プログラミング 18−11
(1998. 3. 23)

命題論理に基づいた並行システムの仕様記述

菅原 佳菜　　高橋 薫

仙台電波工業高等専門学校

〒989-3124 仙台市青葉区上愛子字北原 1
電話: 022-392-4761　FAX: 022-392-3359
E-mail: kaoru@cc.sendai-ct.ac.jp

あらまし　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．そのような記述法の一つとして，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，事後条件によって記述できることに着目した，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その意味記述法が提案されている．本論文では，従来単一システムを対象としていた上記の要求記述法を，いくつかのサブシステムから成る並行システムに拡張する．そのために，サブシステム毎の機能要求記述，およびその集まりとしての並行システム機能要求記述を提案する．更に，これら機能要求記述から形式仕様としての状態遷移システムを合成する方法についても述べる．

キーワード　命題論理，機能要求記述，形式仕様，状態遷移システム

Specification of a Concurrent System Based on Propositional Logic

Kana SUGAWARA and Kaoru TAKAHASHI

Sendai National College of Technology

Kitahara 1, Kamiayashi, Aoba-ku, Sendai, 989-3124, JAPAN
Phone: +81-22-392-4761　FAX: +81-22-392-3359
E-mail: kaoru@cc.sendai-ct.ac.jp

Abstract　As information processing systems become large and complex, formal description methods are needed for specification of systems and their efficient and reliable designs. Since information processing systems can be described by some executed functions and each function for execution can be described by a pre-condition to be satisfied before execution, input, output and a post-condition to be satisfied after execution, a description method based on propositional logic and its semantic description method by a state transition system have been proposed. In this paper, we extend the above mentioned description method whose target is a single system so that a concurrent system consisting of some subsystems can be handled. For this purpose, we give a functional requirement description method of each subsystem and a concurrent system as a collection of subsystems. We also propose a method to synthesize a state transition system as a formal specification from a functional requirement description.

Keywords　propositional logic, functional requirement description, formal specification, state transition system

1 はじめに

情報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設計を達成するための形式記述法 (FDT: Formal Description Techniques) の必要性が認識されている (例えば [1])．FDT の多くは，状態遷移の概念を基本とし，システムの挙動を陽に記述することで，システムの形式仕様を得ている．しかしながら，このような仕様では，システム機能の論理的な性質が不明瞭であり，また，仕様の一部の修正・変更が仕様全体に影響を与えることがある．そこで，システムの論理的な機能要求，および，個々の機能実行のための前提条件，実行における入出力，機能実行による事後条件に着目し，命題論理に基づいて仕様を記述する方法が提案されている [2],[3],[4]．この方法では，システムの局所的な機能の記述が可能であり，仕様の修正・変更への柔軟性が高い．結果的に，この命題論理に基づいた方法による仕様はシステムの要求仕様として，また，状態遷移概念に基づいた方法による仕様はシステムの形式仕様として捉えることができる．

本論文では，単一システムを対象としていた上記の命題論理に基づいた方法を，複数のサブシステムから成る並行システムへ拡張する．そのため，サブシステム毎の機能要求記述，および，その集まりとしての並行システム機能要求記述を提案する．更に，これら機能要求記述から，形式仕様としての状態遷移システムを合成する方法についても与える．

2 要求仕様と形式仕様

ユーザが記述することを第一義に考える要求仕様記述の一手法として，機能を単位として記述する機能要求を用いる．機能要求をサブシステムとした並行システムを並行システム機能要求という．一方，システムの挙動を陽に表現する仕様を形式仕様と位置づける．

この節では，最初に機能の条件として用いられる命題論理式について，構文と意味を定義する．次に，ユーザが記述する並行システム機能要求を定義し，併せて，形式仕様を定義する．

2.1 命題論理

$\mathcal{A}$ を素命題の有限集合とする．素命題は，システムを構成しているそれぞれの機能において，機能が使用可能になるための条件と機能によって変更される条件を表す．それによって，システムの状態，条件を意味する命題は，命題論理式として与えることができる．

定義 1 命題論理式を次のように帰納的に定義する:

1. $A \in \mathcal{A}$ は命題論理式である．
2. fが命題論理式であるとき，$\neg f$も命題論理式である．
3. fと gが命題論理式であるとき，$f \wedge g$, $f \vee g$, $f \Rightarrow g$ も命題論理式である． □

以下では，命題論理式を単に命題ともいう．

定義 2 素命題 A あるいは素命題の否定$\neg A$ を A のリテラルと呼ぶ． □

定義 3 $\mathcal{A}$ から**定義 1**によって生成される命題論理式の集合を $\mathcal{L}$ とする．このとき，命題論理式の意味を解釈 $\mathcal{I}$: $\mathcal{L} \rightarrow \{\mathbf{true}, \mathbf{false}\}$ を用いて表す．ここで，**true**, **false** は命題の真偽値である．$\mathcal{I}$と演算子$\neg$, $\wedge$, $\vee$, $\Rightarrow$ との関係を，A を素命題，fと gを命題論理式として，以下に定義する:

