命題論理に基づいた要求記述法と状態遷移システムによる意味記述

宋 国煥[†] 富樫 敦[†] 白鳥則郎[†]

情報処理システムの大規模化・複雑化に伴い，信頼性の高いシステムを効率よく設計するための形式的仕様記述法の必要性が高まっている．また，通信システムにおいて設計の初期段階では，システム要求の修正変更が頻繁に起こるため，一部の修正変更がシステムの仕様全体に影響を与えている．本論文では，情報処理システムがその果たす機能によって記述できること，また個々の機能がその機能を実行するための前提条件，入力，出力，機能実行後の条件によって記述できることに注目し，命題論理に基づいた情報処理システムの新しい要求記述法とその状態遷移システムによる意味記述法について述べる．また，システム要求の意味は健全で完全な遷移システム，標準モデルであるという意味論を提案し，その妥当性を理論的に議論する．さらに，その意味論は，拡張ペトリネットの一種である論理ペトリネット（LPN）を用いても特性化できることを示し，LPNを用いて状態遷移システムを合成するやや具体的なアルゴリズムを記述する．

A Requirement Description Method Based on Propositional Logic and Its Semantic Description by State Transition System

KUKHWAN SONG,[†] ATSUSHI TOGASHI[†] and NORIO SHIRATORI[†]

As information processing systems become large and complex, formal description methods are needed for specification of systems and their efficient and reliable designs. During the initial phase of design, it is often necessary to modify or change system requirements which may influence the whole design specification. In this paper, we pay attention that information processing systems can be described by some executed functions and each function for execution can also be described by a pre-condition to be satisfied before execution, input, output and a condition to be satisfied after execution. Therefore we propose a new description method of information processing systems based on propositional logic, and discuss a semantic description method by its state transition system. Furthermore, we suggest that the meaning of system requirements is a sound and complete transition system (standard model), and show the correctness of that logically. We also depict that its meaning can be characterized by Logical Petri Net (LPN), that is a kind of extended Petri nets, and state more detailed algorithm for synthesizing state transition system by using LPN.

1. はじめに

情報処理システムが年々大規模化・複雑化するに伴い，信頼性の高いシステムを効率よく設計するための形式的仕様記述法の必要性が高まっている．通信システムの分野では，仕様を形式的に記述するための方法に関する研究が以前からさかんに行われ，形式記述技法（FDT: formal description techniques）として実際にSDL[1]，Estelle[2]，LOTOS[3]といった言語が開発されている．このような形式記述技法は状態遷移システムを基本概念とし，記述技法によっては状態遷移システム自体を形式仕様として用いている方法もある．

しかしながら，状態と状態遷移に基づいた記述法は，最終的な仕様を記述する目的には向いているが，修正変更が頻繁に起こるシステム設計の初期段階にはあまり適さない．なぜなら，一部の修正変更がシステムの仕様全体に影響を与えるからである．修正変更に柔軟に対処するためには，システムの局所的な機能の論理的性質に注目した記述法が重要となる．

本論文では，情報処理システムがその果たす機能によって記述できること，また個々の機能がその機能を実行するための前提条件，入力，出力，機能実行後の条件によって記述できることに注目し，命題論理に基づいた情報処理システムの新しい要求記述法とその状態遷移システムによる意味記述法について述べる．システ

† 東北大学電気通信研究所 情報科学研究科
Research Institute of Electrical Communication, Graduate Scool of Information Sciences, Tohoku University

ム要求は機能の局所的な性質を記述した項目の集合と初期条件からなり，状態遷移システムによる意味論を展開する．主な結果としては，システム要求の意味は健全で完全な遷移システム，標準モデルであるという意味論を提案し，その妥当性を理論的に議論する．またその意味論は，拡張ペトリネット（extended Petri net）の一種である論理ペトリネット（LPN: logical Petri net）を用いても特性化できることを示す．本論文の要求記述法をソフトウェアの開発法の側面からとらえると，システム要求はソフトウェアシステムの要求記述（例：時制論理式，自然言語）に対応し，その意味を与える状態遷移システムはソフトウェアシステムの形式仕様に対応する．本論文は，要求記述と仕様記述の関係を状態遷移システムによる意味論の立場からとらえ直した方法を提案する．

本論文と関連する従来の方法に，時制論理（temporal logic）の論理式による要求記述からタブロー法によりシステムの仕様である状態遷移システムを合成する方法[4),5)]や，自然言語の要求記述に制限を加えることによって形式仕様を生成する方法[6),7)]等がある．しかしながら，前者の方法は記述能力は高いが，その反面標準モデルを構成する際の計算量が多項式領域困難（P space hard）となり実用性に欠ける．後者の方法は自然言語特有の曖昧さの問題があり，実際のシステム記述には適さない．このほかプロダクションシステムを用いた方法[8)]があるが，そこでは本論文のような形式的かつ理論的な議論は行っていない．

本論文は次のように構成されている．まず2章で，要求記述と形式仕様に関して述べ，3章では要求記述から形式仕様を自動的に導出する技法について述べる．また4章では，システム要求を論理ペトリネットでモデル化し，3章で述べた合成法を実現するやや具体的なアルゴリズムを，論理ペトリネットを用いて記述することについて述べる．5章は結論である．

2. 要求記述と形式仕様

ρ を仕様記述の対象に依存した，命題論理における素命題の集合とする．つまり，各素命題は仕様化しようとするシステムの具体的な性質を表す．通常のように，命題論理式は論理積（$\wedge$），論理和（$\vee$），否定（$\neg$）の演算子と素命題によって構成される．解釈 I とは写像 $I:\rho\to\{\text{true},\text{false}\}$ である．

