16. モデル検査ツール(SPIN)の使い方等のヒント

モデル検査の目的に応じて、3 つのステップ(1)モデル検査 C コード生成、(2)コンパイル、(3)モデル検査実行、において用いられるツール・オプションの依存関係を整理し、オプションの組み合わせに関する利用パターンを整理する。

対象読者	開発技術者
目的	モデル検査ツール SPIN の使い方に関して、分かりにくいオプションの組合せについて基本的な使い方を整理する。
想定知識	SPIN の基礎知識
得られる知見等	● モデル検査の基本的な実施順序 ● SPIN によるツール利用の3ステップ(1)モデル検査系 C コード生成、(2)コンパイル、(3)モデル検査実行、において用いられるツール・オプションの依存関係と組み合わせの制約 ● Windows 版 SPIN のインストール注意点

16.1. SPIN モデル検査の基本的な実施手順例

SPIN によるモデル検査の実施手順は、目的やスキルによって様々な流れが考えられるが、一般的な場合で効率的にモデル検査を行うためには、以下のような実施手順を参考にするとよい。

(1) モデル自体の基本的な検査(到達性解析)

モデル(Promela コード)が、備えるべき基本的な条件で、ツールにより自動チェック可能な到達性解析を最初に実行するとよい。モデルが LTL 式により定義される検証性質を満たすか検査したり、シミュレーション実行でモデルの振舞いを確認する前に、自動チェック可能な到達性解析を先に行うとよい。

到達性解析により、対象システムのモデル(Promela コード)のデッドロック、ライブロックなどの好ましくない記述の検出を行うことができる。また、Promela コードの特定の位置で、特定の条件を満たすかどうか確認するための assert 文の検査を行う。

(2) 進行性解析(ループ解析)

Promelaコードのある並行プロセスの特定のコード上の位置(ステートメント)を無限に繰返し訪れる場合、他の並行プロセスの進行性を保証できなくなる。これらは進行性解析により検出することができる。また、不具合とは見なさない想定されるループなどが存在する場合、それらはエラーと見なされないように、ループを含むステートメント上に progressラベルを指定する方法がある。これらの使い方に関しては注意が必要で、文献[176]の 3.1.3 節に具体的に書かれているため参考にするとよい。

(3) LTL 検証性質のモデル検査

システムが提供すべき機能(ライブネス)とシステムが守るべき条件(安全性)を LTL 式で記述して、モデル検査を行う。

上記の検証の補助的な手段として、反例(不具合)が発見された場合のガイド付きシミュレーションやランダムシミュレーションの実行により挙動を確認する。

16.2. SPIN モデル検査のオプション組合せヒント

SPIN モデル検査では、3 ステップ (1)モデル検査系 C コード生成、(2)コンパイル、(3)モデル検

[176] 中島震: SPIN モデル検査 −検証モデリング技法, 近代科学社, 2008 .

査実行、により実行される。前節で示した検査の種類ごとに、これらの3ステップで用いられるオプションには依存関係があり、前のステップで指定したオプションはその後のステップで指定するオプションに制約を与える。それらの制約やオプションの組合せについて以下にまとめる。

16.2.1. モデル自体の基本検査（到達性解析）の場合

モデル検査実行3ステップにおけるオプションの組合せは以下の通りである。

(1)モデル検査系 C コード生成

```
% spin –a ＜モデルファイル＞
```

(2)コンパイル

```
% gcc pan.c
```

(3)モデル検査実行

```
% ./pan
```

16.2.2. 進行性解析（ループ解析）の場合

Progressラベルを設定した後、モデル検査実行3ステップにおけるオプションの組合せは以下の通りである。

(1)モデル検査系 C コード生成

```
% spin –a ＜モデルファイル＞
```

(2)コンパイル

```
% gcc –DNP pan.c
```

(3)モデル検査実行

```
% ./a.out –l
```

16.2.3. LTL 検証性質に関するモデル検査の場合

安全性の検査、ライブネスの検査によってオプションの組合せがことなることに注意が必要であるモデル検査実行3ステップにおけるオプションの組合せは以下の通りである。

16.2.3.1. 安全性検査の場合

(1)モデル検査系 C コード生成

```
% spin –a –f “LTL 式” ＜モデルファイル＞
```

(2)コンパイル

```
% gcc –DSAFETY pan.c
```

(3)モデル検査実行

```
% ./pan
```

16.2.3.2. ライブネス検査の場合

spin コマンドの –DNP オプションなしで実行可能である。

(1)モデル検査系 C コード生成

```
% spin –a –f “LTL 式” ＜モデルファイル＞
```

(2)コンパイル

```
% gcc pan.c
```

(3)モデル検査実行

```
% pan –a –f
```

Fairness を仮定したモデル検査の場合、以下のようになる。

(1)モデル検査系 C コード生成

```
% spin –a –f “LTL 式” ＜モデルファイル＞
```

(2)コンパイル

```
% gcc –DNFAIR=3 pan.c
```

(3)モデル検査実行

```
% ./pan –a -f
```

LTL 式をファイルで入力する場合以下のようにする。

(1)モデル検査系 C コード生成

```
% spin –a –F ＜LTL 式ファイル＞ ＜モデルファイル＞
```

(2)コンパイル

```
% gcc pan.c
```

(3)モデル検査実行

```
% pan –a –f
```

16.2.3.3. Never claim生成と利用

LTL 式をコマンドラインオプションとして与え、Never Claim を生成するためには以下のようなオプションを使う。

(1) Never Claim の生成

```
% spin –a –f “＜LTL 式＞” > ＜never claim ファイル＞
```

(2) Never Claim をコマンドラインとしてモデル検査系 C コード生成

```
% spin –a –N ＜never claim ファイル＞ ＜モデルファイル＞
```

以下は、16.2.3 節と同様である。

16.2.4. SPIN シミュレーション実行と反例解析

Promela コードを入力として、ランダムシミュレーションを実行する場合以下のようにする。

```
% spin ＜モデルファイル＞
```

また、反例が見つかった場合に、その反例に至るガイド付きシミュレーションを実行するのは以下の通りである。

```
% spin –t –g ＜モデルファイル＞
```

-g は、大域変数の表示

また、反例に至る最短パスを探索するためには以下のように実行する。

```
% pan –i
```

16.2.5. モデル検査の計算効率化のオプション

内部で生成するハッシュ表サイズ（デフォルト w=18, 2^18）を設定する場合以下のようにする。

```
% pan -w20
```

ハッシュ表の圧縮を強化する場合、以下のように実行する。

```
% gcc -DCOLLAPSE pan.c
```

ただし、この場合、メモリ使用量は減らせるが、圧縮の計算時間が増える。

状態遷移システムの状態を表す状態ベクトルをハッシュ表の代わりに minimal automaton で保持するためには、以下のように実行する。

```
% gcc -DMA=10 pan.c
```

この場合、メモリ使用量は劇的に減らせるが、計算時間が増える。

物理メモリがあふれスラッシングが発生する場合、それを回避するための方法として、モデル検査に利用するメモリのサイズ上限を設定し、物理メモリを超えないようにする。

```
% gcc -DMEMLIM=n pan.c
```

Promela コードに対してスラッシングを実施する場合、以下のように実行する。

```
spin -A <モデルファイル>
```

16.3. Windows 版インストールの注意点

本節は、モデル検査ツール SPIN（Windows 版）のインストール手順を示すものである。必要に応じて参照すれば良い。

16.3.1. ツールのダウンロードおよびインストール

SPIN のインストーラーは、http://spinroot.com/spin/Bin/index.html の「Windows PC Spin executable」と記載されている箇所からダウンロードができる。2011 年 3 月時点での最新バージョンは 6.0.1 であり、spin601.exeをダウンロードできる。

名前を spin.exe に変更し、任意のフォルダに保存する。ここでは、「C:¥spin」フォルダを作成し、そこに保存する。

gccを利用可能にするため、CygwinまたはMinGW等をインストールする必要がある。ここではMinGWのインストール方法を述べる。MinGWは、http://sourceforge.net/projects/mingw/の「Download」と記載されている箇所からダウンロードできる。ダウンロードしたファイルを実行するとインストーラが立ち上がる。基本的には全てNextを選択して良い。インストール先は任意の場所で良いが、ここではデフォルトである「C:¥MinGW」とする。

また、必須ではないがGUIとしてispin、jSpin、xpinが用意されている。ここではispinの利用方法を示す。ispinは、http://spinroot.com/spin/Src/index.htmlからダウンロードすることができる。特別なインストール作業は必要ない。さらに、Windows環境でtclを実行可能にするため、ActiveTcl等をインストールする必要がある。ActiveTclは、http://www.activestate.com/activetclからダウンロードすることができる。ActiveTclはデフォルトの設定でインストールすれば良い。

16.3.2. 環境設定

マイコンピュータを右クリックして、プロパティを選択。詳細設定タブを選択し、環境変数ボタンをクリックする。次にユーザ環境変数内にある PATH を選択し、編集をクリックする。変数値の欄に「;C:¥spin;C:¥MinGW¥bin」を追記し、OK をクリックして全て閉じる。SPIN や MinGW の保存先を異なる場所にしている場合はそれに合わせる。

16.3.3. 実行

スタートメニューから「ファイル名を指定して実行」を選択し、名前に「cmd」と入力して OK をクリックするとコマンドプロンプトが立ち上がる。コマンドプロンプト上で、「spin」と入力したときに、

```
Spin Version 6.0.1 -- 16 December 2010
spin: error no filename specified
```

のような表示が出力されれば SPIN のインストールは成功である。また、コマンドプロンプト上で、「gcc」と入力したときに、

```
gcc: no input files
```

のような表示が出力されれば MinGW のインストールは成功である。

また、ispin.tcl をダブルクリックし、ispin のウインドウが立ち上がれば、ActiveTcl のインストールは成功である。