様式 C-19

科学研究費補助金研究成果報告書

平成２４年　３月　９日現在

機関番号：１１３０１
研究種目：若手研究（B）
研究期間：2008～2010
課題番号：20700019
研究課題名（和文）　型理論と線形計画法によるマルチスレッドプログラムの安全性検証

研究課題名（英文）　Verification of Multi-thread Programs via Linear Programming

研究代表者
寺内　多智弘（TERAUCHI　TACHIO）
東北大学・大学院情報科学研究科・助教
研究者番号：70447150

研究成果の概要（和文）：
本研究は、線形計画法と型理論を応用した、並行ソフトウェアの安全性（例えば、レース状態が起こらないなど）を静的（つまりコンパイル時に）検証する研究である。具体的には「分数権限計算」(Fractional Capability Calculus)という新しい概念を含んだ型システムを構築し、型推論問題を線形計画問題に帰着することにより高速に自動ソフトウェア検証を行う。

研究成果の概要（英文）：
We propose a software verification framework based on the formalism of fractional capabilities that statically (i.e., at compile time) and automatically checks that certain bad things (e.g., data races) never happen in concurrent programs. The key to the success is the reduction of fractional capability calculi to the problem of linear programming.

交付決定額

（金額単位：円）

	直接経費	間接経費	合　計
2008 年度	1,200,000	360,000	1,560,000
2009 年度	1,100,000	330,000	1,430,000
2010 年度	800,000	240,000	1,040,000
年度			
年度			
総　計	3,100,000	930,000	4,030,000

研究分野：情報科学
科研費の分科・細目：情報学・ソフトウェア
キーワード：ソフトウェア検証、プログラム言語、型推論、権限計算、線形計画法

１．研究開始当初の背景

近年 Intel 社などのマルチコアプロセッサの登場などによりマルチスレッドプログラムの重要性が再認識されている。特に、マルチスレッドプログラムがマルチコア、マルチプロセッサー上で正しく動作するか、つまり個々のスレッドが同時に並列起動した時プログラムが正しく動作するか、が重要課題として考えられる。マルチスレッドプログラムのバグは個々のスレッドの微妙な実行スケジュール条件で起こる場合がありテストだけでデバッグするのは大変難しい。そこでマ

ルチスレッドプログラムの不具合を発見・防止する手法としてモデル検査や型理論に基づいた静的コード解析などによりある定式化された不具合（例えばレースコンディション）が絶対に起こらないことを数理論理学的に検証するアプローチなどが注目されている。例えばレースコンディション検証はカリフォルニア大学サンディアゴ校 Ranjit Jhala 教授などによる「Static Race Detection on Millions of Lines of Code」が 2006 年 FSE に採録され、メリーランド大学カレッジパーク校 Michael Hicks 教授などによる「Context-Sensitive Correlation Analysis for Race Detection」が 2006 年 PLDI に採録されるなど世界的研究者に注目されている。

しかし大規模マルチスレッドソフトウェアの検証はシングルスレッド、つまり逐次プログラムの検証に比べ計算量的に大変困難であり、高速化が重要な研究課題となっている。本課題の応募者は前年、ある特定のマルチスレッドプログラム検証（プログラム決定性問題）は型理論を通じて帰着した線形計画問題の解から求められる、という新しい検証方法を考案し、その手法正しさを数理論理学的に証明した。

２．研究の目的

線形計画問題（Linear Programming）はオペレーションリサーチや工学など幅広い分野で重要な、アルゴリズム理論で最も研究が進んでいる問題の一つで、ｼﾝﾌﾟﾚｯｸｽ法を始め、近年では計算量理論的にも実用的にも優れたな内点法などがあり、巨大な線形計画問題を高速で解く優れた実装（例えば GNU の GLPK や IBM 社の COIN-LP）も多数存在する。つまり、これらの線形計画問題アルゴリズムを利用し高速にマルチスレッドソフトウェアの検証を行うことが可能になる。本課題の目標はこの手法を発展させ、プログラム決定性検証の改良、プログラム決定性問題以外のマルチスレッドプログラム検証などである。具体的には以下のような新たな課題に取り組む。