[定義 2.1] f，g を論理式とする．

(1) f が矛盾するとは，f を満たす解釈が存在しないことである．

(2) f が無矛盾であるとは，f が矛盾しないことである．

(3) f が g に依存するとは，g を満たす解釈はすべて f，または $\neg f$ を満たすことである．

(4) f と g が独立であるとは，f が g に依存しないこと，かつ g が f に依存しないことである．

□

γ を無矛盾なリテラル☆の論理積とする．定義から明らかに，リテラル A（$\neg A$）が γ に対して独立であることと，A と $\neg A$ が γ の中に現れないことは同値である．

[定義 2.2] 機能要求は，5項組 $\phi=\langle id,i,f_{\text{in}},o,f_{\text{out}}\rangle$ である．ここで，id：機能の名前，i：入力，f_{in}：機能を実行する前の条件，o：出力，f_{out}：機能を実行した後の条件である．ここで，f_{in} と f_{out} はそれぞれ ρ を用いた無矛盾な命題論理式とリテラルの論理積として記述される． □

機能要求の条件 f_{in} は，一般性を失わず，加法標準形 $\gamma_1\vee\ldots\vee\gamma_n$ で表されているとする．ここで，γ_i はリテラルの論理積である．機能要求 ϕ は，$\phi: f_{\text{in}}=i/o\Rightarrow f_{\text{out}}$ とも記述され，形式的意味は“機能 ϕ はシステムが f_{in} の前提条件を満足していたら，i を入力することによって o を出力した後，システムを f_{out} の条件に変更させる”と解釈される．また，そのとき，機能要求の条件 f_{out} に書いていない素命題の解釈においては，機能に関与するか否かにかかわらず，機能を実行する前に成り立つ（成り立たない）素命題は，機能実行後の条件として直接に記述していなければ機能実行後も成り立つ（成り立たない）．

[定義 2.3] 要求記述は，$R=\langle\Phi,\gamma_o\rangle$ である．ここで，Φ：機能要求の有限集合，γ_o：初期条件である．γ_o は ρ を用いた無矛盾なリテラルの論理積として記述される． □

要求記述法では要求を機能の集合として論理式によって扱うことにより，機能が独立単位として記述できるし，遷移システムにおいて入出力が同一である複数の機能（状態遷移）を1つにまとめて記述できる．本論文では，通信システムにおける形式記述技法の基本概念になっている状態遷移システムを形式仕様とする．

[定義 2.4] （非決定性）状態遷移システムは，$M=\langle q,\Sigma,\Delta,\to,q_o\rangle$ である．ここで，Q ：状態の有限集合，Σ ：入力の有限集合，Δ ：出力の有限集合，$\to\subset Q\times(\Sigma\times\Delta)\times Q$：状態遷移（・出力）関係，$q_o$：初期状態である． □

☆ 素命題および素命題の不定をリテラルと呼ぶ．

状態遷移システム M において，状態遷移 $p-i/o\rightarrow q$ とは，$(p,i,o,q)\in\rightarrow$ のことであり，"状態 p で i が入力されたら，o を出力してシステムの状態を p から q に遷移する" と解釈される．

ここで，状態遷移システムのすべての状態は初期状態から到達できると仮定する．また，素命題 A が状態遷移システムの状態 $p\in Q$ で成り立つか $(p\vDash A)$ 否か $(p\nvDash A)$ が事前に決まっていると仮定する．以上より，与えられた状態のもとで任意の命題論理式の真偽を与えることができる．状態遷移システム M において，素命題 A に対する状態 q の解釈 $I(q)$ を以下のように定義する．

$I(q)(A)=\text{true}$ iff $q\vDash A$,

$I(q)(A)=\text{false}$ iff $q\nvDash A(q\neg A)$.

また，状態遷移システムに対して次の条件を仮定する．

$p=q$ iff $I(p)=I(q)$.

つまり，すべての素命題について命題の解釈が同じ状態は同一視するという仮定である．

[定義 2.5] 状態遷移 $t=\langle p-i'/o'\rightarrow q\rangle$ が機能要求 $\phi=\langle id,i,f_{\text{in}},o,f_{\text{out}}\rangle$ に関して正しい（または，t が ϕ を満足する）とは，次の条件を満たすことである．

(1) $i=i',o=o'$

(2) $p\vDash f_{\text{in}},q\vDash f_{\text{out}}$

(3) 素命題 A が f_{out} と独立ならば，

$p\vDash A\Leftrightarrow q\vDash A$ □

〈例題 2.1〉 要求記述

$\Phi_1=\langle\{\phi_1:A=a_1/a_2\Rightarrow\neg A\wedge\neg B,$

$\phi_2:(\neg A\wedge\neg B)\vee(A\wedge C)=b_1/b_2\Rightarrow\neg C,$

$\phi_3:\neg C=c_1/c_2\Rightarrow C,$

$\phi_4:C=d_1/d_2\Rightarrow A\},$

$A\wedge\neg B\wedge\neg C\rangle$

（便宜的に a_1/a_2 を a，b_1/b_2 を b，c_1/c_2 を c，d_1/d_2 を d と書く）と，**図 1** の状態遷移システム M_1 を考える．まず，遷移 "$t_1=\langle q_0-a\rightarrow q_1\rangle$" と機能要求 "$\phi_1:A=a\Rightarrow\neg A\wedge\neg B$" の関係を調べる．

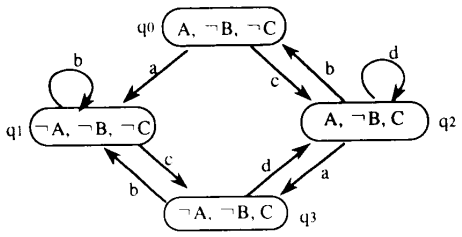


図 1 要求記述 Φ_1 の状態遷移システム

Fig. 1 State transition system for the requirement descriptions Φ_1.

