デッドラインに間に合う
システムを作成するための手法

NuSMVで タイミング解析＆ スケジューリング

藤倉 俊幸

リアルタイム・システムは規定時間内に処理を終了しないとその価値が大きく損なわれる．単純な1タスクのみのシステムならまだしも，現在のリアルタイムOSを使った複数のタスクを実行するシステムでは，本当に規定時間内に処理を終了できるかどうか保障することがとても難しい．

そこで各種モデル検査ツールを使ってリアルタイム・システムを検証する手法が考え出された．ここでは無償で公開されているモデル検査ツールNuSMVを使い，複数のタスクが動作する環境でデッドライン・オーバを起こすかどうかを検証する手法について解説する．（編集部）

1．リアルタイム・システムとタイミング解析

デッドラインなどの時間制約を持ったシステムをリアルタイム・システムという．つまり，計算結果が正しいだけでなく，その計算結果を決められた時間以内に求めなければならないようなシステムがリアルタイム・システムである．このようなシステムでは，タイミング解析やスケジュール可能性解析が重要である．しかし，開発過程の中にこれらの手法を取り入れているという話を日本では聞いたことがない．

タイミング解析を行わないとどんなことが起こるのか．

筆者がかかわっている某大学の組み込みシステムの講義で，ライン・トレーサの制御を行った．ライン・トレーサの動作を滑らかにし，かつ応答性の向上のためにタイマ割り込みの周期を短くしたところ，突然システムの動作が止まり，しばらくするとまた動き出すという奇っ怪な現象に遭遇した．

これはタイミング解析上の問題なので，いくらソース・コードを見ていても解決できない．タイマ割り込み間隔と割り込みハンドラの実行時間からCPU使用率を計算してどの程度余裕があるか計算する方法を，講義時間の関係で簡単に話しただけで，具体的な検証方法を説明しなかったので学生さんには申し訳なかった．

比較的単純なシステムであれば，CPU使用率を計算するだけで経験的に何となくいけそうだくらいは分かる．割り込みのような優先度の高い処理がCPU処理時間の40%を超えると上記のような現象が発生する．

タイミングの解析手法として，昔からあるCPU使用率ベースのものと比較的新しいモデル検査ベースのものの二つがある．前者は結果が悲観的すぎて使えないことが多く，前提条件などを計算に取り入れるのも難しい．例えば，特定の状況でしか動かないタスクとか，絶対に同時には動かないようなタスクの振る舞いを計算に入れることは難しい．これらの扱いに厳密性が欠けるため，大きな安全係数を掛けて最悪の場合を計算するためどうしても悲観的な結果になる．ここでいう悲観的な結果とは，「もっとCPUを早くしないとデッドラインを守れません」，という結果が得られることである．

このような結果しか得られないのでは，使ってみても嬉しくない．あまり手法自身が知られていないこともあり，開発過程の中にこれらの手法を取り入れている話を日本では聞いたことがない．

後者は，アプリケーションのモデリングと同様なアプローチで計算ができるため，タスク設計の一部として実施できる．計算手順もルーチン的である．その上，結果は条件を細かくモデル化できるので悲観的な結果になることもない．

ここでは，後者のモデル検査ツールNuSMV[(1)]を使用したタイミング解析方法[(9)]を紹介する．NuSMVについて紹介した後に，簡単な例を用いてCPU使用率ベースの方法と比較する．

NuSMVはユース・ケースの検証などにも利用できるため，アプリーション部分の検証の一部としてタイミング解析が可能になる．どういうことかというと，アプリケーション部分には，特定の状況でしか動かないタスクとか，絶対に同時には動かないようなタスクの振る舞いが含まれているので，これらを考慮した上でのタイミング解析が可能になる．ただし，NuSMVの本領を発揮するような例題は今

KEYWORD——NuSMV，形式手法，タイミング解析，スケジュール可能性解析