DIVINE 4

Vladimír Štill

Masarykova univerzita

Brno, Česká republika

16th December 2016

Shrnutí prvního roku PhD

Únor až duben

- nástup na PhD, dokončování publikace o podpoře relaxovaných paměťových modelů pro DIVINE 3
  - nakonec nepublikováno z důvodu nalezení chyby v implementaci, odloženo
- prezentace DIVINE 3 na SV-COMP/TACAS 2016 v Eindhovenu
- práce na kompilátoru pro DIVINE
  - integrace clangu
  - robustnější prostředí pro kompilaci
  - nezávislost na systémovém kompilátoru a jeho nastavení
  - nezávislost na systémových hlavičkových souborech

Květen až červenec

- v květnu proběhl první DIVINE sprint, vznikla první interně použitelná verze DIVINE 4
- dokončování kompilátoru
- instrumentace LLVM pro DIVINE 4
- metadata pro potřeby programu/DiOS-u (DIVINE Operating System – poskytuje podporu pro vlákna, část POSIX prostředí verifikovanému programu)
- podpora C/C++ knihoven pro DIVINE 4

Srpen až říjen

- implementace výjimek a `setjmp`/`longjmp` pro DIVINE 4
  - robustnější podpora výjimek založená na vlastním unwinderu
  - funguje bez modifikací C++ knihovny
- statické redukce stavového prostoru na základě detekce lokálních proměnných
- podíl na přípravě článku o architektuře DiVM
  - Ročkai, Mrázek, Štill, Barnat: *A Virtual Machine with Graph-Organised Memory for Model Checking*, odesláno na TACAS

Listopad a prosinec

- refaktoring a optimalizace podpory vláken (`pthreads`)
  - optimalizace Thread Local Storage
  - robustnější kontrola chyb
  - využití symetrie vláken k redukci stavového prostoru
- testy, větší množství oprav
  - testy knihovny `bricks`, kterou DIVINE využívá, částečně verifikovány DIVINE
  - díky tomu identifikace a částečné doplnění chybějící funkcionality
- příprava publikace: Štill, Ročkai, Mrázek, Barnat: *Verifying Exception-Enabled C++ Using Unmodified Standard Library*, odesláno na NASA Formal Methods

Další novinky v DIVINE 4

Vylepšení jádra DIVINE (DiVM)

- vylepšené redukce stavového prostoru
  - sledování, které objekty jsou sdílené mezi vlákny
  - zápis do soukromých objektů nemusí vyvolat přerušení
  - podobná redukce, ale menší režie než DIVINE 3
- lepší reprezentace haldy
  - úspora paměti
  - rychlejší srovnávání stavů

Vylepšení interaktivního debuggeru

- (dostupný jako `divine sim`)
- lepší podpora pro debugovací informace
- mnoho opravených chyb
- breakpointy podle jména souboru a čísla řádku
- automatické testy

Vylepšení DiOS

- DiOS – jednoduchý operační systém uvnitř DIVINE, který poskytuje část POSIX rozhraní verifikovanému programu
- předávání argumentů programu (argumenty funkce `main`, `environ`,...)
- konfigurace chování chybových stavů, simulace chyb
  - lze deaktivovat některé kontroly
  - nastavitelné chování `malloc`, ...
- přepracovaný systém systémových volání

Virtuální souborový systém v DIVINE 4

- DIVINE 4 nyní podporuje základní funkce pro práci se souborovým systémem
- lze zachytit část souborového systému, vstup
- výstup (`printf`/`write`...) zaznamenáván, součástí případného protipříkladu

Symbolizace a abstrakce programu

- cílem je verifikace programů se vstupy
- rozmyšlena reprezentace stavu symbolizovaného programu a reprezentace symbolických hodnot
- algoritmus pro transformaci programu momentálně zvládá transformovat programy s celočíselnými proměnnými
- připraveno jako základ nejen pro přesnou symbolickou reprezentaci, ale také pro abstrakci

Aktuální stav DIVINE 4

- první beta verze 1. 12., druhá 14. 12.
- RC do Vánoc, finální v lednu
  - v plánu dát k dispozici virtuální stroj s předinstalovaným DIVINE
- podán návrh na prezentaci a workshop na DevConf, obojí ale zamítnuto
- od května v průměru > 200 patchů/měsíc
- 7 vývojářů
- rozšířená dokumentace na webu

Pohled do roku 2017

Podpora POSIX systémových volání

- cílem je vytvořit režim spuštění programu v DIVINE, který umožní volat systémová volání Linuxu, na kterém DIVINE běží
  - možné pouze v run módu
  - nyní DIVINE některá systémová volání simuluje
- umožní využít striktních kontrol DiVM i pro programy, které interagují se sítí, čtou větší množství dat z filesystému, ...

Podpora procesů

- přidání podpory pro `fork`
- možnost načtení více programů do DIVINE, `exec`
- meziprocesová komunikace, ochrana paměti

Verifikace programů se vstupem

- dokončení základní implementace
  - nutné úpravy v DiVM
  - nový verifikační algoritmus
- postupné rozšiřování nad rámec symbolické verifikace podporované SymDIVINE
  - floaty, dynamická paměť
  - možná abstrakce/over-aproximace

Redukce stavového prostoru

- optimalizace současných redukcí
- komprese stavového prostoru
- optimalizace reprezentace rámce funkce
- využití detekce ukončení cyklu k odstranění některých přerušení

Monitory, LTL

- podpora složitějších vlastností, například:
  - globální asserty
  - „nenastane chyba a program se ukončí“
- vlastnosti zapsané pomocí monitoru v C(++), který sleduje chování programu, nebo pomocí LTL
- související verifikační algoritmy