VŠB–Technická Univerzita Ostrava

Interaktivní a automatizované dokazování korektnosti programů

studijní opora

Zdeněk Sawa

Verze: 26. února 2023

Kapitola 1

Úvod

Předmět Interaktivní a automatizované dokazování korektnosti algoritmů v sobě kombinuje několik vzájemně provázaných témat — interaktivní a automatizované dokazování, sémantiku programovacích jazyků a dokazování korektnosti programů. Hlavním cílem se seznámit studenty s tím, jakým způsobem je možné pomocí softwarových nástrojů pro interaktivní a automatizované dokazování vytvářet formální počítačem ověřené důkazy korektnosti programů.

Existuje celá řada softwarových nástrojů, které umožňují interaktivně vytvářet důkazy, přičemž automaticky kontrolují správnost těchto důkazů a do určité míry umožňují automatizovat některé mechanické kroky důkazu. Typickým příkladem takového softwarového nástroje je systém Coq, což je nástroj, který bývá označován anglickým termínem ***proof assistent***. Kombinuje v sobě několik věcí. V prvé řadě je to nástroj, který umožňuje interaktivně vytvářet důkazy libovolných matematických tvrzení, přičemž se automaticky kontroluje správnost těchto důkazů. Zároveň je Coq funkcionálním jazykem s velmi mocným typovým systémem, který umožňuje implementovat a reprezentovat libovolné algoritmy, které je možné doplňovat o důkazy jejich korektnosti. Navíc je možné z kódu napsaného v Coqu automaticky vygenerovat odpovídající kód ve více konvenčních funkcionálních jazycích jako jsou OCaml, Haskell nebo Scheme. V neposlední řadě v sobě Coq obsahuje prostředky, které umožňují do značné míry vytváření rutinních částí důkazů automatizovat tak, aby je nebylo třeba vytvářet ručně, ale aby byly automatizovaně vytvářeny systémem.

První část semestru bude věnována primárně základům práce se systémem Coq — jednak programování v jazyce Coqu a jednak vytváření důkazů v Coqu. Coq pak bude v další části semestru využíván pro dokazování korektnosti programů. V této druhé části semestru budou probírána témata související s různými metodami dokazování této korektnosti.

Aby bylo vůbec možné nějaké formální důkazy korektnosti algoritmů vytvářet, musí být nejprve formálně nějakým způsobem reprezentována sémantika programů. Část přednášek proto bude věnovaná této problematice a také teorii typů, což je téma, které s výše uvedeným úzce souvisí. Dále pak budou studovány logiky používané pro dokazování vlastností programů — Hoarova logika a separační logika (což je modernější a obecnější rozšíření Hoarovy logiky o možnost dokazování korektnosti programů, které pracují s ukazateli a dynamicky alokovanými datovými strukturami).

Celý kurz je primárně postaven na následující sérii elektronických knih

> Benjamin C. Pierce et al. — *Software Foundations (volumes 1 – 6)*, Electronic textbook, 2022,

které jsou volně dostupné na webové adrese `https://softwarefoundations.cis.upenn.edu`.

V rámci kurzu budou probírána témata z následujících pěti knih v rámci této série — především ovšem z Volume 1 a 2 (témata, kterými se zabývají Volume 5 a 6 budou zmíněna spíše okrajově):

- ***Volume 1: Logical Foundations*** — která se zabývá základy funkcionálního programování v Coqu a základy vytváření důkazů, tj. jakým způsobem je Coqu reprezentována výroková a predikátová logika, predikáty, funkce, důkazy indukcí, apod.

- ***Volume 2: Programming Language Foundations*** — která se zabývá základy sémantiky programovacích jazyků, Hoarovou logikou a statickými typovými systémy, a tím, jak tyto věci reprezentovat v Coqu.
- ***Volume 3: Verified Functional Algorithms*** — která se zabývá konkrétními příklady toho, jak dokazovat korektnost algoritmů reprezentovaných funkcionálním způsobem v rámci jazyka Coqu. Tyto příklady zahrnují například třídící algoritmy (Insertion Sort, Merge Sort, ...), implementaci různých datových struktur (např. různé druhy vyhledávacích stromů, fronty, ...), barvení grafu, apod.
- ***Volume 5: Verifiable C*** — která se zabývá tím, jakým způsobem je možné všechny dříve popsané techniky dokazování korektnosti programů aplikovat na programy napsané v reálně používaných nízkoúrovňových imperativních jazycích, jako je jazyk C.
- ***Volume 6: Separation Logic Foundations*** — která se zabývá separační logikou.

