FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Posudek oponenta závěrečné práce

Oponent práce:	Mgr. Jan Starý, Ph.D.
Student:	Bc. Martin Čapek
Název práce:	DPLL(MAPF): Integrace řešiče SATu a multi-agetního hledání cest
Obor / specializace:	Znalostní inženýrství
Vytvořeno dne:	28. srpna 2021

Hodnotící kritéria

1. Splnění zadání

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

Implementace, tj. propojení DPLL(T) s existujícím SAT solverem Glucose, není zdá se provedeno zamýšleným způsobem.
("We did not manage to integrate our code with Glucose 4 on such level that DPLL(T) proposes.")

2. Písemná část práce 70 / 100 (C)

Kapitola 1 ("Backround") popisuje teorii. Příklad algoritmu SATPlan (str. 13) je zdá se chybně: ve formuli 3 je disjunkce vpravo splněna (neboť At(T,A)^0 je splněno),
tedy i At(T,A)^1 vlevo má být splněno; přitom formule 3 níže při naznačeném ohodnocení zjevně neplatí. Ve formuli 4 z ničeho neplyne, že by At(T,B)^0 neplatilo,
neboť podmínky nikde neříkají, že ve stejném čase na stejném místě nemohou stát dva.
Poněkud zmatečně působí věta "the number of clauses scale quite well",
čímž se zřejmě myslí exponenciální růst výpočetní náročnosti.

Kapitola 2 popisuje hlavní přínos, totiž _průběžné_ ověřování dosavadního ohodnocení vzešlého ze SAT solveru proti čím dál úplnějšímu fragmentu celé teorie (narozdíl od jednorázového předání celého zadání solveru jakožto blackboxu). Související terminologie je zavedena trochu pochybně: podle definice 1 (str. 14) a definice 4 (str. 22) je každý úplný model neúplný. Na str. 28 se diskutují zásadní problémy takové integrace: stávající solver je navržen pro práci s ohodnocením všech proměnných,
nyní musíme dekódovat i částečné, a zároveň již v něm odhalovat případné kolize. Po konstatování "it probably cannot do that" a "might not be designed for the plan created by

partial assignment" se tyto zásadní překážky spláchnou větou "These new components doesn't break the optimality and completeness". Příklady 1 a 2 na str. 29 dole jsou zdá se prohozené oproti tomu, co se říká v textu (lazy vs eager).

Kapitola 3 popisuje prototyp. Podle pseudokódu jde (řečeno shora) o přidání collisions.clear() do hlavní metody solve() a speciální booleovský stav I_Reset, který znamená "v dosavadním řešení jsou kolize". Kapitola končí nejpochybnější pasáží celé práce: "we did not manage to integrate our code with Glucose on such level that DPLL(T) proposes [...] we were not able to add clauses inside the solve::check_collisions() method without jumping out of solve() ... when conflicts occur some reset needs to be done ... and when we tried that inside solve() it broke". To vzbuzuje pochybnosti o vytvořené implementaci vůbec. Závěr je nicméně "our implementation is close enough", aniž bychom se dozvěděli, v čem ten který problém spočíval: "it broke", ale stále "close enough".

Kapitola 4 popisuje provedené experimenty. Trochu alibisticky působí věta "maps are quite small, this is because our program should perform well on smaller maps"
- otázka je samozřejmě opačná, totiž jak si poradíme s velkými zadáními. Součástí testů je odložení kontroly kolizí (parametry uniform a exponential): místo po každém novém přiřazení (protože každé může a priori způsobit novou kolizi) se kontroluje v desetině, čtvrtině atd volitelně; jako důvod se uvádí "it would be too expensive" bez dalších detailů (jak moc víc "expensive"?) U jednotlivých zadání najdeme časová zlepšení i zhoršení pro rostoucí i klesající hodnoty těchto parametrů.

Tabulka 4.3 uvádí průměrné časy deseti běhů (na různých zadáních) v milisekundách.
S jakou přesností měříme hodnoty v desítkách milisekund a jaká je chyba takového měření?
Ve všech případech vybočuje svou náročností úloha s 20 agenty, což se nijak nediskutuje.
Obrázek 4.3 zobrazuje na třetině stránky prázdný grid.

Práce je psána dosti chabou angličtinou, s chybami na úrovni od spellingu ("standart", "we now exactly") přes lexikologii ("continuous" místo "continues", "as we spoke earlier") morfologii (chybějící členy, "formula represent a solution", "on place A", "eagerly adding of chokepoints") a syntax ("Such variable can be assign truth value that literal is true", "Planning have representation of goal", chybějící podměty, ...) po stylistiku ("to show _you_ how to encode").

3. Nepísemná část, přílohy 70/100 (C)

Jádrem implementace je patřičně pozměněná metoda Solve() stávajícího SAT solveru Glucose,
a související práce s odhalováním kolizí v dosavadním (částečném) řešení. Není bohužel jasně vyznačeno,
jaký objem práce to představuje na úrovni kódu (i když lecos lze vyčíst z diffu vůči tarballu Glucose4).

Podle poznámky na str. 16 dole lze neřešitelnost úlohy zjistit v polynomiálním čase - implementace to ale zdá se neprovádí,
což se zjistí na dvoubodovém grafu, kdy se navždy marně hledá řešení.

Na řešitelných vzorových příkladech implementace prokazatelně funguje.

4. Hodnocení výsledků, jejich využitelnost 70 /100 (C)

O praktickém využitím vylepšeného MAPF solveru není pochyb,
předložená implementace ale vzbuzuje pochybnosti.

Celkové hodnocení 70 /100 (C)

Téma je dosti náročné, ba na špici poznání. Teoretická příprava musela být poměrně náročná, implementace rovněž.
Přes výše uvedené výhrady hodnotím tedy nakonec práci průměrně.

Otázky k obhajobě

1. Kterému underlying grafu odpovídá MDD na straně 31?
2. Odkud pocházejí navržené hodnoty 3, 5, 10, resp. 2, 1.6, 1.4? Jak přesně je oproti tomu drahá kontrola po každé proměnné?
3. V čem je testovaná úloha s 20 agenty skokově náročnější než 18 agentů, ale snazší než 24 agentů (ve všech případech)?
4. Proč testujete jen běhy trvající zlomky sekundy místo mnohem delších, kde by se minimalizovala chyba měření?

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.