1. $\mathcal{I}(A) = \mathbf{true}$ または $\mathcal{I}(A) = \mathbf{false}$
2. $\mathcal{I}(f) = \mathbf{true}$ のとき，$\mathcal{I}(\neg f) = \mathbf{false}$
3. $\mathcal{I}(f) = \mathbf{false}$ のとき，$\mathcal{I}(\neg f) = \mathbf{true}$
4. $\mathcal{I}(f) = \mathbf{true}$, $\mathcal{I}(g) = \mathbf{true}$ のとき，$\mathcal{I}(f \wedge g) = \mathbf{true}$，それ以外のとき，$\mathcal{I}(f \wedge g) = \mathbf{false}$
5. $\mathcal{I}(f) = \mathbf{false}$, $\mathcal{I}(g) = \mathbf{false}$ のとき，$\mathcal{I}(f \vee g) = \mathbf{false}$，それ以外のとき，$\mathcal{I}(f \vee g) = \mathbf{true}$
6. $\mathcal{I}(f) = \mathbf{true}$, $\mathcal{I}(g) = \mathbf{false}$ のとき，$\mathcal{I}(f \Rightarrow g) = \mathbf{false}$，それ以外のとき，$\mathcal{I}(f \Rightarrow g) = \mathbf{true}$ □

定義 4 $\mathcal{I}(f)$=**true** のとき，命題 fは解釈$\mathcal{I}$の下で真であるといい，$\mathcal{I}(f)$=**false** のとき，命題 fは解釈$\mathcal{I}$の下で偽であるという．解釈$\mathcal{I}$の下で命題 fが真であるならば，$\mathcal{I}$ は fを満たすという． □

定義 5 fと gを命題とする．

1. fが無矛盾であるとは，fを満たす解釈が存在することである．
2. 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．
3. fが gに対して従属であるとは，gを満たす全ての解釈が fを満たすか，または gを満たす全ての解釈が $\neg f$を満たすことである．
4. fと gが独立であるとは，fが gに対して従属でなく，また gが fに対して従属でないときである． □

以上の定義より，次の命題が成立する:

命題 1 γはリテラルの連言であり，無矛盾であるとする．素命題 A がγに対して独立であるとは，A, $\neg A$ がγの中に現れていないことである．また，$\neg A$ がγに対して独立であるということは，A, $\neg A$ がγの中に現れていないことである． □

2.2 機能要求

並行システムは互いに相互作用を行ういくつかのサブシステムから構成されると考える．

サブシステムをそれぞれ関連させるために，ポートの概念を導入する．また，並行システムの外部とも関連させるために，サブシステム kのポートを内部ポートの集合 P_{k_i}と環境ポートの集合 P_{k_e}に分ける．内部ポートを通してサブシステム間で相互作用 (入出力) を行い，環境ポートを通して並行システムの外部との相互作用 (入出力) を行うのである．

サブシステム kのポートの有限集合を $P_k = P_{k_i} \cup P_{k_e}$ ($P_{k_i} \cap P_{k_e} = \phi$) と表す．また，入力の有限集合を I_k，出力の有限集合を O_kと表す．これらを用いてサブシステムの入力アクションと出力アクションを以下のように定義する：

定義 6 サブシステム kの入力アクションΣ_kと出力アクションΔ_kは次の集合である:

$$\Sigma_k \subseteq P_k \times I_k$$
$$\Delta_k \subseteq P_k \times O_k \qquad \square$$

サブシステムの動作として，ポートを通した外部からのある入力，ポートを通した外部へのある出力，外部とは独立な内部アクションの 3 つを考えることができる．また，サブシステムが，何らかの条件を満たしているときのみ，動作が可能であると考えられる．このことから，サブシステムの状態の有限集合を考えたときに，ある動作が実行可能な状態の有限集合と，実行不可能な状態の有限集合に分ける事ができる．さらに，動作を実行することにより，条件が変更され，新たに実行可能になる動作と実行不可能になる動作が存在する．

定義 7 サブシステム kの素命題の有限集合を $\mathcal{A}_k$ ($\mathcal{A}_k \subset \mathcal{A}$) とし，$\mathcal{A}_k$から生成される命題論理式の集合を $\mathcal{L}_k$とする．このとき，サブシステム kの機能ρは以下の 3 項組である:

$\rho = \langle f_{in},\ a,\ f_{out} \rangle$

- f_{in} 機能ρを実行するための前提条件 ($f_{in} \in \mathcal{L}_k$)
- a 入力アクションまたは出力アクションまたは内部アクション[1] ($a \in \Sigma_k \cup \Delta_k \cup \{\varepsilon\}$)
- f_{out} 機能ρの実行後に満たされるべき事後条件 ($f_{out} \in \mathcal{L}_k$)

$\square$

分かりやすさのため，機能ρを$\rho : f_{in} \overset{a}{\Rightarrow} f_{out}$と記述する．入力の場合$\rho : f_{in} \overset{a?}{\Rightarrow} f_{out}$と記述し，機能を入力機能と呼ぶ．出力の場合$\rho : f_{in} \overset{a!}{\Rightarrow} f_{out}$と記述し，機能を出力機能と呼ぶ．また，内部アクションの場合$\rho : f_{in} \overset{\varepsilon}{\Rightarrow} f_{out}$と記述し，機能を内部機能と呼ぶ．入力機能および出力機能のとき，アクション aをポートの部分 pと入力/出力の部分 eに分け，$p:e$のように表す．

サブシステム全体の機能要求を機能の集まりとして次のように定義する：

定義 8 サブシステム kの機能要求 $\mathcal{R}_k$は次の 5 項組である：

$\mathcal{R}_k = \langle R_k,\ P_k,\ \Sigma_k,\ \Delta_k,\ \mathcal{A}_k \rangle$

- R_k サブシステム kの機能の有限集合
- P_k サブシステム kのポートの有限集合
- Σ_k サブシステム kの入力アクションの有限集合
- Δ_k サブシステム kの出力アクションの有限集合
- $\mathcal{A}_k$ サブシステム kの素命題の有限集合 ($\mathcal{A}_k \subset \mathcal{A}$)

$\square$

サブシステムの機能要求および初期条件の集まりとして並行システムの機能要求を定義する:

定義 9 並行システム機能要求 $\mathcal{R}$ は次の集合である:

