta_1^n)$
　4(b): $n = n + 1$
5: $S : \theta_S \in \Delta_1^n$ ならば真を，そうでなければ偽を返す

図 3　G の型付け可能性を判定する手続き
Fig. 3　A procedure deciding G is typable.

$$\left.\text{すべての書き換え規則 } X \to \alpha \in R \text{ について } \Delta_1 \mid x : \bigwedge_{i\in I} \theta_i \vdash_M \alpha x : \theta \right\}$$

$\tau_S = \bigwedge_{i\in I} \theta_i \to \theta_S$（ここで $\{\theta_i \mid i \in I\} = \{\theta \mid \theta \vDash^{\$} \Box\}$）と定義する．すると，$S : \tau_S \in \Delta_1$ と $\Delta_1 \mid \emptyset \vdash_M S\$: \theta_S$ が同値になる．次の補題が成り立つことは容易に分かる．

補題 4.17. 次の 2 つは同値である．
(1)　$\Delta_1 \vdash_M G$.
(2)　Δ_1 は $\mathcal{F}_G$ の不動点かつ $S : \tau_S \in \Delta_1$.　□

以上の準備の下，$\Delta_1 \vdash_M G$ となる関数型環境 Δ_1 が存在するか否かを決定するアルゴリズムを述べる．基本となる戦略は $\mathcal{F}_G$ を繰り返し適用することで不動点を求めることである．**図 3** に示した手続きによって，G の型付け可能性が判定できる．ここで K は補題 4.9 に現れる定数である．

まずは，この手続きの各ステップが計算可能で，さらに繰返しが停止することを示す．任意の Δ_1 について $\Delta_1 \supseteq \mathcal{F}_G(\Delta_1)$ であるので，この手続きで構成する $\Delta_1^0, \Delta_1^1, \ldots, \Delta_1^n$ という系列は減少列である．特に $|\Delta_1^0| < K$ であるので $|\Delta_1^i| < K$ が任意の i について成り立つ．したがって補題 4.16 から $\mathcal{F}_G(\Delta_1^n)$ は計算可能であることが分かる．また Δ_1^0 が有限集合であるので，繰返しはたかだか Δ_1^0 の要素数回で停止する．

この手続きの健全性は補題 4.17 の簡単な帰結である．完全性を示す．$L_G \subseteq L_M$ を仮定し，補題 4.9 で存在が保障されている関数型環境を Δ_1 と書く．$|\Delta_1| < K$ であることから，$\Delta_1 \subseteq \Delta_1^0$ であることは容易に分かる．さらに $\Delta_1 \vdash_M G$ であるので，補題 4.17 より Δ_1 は $\mathcal{F}_G$ の不動点である．ここから任意の n について $\Delta_1 \subseteq \Delta_1^n$ であることが分かり，$S : \tau_S \in \Delta_1$ より $S : \tau_S \in \Delta_1^n$ となる．

次にこの手続きの時間計算量について，簡単に述べる．計算量に対して最も大きな影響を与えるのが，繰返し回数の上界である Δ_1^0 の大きさである．まずはこの大きさを見積もる．補題 4.9 に現れる定数を K と書く．

大きさが K より小さい基底型の集合 $\{\theta \mid |\theta| < K\}$ を，ここでは Θ と書くことにする．集合 Θ の大きさは，$|\Theta| < |Q| \times (|\Gamma|+1)^K$ とおさえることができる．関数型 $\tau = \bigwedge_{i\in I} \theta_i \to \theta$ は基底型の集合 $\{\theta_i \mid i \in I\}$ および基底型 θ の組であるから，大きさが K より小さい関数型の集合は $\mathcal{P}(\Theta) \times \Theta$ である．ここで $\mathcal{P}(\Theta)$ は Θ のべき集合を表す．したがって，関数型環境 Δ_1^0 の大きさは，およそ $|N| \times |\mathcal{P}(\Theta) \times \Theta|$ と評価することができる．$|\Theta|$ の大きさが K に対して指数的に増大するため，$\mathcal{P}(\Theta)$ は二重指数的に増大する．すなわち，Δ_1^0 の大きさは K に対して二重指数的に増大する．

前に述べたとおり，ループ回数は Δ_1^0 の要素数でおさえられることを考えると，図 3 の時間計算量は $L \times |\Delta_1^0|$ 程度でおさえることができる．ここで L はループ 1 回あたりにかかる計算量である．L が K について二重指数以下であることは容易に分かるので，結局，時間計算量は K についての二重指数でおさえることができる．K が文法やオートマトンのサイズに対して指数的であったことを合わせると，時間計算量は文法やオートマトンのサイズに対して（たかだか）三重指数的である．Greibach らのアルゴリズム[2] が二重指数的な増大で済むことに比べると，図 3 の手続きは指数一段分遅いということになる．

注意 4.18. 上記のアルゴリズムを拡張することで，$L_G \not\subseteq L_M$ である場合にその反例，つまり $w\$ \in L_G$ かつ $w\$ \notin L_M$ を満たす $w\$$ を構成することができる．単純な方法としては，$S : \theta_S \notin \Delta_1^i$ であることが分かった時点で，高さが i 以下の構文木を持つ語の中に L_M に含まれないものが存在することが分かるので，そのような語 w を列挙して反例であるか否かを検査すればよい．

5. 関 連 研 究

本研究は，言語の包含判定問題に関する 2 つの結果と密接なつながりがある．

1 つは，Greibach ら[2] によって示された，文脈自由言語と超決定性言語の包含判定の決定可能性である．本論文は，この結果の別証明を与えたものである．超決定性言語はこの論文によって導入された言語クラスであり，他の言語クラスとの比較や超決定性言語の特徴付けなどにも触れられている．この結果のほかにも，決定性プッシュダウンオートマトンの受理言語と超決定性プッシュダウンオートマトンの受理言語の同値性判定が決定可能であるこ

とが証明されている．Nguyen ら[5] は Greibach ら[2] の用いた Alternative Stacking を改良し，より簡潔な構成法で証明できることを明らかにした．

その他，超決定性言語の部分クラスではあるが，応用上重要な言語クラスに関する研究も行われている．Minamide ら[4] は，文脈自由言語と XML Grammar の包含判定問題，文脈自由言語と Regular Hedge Grammar の包含判定問題に対して実用的なアルゴリズムを考案し，これをプログラム検証に応用した．また Tozawa ら[3] は与えられた文脈自由言語が生成する言語に対して，括弧が対応するかを検証することは多項式時間でよいこと，Regular Hedge Grammar との包含判定は二重指数時間完全であることを示した．

本研究と密接に関わっているもう 1 つの研究成果は，高階再帰スキームの正規木言語に対する所属性判定問題である．この問題に対する回答として，まず高階再帰スキームが安全であるとの制限の下での決定可能性が Knapic ら[16] によって示され，次いでこの制限がなくても決定可能であることが Ong[17] によって示された．これらの結果をうけて，小林はこの問題の決定可能性に対する型理論を用いた証明を提案した．まず正規木言語を正則な安全性に制限した場合の所属性判定問題に対する型システムを提案し，これが高階の関数型言語に対するプログラム検証に応用できることを示した[8]．その後，この制限を除いた問題に対する型システムが Kobayashi ら[9] によって提案された．Kobayashi らの提案した型による手法は強力なものであり，これを拡張する研究もいくつか行われている．Kobayashi ら[18] では，高階多引数木トランスデューサーという木から木への変換器を定義したうえで，この生成する木が与えられた正則な安全性を満すかの検証に Kobayashi らの型システムを応用できることを示した．Tsukada ら[19] は，型を用いた手法によって，型無し高階再帰スキーム（本質的には型無し λ 計算と同等）が与えられた正則な安全性を満すかを特徴づける型システムを提案している．また，Kobayashi[10] では，高階再帰スキームが与えられた正則な安全性を満たすかを検証するための実用的なアルゴリズムを提案し，その実装を行っている．

