CESLにおける形式手法研究

CATS社 組込みソフトウェア研究所（CESL） 所長
松本　充広

■ CESLとは？

CESLは、キャッツ組込みソフトウェア研究所（Cats Embedded Software Laboratory）の略称です。福岡県福岡市に2007年4月に開設しました。組込みソフトウェア研究、車載組込みソフトウェアコンサルティングを行っています。組込みソフトウェア研究としては、形式手法研究、具体的には後述するGarakabuの研究、リソーススケジューリングの研究を行っています。車載組込みソフトウェアコンサルティングとしては、特に、実装前のモデリング（構造設計）のコンサルティングを行っています。

CESLは、ソフトウェア技術センター（愛知県刈谷市）に続く、CATS社２番目の地域拠点です。CESLも、九州の地域拠点として、セミナー開催、ツール販売を行っています。

この度、新たにCAL（CATS先端研究所）が開設されました。CALが10年後を見据えた先端技術を研究していくのに対し、CESLでは、５年後に実用化する技術を研究しています。また、CESLでは特に、車載組込みソフトウェアを適用領域とした研究を行っています。

■ 形式手法とは？

形式手法は、ソフトウェアの数学モデルを構築する技術と、その数学モデル上で指定した性質が成り立つかどうかを検証する技術の総称です。数学モデルの例として、状態遷移表があり、指定する性質の例として「不可セルに到達しないこと」「変数xの値が常に10以下であること」があります。

■ Garakabuの研究

Garakabuは、文部科学省の知的クラスター創成事業（第Ⅰ期）の中で、筆者が考案した状態遷移表モデル検査ツールです。状態遷移表モデル検査は後述しますが、形式手法の一種です。知的クラスター創成事業（第Ⅰ期）では、2005年6月～2007年3月に、当時筆者が在籍していた、(財)福岡県産業・科学技術振興財団、九州大学、CATS社で、Garakabuを共同研究しました。更に、知的クラスター創成事業（第Ⅱ期）において、Garakabuを車載組込みソフトウェア向けに改良する研究提案が採択されましたので（[1]）、この共同研究を、2007年9月～2012年3月に、(財)福岡県産業・科学技術振興財団、九州大学、CATS社で実施する予定です。

状態遷移表モデル検査では、ソフトウェアの数学モデルとして状態遷移表を使用し「不可セルに到達しないこと」「変数xの値が常に10以下であること」などを検証します。ここで、ソフトウェアが複数の状態遷移表（それぞれがタスクに対応）で記述されているとし、各状態遷移表の状態、状態遷移表に現れる変数の値の組も（ソフトウェアの）状態と呼ぶことにします。状態遷移表モデル検査では、まず、状態遷移表に基づき、初期状態から状態遷移を繰り返すことによりどの状態に到達するかを表す状態空間を構成し、次に、この状態空間上の各状態を、指定した性質が成り立つかどうかを、網羅的に調べます。もし、指定した性質が成り立たなかった場合には、初期状態から、指定した性質が成り立たなかった状態に至る状態までのシーケンス（これを反例と呼びます）を構成します。

Garakabuは、状態遷移表モデル検査を実行するツールで、指定した性質を満たさない状態、つまり、バグを網羅的に見つけます。「ガラカブ」は、筆者の先祖の出身地である熊本県の方言で、カサゴを意味します。カサゴは、バグ（虫）に即座に食いつく魚なので、Garakabuには、バグにすぐ食いついて欲しいという願いを込めて、この名前を付けました。

Garakabuで「不可セルに到達しないこと」を指定して状態遷移表モデル検査を行ったときの実行画面が図１です。この例では、Garakabuが、初期状態から不可セルに到達した状態に至る反

例を構成し、下ペインにその反例を表示します。下ペインの各行は、反例を構成する各状態に対応し、最下行が初期状態、最上行が不可セルに到達した状態に対応します。下ペインでカーソルキーを上下させることで、反例の中の状態を指定し、対応する行を反転表示できます。右上ペインの状態遷移表表示は、この反転表示と対応しており、指定した状態に対応する状態遷移表のセルが点灯します。図1では、不可セルに到達した状態に対応する最上行が反転表示しているので、右上ペインの状態遷移表では、不可セルが点灯しています。下ペインでカーソルキーを上下させると、右上ペインの状態遷移表上で点灯するセルが移動するので、初期状態から不可セルに到達した状態に至る過程の理解を助けます。

