モデル検査における検査対象と外部環境の自動合成手法

Model Synthesis Method Automatically Combining Target System with External Environment in Model Checking

鷲見　毅　　藤本　宏　　森　奈実子

■SUMI Takeshi　■FUJIMOTO Hiroshi　■MORI Namiko

近年，ソフトウェアの大規模化や複雑化により，レビューやテストだけではソフトウェアの品質確保が難しくなってきている。これを解決するための技術として，開発の上流工程で設計検証を行うモデル検査が注目されている。しかしソフトウェアを対象としたモデル検査では，検査対象に加えてシステムの外部環境まで含めた検査が必要になる場合がある。そのため，モデル検査の対象が大規模になり，検査に必要な計算機資源が不足して検査結果を得られなくなるという課題がある。

東芝は，大規模なシステムを対象にした場合でもモデル検査を実行可能にする，検査対象と外部環境の自動合成手法を開発した。この手法を用いることで，外部環境の不要なふるまいだけを除外でき，従来に比べ約10倍の規模の検査対象で検査が可能になり，その有効性を確認できた。

In software development, it has become difficult to assure system quality solely by design reviews and tests due to the increasing scale and complexity of software in recent years. As a solution to this issue, model checking is attracting attention as a means of realizing effective design verification in the upstream processes of software development. However, in the case of large-scale software model checking that requires verification of the external environment in addition to the target system itself, integration of the external environment into the system verification model often leads to the failure of verification due to a lack of the necessary computing resources.

In this context, Toshiba has developed a model synthesis method that automatically combines the models for a target system and its external environment in such a way that inspection processes can be successfully executed even in the case of model checking for large-scale systems. We have confirmed the effectiveness of this model synthesis method through the results of tests showing that it can be applied to systems approximately 10 times larger than those handled by conventional methods.

1　まえがき

近年，ソフトウェアは様々なシステムに搭載され，社会インフラシステムをはじめとする多様な事業領域で利用されている。それに伴い，ソフトウェアに起因する不適合の社会的影響が大きくなっており，機能性に加えて信頼性や安全性といった品質の向上がこれまで以上に求められている。その一方で，システムに搭載されるソフトウェアの複雑化や大規模化は一段と進み，デザインレビューやテストだけでは品質の確保が難しくなってきている。

こうした状況に対して，ソフトウェア品質を確保する効果的な手段の一つとして，モデル検査[1]が注目されている。モデル検査は，検査対象の動作仕様をモデル化した検査用モデルと，検査対象に期待される動作を表す制約条件とを入力とし，検査用モデルを網羅的に探索することで，システムの不適合を早期に発見する技術である[2]。

モデル検査を行うには，検査用モデルや制約条件を，専用言語や論理式を用いて記述しなければならない（**図1**）。モデル検査器SPIN[3]の場合，検査用モデルは専用言語Promelaで記述し，制約条件は線形時相論理[4]（LTL：Linear Temporal Logic）を用いた検査式で記述する。そのためモデル検査を行う場合には，これらに対する専門的な知識が必要になる。

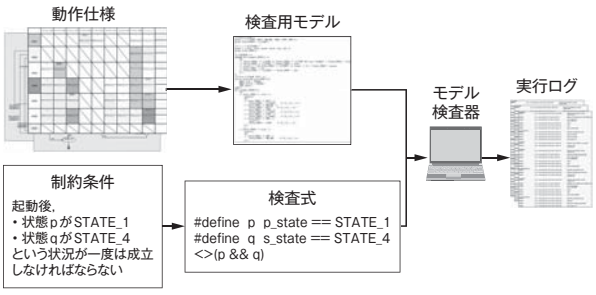

図1．モデル検査実行時の入出力 ― 動作仕様と制約条件から検査用モデルと検査式を作成してモデル検査を実行し，検査式を満たさない実行ログを出力する。

Input and output data for model checking

更に，モデル検査器は検査対象の動作仕様を網羅的に探索するため，検査対象の規模が大きくなるほど検査に必要な計算量が増大する。そのため，例えば検査に影響を与えない範囲で検査用モデルを抽象化するといった，検査対象を適切な規模に収める工夫が必要になる。特に，システムの外部環境を考慮しなければならない場合，外部環境を適切に抽象化することが求められる。従来，こうした作業は検査用モデル

を作成する個人のノウハウやスキルに依存していた。

そこで東芝は，大規模な検査対象であってもモデル検査を実施可能にする外部環境の自動合成手法を開発した。ここでは，モデル検査における外部環境とそれに起因する課題について述べるとともに，今回開発した外部環境の自動合成手法の概要とこの手法を組み込んだモデル検査自動化ツールを用いてモデル検査を行った評価結果について述べる。

