マルチメディア通信と分散処理 58−11
（1992. 11. 19）

LOTOSと時制論理に基づくプロトコル検証

高橋 薫† 加藤 靖†† 安藤 敏彦†† 野口 正一†††

† 東北大学電気通信研究所
†† 仙台電波高専
††† 東北大学応用情報学研究センター

本論文では、形式記述技法LOTOSを用いて記述されたプロトコル仕様に対して、その時間的な性質の検証支援を可能とするモデル論的方法論を提案する。本方法論において、プロトコルはその実際的なシステム構造を反映したLOTOSプロセスで仕様化される。検証されるべき時間的性質は、プロトコルの満たすべき要求としてイベントの観点から与えられ、分岐時間時制論理を用いて定式化される。検証においては、プロトコルのLOTOS仕様から導出される遷移システムのイベント系列(木)をモデルとして着目し、与えられた時制論理式がそのモデルで充足されるかどうかを通して検証を行う。時制論理式としては、原子的なものに加え、プロトコルの検証という観点で有効と考えられるものを併せて提供しており、検証のための有用なインタフェースとなっている。

Protocol Verification using LOTOS and Temporal Logic

Kaoru TAKAHASHI† Yasushi KATO†† Toshihiko ANDO†† Shoichi NOGUCHI†††

† Research Institute of Electrical Communication, Tohoku University
Katahira 2-1-1, Aoba-ku, Sendai-shi, 980 JAPAN

†† Sendai National College of Technology
Kitahara 1, Kami-ayashi, Aoba-ku, Sendai-shi, 989-31 JAPAN

††† Research Center for Applied Information Sciences, Tohoku University
Katahira 2-1-1, Aoba-ku, Sendai-shi, 980 JAPAN

In this paper, we propose a model-theoretic methodology by which it is possible to verify the temporal properties of protocol specifications described in the formal description technique LOTOS. In the proposed methodology, a protocol is specified as a LOTOS process, which reflects the actual system structure of the protocol, and the possible behavior of the whole system is obtained as a corresponding transition system. The required temporal properties is given in terms of events, and formulated using the branching time temporal logic. The verification is performed by checking the satisfiability of the given temporal formulas in the model, which is the event part of the transition system derived from the specification. To facilitate protocol verification, we provide some protocol-specific temporal formulas, in addition to the primitive ones.

1. はじめに

プロトコルなどの通信システムの仕様を曖昧なく厳密に記述するためには形式記述技法の採用が不可欠である。LOTOS[1],[2]はISOで開発された形式記述技法の一つであり、OSIプロトコルや分散システムの形式記述に広く使用されてきている。また、仕様の解析に対して等価性の概念に基づいた強力な数学的枠組みを持つ。その一方、LOTOSはイベント個々を主体にして仕様記述を行うため、システムの進行過程における時間的に離れたイベント間の性質・関係などを直接的に表現することはできない。

このような時間的な性質を直接的に表現する方法としては時制論理が有用である[3],[4]。時制論理を用いることによって、時間に渡る安全性の性質や生存性の性質の仕様化が可能となる。例えば、プロトコルにおいて、ある通信エンティティがメッセージを送信したならば、別の通信エンティティがそのメッセージをいつか受信するというような生存性を表現することができる。

本論文では、LOTOSとプロトコルと時制論理の考え方を有機的に結合することによって、LOTOSを用いて記述されたプロトコル仕様に対して、時制論理で表現された時間的な性質の検証支援を可能とするモデル論的方法論を提案する。本方法論の特徴は、LOTOSがイベントの概念に基づく形式記述技法であるということから、イベントを基本として時制論理の構文と意味を導入し、LOTOSによるプロトコル仕様との円滑な橋渡しを実現していることにある。また、対象がプロトコル仕様であることを考慮した枠組みを提供している。なお、従来、時間的性質の検証のための同様の方法があるが[6]、そこではSDL仕様を対象とし、イベントについての陽な表現は提供していず、また、プロトコル特有な枠組みは有していない。

本方法論においては、プロトコルはその実際的なシステム構造を反映したLOTOSプロセスで仕様化される。時間的性質は、プロトコルの満たすべき要求としてイベントの観点から与えられ、イベントの概念に基づいた分岐時間時制論理(の式)を用いて定式化される。この時制論理では、イベントから構成される木をモデルとし、イベント発生の時間的な流れに対応する木上の可能なパス群を考慮してその論理式の意味を規定している。プロトコルの検証においては、与えられたLOTOS仕様の意味として導出される遷移システムをこの時制論理のモデルに対応づけ、与えられた時制論理式がそのモデルで充足されるかどうかを通して検証を行う。時制論理式としては、原子的なものに加え、プロトコルの検証という観点で有効と考えられるものを枠組みとして提供しており、検証のための有用なインタフェースとなっている。

