

Posudek oponenta diplomové práce

Jméno a příjmení autora posudku: Mgr. Miroslav Kratochvíl

Jméno a příjmení autora práce: Bc. Bít Šefl

Název práce λ -calculus as a Tool for Metaprogramming in C++

Text posudku

Diplomová práce se věnuje návrhu jazyka, který by zjednodušil metaprogramování v jazyce C++-ových šablon. Autor poukazuje na fakt, že ačkoliv jsou šablony Turingovsky úplné a potenciálně by je bylo možné použít na mnoho zajímavých nových programovacích technik a konstrukcí, samotné programování v nich je obtížné — prakticky všechny konstrukce je potřeba zaobalit do typových a šablonových struktur C++, výsledný kód je často nečitelný a lehce se v něm ztratí původní význam.

V práci je zmíněno několik programů, které se pokoušejí různými způsoby řešit podobný problém. Nástroj s přístupem a vyjadřovací silou podobnou navrhovanému řešení zatím ale pravděpodobně neexistuje.

Autor se v práci nejdříve věnuje popisu šablonového systému C++ spolu se základními technikami metaprogramování a důkazem jeho úplnosti. Podobně pak popisuje λ -kalkulus — základní programovací techniky, vlastnosti jazyka a typové systémy včetně Hindley-Milnerova, který je později použit pro typovou kontrolu vytvořeného jazyka.

Další kapitoly pak postupně popisují metodu překlada λ -kalkulu do šablon, syntax nového jazyka, standardní knihovnu, interní zpracování jazyka kompilátorem a implementaci typového systému. Zajímavou součástí je popis (a odstranění) několika netriviálních omezení šablon jazyka C++, která autorovi mírně komplikují přímý překlad, např. implementace fixed-point kombinátoru, která je zkomplikovaná nutností použít pouze striktně pozitivní typy.

Softwarová část práce je napsaná v Haskellu (nový jazyk je pojmenován Norri) a je k ní poskytnuta dostatečná dokumentace jak v rámci práce, tak v samotném programu. Zdrojový kód je “tradičně funkcionálně nahuštěný”, přesto přehledný a dostatečně komentovaný. Jednoznačným pozitivem je schopnost programu poskytnout smysluplnou a dobře čitelnou typovou chybu před tím, než kód zpracuje kompilátor C++ (kterému v případě šablon tato vlastnost většinou zcela chybí). Program je stabilní, chyby v implementaci se mi najít nepodařilo. Kromě několika (málo) okrajových případů zmíněných autorem v práci se mi nepodařilo ani vymyslet způsob, jak kompilátor donutit vygenerovat kód, který by v C++ nebyl validní.

Spojení programování v C++ a Norri je demonstrováno několika jednoduchými programy dodanými jako příklady.

Autor v závěru poukazuje na některé nedostatky jazyka, většinou absenci funkcí a vlastností, které jsou již přítomné v jiných jazycích a praktický význam a metody jejich implementace jsou již známé. Trochu mi chyběl “krok do neznáma”, odkaz na nějaký problém současného programování v C++, který by bylo autorovým přístupem vhodné zkusit vyřešit. Bylo by tímto přístupem například možné generovat přímo kód jazyka C++, a tím definitivně (a včetně typové kontroly) nahradit některé případy, kde se dodnes používají makra? Nešlo by, potenciálně ve spojení s předchozím, nějakým praktickým způsobem využít novou “dvouúrovňovou” typovou kontrolu pro odvození většího množství statických vlastností kódu?

Celá práce je velmi pěkně vysázená, psaná kvalitní angličtinou, čte se dobře, všechny popsané úvahy jsou srozumitelné a jednoznačně a rychle pochopitelné. Mírnou útržkovitost částí textu vyskytujících se kolem několika příkladů kódu a popisu typových systémů lze přičíst značné komplexnosti tématu, celkový dojem z kvalitní práce nijak nenarušuje.

Doporučení k obhajobě

Z výše uvedených důvodů práci *doporučuji* k obhajobě.

Soutěž studentských prací

Vynikající práce vhodná soutěže studentských prací: **NE**.

V Praze dne 16. 8. 2016

Podpis: