氏　　名　内藏 理史

学位(専攻分野)　博士(情報学)

学 位 記 番 号　総研大甲第 2326 号

学位授与の日付　2022 年 3 月 24 日

学位授与の要件　複合科学研究科 情報学専攻
学位規則第6条第1項該当

学 位 論 文 題 目　Semantic Refinements for Program Verification

論 文 審 査 委 員　主　査　蓮尾 一郎
情報学専攻 准教授
龍田 真
情報学専攻 教授
関山 太朗
情報学専攻 助教
亀山 幸義
筑波大学 大学院システム情報工学研究群 教授
Sam Staton
オックスフォード大学 Department of Computer Science 教授

（様式３）

博士論文の要旨

氏　　名 内藏 理史

論文題目 Semantic Refinements for Program Verification

Semantics of programming languages is an essential theory not only for defining the meaning of programs but also for studying properties of programs. This thesis aims to apply program semantics to verification of programs. Specifically, we study refinements of semantics of programs to enhance expressivity and verification power. The refined models maintain essential mathematical structures of the original models and have more information about properties of programs. By replacing the original models with the refined models, we obtain improved verification methods. In the thesis, we consider four applications of semantic refinements.

The first one is a semantic construction of dependent refinement type systems. A dependent refinement type system is a type system that admits both refinement types and dependent types. Refinement types are types whose values are restricted by predicates and can be used to specify preconditions and postconditions of functional languages. Dependent types are types that depend on other terms and allow us to use postconditions that depend on the input value. We study categorical semantics of dependent refinement type systems. Our construction is based on the following intuition: a dependent refinement type system is obtained from an underlying type system (a type system that does not contain refinement types) by refining types by predicates. We formalize a semantic counterpart of this intuition. Given a model of the underlying type system (a closed comprehension category and a fibred monad) and a model of predicate logic (a posetal fibration with some conditions), we construct a model of the dependent refinement type system (a closed comprehension category and a fibred monad that refines the model of the underlying type system). We show that we can define the interpretation of refinement types using our construction. We also provide several examples of our construction.

The second one is a program logic for effect handlers. Effect handlers are a programming language feature that allows programmers to implement user-defined computational effects. However, verification of effect handlers is not yet well-studied. We provide a program logic for effect handlers by considering refined semantics of effect handlers. Specifically, we consider Hoare triples for effect handlers and interpret them as liftings of the interpretation of effect handlers along a fibration. We consider sufficient conditions under which we can construct those liftings from liftings of each operation in effect handlers. Such conditions lead to inference rules of our program logic that provide compositional reasoning about effect handlers.

The third one is decision tree-based ranking function synthesis. Ranking functions are an essential notion for termination analysis. We propose an example-based termination analyzer that can synthesize piecewise affine ranking function. Our analyzer finds a piecewise affine ranking function for a given program by repeatedly guessing a candidate solution from a finite set of examples of the transition relation and accepting the genuine solution from the candidates. We refine an existing example-based method that synthesizes affine ranking functions so that our method can synthesize piecewise affine ranking functions. Our method uses decision trees to express affine ranking functions and extends the decision tree learning algorithm for transition examples.
Our extended decision tree learning algorithm detects a certain kind of cyclic constraints in transition examples and resolves them by appropriately splitting the state space. We implemented our synthesizer and compared our tool with other tools.
The fourth one is a method for overapproximating tail probabilities of runtime of randomized programs. It is known that ranking supermartingales can give an upper bound of the expected runtime of randomized programs. This fact can be used to overapproximate tail probabilities of runtime of randomized programs by applying concentration inequalities like Markov's inequality. We refine the existing notion of ranking supermartingales so that they can also give upper bounds of higher moments of runtime. Technically, our improvement is based on the order-theoretic characterization of ranking supermartingales. It is known that the expected runtime is the least fixed point of a certain monotone function, and ranking supermartingales are prefixed points of the monotone function. We extend this to characterize higher moments of runtime as the least fixed point and define ranking supermartingales for higher moments as prefixed points. This extension allows us to improve upper bounds of tail probabilities. We implemented a synthesizer of our notion of ranking supermartingales and conducted experiments.

Form8 : Separate Sheet
（様式８・別紙１）

Results of the doctoral thesis screening

博士論文審査結果

Name in Full
氏 名　　内藏　理史

Title
論文題目　　Semantic Refinements for Program Verification

本論文は，近年さまざまな機能によって拡張されつつある種々のプログラミング言語に対して，プログラムの数理的意味論の理論的詳細化を通じて効率的形式検証手法を追求した出願者の研究内容をまとめたものである．

プログラムの形式検証とは，所与のプログラムが所与の論理的性質（仕様）を充足することを数学的に証明する営みを指し，ソフトウェアの正しさや安全性に数学的証明という強い保証を与えるソフトウェア科学の方法論である．形式検証においては，プログラムの振る舞いを数学的に定義しなければ，その性質に対して数学的な言明を行うことができない．このような数学的定義をプログラム意味論（semantics）とよぶ．