本論文の構成は次の通りである。2.ではイベントの概念に基づいた分岐時間時制論理を与え、3.では対象とするプロトコルのLOTOSによる仕様化の考え方を与える。4.では2.と3.の議論に基づき、検証の具体的方法論を提案する。

なお以下では、紙面の都合上、LOTOSについての基本知識は前提として議論を進める。また、以下で与える諸定義において用いる論理式や集合などの表現記法は主としてZ記法[8]に基づいている。

2. イベント概念に基づいた時制論理

本節では、LOTOSによるプロトコル仕様の検証のためのイベントの概念に基づいた時制論理を与える。

この時制論理(以下、*EBBTTL*と呼ぶ)は[7]の分岐時間時制論理を基本とし、これをイベントの観点からの解釈の枠組みで拡張している。1.で述べたように、本論文における検証の考え方はモデル論的方法であり、採用するEBBTTLのモデルはイベントから構成される木である。原子命題は木の枝に関して定義され、可能な枝のパス群を考慮してEBBTTL論理式の構文と意味解釈が与えられる。枝のパスは時間の流れに対応すると考えることができる。

2.1 EBBTTLの構文

基本となる原子命題の有限集合をΩとおく。Ωの要素はLOTOSにおけるイベント(アクション)と対応を成すものであり、適当に与えられた観測可能イベント、内部イベントi、正常終結を表す特別なイベントδからΩが構成される。ΩのもとでEBBTTLの論理式を次のように帰納的に与える。

【定義1】 EBBTTL論理式

原子命題の集合をΩとする。このとき、以下で定義される記号列はEBBTTLの論理式である。

(i) $\alpha \in \Omega$について、α(イベント)は論理式。

(ii) ϕが論理式のとき、$\neg\phi$ (negation)は論理式。

(iii) ϕとψが論理式のき、$\phi \wedge \psi$ (conjunction)、$\phi \vee \psi$ (disjunction)、$\phi \oplus \psi$ (exclusive disjunction)、$\phi \Rightarrow \psi$ (implication)、$\phi \Leftrightarrow \psi$ (bi-implication)は論理式。

(iv) ϕが論理式のとき、$\Box_\forall \phi$ (universally always)、$\Box_\exists \phi$ (existentially always)、$\Diamond_\forall \phi$ (universally sometime)、$\Diamond_\exists \phi$ (existentially sometime)は論理式。

(v) ϕとψが論理式のとき、$\bigcirc_\forall \phi$ (universally weak next)、$\bigcirc_\exists \phi$ (existentially weak next)、$\circledcirc_\forall \phi$ (universally strong next)、$\circledcirc_\exists \phi$ (existentially strong next)、$\phi \nabla_\forall \psi$

(universally until) 、$\phi\ \nabla_{\exists}\ \psi$ (existentially until) は論理式。 □

上の定義において、(i)-(iii) は命題論理における論理式である。(iv) と (v) は時制演算子 (temporal operator) を含む論理式である。特に、(iv) は伝統的な時制論理式である。$\Box_{\forall}$、$\Diamond_{\forall}$、$\bigcirc_{\forall}$、$\circledcirc_{\forall}$、$\nabla_{\forall}$ は、木のすべてのパスに渡るものであり、$\Box_{\exists}$、$\Diamond_{\exists}$、$\bigcirc_{\exists}$、$\circledcirc_{\exists}$、$\nabla_{\exists}$ は、木のあるパスを対象とするものである。従って、例えば、$\Box_{\forall}$ a は、どんな場合においてもいつでも a が成り立つ (イベント a が生起可能である) ことを表す。一方、$\Diamond_{\exists}$ a は、ある場合においていつか a が成り立つ (イベント a が生起可能である) ことを表す。

S を論理式の集合としたとき、$\vee$ と $\wedge$ に関して、次の省略記法を導入する:

$\vee$S ——— S中の各論理式の $\vee$ による結合
$\wedge$S ——— S中の各論理式の $\wedge$ による結合

2.2 EBBTTLの意味

EBBTTL論理式の意味を形式的に定義するための準備としてEBBTTLのモデルを導入する。

【定義2】 EBBTTLのモデル

EBBTTLのモデル τ は次の4項組である。

$$\tau = <V, \Omega, E, v_0>$$

ここで、V は節点の集合、Ω はイベントの集合、E は枝の集合 ($E \subseteq V \times \Omega \times V \ \wedge\ E \neq \varnothing$)、$v_0$ は根節点 ($v_0 \in V \ \wedge\ \neg(\exists v:V \bullet \exists a:\Omega \bullet (v, a, v_0) \in E)$) である。但し、$\tau$ は有向閉路を持たず連結である。 □

すなわち、モデル τ は v_0 を根とし各 (有向) 枝に1つのイベントをラベルとして持つ無限または有限の木である。

【定義3】

$\tau = <V, \Omega, E, v_0>$ を EBBTTL のモデルとする。この時、次の記法、用語を導入する。

枝 e=(v, a, u) を見やすさのため v—a→u と書く。a を e のラベルといい、*label(e)* と表す。また、v、u を、それぞれ、e の始点、終点と呼び *src(e)*、*dst(e)* で表す。

枝 e のすぐ後ろに連結しうる枝の集合を *Next(e)* と表す。 $f \in Next(e)$ *iff* src(f)=dst(e)

連結した枝の1個以上の系列 $<e_1, e_2, e_3, \cdots\cdots>$ を枝のパスという。$p=<e_1, e_2, e_3, \cdots\cdots>$ が枝のパスであるとき、その長さ (枝の数) を *length(p)* で表す。また、p を構成する枝 e_i を p の要素という。e_i が p の要素であるとき、$e_i \in p$ と表す。

$p=<e_1, e_2, \cdots\cdots, e_{m-1}, e_m, \cdots\cdots>$ が枝のパスであるとき、p の有限部分パス $<e_1, e_2, \cdots\cdots, e_{m-1}>$ を、枝のパス p における枝 e_m のプレフィクスといい、*prefix(p, e_m)* と表す。

枝 e_1:E から始まる枝のパス $p=<e_1, e_2, e_3, \cdots\cdots>$ が以下の① または② を満たすとき、p を枝 e_1 からの枝の最大パスという。

① p は無限のパス (length(p) = ∞)
② p が有限のパス $<e_1, e_2, e_3, \cdots\cdots, e_n>$ ならば、e_n のすぐ後ろに連結しうるどんな枝も E に存在しない

枝 e:E に対して、*Path(e)* を次のように定義する。

Path(e) = { p | p は枝 e からの枝の最大パス } □

EBBTTL論理式の意味を充足関係を定義することで与える。これはモデルとその枝および論理式の間の関係であり、論理式がその状況 (そのモデルとその枝) において真 (true) であることを表す。

【定義4】 EBBTTL論理式の意味

$\tau = <V, \Omega, E, v_0>$ をモデル、e を E の要素、α を Ω の要素とする。また、ϕ、ψ を論理式とする。このとき、充足関係 $\models$ を次のように定義する。

(1) $\tau, e \models \alpha$ *iff* α = label(e)
(2) $\tau, e \models \neg\phi$ *iff* $\tau, e \models \phi$ ではない
(3) $\tau, e \models \phi \wedge \psi$ *iff* $(\tau, e \models \phi)$ かつ $(\tau, e \models \psi)$
(4) $\tau, e \models \phi \vee \psi$ *iff* $\tau, e \models \neg(\neg\phi \wedge \neg\psi)$
(5) $\tau, e \models \phi \oplus \psi$
iff $\tau, e \models (\phi \wedge \neg\psi) \vee (\neg\phi \wedge \psi)$
(6) $\tau, e \models \phi \Rightarrow \psi$ *iff* $\tau, e \models \neg\phi \vee \psi$
(7) $\tau, e \models \phi \Leftrightarrow \psi$
iff $\tau, e \models (\phi \Rightarrow \psi) \wedge (\psi \Rightarrow \phi)$
(8) $\tau, e \models \Box_{\forall}\phi$
iff $\forall p:Path(e) \bullet \forall f:E \bullet$
$(f \in p$ ならば $\tau, f \models \phi)$
(9) $\tau, e \models \Box_{\exists}\phi$
iff $\exists p:Path(e) \bullet \forall f:E \bullet$
$(f \in p$ ならば $\tau, f \models \phi)$
(10) $\tau, e \models \Diamond_{\forall}\phi$
iff $\forall p:Path(e) \bullet \exists f:E \bullet$
$(f \in p$ かつ $\tau, f \models \phi)$
(11) $\tau, e \models \Diamond_{\exists}\phi$
iff $\exists p:Path(e) \bullet \exists f:E \bullet$
$(f \in p$ かつ $\tau, f \models \phi)$
(12) $\tau, e \models \bigcirc_{\forall}\phi$
iff $\forall f:E \bullet (<e, f>$ が枝のパス
ならば $\tau, f \models \phi)$
(13) $\tau, e \models \bigcirc_{\exists}\phi$
iff $(\exists f:E \bullet (<e, f>$ が枝のパス

