複数時間オートマトンによる仕様記述と検証

6J－3

岡野 浩三　服部 哲　山本 亮　東野 輝夫　谷口 健一

大阪大学 大学院基礎工学研究科 情報数理系専攻

1　はじめに

本稿では，時間制約を持つ分散システムを記述するモデルとして，拡張されたタイムオートマトンの組による記述方法を提案し，この記述に対する検証方法を与える．

実時間システムを記述するためのモデルとしてタイムオートマトン，タイムペトリネットなど多くのモデルが提案されている [1, 3, 4]．複数のコンポーネントからなる実時間システムを記述する際，例えば，複数のタイムオートマトンの組を用いるのが自然である．Alur のタイムオートマトン [2] の積オートマトンでは受理言語を規定する形で動作定義を行なうため，各オートマトン間の同期動作の意味としては，LOTOS のマルチランデブと同様の非常に強い同期機構を意味する．このことは利点でもある一方で，システムを実装する際などに，大きな制約になることがある．そこで，本稿では，異なるコンポーネントの異なるイベント間の実行順序と時間制約を自然に記述できるモデルを提案する．

2　記述モデル

2.1　拡張タイムオートマトン

Alur のタイムオートマトンに対して，命題変数を参照する条件判定と命題変数の真偽値の変更を付加した拡張タイムオートマトンを記述モデルに用いる．Alur のタイムオートマトンで用いている時間変数の役割は拡張タイムオートマトンでも同様である．すなわち，遷移条件 (時間変数と定数の大小比較) としての参照および任意の遷移での任意の時間変数のリセットができる．時間は有理数値をとり，各時間変数に対して時間の経過は一様である．一方，新たに導入した命題変数に関しては，遷移条件としての参照および任意の遷移での真偽値の変更ができる．拡張タイムオートマトンにおける遷移条件は，したがって，従来の時間条件と命題論理式の論理積で表す．簡単のため，命題の真偽値の変更は，定数 (t,f) の代入のみに限定する．命題の更新は同一遷移のイベントと同時に行なわれ，かつ処理時間は 0 とする．

Specification and Verification using Extended Timed Automata
Kozo OKANO, Satoshi HATTORI, Akira YAMAMOTO, Teruo HIGASHINO and Kenichi TANIGUCHI
Division of Infnformatics and Mathematical Science, Osaka University, Machikaneyama 1-3, Toyonaka, Osaka 560-8531, JAPAN

2.2　拡張タイムオートマトン群

Alur のタイムオートマトンの積オートマトンの動作は簡単には次のように定義される: 積オートマトンの受理ファミリ (受理状態の集合の集合) は，各オートマトンの状態への射像がそのオートマトンの受理ファミリとなるような積オートマトンの集合である．直感的には，あるイベントによって遷移可能なオートマトンはそのイベントによって同時に遷移しなければいけないが，遷移不可能なオートマトンは状態を変えない．

拡張タイムオートマトンにおいても同様の動作を行なうように定義にする．ただし，時間変数の共有は (Alur のタイムオートマトンと同様に) 認められない一方で，命題変数に関しては，大域変数的に各オートマトンで共通に参照および変更できるものとする．対象システムの外部から発生したイベントを外部イベントと呼ぶ．一方，システムが自発的に発生させるイベントを内部イベントと呼ぶ．外部イベントと内部イベントを合わせて単にイベントと呼ぶ．拡張タイムオートマトン群を記述に用いる際，ここでは各拡張タイムオートマトン間で内部イベント記号の共有はないものとし，また命題更新に関する矛盾の生じないクラスのみを考える．

2.3　記述能力

提案するモデルでは，命題変数の論理式を陽に遷移条件に記述できることから，明らかに，Alur の積タイムオートマトンより少ない状態数で記述できるものが存在する．また，命題変数を介して，複数のオートマトン間で通信することも自然に記述できる．

3　記述例

図 1 は，単方向通信を行なう 2 つのクライアントと，その間の処理を受け持つサーバの制御状態を拡張タイムオートマトンの組で表したものである．ここでは，外部イベントをすべて大文字で表している．一方，内部イベントは最初の文字のみ大文字で表している．命題記号はすべて小文字で表す．ここでは，動作上意味のないイベントや条件式，更新式は略している．

システムは，通信サーバとデータ送信部，データ受信部からなり，それぞれ，60 秒，30 秒処理すべきデータが発生しなければ強制終了するようになっている．まず，データ送信部，データ受信部の 2 つはユーザの INVOKE_A,

