4A-04

SysML ステートマシン図から簡素な SPIN モデルへの変換手法

安藤 崇央 * 宮本 裕也 * 谷津 弘一 * 久住 憲嗣 † 福田 晃 * 道浦 康貴 ‡
酒見 慶太 ‡ 松本 充広 ‡
* 九州大学 大学院システム情報科学研究院 / 大学院システム情報科学府
‡ 九州大学 システム LSI 研究センター § 有人宇宙システム株式会社

1 はじめに

モデル検査 [1] などを用いたソフトウェアの形式検証についてはそのコストが問題となることが多く、検証過程の初期段階においては検証コストを抑えるため簡素なモデルの利用が期待される。また、検証用のモデルの作成に関しても、SysML[2] ステートマシン図などの設計情報から検証用モデルを自動生成する技術の研究がなされている。本稿では、検証過程の初期段階で利用可能な簡素な検証用モデルをステートマシン図から自動生成する手法について議論する。特に、検証用モデルとしてモデル検査器 SPIN[3] で利用されるモデルについて扱い、ステートマシン図から SPIN 用の簡素な検査モデルへの変換規則について紹介し、その有効性について議論する。

2 SPIN モデルへの変換

本節では、ステートマシン図を SPIN で用いられる Promela 言語で記述された簡素な検査モデルに変換する変換規則について述べる。変換により生成される検査モデルは、ステートマシン図の各要素との対応がとりやすいよう、要素毎にひとかたまりのコードとなるよう工夫している。

また、本稿で提案する簡素な検査モデルは、並行動作するプロセスを持つステートマシン図を、ひとつのプロセスにまとめた検査モデルとなる。この検査モデルでは、ステートマシン図の表現する並行プロセスのすべての振る舞いを表現するものとはなっていないが、状態爆発を防ぎ、検証コストを抑えることができるため、検証プロセスの初期段階での利用に適する。

変換規則の概要 提案する変換規則では、XMI 形式で表現されたステートマシン図と、ステートマシン図上に現れる変数について、その型や初期値などの情報を記述した変数情報を入力とする。各変換規則は、対応するステートマシン図上の要素や変数情報などを検査モデルを構成する Promela コードに変換する規則となる。以下に、その変換規則の概要を示す。

1. ステートマシン図の各状態名に対応する変数値を mtype の値として定義
2. 現在の状態を表現するための変数を、ステートマシン図上の領域の数だけ用意
3. イベント名に対応する変数値を mtype の値として定義
4. 生起したイベントを表現する変数を用意
5. ステートマシン図上の操作やガード条件中に出現する変数について、入力の変数情報を基に、対応する Promela コード用の変数を用意
6. 初期状態を inline 文を用いたマクロ定義に変換
7. 並行合成状態を含む状態を、3 つの inline 文マクロに変換
8. 遷移を inline 文マクロに変換
9. 領域を 2 つの inline 文マクロに変換
10. 終了状態を inline 文マクロに変換
11. イベントの生成を表す inline 文マクロを生成
12. ステートマシンの振る舞いを表すプロセスを生成

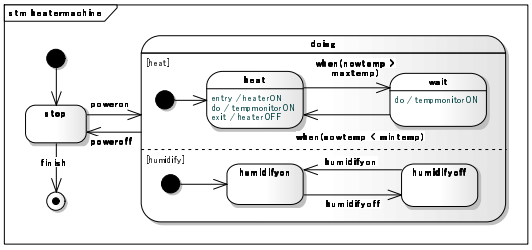

図 1: 空調システムのステートマシン図

以下では、上記の変換規則のうち主要な規則について変換例を示しながら詳細を述べる。変換例はいずれも、図 1 に示すステートマシン図の各要素を変換したものである。

状態の変換 ステートマシン図上の各状態は、entry 部、do 部、exit 部の 3 つのマクロ定義に分割されて変換される。それぞれ、状態への入状時振る舞い、在状時振る舞い、退状時振る舞いの 3 つに対応する。また、内部に領域を持つ合成状態の場合、entry 部に内部領域の初期状態の inline 文マクロ呼び出し、do 部に内部領域の inline マクロ呼び出し、exit 部に内部領域の退状時振る舞いの inline マクロ呼び出しが追加される。

```
inline S_doing_entry() {
  topState = top_doing;
  T_doing_heat_init(); T_doing_humidification_init()
}

inline S_doing() {
  R_doing_heat(); R_doing_humidification()
}

inline S_doing_exit() {
  R_doing_heat_exit(); R_doing_humidification_exit();

  doing_heatState = doing_heat_init;
  doing_humidificationState = doing_humidification_init
}
```

Translation Method into Simple SPIN Model for SysML State Machine Diagram
* Takahiro Ando, Yuya Miyamoto, Hirokazu Yatsu and Akira Fukuda,Graduate School and Faculty of Information Science and Electrical Engineering, Kyushu University
†Kenji Hisazumi, System LSI Research Center, Kyushu University
‡Yasutaka Michiura, Keita Sakemi and Michihiro Matsumoto, Japan Manned Space Systems Corporation

遷移の変換　本稿で提案する変換規則では、ひとつの状態からの複数の出力遷移をひとつの inline 文マクロに変換する。各遷移は、if 文を用いて遷移のトリガとなるイベントによる条件分岐により分けて記述される。また、状態の内部遷移もその状態の出力遷移と同じ inline 文マクロにまとめられる。ステートマシン図の遷移の表す意味論から、遷移元の状態の exit 部、遷移先の状態の entry 部の inline 文マクロの呼び出しもまとめられる。

```
inline T_stop() {
  if
  :: (event == poweron) -> event = NULL;
     S_stop_exit(); S_doing_entry()

  :: (event == finish) -> event = NULL;
     S_stop_exit(); S_final()

  :: else -> skip
  fi
}
```

領域の変換　ステートマシン図の領域毎に、その構成要素となる状態と遷移の inline 文マクロをまとめ、その領域での状態遷移を表現する inline 文マクロに変換する。また、その領域外への遷移時における操作も別の inline 文マクロに変換される。

```
inline R_doing_heat() {
  if
  :: (doing_heatState == doing_heat) ->
     S_doing_heat(); T_doing_heat()

  :: (doing_heatState == doing_wait) ->
     S_doing_wait(); T_doing_wait()
  fi
}

inline R_doing_heat_exit() {
  if
  :: (doing_heatState == doing_heat) ->
     S_doing_heat_exit()

  :: (doing_heatState == doing_wait) ->
     S_doing_wait_exit()
  fi
}
```

イベント生起モデルの変換　本稿で提案する検査用モデルでは、イベント生起のモデルとして、“内部状態を含めた現在の状態のいずれかの遷移のトリガとなるイベントのみが生起する” というモデルを採用する。

```
inline eventOccur() {
  event == NULL;
  if
  :: (topState == top_stop) -> event = poweron
  :: (topState == top_stop) -> event = finish
     ...
  fi
}
```

ステートマシンの振る舞いを表すプロセス　本稿で提案する検査モデルでは、ステートマシン図の表す振る舞いを表現するプロセスをひとつだけ用意する。このプロセスは、ステートマシン図の初期状態から実行が始まり、イベント生起とそれに伴う状態遷移を交互に繰り返し実行する。

```
active proctype stm() {
  T_init();
  do
  :: eventOccur();
     R_top()
  od
}
```

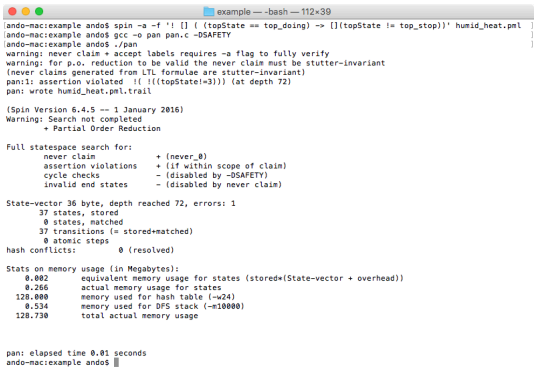

図 2: SPIN でのモデル検査実行例

3　適用

本節では、提案した変換規則によりステートマシン図から生成した検査用モデルが、SPIN でのモデル検査に利用可能であることを確認する。

適用例には、図 1 のステートマシン図を利用し、変換規則を用いて生成した検査モデルを SPIN の入力として LTL モデル検査を行う。検査式として、“doing 状態から stop 状態への遷移が起きない” ことを表す次の線形時相論理式を用いて、この式を満足するか検査する。

```
[]( (topState == top_doing) -> [](topState != top_stop) )
```

この適用実験でのモデル検査の実行は、図 2 の通りであり、その検査結果は、stop 状態への遷移が見つかったことを示している。これは、本稿の変換規則により生成した検査モデルがモデル検査に実際に利用可能であること、その検査結果が直観に合うものであることを示しており、ステートマシン図のモデル検査に本稿の変換規則が有用であることを示している。

4　まとめ

本稿では、並行動作するプロセスを持つステートマシン図に対して、SPIN モデル検査用の単一のプロセスを持つ簡易モデルへの変換手法について述べた。本稿で提案した簡易検査モデルは、inline 文マクロを利用することで変換元のステートマシン図の各要素に対応する要素と、Promela 言語上でのコードとの対応がとりやすくなっている。また、実際に変換規則を用いて生成した検査用モデルを用いたモデル検査が可能であることを示し、その有効性を確認した。今後の課題として、本稿で提案した変換規則を利用した自動変換器の実装を行い、より多くの例題に対して適用することで、変換規則の洗練を行うことが挙げられる。

参考文献

[1] Jr. Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. *Model Checking*. The MIT Press, 1999.

[2] OMG. OMG Systems Modeling Language Version 1.3. `http://www.omg.org/spec/SysML/1.3/PDF`, June 2012.

[3] G. Holzmann. The Model Checker SPIN. *IEEE Trans*, Vol. 23, No. 5, pp. 279–295, 1997.