かつ $\tau, f \models \phi$))

あるいは

(Next(e) = ∅)

(14) $\tau, e \models \bigcirc_\forall \phi$

iff (Next(e) ≠ ∅)

かつ

(∀ f:E • (<e, f> が枝のパス

ならば $\tau, f \models \phi$))

(15) $\tau, e \models \bigcirc_\exists \phi$

iff ∃ f:E • (<e, f> が枝のパス

かつ $\tau, f \models \phi$)

(16) $\tau, e \models \phi \nabla_\forall \psi$

iff ∀ p:Path(e) • ∃ f:E •

(f ∈ p かつ

$\tau, f \models \psi$ かつ

(∀ g:E • g ∈

prefix(p, f) ならば $\tau, g \models \phi$))

(17) $\tau, e \models \phi \nabla_\exists \psi$

iff ∃ p:Path(e) • ∃ f:E •

(f ∈ p かつ

$\tau, f \models \psi$ かつ

(∀ g:E • g ∈

prefix(p, f) ならば $\tau, g \models \phi$))

□

充足関係 ⊨ の (1) の定義は、イベントのアトミック性 [1], [2] の仮定に基づくものである。(12)-(15) における weak next と strong next の違いは、枝 e に連結しうる枝がなくても良いか必ずなくてはならないかの違いである。

モデル $\tau = <V, \Omega, E, v_0>$ のある枝 e:E に対して $\tau, e \models \phi$ であるとき、論理式 ϕ は τ において充足可能であるといい、$\tau \models \phi$ と書く。

$src(e) = v_0$ であるようなある枝 e:E に対して $\tau, e \models \phi$ であるとき、論理式 ϕ は τ においてあるパスで初期的に充足可能 (existentially-initially-satisfiable) であるといい、$\tau \models_\exists \phi$ と書く。$src(e) = v_0$ であるようなすべての枝 e:E に対して $\tau, e \models \phi$ であるときは、論理式 ϕ は τ においてすべてのパスで初期的に充足可能 (universally-initially-satisfiable) であるといい、$\tau \models_\forall \phi$ と書く。これらの概念は、演算子 □ との組合せで、システム全体としての不変な挙動を EBBTTL 論理式により特性化するのに有用である。

3. プロトコルの LOTOS によるモデル化

本節では、検証対象となるプロトコルの LOTOS による仕様化のモデルを与える。

一般にプロトコルは階層的な構成をとっており、ある階層においては、対応するプロトコルの規定に従ってその動作を遂行するプロトコルエンティティ (PE, Protocol Entity) が存在する。層を構成するプロトコルエンティティは (通常は) 二つであり、それらの間で通信オブジェクト (メッセージ、PDU など) の交換が行われる。プロトコルエンティティ間のメッセージは下位層のサービスを利用して交換される。従って、プロトコルを図 1 のようにモデル化することができる。

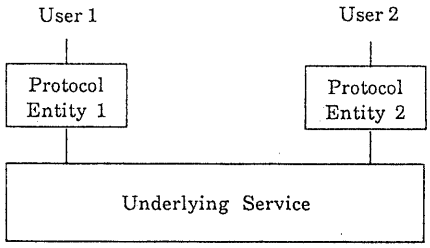

図 1　プロトコルのモデル化

LOTOS の観点からは、プロトコルを次のような LOTOS プロセスとして定式化できる :

```
process Protocol [U1, U2, L1, L2] :=
      PE1 [U1, L1]
   |[L1]|
      UService [L1, L2]
   |[L2]|
      PE2 [U2, L2]
endproc
```

ここで、U_i (i=1, 2) は、プロトコルエンティティ i (PE_i) とユーザ i (上位層の User i) の間のインタラクションポイントを表すゲートである。L_i (i=1, 2) は、プロトコルエンティティ i (PE_i) と下位層サービス (UService) の間のインタラクションポイントを表すゲートである。

この層のサービスは、上記の Protocol プロセスにおける各 L_i でのイベントをユーザから隠蔽するか、U_i (i=1, 2) にのみ着目してユーザとのインタラクション (イベント) を記述することで定義できる :

```
process Service [U1, U2] :=
   hide  L1, L2  in  Protocol [U1, U2, L1, L2]
endproc

process Service [U1, U2] :=
   --------------
endproc
```

本論文では、LOTOS によるこのモデル化において、PE1、PE2、UService の内部には立ち入らない。これは対象とするプロトコルに依存するからである。ゲートに関しては、上記のものを上記の意味で

固定して取り扱うこととする。なお、柔軟性のため、PE1、PE2、UServiceの中で必要に応じてゲートを導入しても良いこととする。

このようなモデル化のもと、以下のような項目についての検証を期待できる。

(1) サービスユーザ間でのイベントの間の時間関係
(2) サービスユーザと下位層サービスの間のイベントの間の時間関係
(3) プロトコルエンティティ間(L1とL2の間)のイベントの間の時間関係
(4) 上記の組合せによるイベントの間の時間関係

(1)はサービスに関連し、$User_i$のU_iのところでのイベントと$User_j$ ($j \neq i$)のU_jのところでのイベントの間の関係を表現するものである。(2)はサービスとプロトコルに関連し、U1のところでのイベントとL1のところでのイベントの間の関係、U2のところでのイベントとL2のところでのイベントの間の関係、U1のところでのイベントとL2のところでのイベントの間の関係、U2のところでのイベントとL1のところでのイベントの間の関係を表現する。(3)はプロトコルに関連し、PE_iのL_iのところでのイベントとPE_j ($j \neq i$)のL_jのところでのイベントの間の関係を表現するものである。そして(4)は、(1)～(3)による関係を利用してイベントの間の関係を表現するものである。

また、上記のようなイベントの間の関係の他に、次のような一般的な項目の検証も期待される。

(5) 終了性
(6) デッドロック性
(7) ライブロック性

(5)は、プロトコル動作がひととおり正常に終了するような状況を表している。これはLOTOSのδイベントでモデル化できる。(6)は、プロトコル動作が途中で(異常に)停止するような状況を表している。(7)は、サービス動作の刺激によるプロトコル動作の結果がサービス動作として現れず永久にプロトコルの内部動作を繰り返すような状況を表すものである。

以上の検証項目の具体的な表現についての議論は次節で与える。

4. プロトコル検証方法論

本節では、2.と3.における議論を土台として、LOTOSで記述されたプロトコル仕様の時制論理を用いた検証のための方法論を与える。まず4.1では、プロトコルにおける時間的性質についての検証内容を有用で簡潔に表現する枠組みを述べる。そして4.2では、検証のための具体的な方法論を与え、4.3でその適用例を示す。

4.1 プロトコル検証のための有用表現

プロトコル検証のための表現には基本的に2.で与えたEBBTTLの論理式を用いる。しかし、それらは原子的あるいは一般的なものであるから、3.で述べた典型的な検証内容の表現に適切でない場合がありうる。従って、それらに加えて、より簡潔(便利)で理解性が高い有用表現の導入が望まれる。以下では、考えられるそのような有用表現(イタリック体の語で定義)のいくつかを原子的なものの組合せで与える。

- イベント

3.で述べたプロトコルモデル中の各インタラクションポイント(ゲート)でのイベントを分かりやすく次のように表現する。

A.v

ここで、Aはゲート名であり、vは値である。これにより、イベントA<v>:Ωを表現する。A<v>はLOTOSに準じたイベント表現であり、ゲートAで値vが生起するということを表す。

- イベント発生の因果関係

各インタラクションポイント(ゲート)でのイベント間の時間的性質(イベント発生の因果関係)を容易に表現するため、次のような有用表現を導入する。

$$A.v\ \textit{ENABLES}\ (\otimes_1 B.u_1, \cdots, \otimes_n C.u_n) \ == \ A\langle v\rangle \Rightarrow \Diamond_\forall (\otimes_1 B\langle u_1\rangle \vee \cdots \vee \otimes_n C\langle u_n\rangle)$$

$$A.v\ \textit{enables}\ (\otimes_1 B.u_1, \cdots, \otimes_n C.u_n) \ == \ A\langle v\rangle \Rightarrow \Diamond_\exists (\otimes_1 B\langle u_1\rangle \vee \cdots \vee \otimes_n C\langle u_n\rangle)$$

ここで、$\otimes_h$ ($1 \leq h \leq n$)は有用表現用の記号ではなく、その出現場所には何も書かれないか否定記号¬が書かれる。*ENABLES*および*enables*の第2引数の要素が1つの場合は括弧を省略してよい。==はその左辺が右辺の省略形(左辺は右辺によって定義される)であることを表す。

これらの有用表現により、例えば、サービスプリミティブ間の関係を次のように分かりやすく簡潔に表現できる:

U1.CONReq *enables* U2.CONInd
U1.CONReq *ENABLES* (U1.CONConf, U1.DISInd)