(1) スレッド同士がメッセージパッシングで通信・同期をとる、いわゆるメッセージパッシング並列プログラムではメッセージがバッファされる場合がある。この場合バッファの最大サイズが有限であることを検証する。
(2) 競合状態（レースコンディション）の防止、つまり複数のスレッドが同時に共有変数をアクセスすることを防ぐ。
(3) 無害な非決定性を許容できるプログラム決定性検証。

線形計画問題を使用したマルチスレッド検証方法の正しさは型理論に基づき数理論理的に証明が可能であるが、上記(1, 2, 3)の解析に対し証明法が微妙に異なる。特に限られた非決定性を許す決定性検証の正しさの証明は複雑であると考察。この手法をさらに深く理解するとともに、証明法の形式化を試みる。

また、線形計画問題を使用したマルチスレッド検証方法が大規模プログラム検証をどれだけ高速化することができるかを明らかにするために、上記の課題（1, 2, 3）の解析の理論的計算量（多項式時間で解けるかなど）を数理論理的に証明するとともに、実際の Linux Kernel などの大規模プログラムに対して検証実験を行う。

本研究が成功すれば、マルチスレッドソフトウェアの安全性の大きな改善が期待できる。

また、(1,2,3) の課題のような様々な検証問題に応用し、検証健全性の証明をさらに形式化することによってこの手法がさらに深く理解され、他の研究者が取り組んできたソフトウェア検証の研究にも本課題の手法が応用できことが期待されるなど、学術的意義も極めて大きい。

３．研究の方法

下記の点を重点に研究を進める。

- 検証システムの定式化。定式化、証明を迅速に進めるために、初期段階ではプログラム決定性問題検証で提案した計算モデル「Simple Concurrent Language」をベースにした計算モデル上で定式化する。
- 検証システムの正しさの証明。型理論に基づき安全性を型安全性の性質として定義することにより証明する。(3)の非決定性を許容できるプログラム決定性検証については従来の型が計算モデル全体のコンフルエンス性質を保つという証明法がそのままでは使えないので証明法を拡張する必要がある。
- 検証システムの理論的計算量を求める。線形計画問題は問題の大きさに対し多項式時間で解が求められることが証明されている。そのため検証システムの計算量は線形計画問題に帰着することによって求められる。
- 検証機プロトタイプ実装と検証実験。

これらの検証は Sun Microsystems 研究所の Ivan Sutherland 博士などが関わる FLEET プロジェクトの FLEET アーキテクチャ上で稼動するプログラムに対する検証として使用できると思われ、FLEET プロジェクトグループの一員であるカリフォルニア大学バークレー校の Adam Megacz 氏らと共同で研究を行う。

検証実験は POSIX Thread ライブリを使用するオープンソースプログラムや Linux Kernel のデバイスドライバのコードに対して行う。このため C 言語で書かれたプログラムを解析できる検証機を実装する必要がある。C 言語にはポインタ、構造体、関数など Simple Concurrent Language にはない機能があるが、これらは従来の型理論ベースの静的解析で研究された手法を応用することによって本課題の検証システムに組み込めると考えられる。

具体的には Alias Analysis を使用する。Alias Analysis (Points to Analysis) は静的コード解析の分野でも最も研究が進んでいる解析のひとつで、CFL(文脈自由言語)-Reachability の理論に基づく多層型解析、Binary Decision Diagram(二分決定図)を利用した Context-Sensitive な解析など精度、速度ともに優れた解析手法が存在する。

また POSIX Thread ライブラリにはコンディション変数、リードライトロックなど Simple Concurrent Language にはない同期プリミティブが存在するため、検証システムの計算モデルを拡張する必要があると考えられる。