図1　Garakabu実行画面

Garakabuの研究では、特許を2件出願しました（[2]，[3]）。[2]は、不可セルに到達しないことの検証に関する特許です。[3]は、検証の効率化に関する特許です。（状態遷移表）モデル検査では、状態空間上の各状態を、指定した性質が成り立つかどうかを、網羅的に調べるので、複数の大きな状態遷移表では、使用するメモリ量、検証時間が膨大になります。このため、指定した性質が成り立つかどうかを検証するのに充分な部分状態空間を構成し、この部分状態空間上で検証を行うPartial Order Reductionという方法（[4]）が提案されています。モデル検査のソフトウェアへの適用の際に良く使われるSPIN（[5]）では、このPartial Order Reductionを使用しています。Partial Order Reductionは、プロトコル検証のように、複数のプロセスが独立して動作する場合には、部分状態空間が状態空間よりもかなり小さくなるのですが、組込みソフトウェアのように、複数のタスクがスケジューリング機構の下、非独立に動作する場合には、部分状態空間はあまり小さくなりません（図2）。[3]は、組込みソフトウェアのように、複数のタスクがスケジューリング機構の下、非独立に動作する場合に、部分状態空間が状態空間よりもかなり小さくなる方法の特許です（図2）。

	状態数	状態遷移数
状態空間	28224	56304
PORによる部分状態空間	28224	55489
[3]の方法による部分状態空間	8784	16393

図2　複数のタスクがスケジューリング機構の下、非独立に動作する場合の状態空間の例

なお、Garakabuの研究に関しては、雑誌、新聞でも紹介され（[6]，[7]）、記事を執筆し（[8]，[9]）、産業技術総合研究所などでの3件の招待講演を行っています。

■ リソーススケジューリングの研究

車載組込みソフトウェア、デジタル家電ソフトウェアなどの分散組込みソフトウェアでは、複数のタスクが、ネットワークやバスを通して、マルチキャスト通信をしながら処理を実行します。更に、これらのタスクは、リソースを共有します。タスクは共有リソースを定められた時間内に確保・解放する必要があり、また、リソース競合が起こってはいけません。例えば、マルチメディアプレーヤーがHDD、DVD、ビデオテープなどの複数のメディアに対し同時に再生や録画を行うデジタル家電ソフトウェアにおいて、複数のタスクが画像処理用LSI（リソース）を共有している場合、各タスクは時間制約を満たしながら、競合せずに画像処理用LSIを

共有しなければなりません。

リソーススケジューリングとは、分散したタスクが、時間制約を守りながら、競合せずに共有リソースを使用できるようにする、スケジューリングのことです。タスクは、マルチキャスト通信をしながら処理を実行しますが、ネットワークやバスにはメッセージ遅れがあるので、各タスクが時間制約を満たしながら、競合せずに共有リソースを使用していることを保証するのは、簡単な作業ではありません。

リソーススケジューリングの検証では、後述するリソーススケジュール図を数学モデルとし「分散したタスクが、時間制約を守りながら、競合せずに共有リソースを使用している」ことを検証します。

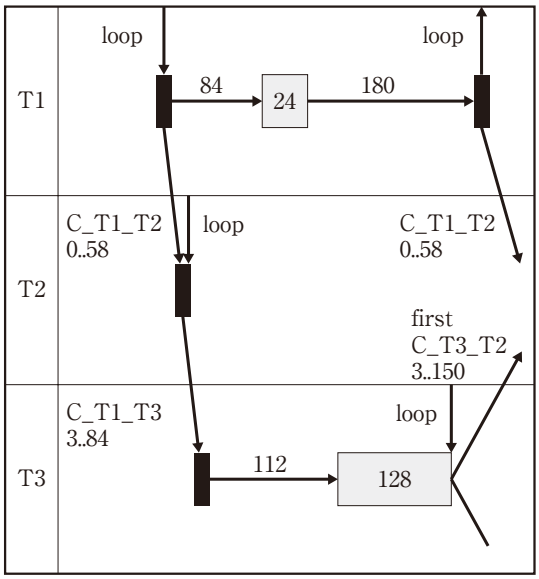

図3　リーススケジュール図

図3は、リーススケジュール図の一例です。リーススケジュール図には、タスクがいつリソースを使用し、いつマルチキャスト通信をするかを記述します。図3には、T1、T2、T3の3レーンがありますが、これらは、それぞれタスクT1、T2、T3の動作記述に対応します。リーススケジュール図の各レーン上の右矢印は、タスクがリソースの確保を待っていることを意味し、白箱は、タスクがリソースを確保していることを意味します。ここでタスクT1の時間制約を「最大300ms毎に、24msリソースを確保する」とします。T1レーンは「タスクT1は、84ms待った後、24msリソースを確保し、180ms待つ」の繰り返しという動作記述で、時間制約を満たす動作の一つです。リーススケジュール図のレーン間の矢印は、マルチキャスト通信を意味します。図3のC_T1_T2、C_T1_T3は、T1からT2、T3へのマルチキャスト通信を意味します。C_T1_T2下の0..58はメッセージ遅れの制約を記述しており、T1からT2へのメッセージは、0msから58msの間に到達することを意味します。

リーススケジューリングの検証では、UPPAAL（[10]）を利用します。リーススケジュール図をUPPAALのモデルに変換し、検証する性質に対応するUPPAALの検証式を与えて検証を行います。図3のうち、タスクT1に関する部分は、図4のUPPAALモデルに変換します。「タスクT1は、最大300ms毎にリソースを確保する」は、

A [] t1.clck <= 300

というUPPAALの検証式を与えて検証を行います。

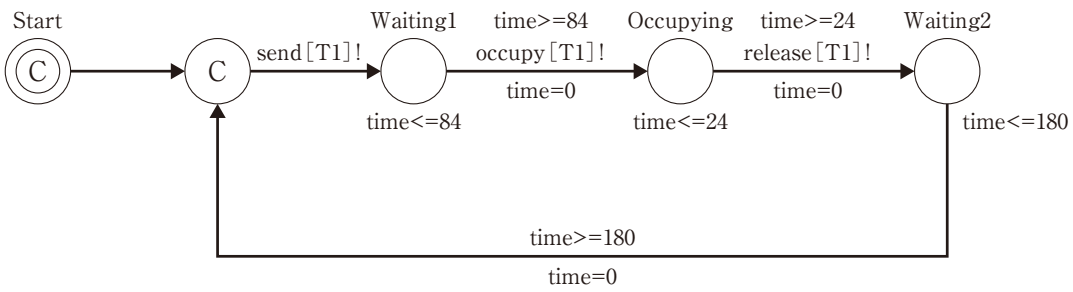

図4　タスクT1に対応するUPPAALモデル

なお、リソーススケジューリングの研究に関しては、論文誌に採録され（[11]）、更に、2007年10月11日～12日に米国ポートランドで開催される国際会議QSIC2007にて共著者の渡辺副社長が発表する予定です（[12]）。

■ 最後に

CESLでは、本稿に記述した形式手法研究だけでなく、車載組込みソフトウェアコンサルティング、九州でのセミナー開催、ツール販売を行っています。ZIPCセミナーは、6月、10月、2月に福岡で開催予定です。車載組込みソフトウェアコンサルティング、九州でのセミナー開催、ツール販売についてご興味のある方は、CESLまでご連絡下さい。

■ 参考文献

[1] http://www.mext.go.jp/b_menu/houdou/19/06/07062638.htm
[2] PCT/JP2006/306199「検証支援装置、検証支援方法、その検証支援方法をコンピューターに実行させることが可能なプログラム、及び、そのプログラムを記録した記録媒体」
[3] 特願2007-86529号「検証装置、検証方法、プログラム、及び、記録媒体」
[4] E.M.Clarke, O.Grumberg, and D.A.Peled, “Model Checking”, The MIT Press, 1999.
[5] http://spinroot.com/spin/whatispin.html
[6] 進藤智則「ソフトウェアは硬い」日経エレクトロニクス　2005/12/19号，p.87-121.
[7]「福岡知的クラスター研など、携帯・家電用ソフトの設計ミスを瞬時に探知」日経産業新聞　2006/12/26.
[8] 穴田啓樹、松本充広「組み込みソフトウェアの検証手法（3）モデル検査ツール「Garakabu」を開発」日経エレクトロニクス　2006/1/30号，p.138-142.
[9] 穴田啓樹、松本充広「組み込みソフトウェアの検証手法（4）モデル検査ツール「Garakabu」を使ってみる」日経エレクトロニクス　2006/2/13号，p.132-139.
[10] http://www.uppaal.com/
[11] 渡辺政彦、福田晃、松本充広、細谷伊知郎、城戸滋之「組込みシステムにおけるリソーススケジューリング設計・検査手法とツール」電子情報通信学会論文誌D,Vol.J90-D, No.3 ,pp.848-861.
[12] M.Watanabe, A.Fukuda, M.Matsumoto, H.Yatsu, I.Hosotani, and S.Kido, “A Resource Scheduling Design Method with Model Checking for Distributed Embedded Software”, to appear.

キャッツ株式会社
組込みソフトウェア研究所（CESL）
〒814-0001
福岡県福岡市早良区百道浜3-8-33
福岡システムLSI総合開発センター514
Tel:（092）832-7131
Fax:（092）832-7132