様式 C-19

科学研究費補助金研究成果報告書

平成２２年３月３１日現在

研究種目：　基盤研究（Ｂ）
研究期間：　２００７～２００９
課題番号：　１９３００００６
研究課題名（和文）時相論理を用いたコンパイラ最適化器の生成・検証と別名を扱えるＳＳＡ形式高度最適化

研究課題名（英文）Generation and verification of compiler optimizers using temporal logic and high-level SSA form optimization considering aliases

研究代表者
佐々　政孝（SASSA MASATAKA）
東京工業大学・大学院情報理工学研究科・教授
研究者番号：２００１６１８２

研究成果の概要（和文）：
時相論理を用いたコンパイラ最適化器の生成系を作成した．これにより，従来は実用にならないと思われていたものが手書きのものに近づいた．また，コンパイラの最適化器が正しいかどうかを，時相論理を用いて検証する方法を開発，実装した．これは，モデル検査によりバグを報告する．一方，プログラムの実行経路によって実行されない式を除去する部分冗長除去の最適化を，要求駆動の性質を用いて，ループ不変式のループ外移動を投機的に行えるように拡張した．本手法を実装し，評価した結果，従来法より効率的に高速なコードを生成できることを示した．

研究成果の概要（英文）：
We made a generator of compiler optimizers using temporal logic. This showed the efficiency of the generated optimizer, that was previously considered unpractical, approaches the hand-made one. We also developed and implemented a method that verifies the correctness of compiler optimizers using temporal logic. The system reports bugs by model checking. On the other hand, we extended the partial redundancy elimination, which can remove expressions executed on only some paths, for speculatively hoisting expressions out of loops using a demand-driven property. We conducted experiments to evaluate our method, so that we showed that it is less costly and generates more efficient code than previous works.

交付決定額

（金額単位：円）

	直接経費	間接経費	合　計
2007 年度	6,200,000	1,860,000	8,060,000
2008 年度	4,800,000	1,440,000	6,240,000
2009 年度	3,500,000	1,050,000	4,550,000
年度			
年度			
総　計	14,500,000	4,350,000	18,850,000

研究分野：プログラミング言語処理
科研費の分科・細目：情報学・ソフトウェア
キーワード：コンパイラ，最適化器，検証，時相論理，モデル検査，投機的部分冗長除去

１．研究開始当初の背景

コンパイラの最適化はプログラムの効率よい実行に欠かせない．我々はコンパイラ・インフラストラクチャ COINS [http://www.coins-project.org/] の最適化器［佐々ほか:情報処理学会論文誌:プログラミング,2006］をベースに，最適化器の研究と実装を行ってきた．しかし，次のような課題が残された．

(1) 実装した最適化器の正しさを保つ作業に多くの苦労があり，解決されていないバグもある．

(2) さらなる最適化のアイディアはあったがまだ実現されていないものも多い．

(1) についての従来の手法は，たとえば，GCC のような実用的に使われているコンパイラにおいても苦労しており，自動テストされる豊富なテスト集合に含まれていない最適化のバグ（虫）が利用者に発見されることが少なくなく，数日の単位で手作業でのバグ修正が CVS 上の改訂の形で行われている．

これに対し，別の側面から最適化の正しさを保証しようとする研究がある．たとえば，Proof Carrying Code (PCC) ［Necula ら:ACM POPL,1997］は，プログラムを配布する際に，プログラムを提供する側がそのプログラムが正しいかどうかの証明を添付し，プログラムを使用する側はその証明が正しいかどうかをチェックしてから実行するという方法である．これは主にプログラムの型に関する正しさを検証する手法で，コンパイラの正しさの一面を保証しようとしている．その他にも Translation Validation ［Pnueli ほか:LNCS,1998］などから始まる多くの研究がある．

一方，通常のコンパイラの最適化器はコードとして書き下すことにより作成されているが，その代わりに，最適化器を論理に基づく記述から生成する研究もある．たとえば，［Lacey ら： Higher-Order and Symbolic Computation, 2004］は，最適化器を時相論理で記述し，そこから最適化器を生成する研究を行っている．ただし，これは理論的研究が主で，実用化からはほど遠い．