6. 結論および今後の課題

本論文では，Greibach らによって証明された，文脈自由言語と超決定性言語の包含判定問題の決定可能性に対して，型理論を用いた別証明を与えた．

型システムを構築するにあたって，小林らによって提案されている型システムのアイデアを用いた．そして，提案する型システムの完全性と健全性に関しては，一般的な型理論における手法を用いることで，素直に証明されることを見た．

また，オートマトンが超決定性である場合には，対応する型システムが決定可能であることを証明した．その鍵となるのは，部分型の導入とポンプの補題の拡張である．部分型関係は，十分に下のスタック記号はプッシュダウンオートマトンの動作に影響を与えないという，明かな直感から得られるものであった．この部分型を使うことで，遷移におけるスタックの大きさそのものは重要ではなく，スタック長の変化にのみ注目すればよくなる．そののち，ポンプの補題を用いて，与えられた十分に長い語に対して，スタック長の変化が本質的に等しくなるような短い語を構成した．いい換えれば，どれだけ長い語を読み込むときにも，スタックの変化の幅は十分に小さいことを意味する．これは Greibach らが指摘したとおりである．これを型理論の言葉で述べたものが我々の証明で鍵となる補題であって，次のように表現することができる．無限種類の基底型は本質的に必要とされることはなく，有限個の基底型（と部分型関係）だけがあれば十分である．決定可能性は，基底型が本質的には有限個しかない（と考えてもよい）ことの直接の帰結として得られる．

今後の研究としては，この手法をより大きな言語クラスに対して適用することがあげられる．

最も単純な着想は，包含判定の左辺の言語クラス $\mathcal{L}_1$ を，文脈自由言語から高階再帰スキームの生成言語に拡張することである．これは小林らの型システムが高階再帰スキームのために提案されたことを考えると自然な拡張の方針である．この場合にも，我々が用いたアイデアと同様のアイデアによって，完全かつ健全な型システムが構築できることが期待できる．問題となるのはやはり決定可能性であって，たとえば $\mathcal{L}_1$ を高階再帰スキームの受理言語のクラス，$\mathcal{L}_2$ を超決定性言語とすると，型システムは決定不能となってしまう．これは型システムの設計の問題ではなく，そもそも高階再帰スキームと超決定性言語の包含判定が決定不能であるためである[20]．

このように，我々の手法を用いて包含判定の決定可能性を証明するためには，一般的に型システムの決定可能性が問題となると考えられる．本論文では，型システムの決定可能性を示すために，部分型関係とポンプの補題を用いた．同様にして，文脈自由言語よりも広いクラス $\mathcal{L}_1$ に対しても，適切な部分型関係とポンプの補題を探すことで，決定可能性を示すことができるのではないかと期待される．

謝辞　原稿を注意深くお読みいただき適切な助言をいただいたことに対して，匿名査読者に感謝する．本研究は科研費（20240004，223842）の助成を受けたものである．

