OPONENTNÍ POSUDEK ZÁVĚREČNÉ KVALIFIKAČNÍ PRÁCE

Autor závěrečné práce: Bc. Victor Trnka

Název práce: Akcelerace vyhodnocování výrazů pomocí vektorových instrukcí

Oponent práce: Mgr. Lukáš Bajer, PhD.
Pracoviště oponenta: Cisco Systems

A. Úplnost abstraktu, klíčová slova odpovídají náplni práce ..... Výborně (1)

B. Kvalita zpracování rešerše ..... Velmi dobře (2)

C. Řešení práce po teoretické stránce ..... Velmi dobře (2)

D. Vhodnost, přiměřenost použité metodiky ..... Výborně (1)

E. Úroveň zpracování výsledků a diskuse ..... Velmi dobře minus (2–)

F. Vlastní přínos k řešené problematice ..... Výborně minus (1–)

G. Formulace závěru práce ..... Velmi dobře (2)

H. Splnění zadání (cílů) práce ..... Splněno

I. Skladba, správnost a úplnost citací literárních údajů ..... Výborně (1)

J. Typografická a jazyková úroveň (vč. pravopisu) ..... Výborně minus (1–)

K. Formální náležitosti práce ..... Výborně (1)
(struktura textu, řazení kapitol, přehlednost ilustrací)

Komentáře či připomínky:

Diplomová práce se zabývá využitím vektorových instrukcí moderních procesorů pro vyhodnocování aritmetických výrazů v rámci již existujícího vyhodnocovače BParser. Student v ní navrhnul, implementoval a otestoval jedno zvolené řešení použití vektorizační C++ knihovny. Současný vyhodnocovač výrazů tedy rozšířil. Pozitivně hodnotím, že vedoucí práce již navržené řešení do hlavní vývojové větve začlenil.

Text práce je napsán čtivě a pouze s malým množství překlepů. Autor zvolil až učebnicový či vyprávěcí jazyk, což je někdy až na úkor detailů nebo včasných definic pojmů, které bych v práci v některých částech uvítal (např. definice BParseru v úvodu, nebo zavedení termínu "knihovna" (nejčastěji vektorová)).

... pokračuje na straně 2

Celkové zhodnocení:

Výběr použité vektorizační C++ knihovny je dobře zdůvodněn. Otestování (rychlosti) i některé další knihovny by bylo zajímavé plus.

Třetí kapitola popisuje navržené změny v kódu BParseru. V této části bych uvítal určitý nadhled, např., že pro vytváření Processoru je v podstatě použitý návrhový vzor Factory method, nebo informaci, jak bylo ke změnám v knihovně přistupováno (co největší zachování funkčnosti, nebo kódu, nebo něčeho jiného?).

Z hlediska implementace bych uvítal důslednější odlišování různých druhů testů, zejména unit testů a skriptů na měření rychlosti. V unit testech bych pak omezil přímý výpis na obrazovku na nutné minimum, a zaměřil se na automatickou testovatelnost a následnou udržitelnost kódu, který těmito testy je pokryt. Pozitivně hodnotím použití C++ šablon.

Poslední kapitola popisuje a srovnává navržené řešení s dosavadním stavem a s referenční implementací. U grafů bych ocenil namísto barevných sloupců nějakou formu box plotů značících také odchylku jednotlivých měření. Dále by bylo zajimavé nejen relativní srovnání rychlostí, ale i numerické průměrné časy na jedno vyhodnocení v různých konfiguracích.

Celkově práci hodnotím kladně, je za ní vidět studetnův přínos a doporučuji ji k obhajobě.

Otázky k obhajobě:

1. Používala varianta "čistý výpočet v C++" nějakou vektorizaci? Jaké optimalizace (např.) dělá překladač?

2. Máte nějakou hypotézu, proč vychází délka vektoru 256 bitů jako nejrychlejší?

3. Dalo by se zjistit přesněji, jaká část měřených a reportovaných časů byla skutečně strávená při zpracování výrazu, a koli času byla ostatní režie (načtení binárního souboru, inicializace apod.)?

4. Proč jsou výsledky na grafech 4.7 a 4.10 odlišné (rychlejší, resp. pomalejší C++)?

Celková klasifikace a doporučení k obhajobě:

Práce splňuje požadavky na udělení akademického titulu, a proto ji doporučuji k obhajobě
Navrhuji tuto práci klasifikovat stupněm: Výborně minus (1–)

Podpisem současně potvrzuji, že nejsem v žádném osobním vztahu k autorovi práce

V Praze dne 27. ledna 2023 .....
podpis oponenta práce