(2)の分類にあたるものとしては，申請者らは，静的単一代入形式(以下，SSA 形式という)上の最適化を研究してきた．SSA 形式は，最適化が誤りなく容易に行える形式として注目されているが，国内外でもまだ十分な研究が行われているとは言えない．①申請者は，冗長かどうか知りたい個々の式の出現に注目し，解析範囲を局所化することによって，効率的に冗長な式を除去する手法を提案した［滝本,佐々ら:プログラミング研究会，2006］.この手法は，個々の式の出現に注目しているので，φ関数をまたがった部分式の冗長性も容易に見つけることができる．②また，通常形式上での部分冗長除去などのやや難しい最適化のアルゴリズムを，SSA 形式で行えるアルゴリズムに機械的に変換する汎用的な手法［今橋，佐々:ソフトウェア科学会大会論文集，2006］などの研究も行ってきたが，まだ研究半ばの段階である．これらの研究は，多くの発展性を有しており，これらをさらに発展させたい．③SSA 形式では難しいとされていた別名解析（ポインタ解析）についても，flow-insensitive な解析を SSA 形式に適用する研究を開始しており，配列の扱いにも取り組みたい．

２．研究の目的

「１．研究開始当初の背景」で述べた課題を解決するため，本研究では次を行う．

(1) コンパイラにおける時相論理を用いた最適化器の生成と最適化器の検証の研究

①コンパイラにおける時相論理を用いた最適化器の生成の研究

②コンパイラにおける最適化器の検証の研究

(2) SSA 形式最適化の新しい方式，通常形式最適化の SSA 形式への変換の研究

３．研究の方法

基本的な流れは次のとおりである．

(1)サーベイを行い，従来法の問題点を洗い出す

(2)プロトタイプのアルゴリズムの設計

(3)プロトタイプの実装

(4)プロトタイプの評価

(5)学会大会等で発表，意見をいただく
(6)本システムの設計
(7)本システムの実装
(8)本システムの評価
(9)査読付き論文誌等に発表
必要に応じて後戻りを行う．
モデル検査器は自作する．
新しい最適化は，コンパイラ・インフラストラクチャ COINS の上に作成する．

４．研究成果

(1) 双方向 CTL による Java 最適化器の生成

近年，時相論理によるコンパイラ最適化器生成の研究が行われている．しかし，実際のプログラムを対象として最適化時間や目的コードの実行時間を提示した研究はない．

本研究は従来研究の弱点を克服し，時相論理による記述からJava 言語の効率良い最適化器を生成するシステムを作成した．論理としては，分岐時相論理の1つで過去時制と未来時制をともに扱える双方向CTLであるCTL-FVを用いた．

最適化変換を記す仕様記述も，従来法と異なり，条件式を満たす特定の番号の個々の命令文ではなく，命令文の集合を計算するようにしたので，複雑な最適化が容易に記述できるようになった．さらに，本研究は一時変数を記述できるようにするなど種々の実用化の工夫を行い，Java 言語に対する典型的なコンパイラ最適化器を実現した．

本研究が実装したモデル検査器は何の変換も行わずに，過去時制と未来時制を直接扱えるため，十分な効率で動作する．

従来，モデル検査器を用いた最適化器は実用的な時間では動作しないと言われていたが，実験により，本研究では SPECjvm98 の7つについて，4 秒～4 分で最適化が行える事を確認できた．

また，通常の最適化器に近づいた性能を持った，CTL による最適化器の実装は本研究が初めてであるため，その可能性と問題点を明らかにした．生成される最適化器の処理時間を短くするための工夫や，記述のノウハウなど，本手法の改善に向けた種々の考察を加えた．

(2) 時相論理を用いたコンパイラ最適化器の実行の正しさの検査

コンパイラの最適化がプログラムの意味を変えず正しく動作することは非常に重要である．しかし，最適化には複雑なものも多く，コンパイラの最適化を正しく実装することは困難である．

本研究では，プログラムが正しく最適化されたかどうかを時相論理を用いて最適化後に検査する手法を提案する．最適化によるプログラムの変形箇所が，プログラムの意味を変えないために満たすべき性質を時相論理で記述し，論理式で記述した性質を満たすかどうかを最適化後のプログラムをモデル検査することで検証する．

この手法には，既存の複雑な最適化器にも適用でき，現実的な時間の範囲内での検査が可能であるといった利点がある．提案手法を実装し検査を実行した結果，COINS コンパイラの最適化器の未知のバグを発見することができた．

(3) 自動的等価性差分の抽出による SSA コンパイラ最適化器の生成するコードの正しさの検証

目的コードの効率を向上させる最適化はコンパイラの重要なフェーズである．しかし最近の進んだ最適化器の多くは複雑なソフトウェアであるため，最適化器に誤りがあることは稀ではなく，そのときその原因を突き止めることが難しい．本研究では，最適化前後の差分を自動的に抽出し，時相論理に基づいて変数などの等価性を評価することにより SSA 形式上のコンパイラ最適化器の正しさを検証する手法を提案する．