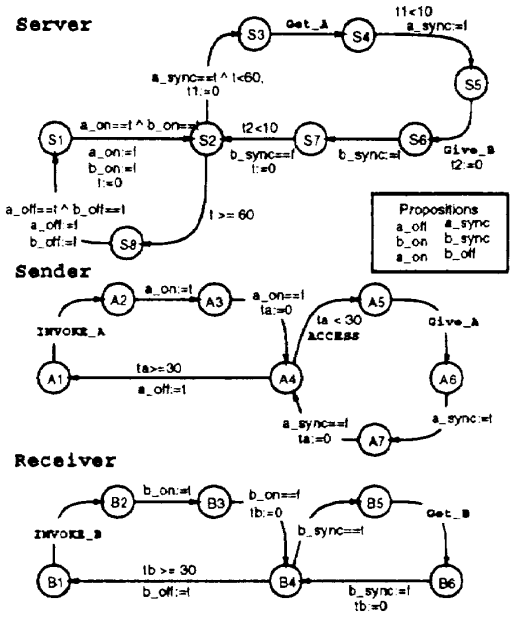

図 1: 単方向通信クライアントの状態遷移

INVOKE_B を受けて起動する．そして それぞれ，A4, B4 に遷移する．すると 通信サーバはこれらの遷移の結果を命題変数 a_on, b_on の値の変化として受取り，状態 S2 に遷移する．それぞれの状態から，データ送信部の，A4, ..., A7, A4 のループ，データ受信部の B4, B5, B6, B4 のループ，さらに，通信サーバの S2 ..., S7, S2 のループを通して，データ送信部からデータ受信部へ，データ送信が行なわれる．ACCESS の発生が，30 秒以上なければ，データ送信部，データ受信部ともに終了し，その結果，命題変数，a_off, b_off の値の更新を受けて，通信サーバが終了する．

このように，2 つのクライアントの非同期的な起動，終了を受けて，サーバが起動，終了するという動作を自然に記述することができる．その他，複数のオートマトンの命題更新を受けて，一つでも更新されたものがあれば，全体で次の動作に進むというような振る舞いも命題の論理和を用いることにより，自然に記述することができる．

4　安全性の検証

図 1 において，データ送信部の時間制約 30 秒を 70 秒に変更したとする．定常状態において 通信サーバ，データ送信部，データ受信部はそれぞれ S2, A4, B4 にいる．この場合，データ送信部でデータ処理のループに入ったにも係わらず，さきに データ受信部が B1 に遷移してしまう，という望ましくない状態になる．与えられたシステム に対して例えば，「データ送信部でデータ送信する状態では常に データ受信部はデータ受信可能な状態にいることが成立する」といった安全性の性質を検証できることがきわめて望まれる．

そこで，下記の方法により安全性が成り立つことの十分条件を機械的に判定する．ここでは積オートマトンを作製する際に時間領域の情報を一部失うことがありえるために，必要十分条件にはならない．一方，3., 4. を行なうことにより，判定の際明らかに不要な状態を取り除く工夫を行なっている．

1. 検証すべき性質をタイムオートマトン PA として記述する．
2. 各オートマトン A_i および PA の補集合を受理するオートマトンに対して，region automaton[2] r_i を作成する．
3. 作成された region automaton の検証性質に現れない各遷移のうち，命題変数を参照あるいは，変更していない遷移で時間変数をリセットしていない遷移を ϵ 遷移に置き換え，状態縮約を行なう．
4. これらの各 region automaton を拡張タイムオートマトンと見なして積マシン M を作成する．この際，時間領域で存在し得ない状態は初期状態からのトレースを行なうなどにより削除する．
5. M の受理言語空集合判定を行なって安全性判定を行なう．

5　あとがき

本稿では、時間制約を持つ分散システムを記述するモデルとして，拡張されたタイムオートマトン群を用いた記述方法を提案し，この記述に対する安全性の検証方法を与えた．この検証方法は十分条件にしたがっているため，必ずしも安全性の判定に成功するとは限らないが，安全性が判定されれば，そのことは保証できる．今後は，この判定法でどこまで安全性が保証できるか，例題を通して調べていきたい．また，判定の高速化も課題である．一方，逆に「望ましくない状況に陥ることはない」ことを効率よく調べる方法を考案することも今後の課題である．

参考文献

[1] Henziger, T.A.: "Symbolic Model Checking for Real-Time Systems," Information and Computation 111, pp.193-244, 1994.

[2] Alur, R. and Dill, L.D.: "A Theory of timed automata," Theoretical Computer Sciences, 126, pp. 183-235, 1994.

[3] Koskimies, K. and Mäkinen, E.: "Automatic Synthesis of State Machines from Trace Diagrams," Software-Practice and Experience, Vol.24(7), pp. 643-658 1994.

[4] Somé, S., Dssouli, R. and Vaucher, J.: "Toward an Automation of Requirements Engineering using Scenarios," Journal of Computing and Information, 2, 1, pp.1070-1092 1996.