車載ソフトウェアの自動並列化における品質保証上の課題と対策に向けて

芹沢 一[1]　中村 宏貴[1]　納富 昭[2]　鷲崎 弘宜[3]

概要：メニーコア化する電子制御ユニットのリソースを高効率に使用するため複数コアの並列実行が必要になる．並列化の開発工数を抑えるため，ツールによる自動並列化技術の適用が必要である．一方で適用にあたり，品質保証が課題となる．そこで筆者らは，課題および解決方針の検討と整理に向けて，並行性・複数コアソフトウェアの欠陥分類と実態を調査したうえで，品質保証上の課題としての等価性違反およびそれに対応する対策および技術的制限をまとめた総合的な枠組み，およびそれに基づく検証プロセスを提案する．

Technical Challenges in Quality Assurance of Automated Parallelized Automotive Software

Kazuyoshi Serizawa[†1]　Hirotaka Nakamura[†1]　Akira Nodomi[†2]
Hironori Washizaki[†3]

1. はじめに

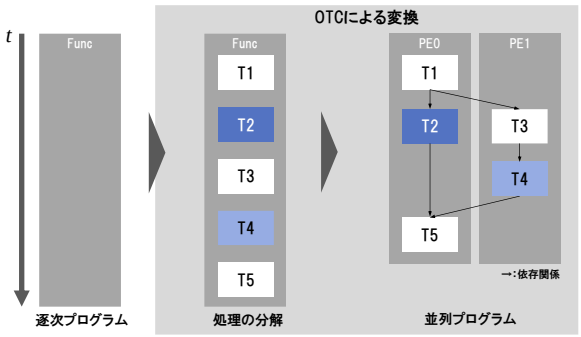
車載電子制御ユニットのプロセッサはメニーコア化しており，リソースを高効率に使用して競争力を確保するためには，複数コアの並列実行が必要になる．一方，並列化にはスキルフルな技術者の工数が必要であり，その開発工数を抑えるために，ツールによる自動並列化技術の適用が必要である，ところが，ツールは適用後の検証まではサポートしないため，品質保証が課題となる．そこで筆者らは，課題および解決方針の検討と整理に向けて並行化後の違反の分類と実態を調査したうえで，品質保証上の課題としての等価性違反およびそれに対応する対策および技術的制限をまとめた総合的な枠組みおよびそれに基づく検証プロセスを提案する．枠組みにおける具体的な対策技術としては，テストおよび静的解析・制約チェックの併用を提案する．

本稿では以降において，2 節で背景として車載ソフトウェアの自動並列化の状況を説明する．3 節では品質保証の枠組みおよびそれに基づく検証プロセスを提案する．4 節で本稿をまとめる．

2. 車載ソフトウェアの自動並列化

車載ソフトウェアにおける並列化の必要性と進展，および，それを進めるツールでの自動並列化の状況を説明する．

自動並列化を行うツールは複数あり，その一つ，OTC（OSCARTech(R) Compiler）は，入力とする逐次プログラムを複数ブロックに分割し，ブロック間での実行順序の依存関係を解析しながら，それを守る範囲で各ブロックを複数コアに割り当てることで並列化し，並列化後のプログラムを出力するツールである．



2.1 並行・複数コアソフトウェアの違反

Asadollah らは並行および複数コアソフトウェアについて観測可能な情報や特性（スレッドの状態，メモリアクセスなど）に基づき，次のように違反を分類している [1][2]．

類似の整理結果としては[3]において，特に性能に影響する並行プログラムの違反として次の分類をまとめている：デッドロック，ライブロック，飢餓状態，データ競合，原子性違反，表明違反，通信違反，同期・並行性違反．

3. 課題と対策の総合的枠組みの提案

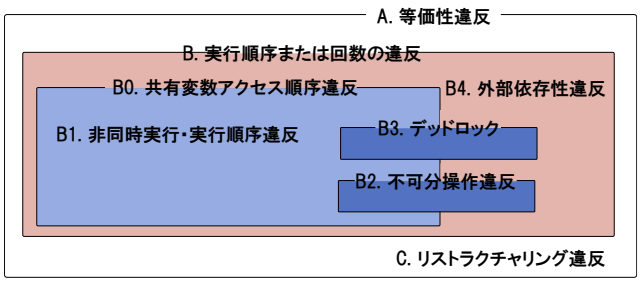
前述の背景に基づき，自動並列化における特に信頼性に

1 日立 Astemo 株式会社
Hitachi Astemo, Ltd.
2 オスカーテクノロジー株式会社
Oscar Technology Corporation
3 早稲田大学
Waseda University