まず，変形箇所がプログラムの意味を保つために満たすべき性質を時相論理の CTL 式で記述しておく．次に，最適化前後の SSA 形式の中間表現を比較照合し，最適化による変形を抽出する．それらの結果に従ってモデルを生成し，すべての変形がその種の変形に応じた CTL 検査式を満たすかどうかをモデル検査により検査する．

本手法により COINS コンパイラの最適化器の誤りや曖昧な変形をいくつも発見した．本手法では，検証者は最適化アルゴリズムを知る必要がなく，テストコードを実行する必要もない点に特徴がある．

(4) 静的単一代入形式上で通常形式部分冗長除去を実現する汎用的手法

部分冗長除去(PRE)は，部分的に冗長な式を除去する変換で，共通部分式除去とループ不変式移動の効果を含んだ効果的なプログラム最適化である．この PRE を，最適化に適した形式である静的単一代入(SSA) 形式上で行おうという従来研究がいくつかあるが，これは一般には容易ではない．その理由は，SSA 形式の特徴である，変数名の一意性に関する規則が PRE の実装の妨げになるからである．例えば，同じ名前であった変数同士が SSA 形式化に伴い異なる名前になり，変数名の同一性の判断ができなくなるといった問題が挙げられる．このような問題に対し，従来手法では，PRE を SSA 形式上で実現するために特別なデータ構造を用いるなど複雑な処理を行っていた．

これに対し本研究では，SSA 形式でありながら通常形式にも近い性質を持つ CSSA 形式と phi congruence class というものを利用すれば，SSA 形式でも通常形式における変数名の同一性が判断できる事に着目した．その事実に基づいて，通常形式の PRE アルゴリズムを SSA 形式に適用する手法を提案した．この手法は汎用的なものであり，挿入点の決定・式の挿入・式の置き換え（冗長性の除去）という通常の手順に従う PRE であれば，原則としてアルゴリズムに依らず SSA 形式に対応させることが出来，また元々の PRE のアルゴリズムの枠組みを変える必要もない．

本手法を確かめる実験として，代表的な PRE アルゴリズムの一つである Lazy Code Motion を SSA 形式上のアルゴリズムに変換し，変換後でも部分冗長除去の効果が変わりなく発揮されていることを確認した．

(5) 要求駆動型部分無用コード除去

コード最適化の１つである無用コード除去をコード移動を用いて効果を高めた手法に部分無用コード除去がある．部分無用コード除去は，一部の実行パス上で無用である代入文を除去し，ループ不変代入のループ外移動を実現する強力な手法である．しかし，その効果を高めるためには，複数の副次的効果を反映させる必要があり，コストが高いことが知られていた．

本研究では，各代入文ごとに必要に応じて部分無用コード除去を行う手法を提案する．本手法は，プログラムの終了点に近い文から順に適用することによって，多くの副次的効果を反映させることができ，低い解析コストで効果を向上させることができる．

(6) 質問伝播に基づく投機的部分冗長除去

コンパイラが行うコード最適化の１つである部分冗長除去は，冗長な式を除去するとともに，ループ不変式をループの外に移動する強力な手法である．部分冗長除去に基づいて行うプログラム変形は，いずれの実行経路上にも式を増加させないことを保証しており，その意味で，安全な最適化であるといわれる．一方，ループ内に存在する計算のように，頻繁に実行されることが予想される式は，たとえ実行経路上の式の数を増加させても，ループの外に移動させる投機的な移動を行う方が，プログラムの効率的な実行に貢献する場合がある．

本研究では，ループ内の式についてだけ，投機的な移動によって，ループの外に移動させる部分冗長除去法を提案する．従来，部分冗長除去において，一部の実行経路上で冗長な部分冗長な式とループ不変式とを見分けることは困難であった．本手法では，質問伝播という要求駆動型の解析法を用いることによって，任意の制御フローグラフに対して，ループ不変式だけを投機的にループ外に移動させることができる．本手法の効果を示すために，本手法をＣコンパイラに実装し，評価を行った．その結果，従来法と比べ，実行効率が17%以上向上する場合があることを確認した．

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１１件）

① 滝本宗宏：質問伝播に基づく投機的部分冗長除去，情報処理学会論文誌　プログラミング，Vol. 2, No. 5, pp. 15-27 (Nov. 2009)．査読有り

② Fang Ling，佐々政孝：自動的等価性差分の抽出による SSA コンパイラ最適化器の生成するコードの正しさの検証，情報処理学会論文誌　プログラミング，Vol. 2, No. 4, pp. 33-52,(Aug. 2009)．査読有り

③ Sassa, M., Ito, Y. and Kohama, M.:

