GSLetterNeo Vol.56
2013年3月

形式手法コトハジメ
- LTSA を使って -

産業第１事業部
熊澤 努
Tsutomu Kumazawa
kumazawa@sra.co.jp

ソフトウェアの品質向上はこれまで幾度も唱えられています。その一つの手段として、数学的な記述に基礎を置く形式手法と呼ばれる方法論が提案されてきました。しかしながら、数学に対する敷居の高さなどが障壁となって、開発現場への普及がなかなか進んでいないのが現状でしょう。

本稿では、気軽に使うことのできるツール LTSA (Labelled Transition System Analyser) を紹介します。LTSA は、ソフトウェアの動作を状態遷移機械(モデル)によって記述し、かつそのモデルの正しさを自動的に検証するツールです(モデル検査器といわれています)。その特徴は、簡潔なモデルの表記法を採用している点に加えて、作成した状態遷移機械が図式化されるので直感的にわかりやすい、ボタン一つで簡単な検証を自動的に実行することができる、などの点にあります。

◆インストール・基本操作

LTSA が強みを発揮するのは、複数の並行動作するプロセスから成るシステムの振る舞いの記述です。このような並行システムにおいてよく問題とされる、デッドロックを自動的に検出することができます。

2013 年 1 月時点で、LTSA はスタンドアローンツール ver. 3.0 と Eclipse 用プラグイン ver.2.0 が開発されており、開発者の Web サイト[3] にて無償公開されています。本稿ではスタンドアローン版を使って説明しています。Windows、Macintosh いずれの場合も、インストールは Web サイトから zip ファイルをダウンロードして解凍するだけです(JRE1.5.0 が必要です)。ツールは解凍したフォルダ内の lltsa.bat をダブルクリックして実行します。

LTSA では互いにやり取りをする複数のプロセスの動作を状態遷移機械で記述します。各プロセスの動作を合成することで、それらを協調動作させたときのシステム全体の振る舞いを表現したことになります。最後に、デッドロックの検出を行い、モデルに問題がないかを調べます。以上の手順は図 1 の起動画面にあるメニューから実行できます。

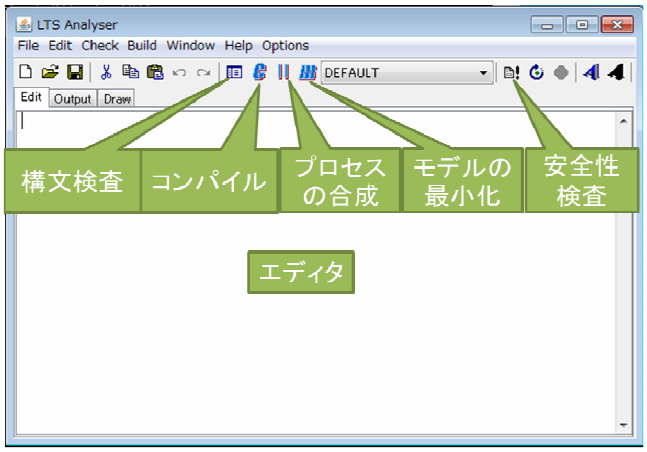

図 1 LTSA の起動画面

- **エディタ**: 対象システムのモデルを FSP という仕様記述言語で記述します。FSP については割愛します。文献[1,4]を参照して下さい。
- **構文検査**: モデルの構文検査を実施します。
- **コンパイル**: モデルをコンパイルし、各プロセスの状態遷移機械を生成します。
- **合成**: コンパイルした各プロセスを合成し、システム全体のモデルを作成します。
- **安全性検査**: モデルが安全性を満たすかどうか検証します。LTSA は全てのプロセスが実行不

可能になるデッドロックを検出します。

◆簡単な例

例として、2 つのプロセスが協調動作するモデルの安全性を検証してみます。2 つのプロセスの状態遷移機械を図 2 に示します。なお、これらの図は、プロセスのコンパイル、合成後に LTSA のプロセス表示画面から確認することができます。

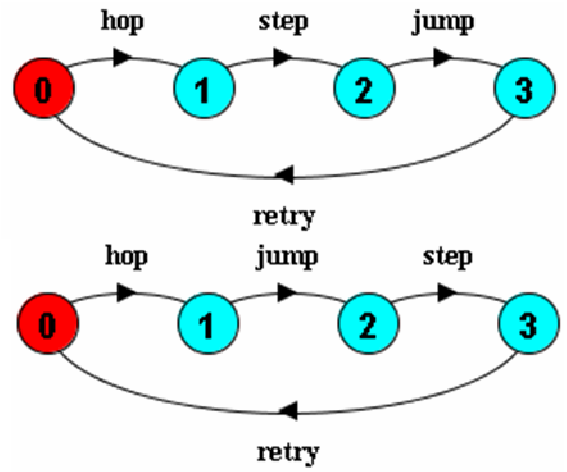

図 2 協調動作する 2 つのプロセス

2 つのプロセスはどちらも状態 0 が開始状態です。どちらも hop, step, jump, retry という一連の事象(アクションという)を繰り返します。

図 2 の 2 つのプロセスを協調動作させると、デッドロックは発生するでしょうか。LTSA の協調動作は並列合成という仕組みによって実現されています。簡単に言うと、並列合成とは、2 つのプロセスに記述された同じ名前のアクションは同時に実行し、そうでないアクションは独立に実行する機構です。図 2 の 2 つのプロセスは最初のアクション hop を同時に実行しますが、step と jump は互いの実行待ちとなります。つまり、デッドロックが起こるはずです。

モデルの合成後に、LTSA の安全性ボタンを押してみましょう。以下の表示がなされ、検出されたデッドロックに至るモデルの実行例が報告されます。

Trace to DEADLOCK:

hop

このように、安全性検査を利用することで、デッドロックの検出を自動的に実行することができます。

◆おわりに

LTSA には他にも自動検証機能が備わっています。ボタンを押すだけで実行できる進行性という性質の検証に加えて、記号論理式で書かれた性質の自動検証機能も実現されており、安全性に限らず広範な検証が可能です。LTSA についての詳細は、書籍[1] や日本語による解説[4]を参照して下さい。

なお、LTSA に関連する応用技術として、上流工程におけるシステムの振る舞いの分析手法[2]が開発されている点も付け加えたいと思います。

LTSA のようなツールを通じて、形式手法に親しんでいただければ幸いです。

◆参考文献

[1] J. Magee and J. Kramer, "*Concurrency: State Models and Java Programming*," 2nd ed., Wiley, 2006.

[2] S. Uchitel, J. Kramer, and J. Magee, "Incremental Elaboration of Scenario-Based Specications and Behaviour Models Using Implied Scenarios," *ACM TOSEM*, 13(1), pp. 37-85, 2004.

[3] http://www.doc.ic.ac.uk/ltsa/

[4]http://www.graco.c.u-tokyo.ac.jp/~tamai/concurrency.html

GSLetterNeo Vol.56
2013 年 3 月 20 日発行
発行者●株式会社 SRA 産業第 1 事業部
編集者●土屋正人、柳田雅子

バックナンバを公開しています●http://www.sra.co.jp/gsletter
ご感想・お問い合わせはこちらへお願いします●gsneo@sra.co.jp

株式会社SRA
〒171-8513 東京都豊島区南池袋２－３２－８