Posudek na diplomovou práci

Překladač Java do C++

Marián Kamenišťák

Cílem práce je vytvořit překladač z jazyka Java v5 do jazyka ANSI C++. To může na první pohled pro nezasvěcené vypadat jako jednoduchý úkol, opak je však pravdou. Motivace je poměrně jasná, přes existující poměrně kvalitní JIT pro Javu poskytuje stále „úplný" překlad optimalizujícím překladačem do nativního kódu lepší výsledky co do rychlosti kódu.

Velmi zajímavá (snad nejzajímavější z celé práce) je kapitola druhá, která analyzuje jednotlivé konstrukce v jazyku Java a navrhuje možnosti pro efektivní překlad do C++. Hned po přečtení kapitoly 2.1, která pojednává o ukazatelích, bych doporučil autorovi podrobněji prostudovat možnosti nabízené v dokumentu „Technical Report on C++Library Extensions TR1". Zejména se mi jedná o knihovní šablonu `shared_ptr`, která by autorovi vyřešila velké množství problému včetně garbage collection. V současné době se již začíná toto rozšíření částečně objevovat u některých překladačů, zatím však není úplně běžně rozšířené. Další zajímavostí je poměrně důsledné se vyhýbání využití standardních knihoven C++ (dříve známé pod názvem STL). Pole řešené v kapitole 2.4 by téměř jistě šlo vyřešit pomocí `std::vector`.

Zajímavou třídou problémů jsou konstrukce, které prostě nejde nijak efektivně přeložit tak, aby zachovávaly plně sémantiku jazyka Java. Autor jich několik ve své analýze odhalil. Očekával bych, že tyto velmi problematické konstrukce budou popsány na jednom místě v textu diplomové práce, takže případný uživatel autorova díla si může snadno zjistit množinu konstrukcí, kterým se má vyhnout při svém programování. Dalším možným řešením je pak detekce takových konstrukcí v implementovaném překladači a jejich hlášení při překladu.

Ve třetí kapitole je rozebírán návrh překladače. Trošku mne překvapilo, že autor zvolil metodu vytvoření si vlastního parseru bez využití nějakých známých nástrojů. Je to, řekněme, neobvyklé.

Kapitola 4 pojednává o implementaci. Zpočátku je poměrně nezajímavá. Zajímavé to začíná být až od části 4.5, kde se hovoří o standardní knihovně Java. Je zřejmé, že není v silách jednoho člověka napsat potřebné ekvivalenty standardních knihoven Java v jazyce C++. Chápu tudíž, že autor vytvořil pouze skutečně velmi omezený výběr z knihovny, který mu umožňoval testovat svůj překladač. Pro překlad reálných projektů však bude nutné tuto standardní knihovnu dodělat. Dalším zajímavým bodem kapitoly je část 4.6 pojednávající o garbage collectoru. To je další velmi rozsáhlý úkol, který je opět asi nad rámec jedné diplomové práce. Zde by bylo nejspíše možné do budoucna využít nativní podpory garbage collectoru v novějších verzích různých překladačů.

V kapitole pět provádí autor porovnání s existujícími překladači Javy, aby experimentálně ověřil úspěšnost své implementace a oprávněnost motivace. Zdá se, že čísla jsou velmi lichotivá ve prospěch autorovy práce.

Trošku mne na textu práce mrzela občasná nepřesnost, např. v 2.3.1 se říká o velikosti typů pro celá čísla v C++: „...dlouhý typ musí být minimálně široký jako jednoduchý typ." Takových nepřesností je v textu více, nemá smysl je zde uvádět všechny.

Ačkoliv to vypadá jako velmi jednoduchý úkol, je to ve skutečnosti mnohem obtížnější cíl. Autor si s tímto cílem dle mého názoru poradil úspěšně. Doporučuji tuto diplomovou práci k obhajobě.

25.8.2006

RNI