2　モデル検査での外部環境の扱いと課題

2.1　外部環境

システムに搭載されるソフトウェアは，それ単独で動作することはなく，システム外部と連携してその責務を果たす。一般的な制御ソフトウェアを例にとると，センサが収集したデータに基づいて状況を判断し，アクチュエータに適切な制御信号を出力する。これらのセンサやアクチュエータを，制御ソフトウェアに対する外部環境と呼ぶ。例えば，アクチュエータに異常が発生した場合も発生した異常に応じた動作をしなければならない。このようにソフトウェアは，外部環境の変化に応じて正しく動作することが求められる。モデル検査とは，こうした動作について検査し，正しさを保証するために用いられる技術の一つである。

モデル検査では，検査用モデルで起こりうる全ての状態を網羅的に探索し，検査式に違反する状態の有無を検査する。したがって，センサやアクチュエータを外部環境として検査用モデルのふるまいに含めることで，外部環境の変化に応じた動作の正しさを検査することが可能になる。このように，検査対象のふるまいに加えて外部環境のふるまいも検査用モデルに記述しなければならない。外部環境のふるまいは，一般に検査対象と同様に状態遷移表を用いて記述する。

2.2　外部環境に起因した状態爆発

モデル検査器では，前述したように検査対象と外部環境を別々の状態遷移表で記述し，それらを合成して，その範囲で全ての状態を探索する。モデル検査器は，別々に記述された状態遷移表をそれぞれ独立に動作するプロセスと捉えるため，**図2**に示す状態遷移表の場合，検査対象の状態が同じS1であっても，外部環境の状態がEx=0であるS1&Ex=0と，外部環境の状態がEx=1であるS1&Ex=1とは別の状態として区別される。また，状態S1&Ex=0において，検査対象の状態を遷移させるイベント（E1，E2，E3，E4）と，外部環境の状態を遷移させるイベント（E5，E6）は，いずれも起こりうるとして状態の探索が行われる。その結果，図2の状態遷移表には，検査対象と無関係な外部環境の処理を繰り返すといったふるまいも含まれることになる。

例えば**図3**に示すように，外部環境だけが動作し検査対象はまったく動作しないようなふるまいや，検査対象には影響しないタイミングで外部環境が動作するようなふるまいがモデル検査時に実行される。これらのふるまいは検査しても結果が変わらないことから，検査対象の検査には不要であると言える。

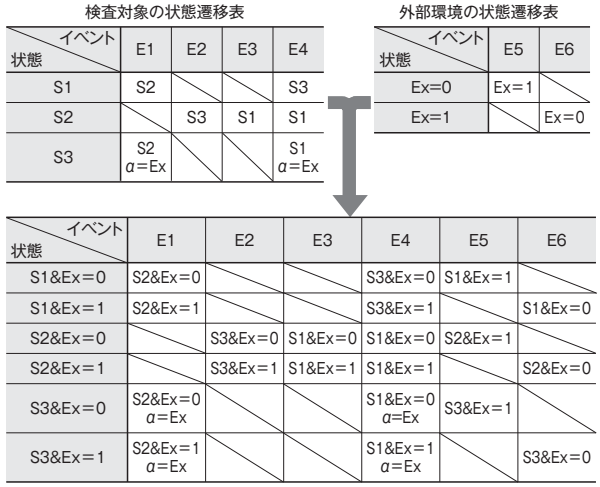

図2．状態遷移表の合成 — 複数の状態遷移表は独立かつ並行に動作するプロセスと捉え，全てのふるまいを検査する。

Synthesis of state transition tables

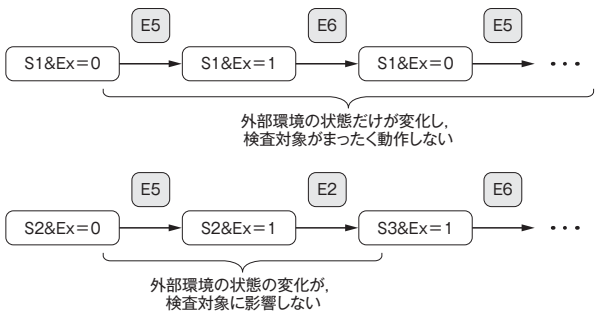

図3．モデル検査に不要なふるまい — 検査対象が動作せず，外部環境だけが動作するふるまいも含まれてしまう。

Negligible behavior in model checking

このように，モデル検査は検査モデルを網羅的に探索して検査するため，規模が大きな検査モデルを対象にすると，探索しなければならない状態が爆発的に増える。こうした状態爆発が発生すると，計算機の資源が不足して検査が中断してしまう。従来の検査用モデルの作成方法では，ここで述べたような不要なふるまいが含まれることから，モデル検査中に状態爆発が発生し，検査結果を得られなくなるという課題がある。

3　外部環境の自動合成手法

ここでは，外部環境に起因する状態爆発を回避するために，外部環境を検査対象に自動的に合成する手法について述べる。

一般論文

前述したように，外部環境のふるまいを検査対象と同様に状態遷移表で記述すると，検査用モデルが複雑化して状態爆発が発生してしまう。一方で，外部環境のふるまいを無視してしまうと，検査対象が外部環境の変化に応じて正しく動作することを検査できなくなってしまう。そこで，外部環境のふるまいのうち，図3に示した不要なふるまいだけを除外する手法を開発した。

図2に示した状態遷移表で検査対象の全てのふるまいを検査するためには，次の二つの条件を満せばよい。

(1) 検査対象の状態遷移表の全ての状態とイベントの組合せを検査している

(2) 検査対象が外部環境を参照する場合，外部環境の全ての状態ごとのふるまいを検査している

図2で検査対象は，状態S3において外部環境Exを参照し，参照した値を変数 α に代入している。このとき，Exの値が0と1の両方の場合について検査できていれば，検査対象の全てのふるまいを検査できたと言える。

そこで，外部環境のふるまいを状態遷移表として作成するのではなく，外部環境の状態を変数として宣言し，外部環境が取りうる状態を変数の値としてモデル化し，それを以下の手順で検査対象に合成することとした。

(1) 外部環境を，検査対象が参照する変数とその取りうる値として記述

(2) 検査対象のモデルである状態遷移表から，外部環境である変数を参照している箇所を抽出

(3) 抽出した箇所に，その変数が取りうる値それぞれの場合の処理分岐を作成

この手順に従ってモデル検査を実行すると，検査対象が外部環境の影響を受ける箇所では，変数の値ごとにふるまいが生成され，検査される。図2で表された検査対象と外部環境は，**図4**に示すような外部環境が0か1かのどちらかの値を取りうる変数Exとして表される。その結果，状態S3でイベントE1が発生した場合，**図5**に示した二通りのふるまいが起こりうる。モデル検査では，起こりうる全てのふるまいを網羅的に探索して検査するため，外部環境の全ての値について検査対象のふるまいが生成され，検査されることになる。

また，検査対象が外部環境を参照しない場合には，Exの値を扱わず，**図6**に示すように不要なふるまいが生成されることはない。

開発した手法では，外部環境を状態遷移表ではなく変数としてモデル化することで，検査対象が外部環境を参照する場合だけ，外部環境が取りうる値を網羅させることを可能にした。これにより，不要なふるまいを除外しつつ，検査対象が外部環境に応じて正しく動作することを検査できる。

更に，この手法の手順(2)と手順(3)をモデル検査自動化ツールに組み込むことで，**図7**に示すように外部環境の合成を自動的に行えるようにした。当社ではこれまでに，状態遷移表から検査用モデルを自動生成するモデル検査自動化ツールを開発しており[1), 2)]，これに外部環境の合成手法を，状態遷移表から検査用モデルを自動生成する過程に組み込むことで実現した。

