氏名	馬谷誠二（うまたにせいじ）
学位の種類	博士（情報学）
学位記番号	情博第122号
学位授与の日付	平成16年3月23日
学位授与の要件	学位規則第4条第1項該当
研究科・専攻	情報学研究科通信情報システム専攻
学位論文題目	Efficient and Scalable Implementation of an Object-Oriented Multi-threaded Language （オブジェクト指向マルチスレッド言語の効率良くスケーラブルな実装）
論文調査委員	（主査）教授 湯淺太一　教授 富田眞治　教授 奥乃博

論文内容の要旨

本研究では，オブジェクト指向マルチスレッド言語において，柔軟な並列処理をサポートするために必要となるスレッド生成，同期処理，および並列処理中の例外処理機能を効率良く，かつ負荷分散の自動化により並列計算機上においてスケーラブルに実装するための手法を開発している。本論文はその成果をまとめたものであり，以下の10章から構成されている。

まず第1章では，高性能計算のための並列言語に関する概観と研究課題，及びそれらの課題に対する本研究のアプローチについて述べている。

第2章では，本研究の背景となる事柄をまとめている。並列処理を記述するための方法を分類し，それらの特徴や利点について比較，検討を行っている。また，例外処理についても，その使用目的，一般的な記述方法とその意味についてまとめている。さらに，並列処理言語を実装する場合，特に処理系の実行効率の観点から留意すべき事柄について述べている。

第3章では，Java言語を拡張したオブジェクト指向並列言語OPAについて説明している。OPAは簡潔なfork-join型並列構文を提供し，構造化された並列計算の記述を可能としている。一方で，JavaやOPAのオブジェクト指向並列計算においては，synchronized構文や条件変数による同期，共有オブジェクトへの排他的アクセスをサポートすることによって，不規則な並列計算の記述も可能としている。さらに，OPAでは洗練された例外処理を実現するため，動的スコープによるjoin構文を採用しており，例外ハンドラは，並列実行中に任意の子スレッドが投げた例外を捕捉することが可能である。

第4章では，OPA言語処理系の構成と実装技術を述べ，従来の処理系におけるマルチスレッドの実現方法とそれらの問題点について触れている。OPA処理系では，synchronized構文を実現するためにスレッドの同一性を維持する必要がある。また，さまざまな同期処理を可能とするためのスレッドの中断および再開の機能も必要であり，fork-join型並列処理のみをサポートする言語処理系よりも複雑な処理を必要とする。

第5章では，OPAのfork-join型並列処理構文をスケーラブルかつ効率良く実装するための方法を提案している。マルチスレッド言語の効率良い実装において，「遅延」は重要な概念である。本研究では，スレッド同一性の維持，さまざまな同期，動的スコープによるjoin構文といったOPAの現代的な言語機能を効率良く実装するために遅延性を利用する方法を提案している。また本研究におけるOPAの実装では，LTC（Lazy Task Creation）という効率良い負荷分散手法を採用しているが，従来のLTCでは考慮されていなかった，上述の現代的な言語機能をサポートしている。

第6章では，ループ構文に対するLTC手法の欠点を克服するための手法を提案している。反復コードの実行中にスレッド生成を行うことも可能であるが，これを通常のLTCを用いて実行した場合，プロセッサ間でバランス良く仕事を分担することができなかった。本研究では，反復コード実行中には通常と異なる方法でLTCを適用することにより，効率の良い負荷分散を可能にした。

第7章では，LTCが導入されたOPAのような処理系において，例外処理を効率良く実装するための手法を提案している。例外が発生したかどうかのチェックを他のチェックと同時に行うことで，通常処理時のオーバヘッドを増やすことなく例外

処理機能を実現することを可能とした。実行中のスレッドは，中断すべきかどうかの判断のためにfork-joinのすべての入れ子レベルで例外の到達をチェックする必要があるが，LTCによる実装のもとで，チェックの回数を大幅に削減する手法も提案している。

第8章では，並列処理，例外処理に関する関連研究を紹介し，言語設計および実装手法の観点からOPAとの比較を行っている。

第9章では，性能評価を行い，本研究の実装手法の有効性を検証している。OPA処理系が著名なマルチスレッド言語であるCilkより優れた実行性能を得ていること，ループを持ったプログラムにおいてもスケーラビリティが低下しないこと，例外処理機能を追加しても処理系の性能が低下しないこと，および中断処理が効率良く実装されていることを確認している。

第10章はまとめであり，本研究で得られた主要な成果について要約している。

論文審査の結果の要旨

本論文は，Java言語の拡張であるオブジェクト指向マルチスレッド言語OPAの並列処理，例外処理をスケーラブルかつ効率良く実装するための手法に関する研究成果を取りまとめたものであり，得られた主な研究成果は次の通りである。

（1）OPAでは，fork-join型並列処理を実装するために，LTC（Lazy Task Creation）と呼ばれる手法を用いている。LTCは，細粒度マルチスレッドをサポートする処理系において効率良い負荷分散を実現するために利用されるが，LTCを実装に用いた従来の言語の多くは，例外処理やオブジェクトベースの同期といった現代的な言語の持つ先進機能をサポートしていないために，柔軟な並列処理の記述が困難であった。OPA言語は，これらの先進機能を提供しているが，本論文が提案する手法により「遅延」を有効に利用することによって，効率の良い実装を可能とした。実際，OPAではこれらの先進機能を許すことで，著名なマルチスレッド言語であるCilkでは記述できない不規則な並列計算を記述可能としながらも，Cilkに勝る実行性能を得ている。

（2）従来のLTCでは，ループ中のforkについては，効率良く負荷分散を行えなかったが，本論文では，LTCによる動的負荷分散の方法を拡張することによって，この問題にも対処可能であることを示した。

（3）OPA言語は，動的スコープによるfork-join型並列処理に適した例外処理を採用しており，それにより不規則な並列計算の記述をさらに容易にしている。本論文では，LTCをベースとしたOPA処理系において，例外処理機能を効率良く実装するための手法を提案した。これによって，例外処理機能をまったく使用しないプログラムの場合には，例外処理機能を処理系に追加したことによるコストは皆無となった。さらに，例外処理の枠組みでスレッドの中断処理を実現する手法を提案し，大規模な並列計算機上での並列プログラムの実行効率を大幅に向上させることを可能にした。

以上要するに本論文は，先進的な機能を備えた現代的な高級プログラミング言語においてもマルチスレッド機能を効率良く実現できることを明らかにしたものであり，学術上，実際上寄与するところが少なくない。よって，本論文は博士（情報学）の学位論文として価値あるものと認める。また，平成16年2月18日に実施した論文内容とそれに関連した試問の結果，合格と認めた。