なお、*ENABLES*および*enables*の両サイドは通常の論理式であっても良い。

$$\phi\ \textit{ENABLES}\ \psi \ == \ \phi \Rightarrow \Diamond_\forall \psi$$
$$\phi\ \textit{enables}\ \psi \ == \ \phi \Rightarrow \Diamond_\exists \psi$$

- イベント系列

連続したイベントの系列 $\alpha_1, \alpha_2, \cdots\cdots, \alpha_n$ を表現したい場合がある。これを原子的なものだけで書くのは容易ではなく、このための有用表現として *seq* を導入する(seq のかわりに < と > でイベント系列を囲んで表しても良い)。

$$
\begin{aligned}
seq(\alpha_1) &== \alpha_1 \\
seq(\alpha_1, \alpha_2) &== \alpha_1 \wedge \bigcirc_{\exists} \alpha_2 \\
seq(\alpha_1, \alpha_2, \cdots, \alpha_n) &== \alpha_1 \wedge \bigcirc_{\exists} (seq(\alpha_2, \cdots, \alpha_n)) \\
& \qquad (n \geq 3)
\end{aligned}
$$

例えば、次のように書くことによって、コネクション確立要求プリミティブに応じて、コネクション確立要求 PDU 、コネクション確立指示プリミティブが連続的に発生する可能性がシステムの不変性質(invariant)であることが表現される:

$$\Box_{\forall}(\text{U1.CONReq } \textit{enables} \ \langle \text{L1.CRPDU, L2.CRPDU, U2.CONInd} \rangle)$$

- 再送

転送したメッセージの紛失の再送による回復制御は、プロトコルに特有な機能の一つである。このための有用表現として次を導入する。

$$\text{A.u } \textit{by\_frequent\_retransmission\_of } \text{B.v} == (\Box_{\exists} \Diamond_{\exists} \text{B.v}) \Rightarrow (\Diamond_{\exists} \text{A.u})$$

例えば、データ転送要求プリミティブの再送の繰り返しにより、いつかデータ転送指示プリミティブが現れることは次のように表現できる:

$$\text{U2.DATAInd } \textit{by\_frequent\_retransmission\_of } \text{U1.DATAReq}$$

- 終了

終了は LOTOS の正常終結イベント δ を用いて定式化し、語 *exit* により分かりやすく表現する。

$$exit == \delta$$

- デッドロック

デッドロックはプロトコル動作が正常でなく停止する状況であるので、次のように定義される *deadlock* で表現する。

$$deadlock == (\neg \bigcirc_{\forall}(\bigvee\{\alpha : \Omega \mid \text{true}\})) \wedge \neg\delta$$

- ライブロック

ライブロックは *livelock* で表現する。つまり、次の式に示すように、観測可能イベントを決してもたらさず内部イベントの発生に終始(プロトコルの内部動作の繰り返し)する状況である。式中の i は、外部(サービス)から隠蔽されたプロトコル動作の各々に対応している (3. を参照)。

$$livelock == \Box_{\forall} \text{i}$$

- 不変性質

システム全体に渡る不変性質は演算子 □ を用いて表現できるが、それをより分かりやすく強調するため、次のような代替表現を導入する。

$$
\begin{aligned}
INVARIANT\ \phi &== \Box_{\forall}\phi \\
invariant\ \phi &== \Box_{\exists}\phi
\end{aligned}
$$

4.2 方法論

本節では、LOTOS で記述されたプロトコル仕様の時制論理を用いた検証のための具体的な(機械処理を可能とする)方法論を与える。

【検証の方法論】

① 検証対象となるプロトコルの LOTOS 仕様 *spec* を与える。
   —— *spec* は、必要に応じたゲートの導入などを別として、基本的に 3. で与えた構造をとるものとする。また、その動的意味としての遷移システムが有限であるようなものを対象とする。これは機械化の上での必須条件である。

② 検証したい時間的性質 *props* を定式化する。
   —— *props* は 4.1 で導入した有用表現も含めた EBBTTL 論理式の集合である。有用表現は本来の EBBTTL 論理式に帰着可能であるから、必要に応じて展開を行う。

③ *spec* に対応する遷移システム *TS(spec)* を導出する。
   —— ここでは、LOTOS の遷移導出体系 [1] を用いて、*spec* を有限な遷移システム *TS(spec)* に変換する。この変換においては、従来の汎用的な LOTOS 関連ツールを有効に使用することができる。

④ *TS(spec)* を論理的に 2.2 で定義した EBBTTL のモデルに対応づけした $TS(spec)_{MOD}$ を構成する。
   —— *TS(spec)* は基本的にグラフ構造を成しており、EBBTTL のモデルには直接合致しない。そこで、なんらかの技法により、対応する木構造 $TS(spec)_{MOD}$ を得る必要がある。簡単な方法は、*TS(spec)* にループがあれば前に戻らず下に展開(開く、unfold)す