Comparison and evaluation of back-translation algorithms for static single assignment forms, Computer Languages, Systems & Structures, Vol. 35, Issue 2, pp. 173-195 (July 2009). 査読有り

④ 滝本宗宏：要求駆動型部分無用コード除去，組込みシステムシンポジウム 2008，pp. 107-114 (Oct. 2008). 査読有り

⑤ 佐原聡一郎，佐々政孝：時相論理を用いたコンパイラ最適化器の実行の正しさの検査，(推薦論文・PPL2007)，コンピュータソフトウェア，Vol. 25, No. 1, pp. 151-166 (Jan. 2008). 査読有り

⑥ 滝本宗宏，佐々政孝：静的単一代入形式を用いた最適化（発展編），コンピュータソフトウェア，Vol. 25, No. 1, pp. 30-46 (Jan. 2008). (解説論文) 査読有り

⑦ 佐々政孝，滝本宗宏：静的単一代入形式を用いた最適化（導入編），コンピュータソフトウェア，Vol. 25, No. 1, pp. 19-29 (Jan. 2008). (解説論文) 査読有り

⑧ 中田育男，渡邊坦，佐々政孝，森公一郎，阿部正佳：COINS コンパイラ・インフラストラクチャの開発，コンピュータソフトウェア，Vol. 25, No. 1, pp. 2-18 (Jan. 2008). 査読有り

⑨ 今橋孝典，伊藤陽，佐々政孝：静的単一代入形式上で通常形式部分冗長除去を実現する汎用的手法，情報処理学会論文誌：プログラミング，Vol. 49, No. SIG 1 (PRO 35), pp. 84-95 (Jan. 2008). 査読有り

⑩ Fang, L. and Sassa, M.：Generating Java Compiler Optimizers Using Bidirectional CTL, Electronic Notes in Theoretical Computer Science, Vol. 190/4, pp. 49-63 (Nov. 2007). 査読有り

⑪ 方玲，佐々政孝： 双方向 CTL による Java 最適化器の生成，情報処理学会論文誌：プログラミング，Vol. 48, No. SIG 10 (PRO 33), pp. 76-89 (June 2007). 査読有り

〔学会発表〕(計１２件)

① 藤原一貴，佐々政孝：時相論理 CTL*を用いた JAVA 最適化器の生成の試み，日本ソフトウェア科学会大会論文集，第２６回，3C-4 (2009 年 9 月 17 日)，島根大学.

② Fang, L. and Sassa, M.：Verification of Compiler Optimization Using Temporal Logic by Checking Value Equality Difference, Eighth International Workshop on Compiler Optimization meets Compiler Verification (COCV 2009), CD-ROM (March 22, 2009), York, U.K..

③ Fang Ling，佐々政孝：自動的等価性差分の抽出による SSA コンパイラ最適化器の正しさの検証，情報処理学会プログラミング研究会 (2009 年 3 月 17 日)，東京大学駒場.

④ 太田眞敬，滝本宗宏：COINSを用いた大域的データサイズ推論の実現，日本ソフトウェア科学会大会論文集，第２５回，4A-2 (2008年9月11日)，筑波大学東京キャンパス.

⑤ Sassa, M. and Sahara, S.: Validating Correctness of Compiler Optimizer Execution Using Temporal Logic, Seventh International Workshop on Compiler Optimization meets Compiler Verification (COCV 2008) pp. 1-17 (April 5, 2008), Budapest, Hungary.

⑥ 滝本宗宏，佐々政孝：質問伝播に基づく要求駆動型大域値番号付け，日本ソフトウェア科学会第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 (PPL2008) 論文集，pp. 233-245 (2008年3月7日)，仙台市.

〔図書〕(計１件)

中田育男，渡辺坦，佐々政孝，滝本宗宏：コンパイラの基盤技術と実践 - コンパイラ・インフラストラクチャ COINS を用いて，朝倉書店，260 ページ (Jun. 2008).

〔その他〕
（一般誌）
滝本宗宏，渡辺坦，中田育男，COINS がもたらすコンパイラ革命，UNIX MAGAZINE 2009 年 7 月号.

（ホームページ）
http://www.is.titech.ac.jp/~sassa/coins-www-ssa/japanese/index.html

６．研究組織
(1)研究代表者
佐々　政孝（SASSA MASATAKA）
東京工業大学・大学院情報理工学研究科・教授
研究者番号：２００１６１８２

(2)研究分担者
滝本　宗宏（TAKIMOTO MUNEHIRO）
東京理科大学・理工学部・講師
研究者番号：００３１８２０５

(3)連携研究者
なし