FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Posudek oponenta závěrečné práce

Oponent práce:	Ing. Tomáš Pecka
Student:	Vojtěch Rozhoň
Název práce:	Debugger SECD virtuálního stroje
Obor / specializace:	Teoretická informatika
Vytvořeno dne:	6. června 2022

Hodnotící kritéria

1. Splnění zadání

- ▸ [1] **zadání splněno**
  [2] zadání splněno s menšími výhradami
  [3] zadání splněno s většími výhradami
  [4] zadání nesplněno

Zadání bylo podle mě splněno.

2. Písemná část práce 90 /100 (A)

Text práce je v anglickém jazyce, jazykové chyby se vyskytují jen občasně. Jednotlivé části jsou informačně bohaté a logicky navazují. Textová část je na velmi dobré úrovni, dobře popisuje průbeh výpočtu SECD strojem.

Obrázek 2.1 podle reprezentace definované v [7, sekce 3.1.5] reprezentuje spíše výraz `( (*) ( (* 1 2) ) (3) )` a ne `(* (+ 1 2) 2)` (zaměněná dvojka a trojka na posledním místě se opravdu vyskytuje v textu). V sekci 4.2.1 by bylo vhodné explicitněji zmínit jak se do překladu zapojuje syntaktická analýza a vůbec lépe projít celý postup překladu.

V sekci 5.1.1.3 je zmíněno, že výstupem kompilátoru (ve třídě pojmenované Parser?) je SECD bytekód a také AST. Myslím. že vhodnější by bylo rozdělit to na generování AST a z něj poté přeložit bytekód. V sekci 5.1.2.1, tj. implementace funkce gensym, je zmíněno, že gensym funkce je překládána na instrukci "LD(-10, -10)", což mi přijde jako hack, který bude zbytečně zvyšovat komplexitu i čitelnost kódu a bude matoucí pro uživatele.

Po typografické stránce je prohřešků již více, nikoliv však zásadních. U plovoucích prostředí s ukázkami kódu není u popisku typ (Tj. např. místo "Code listing 2 A source code" se vyskytuje jen "2 A source code"). Vyskytuje se nekonzistentní číslování ukázek kódu (jeden číselník pro celou práci) a obrázků (číselníky podle kapitol). Některé delší ukázky kódu by si zasložily plovoucí prostředí. Citace jsou používány nekonzistentně, občas se [1] vyskytne za koncem věty, občas před.

Další poznámky:
- str. 3: Konstanty #f a #t by mohly být vysázeny v tt prostředí;
- str. 8: Jazyk se jmenuje "Racket", ne "racket";
- str. 9: Proč není kód uprostřed strany v nějakém plovoucím prostředí? To samé na konci strany 10;
- str. 9: Myslím, že řetězce "!ff! a "!gg" by si také zasloužily jinou typografii (tt?);
- str. 11: lepší použít \cite{a,b} [7,8] než \cite{a}, \cite{b} [7], [8];
- str. 12: Zvláštně posazená reference na zdroj [7] zhruba uprostřed strany;
- str. 14: Za názvem odstavce "Operators" rovnou následuje text, což je nekonzistentní oproti předchozím odstavcům. Dodal bych, po vzoru předchozích odstavců, operace s registry a seznam operátorů, které sem spadají;
- str. 27: Je zmiňována nedefinovaná instrukce DUMMY;
- str. 31-34: Mezi dvěma obrázky vedle sebe na řádku (Fig 5.8, 5.9, 5.10, 5.11) by měla být mezera, jejich popisky (caption) se vizuálně slévají dohromady.

3. Nepísemná část, přílohy 85 /100 (B)

Hodnotím kód v core repozitáři (github lambdulus/tiny-lisp-core ref 2860a0e82e46cae25f7aaefc1379431ccf8f3a74) a frontend repozitáři (github lambdulus/frontend branch tiny-lisp, ref 424e9e7989dca0c9276bcc013eb7dede306849b1).

Mnou testované základní příklady bez problémů (až na výjimky níže) proběhly a dávaly očekávané výsledky. Vizualizace SECD instrukcí a registrů mi přijde pochopitelná. Z tohoto hlediska nemám co vytknout. Kdyby autor dotáhl frontendovou část a kvalita kódu by byla trochu vyšší, bylo by o stupeň vyšší i moje hodnocení.

Integraci do frontendu lambdulu vnímám jako netriviální, ale řešitelnou, úlohu. Bohužel pri používání je vidět, že integrace není zdaleka dokonalá. Tlačítko editace kódu nefunguje, což dělá aplikaci poněkud těžkopádnou na používání, některé autorovy testy z core repozitáře při vyhodnocení z prohlížeče spadnou, např. test s makrem `if-consp`. Očekával bych v odevzdaném díle lepší testování. V kódu core repozitáře se vyskytuje jen 23 testů typu "přelož tento kód a ověř výsledek". Průběh vyhodnocení ve VM se nikde automaticky netestuje.