ることである。この展開は無限になるが、内容は以前のものの繰り返しであるから、機械的に繰り返しをうまく処理するようにすれば良い。例えば、図2の左のような遷移システムを同図の右のように扱う。そして、この展開形のもとで枝のパスなどを算出するようにする。

⑤ $TS(spec)_{MOD}$ の上で *props* の充足可能性を調べる。
—— ここで実際に *props* 中の各々の論理式を評価する。一般には、2. で与えたように、$TS(spec)_{MOD}$ の各枝 e について論理式 ϕ: *props* の充足可能性 ($TS(spec)_{MOD}$, e $\models \phi$) の判定を行うのであるが、特に興味があるのは、システム全体の不変的な時間的性質である。従って、ϕ は *INVARIANT* ψ あるいは *invariant* ψ の形をとるものが主であると考えられる。この場合、$TS(spec)_{MOD} \models_V \phi$ あるいは $TS(spec)_{MOD} \models_\exists \phi$ を調べることが検証の主対象となる。 □

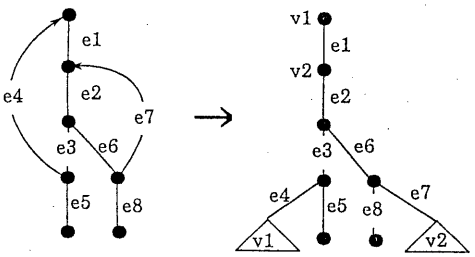

図2　遷移システムの展開

4.3　適用例

本節では、簡単なプロトコルを例として前記の方法論による時間的性質の検証を行う。

ここで扱う例は、図3に示すような、単方向データ転送のプロトコルである。プロトコルのサービス境界では、送り手からのデータが受け手に配送されることの繰り返しが規定される。便宜上、データは1種類(1つの値)だけとする。プロトコルでは、データの転送方向に欠落の生じうる下位サービス(UService)を利用し、その上で、送信側プロトコルエンティティ(PE1)と受信側プロトコルエンティティ(PE2)の間で転送の確認制御、再送制御を行いながら、上記のサービスを実現するような仕様を規定する。PE1は送信用プロセス(Sender)と回復制御用プロセス(Recover)に構造化される。

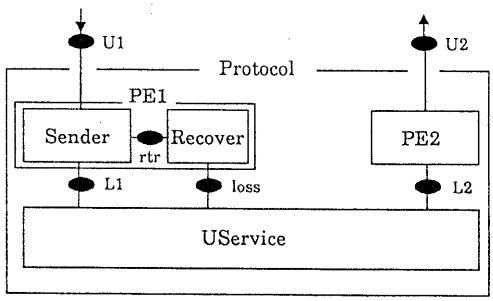

図3　プロトコル例

このプロトコルは、図4の右側に示すLOTOS仕様として記述される。このLOTOS仕様では、U1、U2、L1、L2の他に、欠落を表すゲートlossと再送促進用のゲートrtrを新たに導入している。これらは内部的なものであるから隠蔽している。

この検証対象プロトコルに対して、種々の時間的性質を検証可能であるが、ここでは、例として、有用表現で定式化される次のような生存性と非デッドロック性の検証を行う。

(生存性1)　*INVARIANT* (U1.Ud *enables* U2.Ud)
送信ユーザからのメッセージの送信は、受信ユーザによるそのメッセージの受信をもたらす。

(生存性2)　*INVARIANT* (L2.Ud *enables* L2.ack)
受信側プロトコルエンティティは、メッセージの受信後、いつか、確認を返す。

(非デッドロック性)　*INVARIANT* ¬*deadlock*
プロトコル動作は決して異常終了しない。

図4の右側のLOTOS仕様に対応する遷移システムは同図左側に示すとおりである(なおこれは、既存のLOTOS用解析ツールSAL[9]を用いて得た結果であり、変数の部分(m)は値に置き換えずにそのままの形での変数付き遷移システムを導出しているため、正確には、図中の変数mを値Udに置き換えたものが対応する遷移システムである)。そして、この遷移システムからEBBTTLのモデルに対応した木表現が構成される。結果的には、それぞれがuniversally-initially-satisfiableであることが確認され、所望の時間的性質の検証を達成することができる。

5. むすび

本論文では、LOTOSとプロトコルと時制論理の考え方を有機的に結合することによって、LOTOSを用いて記述されたプロトコル仕様に対して、時制論理で表現された時間的な性質の検証支援を可能とするモデ