$q_0\vDash A$ および $q_1\vDash\neg A\wedge\neg B$ なので，定義 2.5 の条件 (1)，(2) を満たす．また，$\neg A\wedge\neg B$ と独立な素命題は C であり，$q_0,q_1\nvDash C$ なので定義 2.5 の条件 (3) を満たす．したがって，t_1 は ϕ_1 を満たす．同様にして，次を得る．

- 遷移 $q_0-a\rightarrow q_1$，$q_2-a\rightarrow q_3$ は ϕ_1 を満たす；
- 遷移 $q_1-b\rightarrow q_1$，$q_2-b\rightarrow q_0$，$q_3-b\rightarrow q_1$ は ϕ_2 を満たす；
- 遷移 $q_0-c\rightarrow q_2$，$q_1-c\rightarrow q_3$ は ϕ_3 を満たす；
- 遷移 $q_2-d\rightarrow q_2$，$q_3-d\rightarrow q_2$ は ϕ_4 を満たす．

[定義 2.6] 状態遷移システム $M=\langle Q,\Sigma,\Delta,\rightarrow,qo\rangle$ が要求記述 $R=\langle\Phi,\gamma_o\rangle$ に関して健全 (soundness) であるとは，次の条件を満足することである．

(1) ρ のすべての素命題 A に対して，$q_o\vDash A\leftrightarrow A$ が γ_o で肯定的に現れる．つまり，γ_o で否定的に現れる素命題および γ_o に現れない素命題は q_o では成り立たない．

(2) 任意の状態遷移 t に関して，t が満足する機能要求 $\phi\in\Phi$ が存在する． □

M が R に関して健全であっても，R の機能要求 ϕ に対して，ϕ を満足するような状態遷移が M に存在するとは限らない．

[定義 2.7] $M=\langle Q,\Sigma,\Delta,\rightarrow,q_o\rangle$，$M'=\langle Q',\Sigma,\Delta,\rightarrow',q_o'\rangle$ を入出力記号が共通な状態遷移システムとする．M から M' への準同型写像 (homomorphism) $\psi:M\rightarrow M'$ とは写像 $\psi:Q\rightarrow Q'$ である．ここで，ψ は以下の条件を満たす．

(1) $\psi(q_o)=qo'$.

(2) M において $p-i/o\rightarrow q$ ならば，M' において $\psi(p)-i/o\rightarrow\psi(q)$.

(3) M の状態 p と論理式 f に対して $p\vDash f$ ならば，$\psi(p)\vDash f$. □

準同型写像 $\psi:M\rightarrow M'$ が全単射 (bijection) で，また逆関数 (inverse function) ψ^{-1} が M' から M への準同型写像ならば，ψ は同型写像 (isomorphism) と呼ばれる．M から M' への同型写像が存在するならば，M と M' は同型であるという．

[定義 2.8] M を要求記述 R に関して健全な状態遷移システムとする．M が要求記述 R に関して完全 (completeness) であるとは，R に関して健全であるような任意の状態遷移システム M' に対して，準同型写像 $\psi:M'\rightarrow M$ が存在することである． □

機能要求の集合に対して，$\gamma,\gamma_1,\gamma_2,\ldots,\gamma_n$ を無矛盾なリテラルの論理積とすると，次のような変換規則を考える．

- 規則 1： $\Phi\cup\{\gamma_1\vee\ldots\vee\gamma_n=i/o\Rightarrow\gamma\}\notin\Phi$

$\Rightarrow \Phi \cup \{\gamma_1 = i/o \Rightarrow \gamma, \ldots, \gamma_n = i/o \Rightarrow \gamma\}$,

- 規則2： $\Phi \cup \{\gamma_1 \wedge l \wedge \gamma_2 = a \Rightarrow \gamma\} \notin \Phi$
  $\Rightarrow \Phi \cup \{\gamma_1 \wedge l \wedge \gamma_2 = a \Rightarrow \gamma \wedge l\}$,
  ここで l はリテラル A または $\neg A$ を表し，A および $\neg A$ は $\gamma, \gamma_1, \gamma_2$ には現れないとする．

$R = \langle \Phi, \gamma_o \rangle$ を要求記述として，R の Φ に上記の変換規則を適用できなくなるまで適用することによって得られた結果の要求記述を $R' = \langle \Phi', \gamma_o \rangle$ とする．R' を R の標準形（standard form）と呼ぶ．

[命題 2.1] R を要求記述とする．状態遷移システム M, M' が各々 R, R' に関して健全かつ完全であると仮定すると，M と M' は同型である[9]． □

〈例題 2.2〉 本論文で提案するシステム要求の記述法の適用例として，CATV システムを考える．CATV システムは端末がホストコンピュータ（host computer）と連結されていて，端末のリモートコントローラ（remote controller）の操作によって，TV プログラムの多様なサービスを受けることができるシステムである．この例では，次のような一部の機能だけを考える．

(1) 電源 On/Off 機能
(2) チャンネル Up/Down 機能
(3) テンキー（ten key）選局機能
(4) 音量 +/− 機能
(5) ミュート（mute）機能
(6) Force Tune 機能

上の (1)〜(5) の機能は一般の TV の機能と同一で，(6) Force Tune 機能はホストからコマンドで指定されたチャンネルに強制的に選局させられる機能である．