$\mathcal{R} = \{(\mathcal{R}_k,\ \gamma_{0k})\}$
($1 \le k \le n$, n はサブシステムの数)

$\mathcal{R}_k = \langle R_k,\ P_k,\ \Sigma_k,\ \Delta_k,\ \mathcal{A}_k \rangle$
サブシステム kの機能要求
$(\mathcal{A}_i \cap \mathcal{A}_j = \phi,\ i \ne j,\ \bigcup_{l=1}^{n} \mathcal{A}_l = \mathcal{A})$

γ_{0k}
サブシステム kの初期条件であり，$\mathcal{A}_k$のすべての素命題のリテラルからなる無矛盾な連言

$\square$

例 1 並行システム機能要求 $\mathcal{R}$ の例を挙げる.(図 1 参照)

$\mathcal{R} = \{(\mathcal{R}_1,\ A \wedge B),\ (\mathcal{R}_2,\ C),\ (\mathcal{R}_3,\ D)\}$

$\mathcal{R}_1 = \langle \{\rho_1 : A \overset{p_a:a!}{\Rightarrow} \neg A,\ \rho_2 : B \overset{p_b:b?}{\Rightarrow} A,$
$\rho_3 : A \overset{p_c:c?}{\Rightarrow} \neg B\},\ \{p_a,\ p_b,\ p_c\},$
$\{p_b : b,\ p_c : c\},\ \{\{p_a : a\}\},\ \{\{A,\ B\}\} \rangle$

$\mathcal{R}_2 = \langle \{\rho_4 : C \overset{p_a:a?}{\Rightarrow} \neg C,\ \rho_5 : \neg C \overset{p_b:b!}{\Rightarrow} C\},$
$\{p_a,\ p_b\},\ \{\{p_a : a\}\},\ \{\{p_b : b\}\},\ \{\{C\}\} \rangle$

$\mathcal{R}_3 = \langle \{\rho_6 : \neg D \overset{p_a:a?}{\Rightarrow} D,\ \rho_7 : D \overset{\varepsilon}{\Rightarrow} \neg D\},$
$\{\{p_a\}\},\ \{\{p_a : a\}\},\ \{\{\}\},\ \{\{D\}\} \rangle$

$\square$

2.3 形式仕様

形式仕様とはシステムの挙動を陽に記述する仕様のことを指す．一般に，その多くは状態，遷移の概念を用いた状態遷移システムを意味モデルとする．そこで，ここでは合成の対象となる形式仕様として，状態遷移システムを用いる．

[1] 内部アクションをεで示す.

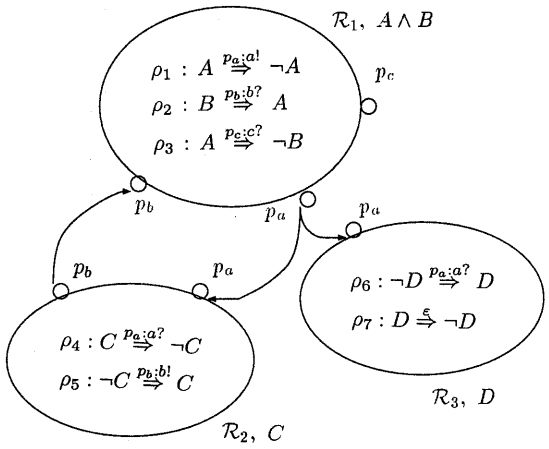

図 1: 機能要求 $\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3$から成る並行システム機能要求 $\mathcal{R}$ の例

定義 10 状態遷移システム Mは次の 4 項組である:

$M = \langle Q,\ E,\ \rightarrow,\ q_0 \rangle$
- Q 状態の有限集合
- E イベントの有限集合
- $\rightarrow$ 遷移関係 ($\rightarrow \subseteq Q \times E \times Q$)
- q_0 初期状態 ($q_0 \in Q$)

□

以下，遷移 $(p,\ a,\ q) \in \rightarrow$を $p \xrightarrow{a} q$と書く．$p \xrightarrow{a} q$は“状態 p において a というイベントが起きたときシステムの状態は qに遷移する”という意味である．ここで状態 p を遷移 $p \xrightarrow{a} q$の前状態，状態 qを遷移 $p \xrightarrow{a} q$の次状態と呼ぶ．

定義 11 任意の $X \in \mathcal{B}$ ($\mathcal{B} \subseteq \mathcal{A}$) に対して，$\mathcal{X} \equiv X$または $\mathcal{X} \equiv \neg X$であるとする[2]．つまり，$\mathcal{X}$は Xのリテラルである．形式仕様 $M = \langle Q,\ E,\ \rightarrow,\ q_0 \rangle$ の状態 $q \in Q$ に命題論理式としての性質を持たせ，$q = \bigwedge_{X \in \mathcal{B}} \mathcal{X}$と定義する．また，状態 qの持つ命題論理式としての性質を，状態 qの示す命題論理式と呼ぶ． □

補題 1 形式仕様 $M = \langle Q,\ E,\ \rightarrow,\ q_0 \rangle$ における任意の状態 $q \in Q$ の示す命題論理式は無矛盾である．
(証明) 状態の定義より，任意の $X \in \mathcal{B}$ ($\mathcal{B} \subseteq \mathcal{A}$) について $\mathcal{I}(\mathcal{X}) = \mathbf{true}$ となるような解釈 $\mathcal{I}$を選べば，$\mathcal{I}(\bigwedge_{X \in \mathcal{B}} \mathcal{X}) = \mathcal{I}(q) = \mathbf{true}$ である． □

命題 2 任意の素命題 $A \in \mathcal{B}$ ($\mathcal{B} \subseteq \mathcal{A}$) は形式仕様 $M = \langle Q,\ E,\ \rightarrow,\ q_0 \rangle$ における任意の状態 $q \in Q$ に対して従属である．
(証明) **命題 1**の対偶より明らか． □

[2]記号$\equiv$は，$\equiv$の左辺と右辺が構文的に等しいことを表す．

定義 12 形式仕様 $M = \langle Q,\ E,\ \rightarrow,\ q_0 \rangle$ の状態 $q \in Q$ の示す命題論理式を fとし，素命題を $A \in \mathcal{B}$ ($\mathcal{B} \subseteq \mathcal{A}$) とする．任意の解釈 $\mathcal{I}$に対して $\mathcal{I}(f \Rightarrow A) = \mathbf{true}$ となるとき，状態 qは素命題 A を満たすといい，$q \models A$ と記述する．満たさないときは $q \models \neg A$ と記述する． □

定義 3で定義した解釈 $\mathcal{I}$を以下のように拡張する．これは状態 $q \in Q$ における素命題 $A \in \mathcal{B}$ ($\mathcal{B} \subseteq \mathcal{A}$) の解釈を表す．

$$\mathcal{I}(q)(A) = \begin{cases} \mathbf{true} & \text{if} \quad q \models A \\ \mathbf{false} & \text{if} \quad q \models \neg A \end{cases}$$

この拡張により，状態 qにおける命題論理式の解釈を定義できる．

定義 13 形式仕様 $M = \langle Q,\ E,\ \rightarrow,\ q_0 \rangle$ の状態 $q \in Q$ において，$\mathcal{B}$ ($\mathcal{B} \subseteq \mathcal{A}$) から生成される命題論理式 gの解釈が **true** となるとき，状態 qは gを満たすといい，$q \models g$ と記述する．満たさないときは $q \models \neg g$と記述する． □

3 健全性と完全性

この節では，並行システム機能要求と状態遷移システムとの関係として，健全性と完全性の定義を与える．

健全性とは，状態遷移システムにおける全ての遷移に対して，それが満たすような機能が存在することであり，完全性とは，並行システム機能要求に関して健全な状態遷移システムの中で同型を除いて最も大きなものであることを指す．

最初に個々の遷移と機能との関係として充足性を定義し，次にそれを用いて，並行システム機能要求と状態遷移システムとの関係である健全性と完全性を定義する．

なお，サブシステムの機能要求と状態遷移システムの間の関係については議論しない．それについては従来の研究 [2], [4] を参照されたい．

定義 14 機能$\rho : f_{in} \overset{a}{\Rightarrow} f_{out}(a = p : e?\ \text{ or }\ a = p : e!\ \text{ or }\ a = \varepsilon)$ の要素について以下の記法を導入する:

$$pre(\rho) = f_{in}$$

$$post(\rho) = f_{out}$$

$$act(\rho) = a$$

$$ev(\rho) = \begin{cases} e & \text{if } a = p : e? \text{ or } a = p : e! \\ \varepsilon & \text{otherwise} \end{cases}$$

$$port(\rho) = \begin{cases} p & \text{if } a = p : e? \text{ or } a = p : e! \\ \mathit{undefined} & \text{otherwise} \end{cases}$$

□

定義 15 $port(\rho_i) \in P_{k_i}$であるようなサブシステム kの出力機能$\rho_i : f_{in_i} \overset{p:e!}{\Rightarrow} f_{out_i} \in R_k$に対して，それに対応する

他のサブシステムの入力機能の集合$peer(\rho_i)$を以下のように定義する:

$$peer(\rho_i) = \{\ \rho_j \mid \rho_j : f_{in_j} \overset{p:e?}{\Rightarrow} f_{out_j},\ \rho_j \in \bigcup_{l \neq k} R_l\}$$

□

定義 16 出力機能ρ_iに対する入力機能の集合$peer(\rho_i)$の任意の空でない部分集合を$\mathcal{IR}(\rho_i)$と表記する. □

定義 17 機能$\rho : f_{in} \overset{b!}{\Rightarrow} f_{out}$が内部機能であるとき，または環境ポートを介する機能のとき，状態遷移$t = \langle p \xrightarrow{a} q \rangle$が機能$\rho$を満たすとは以下が成り立つときである：

1. $p \models f_{in},\ a = b,\ q \models f_{out}$.
2. f_{out}に関して独立なリテラルlに対して，$p \models l$と$q \models l$は同値.

機能$\rho_i : f_{in_i} \overset{b!}{\Rightarrow} f_{out_i}$と$\rho_j \in \mathcal{IR}(\rho_i)$が内部ポートを介する入出力機能のとき，状態遷移$t = \langle p \xrightarrow{a} q \rangle$が機能$\rho_i$と$\rho_j \in \mathcal{IR}(\rho_i)$を満たすとは以下が成り立つときである：

1. $p \models pre(\rho_i) \wedge \bigwedge_j pre(\rho_j)\ (\rho_j \in \mathcal{IR}(\rho_i)),\ a = ev(\rho_i),\ q \models post(\rho_i) \wedge \bigwedge_j post(\rho_j)\ (\rho_j \in \mathcal{IR}(\rho_i))$.
2. $post(\rho_i) \wedge \bigwedge_j post(\rho_j)\ (\rho_j \in \mathcal{IR}(\rho_i))$に関して独立なリテラル$l$に対して，$p \models l$と$q \models l$は同値.

状態遷移tが機能ρを満たすとき，$t \models \rho$と記述する. □

上の定義において，1は現在の状態と次の状態がそれぞれ前提条件と事後条件を満たすことであり，2は事後条件に対して独立なリテラルlの真偽値は，遷移前の状態pと遷移後の状態qにおいて変化しないということである.

定義 18 状態遷移システム$M = \langle Q,\ E,\ \rightarrow,\ q_0 \rangle$が並行システム機能要求$\mathcal{R} = \{(\mathcal{R}_k,\ \gamma_{0k})\}\ (1 \leq k \leq n)$に関して健全であるとは，以下の条件を満たすときである.

1. $q_0 \models \bigwedge_k \gamma_{0k}$
2. 全ての$t \in \rightarrow$に対して，$t \models \rho$となるような$\rho \in \bigcup_k R_k$が存在する. □

完全性の定義は健全であることを使用して行う. 完全であるとは，健全である全ての状態遷移システムの中で必要な状態，遷移が全て存在していることを意味する.

完全性を定義する準備として，状態遷移システム間における準同形写像を定義する.

定義 19 $M = \langle Q,\ E,\ \rightarrow,\ q_0 \rangle$と$M' = \langle Q',\ E',\ \rightarrow',\ q_0' \rangle$を並行システム機能要求$\mathcal{R}$に関して健全な状態遷移システムとする. 写像$\xi : Q \rightarrow Q'$が以下を満たすとき，$\xi$を$M$から$M'$への準同形写像という.

1. $\xi(q_0) = q_0'$
2. $p \xrightarrow{a} q$ならば，$\xi(p) \xrightarrow{a} \xi(q)$
3. Mにおける全ての状態pと命題fに対して，$p \models f$と$\xi(p) \models f$は同値. □

定義19を用いて，完全性を次のように定義する.

定義 20 Mは並行システム機能要求$\mathcal{R}$に関して健全な状態遷移システムとする. Mが$\mathcal{R}$に関して完全であるとは，$\mathcal{R}$に関して健全な全ての状態遷移システムM'に対して，M'からMへの準同形写像が存在することである. □

4 形式仕様の合成

与えられた機能要求から形式仕様を合成する手法を述べる. 最初に合成の準備として，サブシステムの機能要求を正規形に変形する手法を述べる. 正規形ではすべての命題論理式がリテラルの連言で表されている. その後，サブシステムの機能要求から形式仕様の合成を述べる.

さらに，並行システム機能要求から形式仕様の合成法を述べる.

4.1 正規形

サブシステムの機能要求$\mathcal{R}_k = \langle R_k,\ P_k,\ \Sigma_k,\ \Delta_k,\ \mathcal{A}_k \rangle$から状態遷移システム$M = \langle Q,\ E,\ \rightarrow,\ q_0 \rangle$を合成することを考える.

まず，その準備としてR_kに現れるすべての命題を等価性を保ちながら，選言標準形$\gamma_1 \vee ... \vee \gamma_n$に変形する. ここで$\gamma_i\ (1 \leq i \leq n)$はリテラルの連言である. 選言標準形への変形については，例えば，文献[5]を参照されたい.

次に，変形されたR_kに以下の規則を適用する.

rule 1 $R_k \cup \{\gamma_1 \vee ... \vee \gamma_n \overset{a}{\Rightarrow} \gamma\}$を$R_k \cup \{\gamma_1 \overset{a}{\Rightarrow} \gamma, ..., \gamma_n \overset{a}{\Rightarrow} \gamma\}$とする.

rule 2 $R_k \cup \{\gamma_1 \wedge A \wedge \gamma_n \overset{a}{\Rightarrow} \gamma\}$を$R_k \cup \{\gamma_1 \wedge A \wedge \gamma_n \overset{a}{\Rightarrow} \gamma \wedge A\}$とする. ただし，$A,\ \neg A\ (A \in \mathcal{A}_k)$のいずれも$\gamma$中に存在しないとき.

rule 3 $R_k \cup \{\gamma_1 \wedge \neg A \wedge \gamma_n \overset{a}{\Rightarrow} \gamma\}$を$R_k \cup \{\gamma_1 \wedge \neg A \wedge \gamma_n \overset{a}{\Rightarrow} \gamma \wedge \neg A\}$とする. ただし，$A,\ \neg A\ (A \in \mathcal{A}_k)$のいずれも$\gamma$中に存在しないとき.

サブシステム機能要求$\mathcal{R}_k$に対して可能な限り上記の規則を適用し，変形したものを$\hat{\mathcal{R}}_k = \langle \hat{R}_k,\ P_k,\ \Sigma_k,\ \Delta_k,\ \mathcal{A}_k \rangle$とし，$\mathcal{R}_k$の正規形と呼ぶ.

4.2 サブシステムの形式仕様の合成

サブシステムの機能要求$\mathcal{R}_k = \langle R_k, P_k, \Sigma_k, \Delta_k, \mathcal{A}_k\rangle$と初期条件$\gamma_{0k}$が与えられたとき，対応する状態遷移システム$\mathcal{T}_s(\mathcal{R}_k, \gamma_{0k}) = \langle Q, E, \rightarrow, q_0\rangle$を以下の変換$\mathcal{T}_s$によって与える．

【変換$\mathcal{T}_s$】

1. $\mathcal{R}_k = \langle R_k, P_k, \Sigma_k, \Delta_k, \mathcal{A}_k\rangle$から正規形$\hat{\mathcal{R}}_k = \langle \hat{R}_k, P_k, \Sigma_k, \Delta_k, \mathcal{A}_k\rangle$を導出する．
2. $Q = \{\ \gamma \mid \gamma$は$\mathcal{A}_k$のすべての素命題のリテラルの無矛盾な連言$\}$
3. $E = \{\ a \mid \rho : f_{in} \stackrel{a}{\Rightarrow} f_{out} \in \hat{R}_k\}$
4. $\rho : f_{in} \stackrel{a}{\Rightarrow} f_{out} \in \hat{R}_k$に対して以下を満たすような$\gamma \stackrel{a}{\rightarrow} \gamma'$ $(\gamma, \gamma' \in Q)$
   (a) $\gamma \models f_{in}$
   (b) $\gamma' \models f_{out}$
   (c) f_{out}に関して独立なリテラルlに対して，$\gamma \models l$と$\gamma' \models l$は同値．
5. $q_0 = \gamma_{0k}$ □

例 2 例 1で示した機能要求$\mathcal{R}_1$から$\mathcal{T}_s(\mathcal{R}_1, A \wedge B)$の合成を行う．

まず，$\mathcal{T}_s$における 1 を行い，$\hat{\mathcal{R}}_1$を導出する．この場合は，

$\hat{\mathcal{R}}_1 = \langle\{\rho_1 : A \stackrel{p_a;a!}{\Rightarrow} \neg A,\ \rho_2 : B \stackrel{p_b;b?}{\Rightarrow} A \wedge B,\ \rho_3 : A \stackrel{p_c;c?}{\Rightarrow} A \wedge \neg B\},\ \{p_a, p_b, p_c\},\ \{p_b : b,\ p_c : c\},\ \{p_a : a\},\ \{A, B\}\rangle$

である．

次に，2 で状態を生成する．さらに，3 では，イベントの集合を生成し，4 で要素の 1 つ 1 つの遷移を導出する．そして，最後に初期状態を指定する．

このように，機能要求$\mathcal{R}_1$から，$\mathcal{T}_s$により，$\mathcal{T}_s(\mathcal{R}_1, A \wedge B)$が得られる．(図 2 参照) □

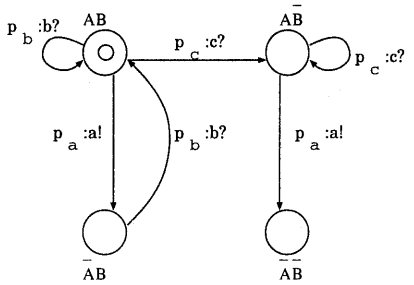

図 2: サブシステムの機能要求$\mathcal{R}_1$からの形式仕様の合成

4.3 並行システムの形式仕様の合成

並行システム機能要求$\mathcal{R} = \{(\mathcal{R}_k, \gamma_{0k})\}$ $(1 \leq k \leq n)$が与えられたとき，対応する状態遷移システム$\mathcal{T}(\mathcal{R}) = \langle Q, E, \rightarrow, q_0\rangle$を以下の変換$\mathcal{T}$によって与える．

【変換$\mathcal{T}$】

1. 各$\mathcal{R}_k = \langle R_k, P_k, \Sigma_k, \Delta_k, \mathcal{A}_k\rangle$から正規形$\hat{\mathcal{R}}_k = \langle \hat{R}_k, P_k, \Sigma_k, \Delta_k, \mathcal{A}_k\rangle$を導出する．
2. $Q = \{\ \gamma \mid \gamma$は$\bigcup_k \mathcal{A}_k$のすべての素命題のリテラルの無矛盾な連言$\}$
3. $E = \{\ ev(\rho) \mid port(\rho) \in P_{k_i},\ \rho \in \bigcup_k \hat{R}_k\} \cup \{act(\rho) \mid port(\rho) \in P_{k_e},\ \rho \in \bigcup_k \hat{R}_k\} \cup \{\varepsilon\}$
4. (a) $port(\rho_i) \in \bigcup_k P_{k_i}$であるような$\rho_i : f_{in_i} \stackrel{p;e!}{\Rightarrow} f_{out_i} \in \bigcup_k \hat{R}_k$と$\mathcal{IR}(\rho_i)$に対して以下を満たすような$\gamma \stackrel{ev(\rho_i)}{\rightarrow} \gamma'$ $(\gamma, \gamma' \in Q)$
   i. $\gamma \models pre(\rho_i)$
   ii. $\forall \rho_j \in \mathcal{IR}(\rho_i) \bullet \gamma \models pre(\rho_j)$
   iii. $\gamma' \models post(\rho_i)$
   iv. $\forall \rho_j \in \mathcal{IR}(\rho_i) \bullet \gamma' \models post(\rho_j)$
   v. $post(\rho_i) \wedge \bigwedge_j post(\rho_j), (\rho_j \in \mathcal{IR}(\rho_i))$に関して独立なリテラル$l$に対して，$\gamma \models l$と$\gamma' \models l$は同値．

   (b) $port(\rho) \in \bigcup_k P_{k_e}$であるような$\rho : f_{in} \stackrel{a}{\Rightarrow} f_{out} \in \bigcup_k \hat{R}_k$に対して以下を満たすような$\gamma \stackrel{a}{\rightarrow} \gamma'$ $(\gamma, \gamma' \in Q)$
   i. $\gamma \models f_{in}$
   ii. $\gamma' \models f_{out}$
   iii. f_{out}に関して独立なリテラルlに対して，$\gamma \models l$と$\gamma' \models l$は同値．

   (c) $\rho : f_{in} \stackrel{\varepsilon}{\Rightarrow} f_{out} \in \bigcup_k \hat{R}_k$に対して以下を満たすような$\gamma \stackrel{\varepsilon}{\rightarrow} \gamma'$ $(\gamma, \gamma' \in Q)$
   i. $\gamma \models f_{in}$
   ii. $\gamma' \models f_{out}$
   iii. f_{out}に関して独立なリテラルlに対して，$\gamma \models l$と$\gamma' \models l$は同値．
5. $q_0 = \bigwedge_k \gamma_{0k}$ □