Linux Kernel のような大規模なソフトウェア検証は精度を上げる、つまり False Positive を減らすためには検証システムを複雑化する必要がある可能性もある。例えばプログラム決定性検証に使用したシステムをそのまま流用しても値や分岐に対しての精度が低いため例えば排他制御が if 文の条件に依存するプログラムなどは正しく検証できない。

この問題を解決するためには線形計画問題と値や分岐についての問題、つまり線形代数式とブール論理式を同時に解くことができ、近年さかんに研究されている SMT (SAT Modulo Theory) Solver の導入などを考慮する。

４．研究成果

以下の成果が得られた。

(1) バッファ使用量の検証

MPI プログラムなど、プロセス同士がメッセージパッシングで通信・同期をとる並行プログラムでメッセージがバッファされる場合、バッファが限度サイズ以上使用されると、バッファオーバーフローが起きたり例外が発生するなどの問題がある。このようなソフトウェアに対し、型推論と線形計画法を用いてバッファの使用量上限を求めるという多項式時間のアルゴリズムを開発した[9]。

(2) レースコンディションの防止

マルチスレッドプログラムにおいて、異なるスレッドが共有変数を同時にアクセスし、少なくとも一方がその変数に書き込んでいる状態をレースコンディションと呼ぶ。我々は、型推論と線形計画法を用い、プログラム実行時にレースが起こらないか検証する多項式時間のレース検証アルゴリズムを開発した。従来のレース検証アルゴリズムに比べ、セマフォやシグナルなどロック以外の同期プリミティブにも対応するなど、性能の向上に成功した[8]。

(3) Observation Determinism の検証

機密情報を含んだ並行ソフトウェアにおいて、スレッドのスケジューリングから外部に機密情報が漏洩してしまう場合がある。機密情報が漏洩しないことを保障する十分条件の一つに、Observational Determinism という概念がある。Observation Determinism は、外部から見える実行結果が機密情報にもスレッドのスケジューリングにも依存しないという性質である。我々は、型推論と線形計画法を応用することによってメッセージパッシングなどを含んだ幅広い並行ソフトウェアの Observational Determinism を効率的に（多項式時間で）検証することに成功した[７]。

(4) 多相型分数権限計算システムの推論

従来の分数権限計算は文脈非依存であるため、異なる文脈で同じ関数が呼び出されるプログラムを正確に検証できない場合があった。我々は、parametric polymorphism という多相型の理論を分散権限計算に導入し、文脈依存であり、かつ多項式時間で推論可能なシステムを構築した[4]。

(5) 依存型の推論

従来の分数権限計算は通常の型システムと同じくプログラムの値に対して非依存なので、例えば、次のようなプログラムをレースがないと正しく検証できない：

```
if (b) {x++} || if (b) {x++}
```

(P||Q は P と Q が並列に動くプログラムを表す。) ブール値 b に依存する分岐を判別できないためである。これを解決する手段として「依存型」(dependent types)という値に依存する型の概念があるが、自動的に推論するのは困難であった。我々は、モデル検査の研究で広められた、「反例を用いた自動抽象洗練化」(counterexample-guided abstraction refinement)の理念に基づいた、自動的に、かつ効率よく依存型を推論する技術を開発

した[3]。

(6) 量的情報流の検証

分数権限計算を用いたプログラム検証の応用のひとつに機密情報を扱うプログラムの情報漏えいを静的に検証するという「情報流」の検証がある。この研究の発展として、情報漏えいの有無だけでなく、その量を見積もる「量的情報流」の検証の研究を行った。具体的にはシャノンのエントロピー、min エントロピー、など既存研究で提案された量的情報流の定義に従い量的情報流を自動的に検証することの困難性（計算量理論的困難性など）を明らかにし、効果的な検証方法（量的でない情報流解析に用いられる self composition の技法の応用）を提案した[1,2]。

(7) 効率的な文脈依存解析

文脈依存なプログラム解析は、関数の出現の文脈を考慮するため、文脈非依存な解析より正確であるが、効率よく解析を行うのが難しいとされていた、我々は、「Thread-based code cloning」という古典的 code cloning を発展させた並列プログラム解析に特化した文脈依存プログラム手法を開発し、上記の分数権限計算によるレース解析に応用した。

５．主な発表論文等
（研究代表者、研究分担者及び連携研究者には下線）

〔雑誌論文〕（計９件）

[1] Hirotoshi Yasuoka and Tachio Terauchi. On Bounding Problems of Quantitative Information Flow. In Proceedings of the 15th European Symposium on Research in Computer Security (ESORICS 2010), Lecture Notes in Computer Science 6345, Springer, 2010. pp. 357-372. 査読有.

[2] Hirotoshi Yasuoka and Tachio Terauchi. Quantitative Information Flow - Verification Hardness and Possibilities. In Proceedings of the 23rd IEEE Computer Security Foundations Symposium (CSF 2010), IEEE Computer Society, 2010. pp. 15-27. 査読有.

[3] Tachio Terauchi. Dependent Types from Counterexamples. In Proceedings of the 37th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL 2010), ACM, 2010. pp. 119-130. 査読有.

[4] Hirotoshi Yasuoka and Tachio Terauchi. Polymorphic Fractional Capabilities. In Proceedings of the 16th International Static Analysis Symposium (SAS 2009), Lecture Notes in Computer Science 5673, Springer, 2009. pp. 36-51. 査読有.

[5] Tachio Terauchi and Alex Aiken. A Capability Calculus for Concurrency and Determinism. ACM Transactions on Programming Languages and Systems (TOPLAS) 30 (5) : (2008) 査読有.

[6] Tachio Terauchi and Alex Aiken. Witnessing Side Effects. ACM Transactions on Programming Languages and Systems (TOPLAS) 30 (3) : (2008) 査読有.

[7] Tachio Terauchi. A Type System for Observational Determinism. In Proceedings of the 21st IEEE Computer Security Foundations Symposium (CSF 2008), IEEE Computer Society, 2008.

pp. 287-300. 査読有.

[8] Tachio Terauchi. Checking Race Freedom via Linear Programming. In Proceedings of the ACM SIGPLAN 2008 Conference on Programming Language Design and Implementation (PLDI 2008), ACM, 2008. pp. 1-10. 査読有.

[9] Tachio Terauchi and Adam Megacz. Inferring Channel Buffer Bounds via Linear Programming. In Proceedings of the 17th European Symposium on Programming (ESOP 2008), Lecture Notes in Computer Science 4960, Springer, 2008. pp. 284-298. 査読有.

〔学会発表〕(計３件)

1. 安岡宏俊 寺内多智弘 Polymorphic Fractional Capabilities JSSST 第 12 回プログラミングおよびプログラミング言語ワークショップ 2010 年 3 月 5 日 香川県 琴平町
2. 寺内多智弘 Dependent Types from Counterexamples JSSST 第 12 回プログラミングおよびプログラミング言語ワークショップ 2010 年 3 月 4 日 香川県 琴平町
3. 寺内多智弘 Checking Race Freedom via Linear Programming JSSST 第 10 回プログラミングおよびプログラミング言語ワークショップ 2008 年 3 月 6 日 宮城県 仙台市

〔図書〕(計 0 件)
〔産業財産権〕
○出願状況（計 0 件）

名称：
発明者：
権利者：
種類：
番号：
出願年月日：
国内外の別：

○取得状況（計 0 件）

名称：
発明者：
権利者：
種類：
番号：
取得年月日：
国内外の別：

〔その他〕
ホームページ等

６．研究組織
(1)研究代表者
寺内 多智弘（TERAUCHI TACHIO）
東北大学・大学院情報科学研究科・助教

研究者番号：70447150

(2)研究分担者
（ ）

研究者番号：

(3)連携研究者
（ ）

研究者番号：