ここでは以下の 3 つの素命題を考える．

- power：端末の電源が On になっている．
- force：ホストコンピュータから決まったチャンネルを通して強制的にメッセージ（message）が送られている．
- mute：端末の音がミュート機能によって聞こえない．

上記の非形式的な記述に基づき，次のような CATV システムの機能要求記述が考えられる．

```
Φ2 = ⟨{
power_off : power∧¬force=pw/-⇒ ¬power∧¬mute,
power_on : ¬power=pw/-⇒power∧¬mute,
channel_up : power∧¬force=chup/print_ch⇒power,
channel_down : power∧¬force=chdw/print_ch⇒power,
channel_change : power∧¬force=tenkey/print_ch⇒power,
mute_on : power∧¬force∧¬mute=mute/print_mute⇒mute,
mute_off : power∧¬force∧mute=mute/print_vol⇒ ¬mute,
force_tune : ¬force=ftune/print_ch⇒power∧force∧¬mute,
force_cancel : power∧force=pw/print_ch⇒ ¬force,
volume_up : power∧¬force=volup/print_vol⇒ ¬mute,
volume_down : power∧¬force=voldw/print_vol⇒ ¬mute},
¬power∧¬force∧¬mute⟩
```

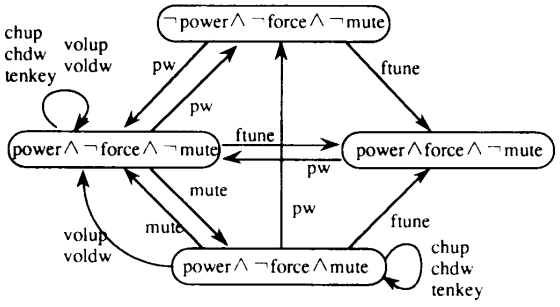


図 2 CATV システム（Φ_2）の状態遷移システム
Fig. 2 State transition system for the CATV system (Φ_2).

また，上の要求記述に関して健全かつ完全な状態遷移システムは**図 2** のようになる．

3. 形式仕様への変換

本章の目標は，与えられた要求記述 $R = \langle \Phi, \gamma_o \rangle$ から R に関して健全かつ完全な状態遷移システム M を導出することである．ここでは，要求記述 R が標準形であると仮定し，標準形 R から M への変換方法 τ について述べる．

$\tau(R) = \langle \Gamma, \Sigma, \Delta, \rightarrow, q_o \rangle$，ここで，

(1) $\Gamma = \{\gamma \mid \gamma$ は素命題の集合 ρ を用いたリテラルの論理積であり，γ には ρ に属する素命題がちょうど 1 回ずつ（肯定か否定で）現れる．$\}$,
(2) $\Sigma = \{i \mid \phi : f_{\text{in}} = i/o \Rightarrow f_{\text{out}} \in R\}$,
(3) $\Delta = \{o \mid \phi : f_{\text{in}} = i/o \Rightarrow f_{\text{out}} \in R\}$,
(4) $\gamma - i/o \rightarrow \gamma'$：次のような機能要求 $\phi : f_{\text{in}} = i/o \Rightarrow f_{\text{out}} \in R$ が存在することである．
  (a) $I(\gamma) \vDash f_{\text{in}}$,
  (b) $I(\gamma') \vDash f_{\text{out}}$,
  (c) 素命題 A が f_{out} と独立ならば，
  $$I(\gamma) \vDash A \leftrightarrow I(\gamma') \vDash A.$$
(5) $q_o = \gamma_o$.

状態 γ において，$I(\gamma) \vDash A$ ならば，素命題 A が状態 γ で成り立つとする．

[定理 3.1] τ によって要求記述 $R = \langle \Phi, \gamma_o \rangle$ から導出された状態遷移システム $\tau(R)$ は，R に対して健全かつ完全である．

〈証明〉

- 健全性： 状態遷移システム $\tau(R)$ の構成法から明らかである．
- 完全性：$M = \langle Q, \Sigma, \Delta, \rightarrow, q_o \rangle$ を R に関して健

全な状態遷移システムとする．Q から Γ への写像 $\psi : Q \to \Gamma$ を次のように定義する．$q \in Q$, $\gamma \in \Gamma$ について $\psi(q) = \gamma$ とは，$I(q) = I(\gamma)$ の場合でありかつこの場合のみとする．

以下の議論によって，ψ は M から $\tau(R)$ への準同型写像であることがわかる．M は健全な状態遷移システムで，M の初期状態 q_o は γ_o の正リテラルのみを満たすので，ψ の定義より $\psi(q_o) = \gamma_o$．$p - i/o \to q$ を M のある遷移とする．また，$\phi : f_{\text{in}} = i/o \Rightarrow f_{\text{out}}$ をこの遷移によって満たされる R の機能要求と仮定する．そうすると，$p \vDash f_{\text{in}}, q \vDash f_{\text{out}}$ が得られて，ψ の定義によって，$\psi(p) \vDash f_{\text{in}}$，$\psi(q) \vDash f_{\text{out}}$．また f_{out} に独立な素命題 A に関する次の条件

$\psi(p) \vDash A$ iff $\psi(q) \vDash A$

は，条件

$p \vDash A$ iff $q \vDash A$

によって導出される．したがって，$\tau(R)$ での遷移 $\psi(p) - i/o \to \psi(q)$ が得られる．最後に ψ の定義により，すべての論理式 f に対して，$p \vDash f$ ならば $\psi(p) \vDash f$ が成り立つ． □