例 3 例 1 で示した並行システム機能要求 $\mathcal{R}$ から $\mathcal{T}(\mathcal{R})$ の合成を行う．

まず，$\mathcal{T}$における 1 を行い，$\hat{\mathcal{R}}_1$,$\hat{\mathcal{R}}_2$,$\hat{\mathcal{R}}_3$を導出する．この場合は，

$\hat{\mathcal{R}}_1 = \langle\{\rho_1 : A \overset{p_a:a!}{\Rightarrow} \neg A,\ \rho_2 : B \overset{p_b:b?}{\Rightarrow} A \wedge B,\ \rho_3 : A \overset{p_c:c?}{\Rightarrow} A \wedge \neg B\},\ \{p_a,\ p_b,\ p_c\},\ \{p_b : b,\ p_c : c\},\ \{p_a : a\},\ \{A,\ B\}\rangle$, $\hat{\mathcal{R}}_2 = \mathcal{R}_2$, $\hat{\mathcal{R}}_3 = \mathcal{R}_3$

である．

次に，2 で状態を生成する．さらに，3 では，イベントの集合を生成し，4.(a) で内部ポートを介して行う要素の 1 つ 1 つの遷移を導出する．この例では，アクションが a である遷移と，b である遷移とについて導出すれば良い．4.(b) でも同じように，環境ポートを介する遷移を導出する．4.(c) でもまた，内部アクションで起こる遷移を導出する．そして，最後に初期状態を指定する．

並行システム機能要求 $\mathcal{R}$ から，$\mathcal{T}$により，$\mathcal{T}(\mathcal{R})$ が得られる．(図 3 参照) □

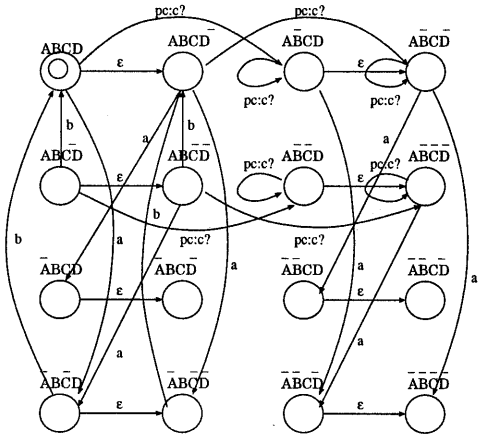

図 3: 並行システム機能要求 $\mathcal{R}$ からの形式仕様の合成

定理 1 変換 $\mathcal{T}$によって並行システム機能要求 $\mathcal{R}$ から合成された状態遷移システム $\mathcal{T}(\mathcal{R}) = \langle Q,\ E,\ \rightarrow,\ q_0\rangle$ は $\mathcal{R}$ に関して健全かつ完全な状態遷移システムである．

(証明) 健全性については，その定義と $\mathcal{T}$の定義との対応から，明らかに成り立つ．

完全性について以下で証明する．

$M = \langle Q',\ E',\ \rightarrow',\ q_0'\rangle$ を $\mathcal{R}$ に関して健全な状態遷移システムとする．また，$q \in Q'$に対し，$q \models \gamma$であるような全ての素命題のリテラルから成る無矛盾な連言γを与え、$\xi(q) = \gamma$となる写像$\xi : Q' \rightarrow Q$ を考える．

証明は，ξが Mから $\mathcal{T}(\mathcal{R})$ への準同形写像であることを示すことによって行う．

1. Mは $\mathcal{R}$ に関して健全なので，$q_0' \models \bigwedge_k \gamma_{0k}$である．
   従って，写像ξと $\mathcal{T}$の定義により，$\xi(q_0') = \bigwedge_k \gamma_{0k}$である．

2. Mにおける任意の遷移 $t = \langle p \xrightarrow{a} q\rangle$ に対して，それが満たすような機能$\rho : f_{in} \overset{a}{\Rightarrow} f_{out} \in \bigcup_k \mathcal{R}_k$を考えると，定義 18より，
   $$p \models f_{in},\ q \models f_{out}$$
   が得られ，$\mathcal{T}$の定義より $\mathcal{A}$ のすべての素命題のリテラルの無矛盾な連言となるような状態が生成されていて，写像ξによって写された状態が満たす命題論理式は変わらないので，
   $$\xi(p) \models f_{in},\ \xi(q) \models f_{out}$$
   が得られる．また，定義18より，f_{out}，または$post(\rho_i) \wedge \bigwedge_j post(\rho_j)\ (\rho_j \in \mathcal{IR}(\rho_i))$ に関して独立なリテラル lに対しては $p \models l$と $q \models l$は同値であり，写像ξの定義より，$\xi(p) \models l$と$\xi(q) \models l$は同値である．よって，$\mathcal{T}(\mathcal{R})$ に$\xi(p) \xrightarrow{a} \xi(q)$ が存在する．

3. Mにおける全ての状態 p と命題 fに対して，写像ξによって写された状態が満たす命題論理式は変わらないので，$p \models f$と$\xi(p) \models f$は同値であることが言える．

従って，ξは Mから $\mathcal{T}(\mathcal{R})$ への準同形写像である． □

〈例題〉 本論文で提案する要求記述法の適用例として，PHS を考える．この例では，並行システムは 2 つのサブシステムから成り，それぞれ，電話機，基地局を表している．2 つのサブシステムは同じポートを持つものとする．ここでは，次のような一部の機能だけを考える．

(1) 電源 On/Off 機能
(2) 移動通信機能
(3) 発信機能

また，それぞれのサブシステムについて以下の素命題を考える．

電話機 phs
- *power*: 電話機 phs の電源が On になっている．
- *link*: 電話機 phs の送信電波が基地局 base で受信可能である．