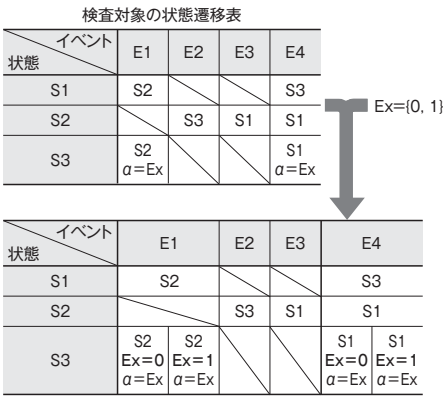

図4. 外部環境の合成手法 — 外部環境を変数としてモデル化することで，不要なふるまいを除外する。

Synthesis method for external environment

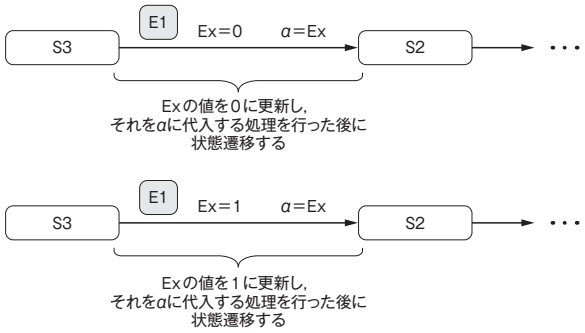

図5. 外部環境の値ごとのふるまい — 外部環境が取りうる値ごとにふるまいを検査する。

Behavior for each value of external environment

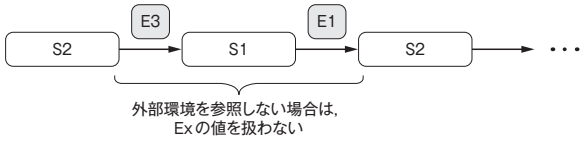

図6. 外部環境を参照しない場合のふるまい — 外部環境を参照しない場合，検査対象のふるまいだけを検査する。

Behavior in case of unreferenced external environment

4 評価

開発した外部環境の合成手法を組み込んだモデル検査自

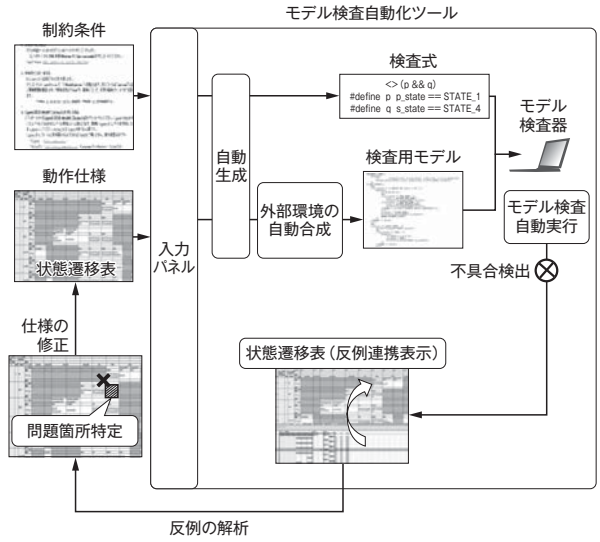

図7. 外部環境の自動合成 — これまでに開発したモデル検査自動化ツールの検査用モデルの自動生成過程に，外部環境の合成手法を追加して自動化する。

Automatic synthesis of external environment

動化ツールを用いてモデル検査を行い，開発した手法の評価を行った。その結果，この手法を用いることで，従来は状態爆発によって結果が得られなかった検査対象についても検査結果を得ることができた。

また，この手法を適用することでどの程度まで規模が縮小できるかを評価するために，**表1**に示す規模の状態遷移表に対して，従来の方法と外部環境の自動合成手法を用いた場合とでそれぞれモデル検査を実行した。その結果，**表2**に示すように，検査のために探索した遷移数が約1/10に，検査時に必要とした計算機資源（メモリ使用量）が約1/2になった。

これらの評価結果から，外部環境を含めた検査において開発した自動合成手法が状態爆発の抑制に効果があることを確認できた。

表1. 評価対象の状態遷移表の規模

Scale of state transition tables

規模 / 対象	状態数（個）	イベント数（個）	遷移数（個）
検査対象	10	14	467
外部環境	2	3	3

表2. 計算機資源の削減効果

Reduction of memory usage

比較項目 / 手法	探索の遷移数（個）	メモリ使用量（Mバイト）
従来の方法	86,684,846	1,163.475
開発手法	7,656,871	501.659

5 あとがき

大規模化し複雑化するソフトウェアの品質確保において，検査対象のふるまいを網羅的に検査するモデル検査の有効性が注目されている。その一方で，大規模なソフトウェアを対象としたモデル検査では，状態爆発によって検査結果が得られないという課題がある。これに対して当社は，外部環境の状態を変数の値としてモデル化し，それを検査対象に合成する手法を開発した。この手法を用いることで，状態爆発の発生を抑制し，より大規模な検査対象に対してもモデル検査を適用することを可能にした。

今後もモデル検査の適用範囲を広げるため，特に検査結果の解析手法について検討を進めていく。

文 献

(1) 池田信之 他. モデル検査によるソフトウェア上流設計の品質向上技術. 東芝レビュー. 64, 4, 2009, p.40－43.
(2) 高田沙都子 他. モデル検査技術による上流工程での効率的な設計検証. 東芝レビュー. 67, 11, 2012, p.45－49.
(3) Holtzmann, G. J. The SPIN Model Checker: Primer and Reference Manual. USA, Addison-Wesley Professional, 2003, 608p.
(4) 中島 震. SPINモデル検査－検証モデリング技法. 東京, 近代科学社, 2008, 238p.

鷲見 毅 SUMI Takeshi
ソフトウェア技術センター ソフトウェア設計技術開発担当主務。ソフトウェア上流設計・検証技術の開発に従事。情報処理学会会員。
Corporate Software Engineering Center

藤本 宏 FUJIMOTO Hiroshi
ソフトウェア技術センター ソフトウェア設計技術開発担当主務。ソフトウェア上流設計・検証技術の開発に従事。
Corporate Software Engineering Center

森 奈実子 MORI Namiko
ソフトウェア技術センター ソフトウェア設計技術開発担当。ソフトウェア上流設計・検証技術の開発に従事。情報処理学会会員。
Corporate Software Engineering Center