I v nejzákladnějším code linteru vidím četná upozornění na nepoužité importy a deklarované nepoužité proměnné. Některé části kódu jsou hůře čitelné (např. parser). V historii ve verzovacím systému se stále opakují nic neříkající commit messages typu "refactor", "a lot of changes", "improvements", bez jakéhokoliv dalšího komentáře. Je tak velmi obtížné sledovat postupný vývoj za účelem porozumění kódu.

Další poznámky:
* Kód `(cons 1 null)` je asi špatně přeložen. Lambdulus ho reportuje jako `(cons 1 1))`.
* Kód `(begin 1 2)` nejde zparsovat, ale `(begin (+ 1 1) (+ 2 2))` ano.
* Zvolený npm stack v core repozitáři neobsahuje v package.json závislosti pro `npm install` a ani po jejich ruční instalaci příkaz `npm run test` nefunguje. Vzhledem k tomu, že nebyla přiložena žádná příručka, očekával bych, že bude fungovat standardní postup. Pro spuštění testů jsem musel udělat manuální zásah do kódu.

4. Hodnocení výsledků, jejich využitelnost 88 /100 (B)

Nasazení projektu v systému lambdulus je v budoucnu rozhodně reálné a cílí zejména na výuku funkcionálních jazyků (BI-PPA). Před reálným nasazením a využitím v PPA je ale

ještě potřeba zapracovat na integraci do Lambdulu. Uvítal bych, kdyby toto autor dotáhl, je škoda, že se to nepodařilo ještě teď. Pokud se to podaří, myslím, že je šance používat tuto implementaci jazyka k automatickému vyhodnocování studentských úkolů či v kombinaci s frontendem jako vývojové prostředí u zkoušek.

Celkové hodnocení 89 /100 (B)

Autor odvedl dobrou práci. Textová část práce je obsahově nadprůměrná. Výsledné SW dílo má v aktuální podobě bohužel ještě mouchy, což nejvíce ovlivnilo moje hodnocení, když jsem se rozhodoval mezi hodnocením známkou A a B. Před reálným použitím ve výuce je na projektu potřeba ještě trochu zapracovat, ale doufám, že se dílo podaří dotáhnout.

Hodnotím 89 body (B, velmi dobře).

Otázky k obhajobě

Jaký je důvod k překladu volání gensym na instrukci LD(-10, -10)?

Instrukce

Splnění zadání

Posuďte, zda předložená ZP dostatečně a v souladu se zadáním obsahově vymezuje cíle, správně je formuluje a v dostatečné kvalitě naplňuje. V komentáři uveďte body zadání, které nebyly splněny, posuďte závažnost, dopady a případně i příčiny jednotlivých nedostatků. Pokud zadání svou náročností vybočuje ze standardů pro daný typ práce nebo student případně vypracoval ZP nad rámec zadání, popište, jak se to projevilo na požadované kvalitě splnění zadání a jakým způsobem toto ovlivnilo výsledné hodnocení.

Písemná část práce

Zhodnoťte přiměřenost rozsahu předložené ZP vzhledem k obsahu, tj. zda všechny části ZP jsou informačně bohaté a ZP neobsahuje zbytečné části. Dále posuďte, zda předložená ZP je po věcné stránce v pořádku, případně vyskytují-li se v práci věcné chyby nebo nepřesnosti.

Zhodnoťte dále logickou strukturu ZP, návaznosti jednotlivých kapitol a pochopitelnost textu pro čtenáře. Posuďte správnost používání formálních zápisů obsažených v práci. Posuďte typografickou a jazykovou stránku ZP, viz Směrnice děkana č. 52/2021, článek 3.

Posuďte, zda student využil a správně citoval relevantní zdroje. Ověřte, zda jsou všechny převzaté prvky řádně odlišeny od vlastních výsledků, zda nedošlo k porušení citační etiky a zda jsou bibliografické citace úplné a v souladu s citačními zvyklostmi a normami. Zhodnoťte, zda převzatý software a jiná autorská díla, byly v ZP použity v souladu s licenčními podmínkami.

Nepísemná část, přílohy

Dle charakteru práce se případně vyjádřete k nepísemné části ZP. Například: SW dílo – kvalita vytvořeného programu a vhodnost a přiměřenost technologií, které byly využité od vývoje až po nasazení. HW – funkční vzorek – použité technologie a nástroje, Výzkumná a experimentální práce – opakovatelnost experimentů.

Hodnocení výsledků, jejich využitelnost

Dle charakteru práce zhodnoťte možnosti nasazení výsledků práce v praxi nebo uveďte, zda výsledky ZP rozšiřují již publikované známé výsledky nebo přinášející zcela nové poznatky.

Celkové hodnocení

Shrňte stránky ZP, které nejvíce ovlivnily Vaše celkové hodnocení. Celkové hodnocení nemusí být aritmetickým průměrem či jinou hodnotou vypočtenou z hodnocení v předchozích jednotlivých kritériích. Obecně platí, že bezvadně splněné zadání je hodnoceno klasifikačním stupněm A.