研究課題別評価

1. 研究課題名：インターコミュニケーション・プログラミング

2. 研究者氏名：関口龍郎

3. 研究の狙い：

この研究ではインターネット上での人と人との双方向コミュニケーションを支援するソフトウェアを安全で効率良く実行できるプログラミング言語システムの構築を目指した。具体的には、Java 仮想機械や.NET ランタイム(CLR)のような、言語非依存、アークテクチャ非依存のコード表現の開発とその実行環境の実装を目標とした。普遍的なコード表現の開発における最大の問題は実行効率と安全性の両立である。Java 仮想機械では安全性を重視するためにプログラムを実行する直前にプログラム検証を行い、合格したものだけが実行される。C 言語などの効率の良い言語で記述されているコードパターンにはポインタ演算等のプログラム検証の制限によって許可されない演算があるため Java 仮想機械では C 言語ほど効率の良い実行ができない。.NET ランタイムではこの制限を回避するためにプログラム検証を行わずに実行するオプションがある。プログラム検証を行わなかった場合にはもちろん実行の安全性は保証されないことになる。インターネットで使われているメールやウェブのサーバやブラウザに対する攻撃方法のほぼ半数はプログラミング言語の仕様の欠陥である不正メモリアクセスを利用しており、言語としての安全性の保証は重要である。この研究では、高い実行効率を実現するために極めて重要な演算であるが Java のプログラム検証では除外されているポインタ演算に着目し、ポインタ演算を含むプログラムの検証を可能にする検証技術を開発し、実行効率と安全性が両立されたプログラミング言語システムの構築を目指した。

4. 研究結果：

本研究の成果は、①ポインタ演算を記述できる言語非依存、アーキテクチャ非依存の低水準言語、②前項の低水準言語の SPARC と IA32 プラットフォームへのコンパイラ、③ポインタ演算を含むプログラムの検証手法の設計と検証器の実装、④標準的なベンチマークである SPECCPU2000 を利用した検証アルゴリズムの性能データ、以上 4 点である。

(1) 我々の設計した低水準言語と Java バイトコードとの大きな違いは 2 点ある。一つはポインタ演算を許可していることであり、もう一つは Java のように例外を言語 primitive として提供するのではなく、「例外を実装できる機構」を提供していることである。ポインタ演算を記述できるので様々な高級言語のプログラムを効率を落さずに我々の低水準言語にコンパイルすることができる。現代的なプログラミング言語では例外機構は不可欠の機能であるため、我々の低水準言語では例外を効率良く実装できる機能を提供している。これは、高級言語から C 言語への変換によって高級言語を実装するときにうまく記述できない箇所である。我々の言語は C 言語に近い syntax と BCPL に近い semantics を持っている。

(2) 我々はこのような低水準言語を単に設計するだけではなく、Solaris と Windows 上で実行可能な SPARC と IA32 の機械語コードを生成するコンパイラの実装を行い、設計の有効性の検証を行った。

コンパイラは Objective Caml で記述されており、コード量はおよそ 18000 行である。同じ意味を持つコードの実行効率をGNU C Compiler と MS C の出力するコードと比較を行った。多くの場合は同程度であり、いくつかのケースでは我々のコンパイラの方が 2? 3 割速いコードを生成した。

(3) 我々は、ポインタ演算を記述できる我々の低水準言語上でのプログラム検証を行う検証アルゴリズムを開発し、検証器の実装を行った。この検証アルゴリズムはいわゆるポインタ解析と呼ばれるプログラムの静的解析手法に基づいている。この解析によって変数の指す可能性のある値の範囲とメモリに格納される可能性のある値の範囲を静的に見積もることができる。変数の取り得る値を静的に決定できるため、実行時にヌルチェック、境界チェックなどの種々のチェックを省くことができ、実行効率を向上させることができる。解析器は Objective Caml で記述されており、コード量はおよそ 2000 行である(ただし予備処理を行うコードが加えて 14000 行ある)。