4. 論理ペトリネットによる形式仕様の導出

本章では，システム要求として獲得された機能要求を対象として，この機能要求を論理ペトリネットによりモデル化することによっても，形式仕様を自動的に生成できることを述べる．

4.1 論理ペトリネット

ペトリネット[10),11)]はダイナミックな表現が可能で，通信システムの仕様記述などにも利用されている[12)~14)]．本論文で用いるペトリネットは，既存の拡張ペトリネットの中で，抑止アーク（inhibitor arc）の概念をさらに拡張したペトリネットである．機能要求の命題論理式が表現できるようにするために，抑止アークをトランジション（transition）の入出力の両方に許した拡張ペトリネットとして定義する．このペトリネットは特に命題論理式を表現するためのペトリネットなので，論理ペトリネットと呼ぶ[15)]．

[定義 4.1] 論理ペトリネット（LPN: logical Petri net）は 5 項組 LPN $= (P, T, I, O, \mu_o)$ である．ここで，

(1) P はプレース（place）の有限集合，
(2) T はトランジション（transition）の有限集合，
(3) $I = (I_p, I_n)$ は入力関数 $I_p, I_n : T \to 2^P$ の対，
(4) $O = (O_p, O_n)$ は出力関数 $O_p, O_n : T \to 2^P$ の対，
(5) $\mu_o : P \to \{0, 1\}$ は初期マーキング（initial marking）である．

I と O はアーク（arc）を用いて図的に表現され，I_p, O_p は正アーク "→"（positive arc）で，I_n, O_n は負アーク "—o"（negative arc）で表される．また，任意の $t \in T$ に対して，$I_p(t) \cap I_n(t) = \phi$ かつ，$O_p(t) \cap O_n(t) = \phi$ とする． □

P, T, I, O はペトリネットの構造的な性質を示す．一方，マーキング μ は各プレースのトークン（token）の配置状態を示し，論理ペトリネットでは，トークンは 1 つのプレースに 2 個以上は存在しないとする．マーキング $\mu = (\mu_1, \mu_2, \ldots, \mu_n)$ は 0 と 1 のベクトル（vector）として表される．ここで $n = |P|$ であり，プレース p_i にトークンがあるとき $\mu(p_i) = 1$，トークンがないとき $\mu(p_i) = 0$ とする．

論理ペトリネットの発火規則（firing rule）も命題論理の性質を反映するために，次のように定義する．

[定義 4.2] マーキング μ において，トランジション $t \in T$ が発火可能であるとは，$\forall p \in I_p(t), \mu(p) = 1$ かつ，$\forall p \in I_n(t), \mu(p) = 0$ が成り立つことである．トランジション t が発火可能になるとすぐに発火され，$\forall p \in I_p(t)$ に対して $\mu(p)$ を 0 にした後，$\forall p \in O_p(t)$ に対して $\mu(p)$ を 1 にし，$\forall p \in O_n(t)$ に対して $\mu(p)$ を 0 にすることによって，t の発火後のマーキング μ' が生成される． □

図 3 にトランジションの発火規則の例として，発火前と発火後のトークンの動きを示す．

4.2 論理ペトリネットへの変換

この節では，システムの要求をモデル化するために，要求記述から論理ペトリネットへの変換について述べる．変換の過程は以下のように 3 段階に分けられる．**図 4** では，要求記述から論理ペトリネットに変換して，論理ペトリネットの実行によって状態遷移システムを生成する流れを示す．図 4 の "論理和の消去" 段階と "意味的 LPN 変換" 段階は要求記述 R の標準形を得るための作業で，それぞれ 2 章で述べた変換規則 1 と 2 を適用するための段階である．

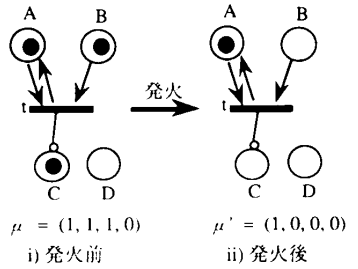


図 3 論理ペトリネットの発火規則の例
Fig. 3 An example of firing rule in logical Petri net.

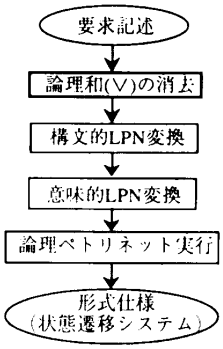


図4 状態遷移システムの生成過程
Fig. 4 Generation process of state transition system.

(1) 論理和（∨）の消去

論理ペトリネットでは，論理演算子の論理積（∧）と否定（¬）はそれぞれアークの組合せと負アークで表現できるので，1個のトランジションを使用して表すことができる．しかしながら，論理和（∨）は1個のトランジションでは直接に表現できないので，変換を通して何個かのトランジションを用いて表現する．そこで，機能要求の集合に対して2章で述べた変換規則1

$$\Phi \cup \{\langle id, i, (\gamma_1 \vee \ldots \vee \gamma_n), o, f_{\text{out}}\rangle\} \Rightarrow \Phi \cup \{\langle id, i, \gamma_1, o, f_{\text{out}}\rangle, \ldots, \langle id, i, \gamma_n, o, f_{\text{out}}\rangle\}$$

を考える．機能要求の条件式には，一般に論理和が含まれているために，まずはじめに，要求記述 $R = \langle \Phi, \gamma_o \rangle$ の Φ に上記の規則を適用できなくなるまで適用することによって，条件式に論理和を含まない要求記述 $\dot{R} = \langle \dot{\Phi}, \gamma_o \rangle$ が得られる．