ル検査(model checking)的な方法論を提案した。プロトコル検証のために提供できる論理式はLOTOSのイベントの概念に密接したものである。また、これらの基本的なEBBTTL論理式に加え、時間的性質の分かりやすい表現として有用表現を併せて導入した。これは、ユーザにとって検証のための有用なインタフェースとなると期待できる。

今後の課題としては、本論中で述べたSALなどの既存の汎用LOTOSツールとの組合せの下、モデル検査を有効なユーザインタフェースとともに提供するソフトウェア支援環境の構築がある。

【参考文献】

[1] ISO : "LOTOS — A Formal Description Technique Based on the Temporal Ordering of Observational Behaviour", *ISO 8807* (Feb. 1989).

[2] 高橋, 神長, 白鳥 : "LOTOS言語の特質と処理系の現状と動向", 情報処理, Vol.31, No.1, pp.35-46 (Jan. 1990).

[3] R.Gotzhein : "Temporal Logic and Applications — A Tutorial", *Computer Networks and ISDN Systems*, Vol.24, pp.203-218 (1992).

[4] R.L.Schwartz and P.M.Melliar-Smith : "From State Machines to Temporal Logic: Specification Methods for Protocol Standards", *Protocol Specification, Testing and Verification* II, pp.3-19 (1982).

[5] A.Fantechi, S.Gnesi and C.Laneve : "An Expressive Temporal Logic for Basic LOTOS", *Formal Description Techniques* II, pp.261-276 (1990).

[6] A.R.Cavalli and F.Horn : "Proof of Specification Properties by using Finite State Machines and Temporal Logic", *Protocol Specification, Testing and Verification* VII, pp.221-233 (1987).

[7] B.-A.Mordechai, Z.Manna and A.Pnueli : "The Temporal Logic of Branching Time", *Proc. 8th Annual ACM Symposium on Principles of Programming Languages*, pp.164-176 (1981).

[8] J.M.Spivey : "The Z Notation — A Reference Manual", *Prentice-Hall* (1989).

[9] A.Sato, K.Kawaguchi, K.Takahashi, N.Shiratori and S.Noguchi : "SAL: Semantics Analyzer for LOTOS Specifications", 情報処理学会マルチメディア通信と分散処理研究会 46-6 (July 1990).

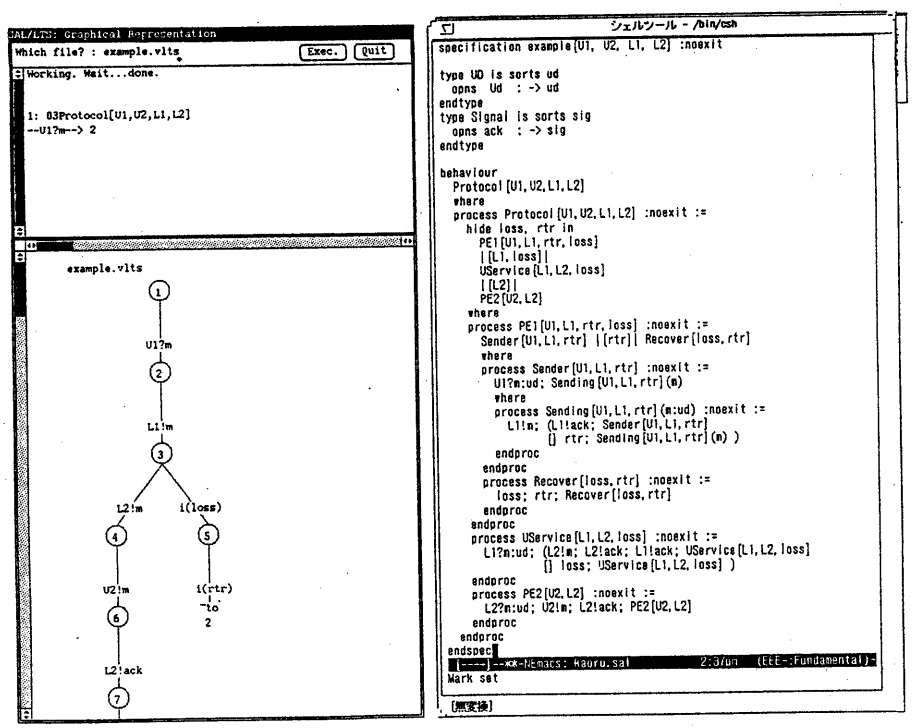

図4 LOTOSによるプロトコル仕様と遷移システム