(4) 我々は標準的なベンチマークである SPEC CPU2000 に対して解析器を適用し、性能を計測した。解析器は数千行の C 言語のプログラムを数秒から十数秒で解析することができ、ほぼ線形時間で解析を行うことができる。この解析時間は充分実用的な水準にある。解析の結果を利用して 2 割から 7 割の実行時チェックを除去することができ、実行時間は 1 割から 4 割向上することが分かった。我々のプログラム検証技術により安全性を損なうことなく実行効率を高めることができた。

5. 自己評価：

Java 仮想機械や .Net 仮想機械とは質的に異なる全く新しい手法により安全性と実行効率を両立させるプログラミング言語システムを作ることができたと思っている。しかし当初想定していたメールサーバなどの現実的なアプリケーションを対象に有効性の検証を行うところまで至らなかったのは残念である。プログラム検証アルゴリズムは従来あるポインタ解析アルゴリズムと比べてかなり複雑なものであり、現実的な時間で実用的なプログラムを解析できるかどうか不安があったが(ポインタ解析には数ギガバイトのメモリと数十時間の解析時間を要するものも多い)結果を見ると充分高速に解析できており、ほっとしている。当初は実用的なシステムを作ることに主眼を置いていたが、実用的な規模のプログラムを解析してみた経験からプログラム検証の新しい方法の着想も得ており、今後理論的に新しい結果も出せそうである。この研究は、一般のユーザが利用できるツールを公開するところをもって完結すると思っており、これからは「きっかけ」と離れることになるが、ツールの公開まで研究を続けるつもりである。コンピュータ科学は1人から数人のグループによる研究結果によって世界に影響を与える仕事を行える分野であり、個人研究型の形態は意味があると思う。

6. 研究総括の見解：

インターネットが社会基盤となっている現在、そこでの安全性の確保は極めて重要であり、安全性を担保可能なインターネットプログラミング言語の研究が活発に行われている。関口研究者はそのような言語の設計と安全性確保のための検証系を組み込んだ言語処理系の設計・構築を行ったが、第 1 級の研究結果を出したと考える。実用性のある言語では安全性上問題のあるポインターを排除することは出来ないことを前提として、ポインター解析に従来より強力な方法を導入することにより、実行速度と安全性の両面において現在開発されている言語としては最高レベルの性能を発揮している。実用的なソフトウェア開発での採用にあともう一歩必要であるが、この分野に非常に大きな貢献をしたと考

える。

7. 主な論文等：

1. Tatsurou Sekiguchi, Takahiro Sakamoto, and Akinori Yonezawa. Portable Implementation of Continuation Operators in Imperative Languages by Exception Handling. Advances in Exception Handling Techniques, LNCS2022, Springer Verlag, May 2001.
2. 関口 龍郎、大岩 寛、米澤 明憲。オブジェクト指向言語によって記述された、携帯電話、PDA のアプリケーションプログラム 圧縮方式. 第 3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. 2001 年 3 月 (コンピュータソフトウェア． Vol.19, No.1． 2002 年 1 月)．
3. Takeo Imai, Tatsurou Sekiguchi, Hidehiko Masuhara, and Akinori Yonezawa. Dynamic Access Control of Mobile Objects by Switching Namespaces. OOPSLA Workshop of Patterns and Techniques for Designing Object-Oriented Mobile Wireless Systems. October 2001.
4. Tatsurou Sekiguchi, Yutaka Oiwa, and Eijiro Sumii. Software: Rog-O-Matic III , The 2002 International Conference of Functional Programming (ICFP '02) Programming Contest Playoff, October 2002.
5. Yutaka Oiwa, Tatsurou Sekiguchi, Eijiro Sumii, and Akinori Yonezawa. Fail-Safe ANSI-C Compiler: An Approach to Making C Programs Secure - Progress Report. International Symposium on Software Security, November 2002 (pp133-153, LNCS2609, Springer Verlag, April 2003).
6. 住井 英二郎、関口 龍郎、細谷 春夫． PLI2002 報告． コンピュータソフトウェア． Vol.20, No.2, pp79-84, 2003 年 3 月．
7. 関口 龍郎． Java のための二つのモバイル技術． オブジェクト指向 2003 シンポジウム． 2003 年 8 月．
8. 関口龍郎． C 言語のための現実的なポインタ解析． 第 6 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 PPL2004(予定)． 2004．