Poznámka: Výše uvedené knihy jsou mimo jiné zajímavé též tím, že jsou samy napsány v podobě zdrojových kódů pro Coq. Z těchto zdrojových kódů jsou vygenerovány jednak texty knih ve formě HTML, které je možné číst, jednak je možné tyto zdrojové kódy otevřít v rámci Coqu a interaktivně provádět popisované příklady, případně je i upravovat. Příklady ve cvičeních v rámci těchto knih mají často podobu, kdy je nutné přímo do zdrojového kódu doplnit určitou část kódu a pomocí Coqu ověřit, že je toto řešení skutečně správné.

1.1 Osnova předmětu

Přednášky:

1. Úvod. Funkcionální programování v systému Coq.
2. Důkazy indukcí. Práce s datovými strukturami.
3. Polymorfismus a funkce vyššího řádu.
4. Základy logiky v systému Coq. Používání taktik v důkazech.
5. Induktivně definovaná tvrzení. Dokazování vlastností relací.
6. Formalizace syntaxe a sémantiky jednoduchého imperativního jazyka.
7. Strukturální operační sémantika. Ekvivalence programů.
8. Logika pro dokazování vlastností programů — Hoarova logika.
9. Separační logika.
10. Typové systémy. Dokazování vlastností typových systémů.
11. Dokazování vlastností funkcionálních programů.

Cvičení:

1. Seznámení se se systémem Coq.
2. Základy funkcionálního programování v Coqu.
3. Jednoduché důkazy v systému Coq.
4. Základy logiky v systému Coq. Používání taktik v důkazech.
5. Další techniky důkazů v Coqu.
6. Formalizace syntaxe a sémantiky jednoduchého imperativního jazyka.
7. Formalizace sémantiky jednoduchého imperativního jazyka v Coqu.
8. Dokazování některých vlastností programů.
9. Formalizace Hoarovy logiky v Coqu, důkaz korektnosti konkrétního jednoduchého algoritmu.
10. Další důkazy korektnosti algoritmů pomocí Hoarovy logiky.
11. Dokazování vlastností funkcionálních programů.

1.2 Přehled literatury

Kromě výše uvedené série knih ***Software Foundations*** bude náplň kurzu vycházet z následujících knih:

- Problematice práce s Coqem a vytváření důkazu v něm jsou věnovány knihy [1] a [2], přičemž zejména druhá z nich (ale částečně i první) se věnuje také speciálně použití Coqu pro dokazování korektnosti programů.
- Použití různých logik pro dokazování korektnosti programů a speciálně také použití Coqu pro tyto účely je věnována kniha [3].
- Problematikou teorie typů a jejího použití při definování sémantiky programů a dokazování jejich korektnosti se zabývají knihy [5] [6].
- Problematickou sémantiky programovacích jazyků se zabývají knihy [7] a [4].

Literatura

[1] Yves Bertot and Pierre Castéran. *Interactive Theorem Proving and Program Development — Coq'Art: The Calculus of Inductive Constructions*. Springer, 2004.

[2] Adam Chlipala. *Certified Programming with Dependent Types: A Pragmatic Introduction to the Coq Proof Assistant*. The MIT Press, 2013.

[3] Andrew W. Appel et al. *Program Logics for Certified Compilers*. Cambridge University Press, 2014.

[4] Hans Hüttel. *Transitions and Trees: An Introduction to Structural Operational Semantics*. Cambridge University Press, 2010.

[5] Benjamin C. Pierce. *Types and Programming Languages*. MIT Press, 2002.

[6] Benjamin C. Pierce, editor. *Advanced Topics in Types and Programming Languages*. MIT Press, 2004.

[7] Glynn Winskel. *Formal Semantics of Programming Languages*. The MIT Press, 1993.