かかわる品質問題として等価性違反を取り上げて，それに対する静的解析・制約チェックおよびテストを中心とした対策の枠組みを提案する．

3.1 等価性違反の定義と分類

並列化に伴う違反(リスク)種別の包含関係の全体を図に示し，それぞれの定義を表に示す．



分類	説明
B.	並列化前後のプログラムが一定の等価性基準を満たさなくなる違反
B0.	ある共有変数へのライトを含むアクセス処理の順序やタイミングが不正になる違反
B1.	ある共有変数へのライトを含むアクセス処理の順序が不正に入れ替わる違反(B0 のうち，B2，B3 以外).
B2.	ある 2 つの操作が同時に起こったとき，一方の操作にもう一方の操作が割り込み，等価でなくなる違反
B3.	複数コア上の処理が，互いにロックを取り合い，処理停止する違反
B4.	ブラックボックス（変数の依存解析から特定できない）な順序制約を満たさない違反
C.	命令列が不正に変更される違反

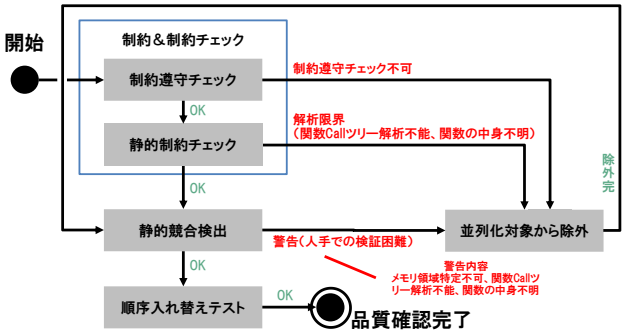
これらの等価性違反の種別は，2.1 節に示した既存研究における違反分類に対して表に示すように対応する．

等価性違反	デッドロック	ライブロック	飢餓	待機中断	データ競合	順序違反	原子性違反
B. 実行順序違反							
B0. 共有変数アクセス順序違反	✔				✔	✔	✔
B1. 非同時実行・実行順序違反					✔	✔	
B2. 不可分操作違反						✔	✔
B3. デッドロック	✔						
B4. 外部依存性違反						✔	
C. リストラクチャリング違反							

3.2 対策の総合的枠組みと検証プロセス

前述のそれぞれの等価性違反に対して，欠陥(違反そのもの)と故障(違反により引き起こされる異常な現象)を分析し，それぞれの違反に対して静的解析・静的チェックおよびテストを中心とした対策，ならびに，技術的制限を整理し，枠組みとしてまとめた．基本的な戦略として，欠陥の検出が可能な違反には静的解析を，故障としてしか検出困難な違反には動的テストを用いて検証を実施する．動的テストでは，並行タスクの実行順序を入れ替えて実行し，様々な順序で実行された結果を比較し，B1 および C を検出する．静的テストでは，静的競合検出により，同時実行された場合に発生するデータ競合を検出する．しかし，動的テストにはテストケースが不十分，静的テストには，ツール等の解析限界などが存在する．そこで，コーディング制約により，検証困難な実装を防ぐことで，並列プログラムの品質を確保する．

この枠組みに基づく検証プロセス全体の概要図を示す．



まずはコーディング制約とそのチェックを行う．その後，静的競合検出と順序入れ替えテストを実施する．なお，各ステップにおいて，何らかの理由で制約遵守が不可能な場合や，ツールの解析限界に達してしまった場合は，その個所を並列化対象から除外することで，品質を確保する．

3.3 課題の扱いの技術的制限

ライブロックや飢餓，待機／中断はツール適用範囲の外でのみ発生するため，対象外とした．

4. おわりに

筆者らは，課題および解決方針の検討と整理に向けて並行性欠陥の分類と実態を調査したうえで，等価性違反およびそれに対応する対策および技術的制限をまとめた総合的な枠組みおよびそれに基づく検証プロセスを提案する．枠組みにおける具体的な対策技術としては，テストおよび静的解析・制約チェックの併用を提案した．

本稿における成果は，自動並列化対象ソフトウェアの開発運用プロジェクトにおいて課題および解決方針の整理と適用に役立てられることを期待できる．今後は，違反および対策の整合性・網羅性の立証を推進する．

参考文献

[1] S.A. Asadollah, H. Hansson, D. Sundmark, S. Eldh, “Towards Classification of Concurrency Bugs Based on Observable Properties,” 1st IEEE/ACM International Workshop on Complex Faults and Failures in Large Software Systems (COUFLESS), 2015.

[2] S.A. Asadollah, D. Sundmark, S. Eldh, H. Hansson, W. Afzal, “10 Years of research on debugging concurrent and multicore software: a systematic mapping study,” Software Quality Journal, Vol. 25, No. 1, 2017.

[3] V. Arora, R. Bhatia, M. Singh, “A systematic review of approaches for testing concurrent programs,” Concurrency and Computation: Practice & Experience, Vol. 28, No. 5, 2016.