審 査 の 結 果 の 要 旨

氏　　名　　アフェルト レナルド

本論文は、定理証明器を用いて現実的な並行プログラムの検証を可能にするための２つのアプローチを提案し、定理証明器Coqを用いてその評価を行っている。本論文は９章から成り、第１章は導入、第２章は準備である。第３章と第４章で１番目のアプローチについて述べ、第５章から第８章まで２番目にアプローチについて述べている。第９章は結論である。

１番目のアプローチでは、並行プログラムを関数プログラムとしてモデル化し、定理証明器を用いて関数プログラムの検証を行う。第３章において関数プログラムによるモデル化の方法について述べられている。第４章では、このアプローチに基づき、現実的なクライアント／サーバアプリケーションの例として、SMTPサーバの受信部分の検証を行った事例について述べられている。SMTPサーバの受信部分（700行のJavaプログラム）が満たすべき４つの性質が定理証明器Coqを用いて形式的に検証された。この検証により、プログラムのバグを発見するとともに、証明の複雑さが適切な範囲内におさまることが確認された。

第２のアプローチでは、並行プログラムを直接定理証明器上にモデル化し、以下のようなライブラリを用いて定理証明器上で検証を行う。ライブラリは、(1)モデル記述言語、(2)仕様記述言語、(3)検証に利用する補題群からなる。モデル記述言語は、Coqのデータ型と関数を用いて拡張されたπ計算として構築される。

第５章では、まず、仕様記述言語とモデル記述言語が非形式的に導入される。そして、この枠組みのもとで、並行プログラムが多くの状態をもつ場合に状態空間が爆発するとい問題に対処するため、検証のために探索する状態空間を削減する技術であるpartial order reductionに必要な定理が証明されている。

第６章では、モデル記述言語が定理証明器Coq上で形式化され、特に公平性の概念が形式的に定義されている。次の第７章では仕様記述言語が形式化され、並行プログラムの検証に必要な補題群が検証されるとともに、上述のpartial order reductionも公理化されている。仕様記述言語は公平性の概念を導入して拡張された空間論理として構築されており、種々の現実的な並行プログラムをモデル化・定式化するのに役立つと考えられる。

最後に、第８章において、ライブラリ（Coqのスクリプトでおよそ1万5 千行）を用いて、第１のアプローチで用いたメールサーバプログラムのモデル化が再度行われ、その重要な性質が形式的に検証されている。この実験により、複雑なモデル化の技術を用いることなく、現実的な並行プログラムを定理証明器上で検証できることが示された。このアプローチは、第１のアプローチと異なり、並行プログラムを直接的にモデル化・検証することができるとともに、ライブラリによる検証の再利用性が高いという利点がある。

結論として、以上の研究は、現実的な並行プログラムの形式的な検証技術にとって多大な貢献と考えられる。よって本論文は博士（情報理工学）の学位請求論文として合格と認められる。