

並行・分散ソフトウェアのモデルと実装に関する研究

学位論文内容の要旨

複数の計算機，あるいは，単一の計算機の疑似並列機構の上で，協調的に動作するソフトウェアを構築する際，そのソフトウェアには同期・相互排除等，並行計算に特有な幾つかの解決すべき問題が発生する．それらの問題は 1960 年代より指摘されており，それらを解決するための様々な計算モデル/通信モデルが提案され続け，現在においてもなお並行計算モデルの研究は盛んに行われている．

並行計算に関わる研究の初期段階においては，密結合共有記憶型の計算形態を想定したモデルが数多く提案されたが，近年のネットワーク技術の発展と共に，分散ネットワークに適合したモデルが台頭し，それらは現在ではプログラミング言語の標準的な機能として実装されるまでに至っている．

このような並行計算モデルを発案する際に重要となる要素は，その機構を実装する際の効率性というよりはむしろ，並行ソフトウェア設計の上流工程とコーディング工程との間のセマンティックギャップを解消することに貢献する度合，言い替えれば，抽象化の度合であると考えられる．

しかし残念なことに，並行計算の概念が提案されてから 20 年を超える現在に至っても，並行計算における通信機構は十分に抽象化されているとはいえず，構造化を経てオブジェクト指向へと進化してきたプログラミング言語の発展と比較して，大幅に遅れをとっていると考えられる．

本研究の中心的なテーマは，複雑な相互作用を有した並行問題を柔軟に設計／表現することが可能な並行計算モデルを構築し，さらに，提案したモデルを実現した並行プログラミング環境を実装することである．そのために，まず本研究では，抽象度を高めた幾つかのプロセス間相互作用モデルを提案している．

Kamui モデルでは，当時研究が盛んであった並行オブジェクトモデルをベースとして，それに場の概念および放送の概念をいち早く導入することで，現実世界の様々な相互作用を自然に表現することに焦点をあてた．また，到着順序モデルでは，分散計算におけるメッセージ到着順序の非決定性を効果的に解消するための分散アルゴリズムを提案し，また，そのアルゴリズムの保証の基に，順序を

宣言的に表現可能なモデルの提案を行った。

提案した上記 2 モデルの発展として、本研究では Generic Interaction Model を提案した。このモデルは Linda モデルを拡張し、相互作用を規定するための書き換え可能なルール群を導入することにより、問題に応じて相互作用の形態を変化させることを可能とした。そしてその結果として、複雑なプロセス間相互作用の柔軟な表現、および独立性の高いプロセスの設計が可能となっている。

さらに本研究では、Generic Interaction Model に基礎を置く並行プログラミング言語を開発し、その実装を行った。定量的な実験による評価が困難である計算モデルの研究領域においては、モデルを反映した言語処理系を実装し、現実の様々な問題に適用することによってモデルの評価を行う必要がある。モニタモデルを具現した Concurrent Pascal, CSP を具現した Occam, Actor モデルを具現した Act1 や ABCL/1 はその例である。

本研究ではまた、Generic Interaction Model の開放分散系への拡張を行った。開放分散系は大域的情報が本質的に得られない系であり、動的に変化する分散ネットワークなどがその例にあたる。

本研究によって、複雑な並行システムの設計時あるいは実装時における、仕様と実装との間のセマンティックギャップが短縮され、より高品質な並行・分散システムの構築が可能となった。

学位論文審査の要旨

主 査	教 授	宮 本 衛 市
副 査	教 授	嘉 数 侑 昇
副 査	教 授	大 内 東
副 査	教 授	和 田 充 雄

学 位 論 文 題 名

並行・分散ソフトウェアのモデルと実装に関する研究

複数の計算機，あるいは単一の計算機の疑似並列機構の上で，協調的に動作するソフトウェアを構築する際，そのソフトウェアには同期・相互排除等，並行計算に特有な幾つかの解決すべき問題が発生する。それらの問題は 1960 年代より指摘されており，それらを解決するための様々な計算モデル/通信モデルが提案され続け，現在においてもなお並行計算モデルの研究は盛んに行われている。

並行計算に関わる研究の初期段階においては，密結合共有記憶型の計算形態を想定したモデルが数多く提案されたが，近年のネットワーク技術の発展と共に，分散ネットワークに適合したモデルが台頭し，それらは現在ではプログラミング言語の標準的な機能として実装されるまでに至っている。

このような並行計算モデルを発案する際に重要となる要素は，その機構を実装する際の効率性というよりはむしろ，並行ソフトウェア設計の上流工程とコーディング工程との間のセマンティックギャップを解消することに貢献する度合，言い替えれば，抽象化の度合であると考えられる。

本研究の中心的なテーマは，複雑な相互作用を有した並行問題を柔軟に設計／表現することが可能な並行計算モデルを構築し，さらに，提案したモデルを実現した並行プログラミング環境を実装することである。そのために，まず本研究では，抽象度を高めた幾つかのプロセス間相互作用モデルを提案し，さらにそれらのモデルに基礎を置く並行プログラミング言語を開発し，その実装を行っている。定量的な実験による評価が困難である計算モデルの研究領域においては，モデルを反映した言語処理系を実装し，現実の様々な問題に適用することによってモデルの評価を行う必要があるため，本研究のアプローチは極めて妥当であると考えられる。

第 1 章では，本研究の目的および概要について述べている。第 2 章では，既存の並行計算モデルを列挙し，それらの特徴や問題点を指摘している。第 3 章では，本研究に間接的に関連す

る分野（プラットフォームや応用）に関しての概説を行っている。

第4章では、通信場と放送に基礎を置く並行計算モデルを提案し、その有効性や他のモデルとの比較検討を行っている。通信場と放送という概念は現実の世界における情報伝達の多くを含むと考えられるため、従来の1対1の通信に比べて問題をより自然に表現することが可能となっている。

第5章では、メッセージの到着順序を送信者が制御できる並行計算モデルを提案し、いくつかの並行問題に応用している。このモデルを用いることにより、複雑な同期制御の削減が達成され、並行問題の複雑さが軽減されることが期待できる。

第6章では、共有空間に基づく相互作用の一般化モデルを提案している。このモデルでは、書き換え可能な相互作用規則を相互作用空間の属性とすることによって、様々なプロセス間相互作用を設計できることに特徴がある。

第7章では、第6章で提案したモデルに基づき、並行プログラミング言語の設計と実装を行っている。言語を実装し、実際の問題への摘要を実践したことによって、提案したモデルの有効性を検証することが可能になった。

第8章では、閉じた並行計算の世界から大規模開放系へのモデルの拡張を行い、さらにそれに対応するプログラミング言語の設計と実装を行っている。大規模開放系は、複雑化する並行・分散システムの極限的形態であり、今後ネットワーク社会が向かっていく形態と考えられる。したがって、開放系に対応した計算モデルは非常に重要な研究テーマである。

第9章では、並行計算モデルの応用としてのエージェントモデル、および基礎としてのプロセス代数をそれぞれ考察している。最後に第10章において今後の展望が述べられている。

これを要するに、著者は、抽象度の高い並行計算モデルを提案し、より高品質な並行・分散システムの構築を可能としたもので、並行・分散計算とプログラミング言語に関する多くの新知見を得ており、計算機言語学、ソフトウェア工学に寄与するところ大なるものがある。

よって著者は、北海道大学博士(工学)の学位を授与される資格あるものと認める。