Posudek oponenta diplomové práce

Název práce: **Mikrobenchmarky Javy**

Autor práce: **Marek Rojík**

Práce Marka Rojíka se zaobírá vytvořením nástroje pro uživatelské spouštění výkonnostních testů malých fragmentů kódu, tzv. mikrobenchmarků, napsaných v jazyce Java. V textu práce se autor v prvních dvou kapitolách zabývá problematikou měření a benchmarkování výkonu software obecně, s důrazem na specifika platformy Java (mj. efekty just-in-time kompilace a správy paměti). Následně je z více kandidátů vybrán rámec pro realizaci měřicí infrastruktury, Java Microbenchmark Harness. Druhá polovina práce je věnována popisu servisně orientované aplikace s webovým rozhraním, která uvedený nástroj implementuje. Součástí je také vyhodnocení testů aplikace, ověřujících její správnou funkčnost. V přílohách se nachází ukázky, technické podrobnosti implementace a popis sestavení aplikace. Rozsah práce a příloh je odpovídající, po formální stránce je text také v pořádku, snad až na příliš časté použití výrazu „zanalyzovat".

Velkým kladem práce je kvalita architektury a realizace aplikace, včetně velmi vhodně zvolených a kvalifikovaně použitých moderních technologií – mj. Spring Boot a Swagger pro REST službu, Docker pro spouštění a izolaci (potenciálně nebezpečných) testovaných fragmentů, node.js/TypeScript pro webovou aplikaci. Jejich výběr je v práci dobře zdůvodněn, implementace sama je přehledná a vcelku dobře komentovaná. Výsledná aplikace je přeložitelná v podstatě podle návodu a funkční.

Celkovou úroveň práce bohužel snižuje analytická část a slabá práce se zdroji (autor používá jen minimum autoritativních, zejm. recenzovaných zdrojů). Pojmy a koncepty týkající se měření a benchmarkování v kapitole 2 jsou definovány spíše volně, přestože jejich přesné chápání a použití je důležité pro vlastní práci. Kromě dobře popsaných specifik platformy Java nejsou dostatečně rozebrány parametry prostředí a okolnosti běhu výkonnostního testu, které mohou výrazně ovlivnit získané hodnoty; jediný aspekt (souběžné provádění více benchmarků) je řešen až v rámci testování aplikace, str. 63. Mezi výraznější věcné nepřesnosti patří rozpor mezi definicí „hot" kódu (str. 20) a tvrzením, že kompilátor může jedenkrát volanou metodu označit pro optimalizaci (str. 21).

V realizační části není vůbec diskutováno, jak pracovat s fragmenty kódu vyžadujícími knihovny neobsažené v distribuci Java platformy (resp. v mapě použité analyzátorem importů). Připravená webová aplikace tak umožňuje testovat pouze fragmenty využívající standardní knihovny zahrnuté v JDK/JRE, což velmi silně omezuje praktickou použitelnost nástroje. Ověření aplikace je v diplomové práci ukázáno jen na triviálním příkladu, mělo být provedeno na více fragmentech různých typů. V práci také chybí aspoň základní metodika tvorby mikrobenchmarků, která by uživatele nástroje vedla při vytváření fragmentů kódu a testovacích dat tak, aby získané výsledky byly validní a správně interpretovatelné.

Západočeská univerzita v Plzni
Fakulta aplikovaných věd
katedra informatiky a výpočetní techniky

Celkově konstatuji, že práce splňuje všechny body zadání. Slabší úroveň části týkající se fundamentů a analytické práce je vyvážena dosti kvalitní realizací výsledného nástroje. Navrhuji hodnocení známkou **velmi dobře** a práci doporučuji k obhajobě.

Dotazy k práci:

1. Jaký je rozdíl mezi testem výkonnosti a benchmarkem?
2. V práci detailně rozebíráte potřebu „warm-up" fáze benchmarkování v případě testování na platformě Java. Je tato fáze vždy nutná; tedy, může být validní ji záměrně vynechat, a případně za jakých okolností?
3. V práci je řešen de facto jediný cíl benchmarkování a související metriky, a to „wall clock" čas resp. související propustnost. Jaké jiné parametry systému či fragmentu kódu je možné měřit?

V Plzni, 3.6.2019

doc. Ing. Přemysl Brada, MSc. Ph.D.

Západočeská univerzita v Plzni
Fakulta aplikovaných věd
katedra informatiky a výpočetní techniky

SOUHLASÍ S ORIGINÁLEM