参 考 文 献

1) Asveld, P.R.J. and Nijholt, A.: The Inclusion Problem for Some Subclasses of Context-Free Languages, *Theor. Comput. Sci.*, Vol.230, No.1-2, pp.247–256 (2000).
2) Greibach, S.A. and Friedman, E.P.: Superdeterministic PDAs: A Subcase with a Decidable Inclusion problem, *J. ACM*, Vol.27, No.4, pp.675–700 (1980).
3) Tozawa, A. and Minamide, Y.: Complexity Results on Balanced Context-Free Languages, *FoSSaCS*, Seidl, H. (Ed.), Lecture Notes in Computer Science, Vol.4423, pp.346–360, Springer (2007).
4) Minamide, Y. and Tozawa, A.: XML Validation for Context-Free Grammars, *APLAS*, Kobayashi, N. (Ed.), Lecture Notes in Computer Science, Vol.4279, pp.357–373, Springer (2006).
5) Nguyen, V.T. and Ogawa, M.: Alternate Stacking Technique Revisited: Inclusion Problem of Superdeterministic Pushdown Automata, *IPSJ Trans. Programming*, Vol.1, No.1, pp.36–46 (2008).
6) Berstel, J. and Boasson, L.: Formal properties of XML grammars and languages, *Acta Inf.*, Vol.38, No.9, pp.649–671 (2002).
7) Pair, C. and Quéré, A.: Définition et Etude des Bilangages Réguliers, *Information and Control*, Vol.13, No.6, pp.565–593 (1968).
8) Kobayashi, N.: Types and higher-order recursion schemes for verification of higher-order programs, *POPL*, Shao, Z. and Pierce, B.C. (Eds.), pp.416–428, ACM (2009).
9) Kobayashi, N. and Ong, C.-H.L.: A Type System Equivalent to the Modal Mu-Calculus Model Checking of Higher-Order Recursion Schemes, *LICS*, pp.179–188, IEEE Computer Society (2009).
10) Kobayashi, N.: Model-checking higher-order functions, *PPDP*, Porto, A. and López-Fraguas, F.J. (Eds.), pp.25–36, ACM (2009).
11) Kozen, D.C.: *Automata and Computability*, Springer (1997).
12) Harrison, M.A. and Yehudai, A.: Eliminating Null Rules in Linear Time, *Comput. J.*, Vol.24, No.2, pp.156–161 (1981).
13) Takahashi, M.: Generalizations of Regular Sets and Their Applicatin to a Study of Context-Free Languages, *Information and Control*, Vol.27, No.1, pp.1–36 (1975).
14) van Bakel, S.: Intersection Type Assignment Systems, *Theor. Comput. Sci.*, Vol.151, No.2, pp.385–435 (1995).
15) Ginsburg, S. and Greibach, S.A.: Deterministic Context Free Languages, *Information and Control*, Vol.9, No.6, pp.620–648 (1966).
16) Knapik, T., Niwinski, D. and Urzyczyn, P.: Higher-Order Pushdown Trees Are Easy, *FoSSaCS*, Nielsen, M. and Engberg, U. (Eds.), Lecture Notes in Computer Science, Vol.2303, pp.205–222, Springer (2002).
17) Ong, C.-H.L.: On Model-Checking Trees Generated by Higher-Order Recursion Schemes, *LICS*, pp.81–90, IEEE Computer Society (2006).
18) Kobayashi, N., Tabuchi, N. and Unno, H.: Higher-order multi-parameter tree transducers and recursion schemes for program verification, *POPL*, Hermenegildo, M.V. and Palsberg, J. (Eds.), pp.495–508, ACM (2010).
19) Tsukada, T. and Kobayashi, N.: Untyped Recursion Schemes and Infinite Intersection Types, *FOSSACS*, Ong, C.-H.L. (Ed.), Lecture Notes in Computer Science, Vol.6014, pp.343–357, Springer (2010).
20) Kobayashi, N.: Unpublished.

(平成 22 年 9 月 28 日受付)
(平成 23 年 1 月 6 日採録)

塚田　武志

1983 年生まれ．2008 年京都大学工学部情報学科卒業．2010 年東北大学大学院情報科学研究科情報基礎科学専攻修士課程修了，同年博士課程進学．同年より日本学術振興会特別研究員．多段階計算や型理論に興味を持つ．

小林　直樹（正会員）

1968 年生．1991 年東京大学理学部情報科学科卒業．1993 年同大学大学院理学系研究科情報科学専攻修士課程修了，同年博士課程進学．東京大学大学院理学系研究科情報科学専攻助手，講師，東京工業大学大学院情報理工学研究科助教授を経て 2004 年より東北大学大学院情報科学研究科教授，現在に至る．博士（理学）．型理論，プログラム解析，並行計算等に興味を持つ．ACM，日本ソフトウェア科学会各会員．2001 年 IFIP TC2 Manfred Paul Award，2003 年日本 IBM 科学賞，2009 年日本学術振興会賞等を受賞．