本論文では，近年注目されるいくつかのプログラム機能及び性質記述方式に対して，これらに対しての形式検証を可能またはより効率的にするような意味論の詳細化及び拡張を導入している．また，これらの理論的詳細化に基づく形式検証の具体的手法を提案し，いくつかについてはこれを実装してその性能を実験的に評価している．

本論文は５つの章から構成され英語で書かれている．第１章ではプログラムの形式検証及び意味論についての導入が述べられたあと，続く４章の技術的な内容（理論および検証手法の樹立）の概要が述べられている．

第２章では，関数型プログラミング言語の基盤となる型システムの意味論に対して研究を行っている．具体的には，近年いくつかのプログラミング言語で用いられている先進的型システムであるところの dependent type system に対し，形式検証における仕様記述の表現能力をさらに大幅に向上させる refinement type の拡張を施した dependent refinement type system (DRTS) について，その数学的意味論を構築する一般的方法論を提案している．ここでは，dependent type system の意味論をファイバー圏を用いて与える Jacobs, Ahman らの既存研究を踏まえ，refinement type system の意味論たる poset ファイバー圏を組み合わせる構成が導入されている．主要な定理は DRTS の健全性であり（「型が導出できれば意味論が定義される」），DRTS の型導出を用いたプログラムの形式検証の意味論的正当化を与えている．

第３章では，関数型プログラムの副作用のハンドラに対する形式検証を研究している．具体的には，Plotkin, Pretnar らが導入した代数的ハンドラの一般論の上に形式検証のためのプログラム論理体系を導入して，第２章と同じくファイバー圏を用いた意味論を与えている．ここでの主要な定理は，代数的ハンドラのための論理的推論が代数的演算子ごとに分解して行えるという意味論的 operation-wise lifting 定理と，これを踏まえたプログラム論理の導出規則の健全性定理である．提案したプログラム論理を用いたプログラム形式検証の例も与え，その具体的応用を議論している．

第４章では,手続き型プログラムの停止性の自動形式検証アルゴリズムを研究している．停止性の形式検証のためにはランク関数とよばれる関数を証拠として合成するのが標準的な手法であるが，ランク関数の探索範囲を表すテンプレートとして近年盛んに研究されているのが，場合分けとアフィン関数を組み合わせた piecewise affine function である．本章ではこの piecewise affine function によるランク関数合成を CEGIS とよばれるデータ駆動形合成手法に基づいて効率的に行うための意味論的考察を行っている．具体的には，CEGIS における反例がなす implicit cycle とよばれる現象の特徴づけと，その効率的対処による CEGIS の効率化を導入している．提案アルゴリズムは実装がなされ，当該コミュニティで標準的に用いられるベンチマークセットに対して (1) state-of-the-art のツールと比肩するパフォーマンスが見られること，(2) 他のツールには解けないが提案手法には解けるベンチマークが存在すること，以上が実験的に示されている．

第５章では，確率的プログラムの停止時間の tail probability の自動近似計算手法を研究している．ここでは ranking supermartingale とよばれる関数を証拠として合成するのが標準的手法であるが，本論文では意味論的考察のさらなる進展により，(1) 停止時間の（１階だけでない）高階モーメントを表す ranking supermartingale の定式化，(2) 当該 ranking supermartingale 概念の線形計画法等による効率的探索，(3) これらを Markov 集中不等式と組み合わせた tail probability の導出法，以上が理論的貢献として示されている．この手法は実装がなされ，高階モーメントを用いることで近似精度が実際に大きく上昇することが実験的に示されている．

本論文にまとめられた研究の成果は査読付き国際会議予稿集のフルペーパー３篇として発表されている．これらの論文を含めた出願者の出版業績は以下の通りである：主著者となる査読付きトップ国際会議予稿集フルペーパー４篇（うち２篇は単著論文）．

公開発表会では博士論文の章立てに従って発表が行われ，その後に行われた論文審査会及び口述試験では，審査委員からの質疑に対して適切に回答がなされた．

以上を要するに本論文は，プログラムの形式検証というソフトウェア科学の重要な研究トピックにおいて，意味論の発展という明確に数学的・理論的な立場から重要な貢献を行ったものである．第２・３章の成果は，関数型プログラミング言語の型システムという複雑な応用対象に対して，圏論を用いた意味論の導入を通じて確固とした数学的基礎を樹立するものであり，同時にプログラム論理の新たな推論規則の導入によりいくつかの形式検証をはじめて可能にするものである．第４・５章の成果は，自動形式検証アルゴリズムの性能を向上させるために意味論的展開を活用するものである．これら２種の成果の相互活用も構想されており，今後理論・実践両面でのさらなる発展が見込まれる．よって，本論文の研究成果の理論・数学的価値及び応用・ソフトウェア科学的価値は双方とも高いと認められる．

以上の理由により審査委員会は，本学位論文が学位の授与に値すると判断した．