(2) 構文的 LPN 変換

要求記述に表れている機能の意味は考えず，その構文的（syntactic）表現だけを考慮して論理ペトリネットに変換する．$\dot{R} = \langle \dot{\Phi}, \gamma_o \rangle$ を論理和を消去した後の要求記述とすると，$\dot{R}$ を以下に示すように，論理ペトリネット（P, T, I, O, μ_o）に変換する．

(a) $P \leftarrow \rho$（素命題全体の集合）

(b) $T \leftarrow \Phi$

(c) $I = (I_p, I_n)$, $O = (O_p, O_n)$ は次のように定義される．

$t = \langle id, i, f_{\text{in}}, o, f_{\text{out}}\rangle$, ここで

$f_{\text{in}} = A_1 \wedge \ldots \wedge A_n \wedge \neg B_1 \wedge \ldots \wedge \neg B_m$,

$f_{\text{out}} = C_1 \wedge \ldots \wedge C_j \wedge \neg D_1 \wedge \ldots \wedge \neg D_k$

に対して

(c1) $I_p(t) = \{A_1, \ldots, A_n\}$,

(c2) $I_n(t) = \{B_1, \ldots, B_m\}$,

(c3) $O_p(t) = \{C_1, \ldots, C_j\}$,

(c4) $O_n(t) = \{D_1, \ldots, D_k\}$.

(d) $\gamma_o = A_1 \wedge \ldots \wedge A_n \wedge \neg B_1 \wedge \ldots \wedge \neg B_m$ とすると，

$$\mu_o(A) = 1 \quad \text{if} \quad A = A_i \quad 1 \leqslant i \leqslant n$$
$$0 \quad \text{otherwise}$$

(3) 意味的 LPN 変換

この段階では，要求記述に表れている機能の意味（semantics）を考慮して，2章で述べた変換規則2を適用することによって前段階で変換された構文的 LPN を書き直す作業をする．たとえば，$\rho = \{A, B, C, D\}$ とし，機能要求 $(A \wedge B) = i_1/o_2 \Rightarrow (A \wedge \neg C)$ に対応する論理ペトリネットのトランジション t を考える（図3参照）．その際，この機能を実行した後の条件 f_{out} は定義 2.2 の形式的意味によって $(A \wedge B \wedge \neg C)$ の意味を持つ．しかしながら，構文的 LPN においては，リテラル B の場合（機能実行前に成り立っているし，実行後には述べていない）は，図3で示すように，プレース B からトランジション t への正アークが存在するが，t から B への正アークは存在しないので，t の発火後，プレース B にはトークンを生成しない．したがって，構文的 LPN では発火後の論理表現は $A \wedge \neg B \wedge \neg C$ になっていて素命題 B を満足しない．一方，素命題 D の場合（機能実行前後とも に述べていない）は，プレース D からトランジション t へかつ t から D へのアークが存在しないので，t が発火されてもプレース D のトークンの状態は変化しない．

したがって，構文的 LPN において，$I_p(t)$ には属すが，$O(t)$（$O_p(t)$ または $O_n(t)$）には属さないプレースに対して，トランジション t からそのプレースへの正アークを追加する．この操作によって，2章の変換規則2を LPN の上に適用させる．ここで得られた意味的 LPN は，要求記述 R の標準形を論理ペトリネットに変換したネットと一致する．

4.3 状態遷移システムの生成

本節では，モデル化された論理ペトリネットの実行により，状態遷移システムを生成する方法について述べる．ペトリネットの状態空間を表現する一方法として可達木（reachability tree）[10]が研究されている．本論文では，既存の可達木を拡張して，状態遷移システムと同値な拡張可達木を定義する．そして，論理ペトリネットの実行により，拡張可達木を生成することによって，3章で構成した状態遷移システムと同値なシステムが得られる．

可達木はペトリネットの初期マーキングを根として，

到達可能なマーキングを各節点に配置した木構造である．可達木の大きさを有限にし，また状態遷移システムと同値になるようにするために，拡張可達木を以下のように定義する．

[定義 4.2] 論理ペトリネットの拡張可達木 (extended reachability tree) はラベル付きの木であり，以下のように帰納的に構成される．拡張可達木の各節点 ξ は，$N=\{0,1\}$ 上の n 次元のベクトル $l(\xi)$ でラベル付けされる．ここで，n は異なるプレースの数である．

(1) 木の根 (root) は初期マーキング μ_o でラベル付けされる．

(2) 各節点を結ぶアークには，トランジションを表示する．

(3) η を節点としたとき，

(a) もし $l(\eta)=l(\xi)$ であるような節点 ξ が構成した節点としてすでに存在するならば，節点 η は終端節点 (terminal node) である．すでに存在する $l(\xi)$ と区別するために，$l(\eta)$ には括弧を付ける．

(b) その他の場合，$l(\eta)$ のマーキングのもとで発火可能な各トランジション t に対して，t を発火した後のマーキングを η の子節点とし，そのアークに t を付加する． □

拡張可達木に対応する状態遷移システムは次のようになる．拡張可達木の節点とアークは各々，状態遷移システムの状態と遷移に対応させる．その際，木の根に対応する状態が初期状態になる．また，遷移のラベルである入出力 (i/o) は，拡張可達木のアークに付加された t に対応する i/o になる．状態遷移システムにおいて状態の集合 Q は，拡張可達木において括弧が付いていない節点の集合になる．つまり，括弧が付いている終端節点 $(l(\eta))$ と既存節点 $l(\eta)$ を同一視する（図 7 参照）．

要求記述が n 個の素命題を用いて記述されているとき，得られる状態遷移システムには最悪の場合 2^n 個の状態が存在することになる．しかしながら，2^n 個の状態の中には初期状態からは到達できない状態もある．3 章の変換方法 τ が 2^n 個の状態を対象として遷移を考えたことに比べて，LPN による状態遷移システムの合成法では，作り方から明らかに，初期状態から到達可能な状態間での遷移を考えるので，より実質的に効率的である．

図 5 に論理ペトリネットの実行アルゴリズム (algorithm) を示す．このアルゴリズムは，深さ優先で論理ペトリネットを実行して，拡張可達木を生成す

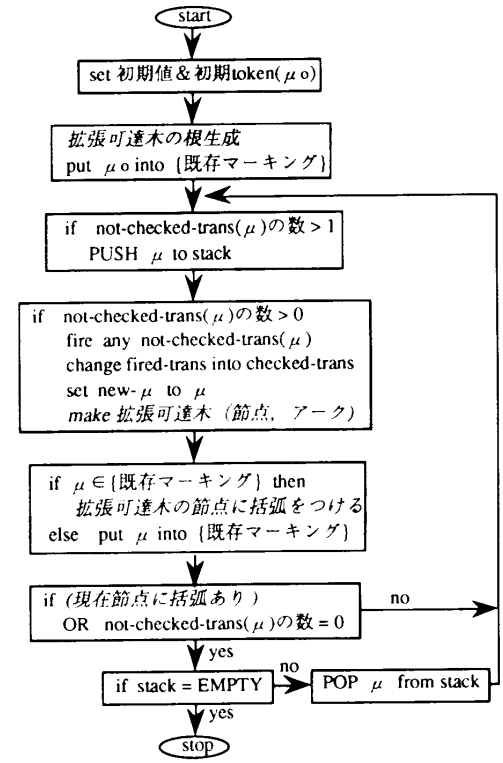


図 5 論理ペトリネットの実行アルゴリズム
Fig. 5 An algorithm for executing logical Petri net.

るアルゴリズムである．

本章で，LPN を導入することによって以下のような利点が考えられる．

(1) LPN を用いることによって，システム要求記述の静的（あるいは動的）な論理エラー検証ができる[15]．その際システム記述が LPN によって記述でき，その標準モデルが LPN の実行と密接な関係がある．

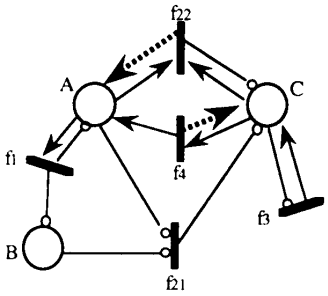
(2) LPN では R の静的表現と，発火によって M の動的振舞いの表現が同時にできるので，R と M の関係を直接調べることもできる．たとえば，ある条件が与えられたとき，実行可能な機能とその機能に対応する状態遷移がすぐ分かる．

(3) R から M への変換を初期状態から順序的に行う．したがって，初期状態から到達できない状態と遷移は生成しないので，計算量が減少する．

(4) LPN は表現能力が高い仕様なので，後段階の作業によってそれ自体が形式仕様にもなる．

4.4 状態遷移システム生成の例題

本節では，本章で述べた変換の例を示す．すなわち，システムの要求記述から論理ペトリネットへ変換した後，実行アルゴリズムに従って発火させて，拡張可達



- 構文的LPNのアーク：実線
- 意味的LPNのアーク：実線と破線

図6 機能要求の構文的LPNと意味的LPN
Fig. 6 Syntactic and semantic LPN of functional requirements.

木を生成する．その後，拡張可達木から状態遷移システムを得る過程について，例題2.1と同一例題をあげて述べる．

まず，例題2.1の機能要求の条件に現れる論理和演算子 (∨) を消去する．論理和演算子が現れる機能 ϕ_2 は，$\phi_{21} : (\neg A \wedge \neg B) = b_1/b_2 \Rightarrow \neg C$ と $\phi_{22} : (A \wedge C) = b_1/b_2 \Rightarrow \neg C$ の2つの論理和演算子が現れない機能要求に分割される．

次に，要求記述を構文的LPNへ変換する．機能要求 ϕ に対応する論理ペトリネットのトランジションを f とすると，変換された構文的LPNは**図6**の実線のアークで表された論理ペトリネットのようになる．

その後，構文的LPNから意味的LPNへの変換を行うことによって，論理ペトリネットを完成させる．機能要求 ϕ_{22}, ϕ_4 は条件の保存性によって，それぞれ $\phi_{22} : (A \wedge C) = b_1/b_2 \Rightarrow (A \wedge \neg C)$, $\phi_4 : C = d_1/d_2 \Rightarrow (A \wedge C)$ と意味的解釈される．意味的LPNは構文的LPNの上にさらに，$I_p(t)$ には属すが，$O(t)$ ($O_p(t)$ または $O_n(t)$) には属さないプレースに対して，論理ペトリネットのトランジション f_{22} からプレース A へ，またトランジション f_4 からプレース C への正アークを追加することによって生成される．以上より，意味的LPNとして図6の実線と破線のアークの両方を持つ論理ペトリネットが生成される．

最後は，図5で示した実行アルゴリズムを用いて論理ペトリネットを実行させることにより，拡張可達木を生成し最終的に形式仕様である状態遷移システムを得る．**図7**(a)は拡張可達木を示し，(b)は最終的な状態遷移システムを示す．

この例によっても，論理ペトリネットにより自動生成された状態遷移システム（図7(b)）は，2章で示した状態遷移システムM1と同値になることが確認できた．

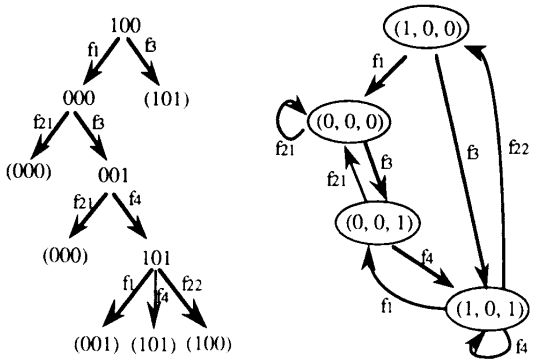


図7 拡張可達木と状態遷移システム
Fig. 7 Extended reachability tree and state transition system.

5. おわりに

本論文では，修正変更に柔軟に対処可能な方法として，獲得されたシステムの機能に関する要求記述から形式仕様を導出する方法を提案し，その妥当性として導出された形式仕様が要求記述に対して健全かつ完全であることを示した．また，従来のペトリネットを拡張した論理ペトリネットを定義して，論理ペトリネットを用いて記述された要求記述から，その実行によっても同型な形式仕様が自動的に生成できることを示した．

従来の研究は，要求を記述する際に状態を直接考慮して記述する方法がほとんどであるが，これらの手法は状態数が多い大規模システムの記述や，修正変更が多いシステム設計には適さない．

現在本手法に基づいたシステム設計支援システムを作成している．支援システムの一部の機能として，状態遷移システムからSDLへの変換器などを検討している．今後の研究としては，複雑なシステム要求を階層的に詳細化・分割化する方法について研究していく予定である．

謝　辞

熱心に討論していただいたパイオニアのゼミグループの皆様，ならびに東北大学白鳥研究室の諸氏に深く感謝します．

参考文献

1) CCITT: Functional Specification and Description Language, Recommendation Z.100 (1989).
2) ISO: Estelle: A Formal Description Technique

based on an Extended State Transition Model, ISO 9074 (1989).
3) ISO: LOTOS: A Formal Description Technique based on Temporal Ordering of Observed Behaviour, ISO 8807 (1989).
4) Manna, Z. and Wolper, P.: Synthesis of Communicating Processes from Temporal Logic Specifications, *ACM Trans. on Programming Languages and Systems*, Vol.6, No.1, pp.68–93 (1984).
5) Gotzhein, R.: *Specifying Communication Services with Temporal Logic, Protocol Specification, Testing and Verification*, Elsevier Science Publishers, pp.295–309 (1990).
6) Kobayashi, Y., Ohta, T. and Terashima, N.: A Requirement Description and Acquisition Method Based on Communication Service Knowledge, 7th International Conference on Systems Research Informations and Cybernetics, Aug. (1994).
7) 原林利幸，河合敦夫，椎野 努，武内 惇：制限自然言語によるソフトウェア要求記述とその解析，ソフトウェア工学 95-3, pp.15–22 (1993).
8) Hirakawa,Y. and Takenaka, T.: Telecommunication Service Description Using State Transition Rules, *Proc. 6th Int. Work. Software Specification and Design*, pp.140–147 (1991).
9) Togashi, A., Usui, N., Song, K. and Shiratori, N.: Synthesis of Formal Specifications from User Requirement and Its Flexibility, to appear in *Tech. Rep. of IEICE*, IN-95 (1995).
10) Peterson, J.L.: *Petri Net Theory and the Modeling of Systems*, Prentice-Hall, p.290 (1982).
11) Murata, T.: Petri Nets: Properties, Analysis and Applications, *IEEE Proc.*, Vol.77, No.4, pp.541–580 (1989).
12) 長谷川晴朗：通信ソフトウェア要求仕様化設計へのペトリネットの応用，情報処理，Vol.34, No.6, pp.770–777 (1993).
13) Danthine, A.S.: Protocol Representation with Finite-State Models, *IEEE Trans. on Communication*, Vol.COM-28, No.4, pp.632–643 (1980).
14) 青山幹雄，平石邦彦，内平直志：高水準ペトリネットによるソフトウェア開発方法論，コンピュータソフトウェア，Vol.11, No.4, pp.3–19 (1994).
15) 宋 国煥，富樫 敦，白鳥則郎：論理ペトリネットを用いた形式仕様の自動変換と検証，信学技報，CST94-29, pp.101–108 (1994).

(平成7年2月10日受付)
(平成8年2月 7 日採録)

宋 国煥（正会員）

1959年生．1988年韓国国防大学院コンピュータ科学科修士課程修了．現在，東北大学大学院情報科学研究科博士課程在学中．ソフトウェア開発法，通信システムの仕様記述法に関する研究に従事．情報処理学会会員．

富樫 敦（正会員）

1984年東北大大学院工学研究科博士課程修了．同年東北大学通信研究所助手．1991年同助教授．1996年4月より静岡大学情報学部助教授．現在に至る．工学博士．現在，並行プロセス計算の意味論や型理論等の研究に従事．並列・分散処理基礎論，プログラム意味論や帰納推論等の推論機構にも興味を持つ．電子情報通信学会，日本ソフトウェア科学会，ACM各会員．

白鳥 則郎（正会員）

1946年生．1977年東北大学大学院博士課程修了．同年，東北大学電気通信研究所勤務．1984年，同大助教授（電気通信研究所）．1990年，同大教授（工学部情報工学科）．情報通信システムの構成論，ソフトウェア開発法，ヒューマンインタフェースの研究に従事．情報処理学会25周年記念論文賞授賞．IEEE，電子情報通信学会，人工知能学会各会員．