基地局 base
- *local*: 電話機 phs の位置情報を持っている．
- *comm*: 電話機 phs と通信可能である．

上記の非形式的な要求記述に基づき，次のような PHS の並行システム機能要求 $\mathcal{R}$ が考えられる．

$\mathcal{R} = \{(\mathcal{R}_p,\ \neg power \wedge link),$
$(\mathcal{R}_b,\ \neg local \wedge comm)\}$
$\mathcal{R}_p = \langle\{\rho_1 : \neg power \wedge link \overset{p:on!}{\Rightarrow} power,$

$$\rho_2 : \neg power \wedge \neg link \overset{\varepsilon}{\Rightarrow} power,$$
$$\rho_3 : power \overset{p:off!}{\Rightarrow} \neg power,$$
$$\rho_4 : link \overset{p:lo!}{\Rightarrow} \neg link,$$
$$\rho_5 : \neg link \overset{p:loc!}{\Rightarrow} link,$$
$$\rho_6 : power \wedge link \overset{p:tel!}{\Rightarrow} power \wedge link,$$
$$\{p\},\ \{\},\ \{p:on,\ p:off,\ p:lo,\ p:loc,$$
$$p:tel\},\ \{power,\ link\}\rangle$$

$$\mathcal{R}_b = \langle\{\rho_7 : \neg local \overset{p:on?}{\Rightarrow} local,$$
$$\rho_8 : local \overset{p:off?}{\Rightarrow} \neg local,$$
$$\rho_9 : local \overset{p:lo?}{\Rightarrow} \neg local\},$$
$$\rho_{10} : \neg local \overset{p:loc?}{\Rightarrow} local\},$$
$$\rho_{11} : local \wedge comm \overset{p:tel?}{\Rightarrow} local \wedge comm\},$$
$$\{p\},\ \{p:on,\ p:off,\ p:lo, p:loc,$$
$$p:tel\},\ \{\},\ \{local,\ comm\}\rangle$$

また, 上の並行システム機能要求 $\mathcal{R}$ に関して健全かつ完全な状態遷移システム ($T(\mathcal{R})$) は図 4 のようになる. 図 4 で状態番号とその状態が満たす命題論理式の関係を次の表に示す.

0	$\neg power \wedge link \wedge \neg local \wedge comm$
1	$power \wedge link \wedge \neg local \wedge comm$
2	$\neg power \wedge link \wedge \neg local \wedge \neg comm$
3	$power \wedge link \wedge \neg local \wedge \neg comm$
4	$\neg power \wedge link \wedge local \wedge comm$
5	$power \wedge link \wedge local \wedge comm$
6	$\neg power \wedge link \wedge local \wedge \neg comm$
7	$power \wedge link \wedge local \wedge \neg comm$
8	$\neg power \wedge \neg link \wedge \neg local \wedge comm$
9	$power \wedge \neg link \wedge \neg local \wedge comm$
10	$\neg power \wedge \neg link \wedge \neg local \wedge \neg comm$
11	$power \wedge \neg link \wedge \neg local \wedge \neg comm$
12	$\neg power \wedge \neg link \wedge local \wedge comm$
13	$power \wedge \neg link \wedge local \wedge comm$
14	$\neg power \wedge \neg link \wedge local \wedge \neg comm$
15	$power \wedge \neg link \wedge local \wedge \neg comm$

5 むすび

本論文では, 修正変更に柔軟に対処可能な方法として, 並行システムへ拡張した機能に関する要求記述から, 形式仕様としての状態遷移システムを導出する手法を提案し, その妥当性として導出された形式仕様が要求記述に関して健全かつ完全であることを示した. また, 並行システムが持っているポートを内部ポートと環境ポートに分けたことによって, 並行システムを更に階層化し, より大きな並行システムに拡張することができるようになっている.

今後の課題としては, 本手法に基づいたシステム仕様記述支援ツールを開発することが考えられる.

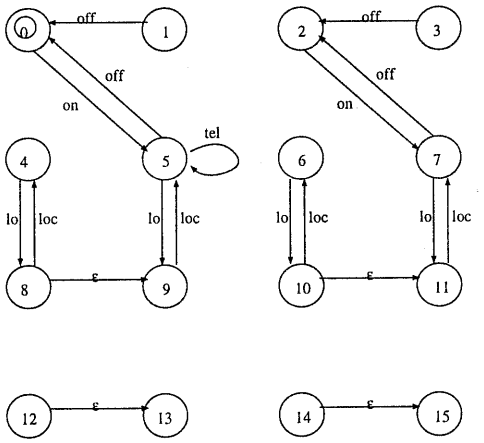

図 4: PHS($\mathcal{R}$) の状態遷移システム

参考文献

[1] K.J.Turner, "Using Formal Description Techniques," JOHN WILEY & SONS, 1993.

[2] 宋国換, 富樫敦, 白鳥則郎, "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," 情報処理学会論文誌, Vol.37, No.4, pp.511-519, 1996.

[3] A.Togashi, N.Usui, K.Song and N.Shiratori, "Synthesis of Formal Specifications from User Requirements and its Flexibility," Tech. Rep. of IEICE, IN94-200, 1995.

[4] 臼井伸幸, 高橋薫, 神長裕明, 白鳥則郎, "形式仕様の開発における機能要求への反映," 電子情報通信学会技術報告 SSE95-65, pp. 67-72, 1995.

[5] M.L. シャグリン, W.J. ラバポート, R.R. ディバート著, 大矢建正訳, "論理とアルゴリズム," マグロウヒル, 1986.