

Statička analiza primenom apstraktne interpretacije i programskog okvira IKOS

Seminarski rad u okviru kursa

Verifikacija softvera

Matematički fakultet

Ozren Demonja

mi12319@alas.matf.bg.ac.rs

24. maj 2018.

Sažetak

RTCA standardi za razvijanje softverskih sistema u avioindustriji opisuju na koji način se statička analiza koristi za utvrđivanje ispravnosti. U ovom radu dat je pregled programskog okvira IKOS, koji vrši preciznu i skalabilnu statičku analizu. IKOS koristi moć tehnike apstraktne interpretacije. Prednost ovog pristupa demonstrirana je analizom prekoračenja bafera u softverskim sistemima.

Sadržaj

1	Uvod	2
2	Programski okvir	2
3	Programski okvir ARBOS	2
4	LLVM i AR medureprezentacija koda	3
5	Biblioteka IKOS	3
6	Analiza prekoračenja bafera	3
7	Slični radovi	4
8	Zaključak	4
	Literatura	4

1 Uvod

Cilj rada je upotreba statičke analize za utvrđivanje ispravnosti softverskih sistema u avioindustriji. Nažalost, danas postoji mali broj alata za statičku analizu koji su zasnovani na apstraktnoj interpretaciji. Šta više, ovi alati su uglavnom ograničeni na utvrđivanje ispravnosti programa pisanih u programskom jeziku C. Cilj je implementacija programskog okvira (engl. *framework*) koji će biti korišćen za statičku analizu softverskih sistema u avioindustriji primenom tehnike apstraktna interpretacija.

Apstraktna interpretacija[4] predstavlja metodu statičke analize u kojoj se semantika programa opisuje matematičkim modelom mogućih ponašanja programa. Ponašanje programa opisuje se konkretnim domenom i relacijama nad njim. Potencijalni problem ovog pristupa javlja se u slučajevima jako velikih domena i tada se vrši aproksimacija konačnog domena apstraktnim. Aproksimacijom domena mogu se izgubiti važne informacije. Iz tog razlga, posebna pažnja se posvećuje biranju adekvatne aproksimacije.

S obzirom da apstraktna interpretacija primenjuje različite aproksimacije prilikom pravljenja modela programa, sistemi za ispitivanje ispravnosti mogu dovesti do pojave lažnih upozorenja (engl. *false positives*). Lažno upozorenje je izveštaj da program ima grešku koja zapravo ne postoji. Zbog pojave lažnih upozorenja, potrebno je proveriti njihovu validnost.

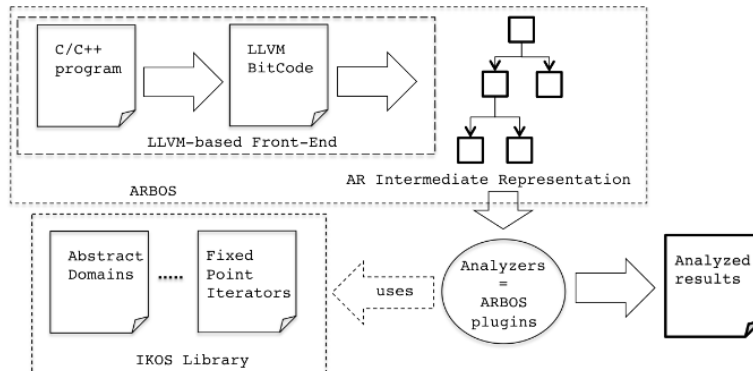
IKOS [2] (engl. *Inference Kernel for Open Static Analyzers*) je javno dostupan programski okvir razvijen u istraživačkom centru NASA sa ciljem precizne i skalabilne statičke analize. IKOS primenjuje koncepte apstraktna interpretacije. Pogodan je za ispitivanje ispravnosti softverskih sistema u avioindustriji pošto najveću pažnju poklanja analizi grešaka prekoračenja bafera koje su posebno važne za detektovanje.

2 Programski okvir

IKOS programski okvir, prikazan na slici 1, se u osnovi oslanja na ARBOS programski okvir za parsiranje izvornog koda, izvođenje semantičke rezolucije i prevođenje u AR međureprezentaciju koja je pogodnija za dalju analizu. Statička analiza korišćenjem apstraktna interpretacije ostvaruje se upotrebom IKOS biblioteke koja sadrži kolekciju apstraktnih domena i iterativnih algoritama. Rezultati analize predstavljaju potencijalno nebezbedne ulazne vrednosti za koje je potrebno proveriti da li zaista dovode do softverske greške. Dobijeni rezultati se čuvaju u skladištima u formi tekstualnih fajlova ili SQL bazi podataka. SQL baza podataka je pogodna za izdvajanje specifičnih informacija iz rezultata. Takođe, možemo vizualizovati rezultate korišćenjem alata IkoView, gde se bezbedne vrednosti označavaju zelenom bojom, nebezbedne vrednosti crvenom bojom a upozorenja žutom bojom.

3 Programski okvir ARBOS

ARBOS je programski okvir koji vrši statičku analizu koristeći AR međureprezentaciju. Trenutno, ARBOS vrši prevođenje programa napisanih u programskim jezicima C i C++ u međukod. Na slici 1 prikazane su faze prevođenja programa pisanih u jezicima C i C++ koje je potrebno analizirati transformacijom u AR međukod.



Slika 1: Arhitektura programskog okvira IKOS

4 LLVM i AR međureprezentacija koda

Analiza programa pisanih u višim programskim jezicima dodatno je otežana brojnim faktorima, kao što su konverzije tipova i bočni efekti. Zbog toga se analiza programa obično izvodi nad kodom dobijenim transformacijom izvornog koda u oblik koji je pogodan za analizu. Kao alat za transformaciju koriste se LLVM, koji operise nad bitkodom generisanim specijalnom verzijom GCC kompajlera.

Osnovu LLVM-a čine biblioteke koje obezbeđuju razne vrste transformacija i analize programa. Ove biblioteke rade nad jezikom koji se zove LLVM međujezik (engl. *LLVM Intermediate Representation*).

LLVM međujezik ima jednostavne instrukcije koje su slične asembler-skim. Za razliku od asembler-skih instrukcija, LLVM instrukcije sadrže informacije o tipovima podataka operanada instrukcija kao i o njihovoj upotrebi u programu.

AR međukod generisan od strane ARBOS programskog okvira na drugačiji način izražava semantiku C programa u poređenju sa LLVM međukodom. U okviru AR reprezentacije, kontrolni tok se izražava na deklarativni način koristeći nedeterminističke izbore i ograničenja.

5 Biblioteka IKOS

IKOS je razvojna platforma za statičku analizu zasnovana na apstraktnoj interpretaciji. IKOS je zapravo velika biblioteka optimizovanih algoritama. Može joj se pristupiti korišćenjem API-ja za različite programske jezike. Trenutno, IKOS sadrži implementacije za sledeće numeričke apstraktne domene: konstante, intervale, aritmetička zatvorenja i diskretne simboličke domene. Ostali domeni su u fazi razvoja.

6 Analiza prekoracenja bafera

U ovom poglavlju, biblioteka IKOS je korišćena za analizu programa pisanih u programskom jeziku C koji se koriste u sistemima za kontrolu leta. Izvršena je analiza koda od oko 600 linija upotrebom računara sa

16GB memorije i Intel Core i7 procesorom na 2.8 GHz. Rezultati su predstavljeni u tabeli 6. Fokus je na preciznosti koja se meri procentima bezbednih i nebezbednih vrednosti u odnosu na vrednosti koje predstavljaju upozorenja. Cilj analize je da vrednosti označene kao upozorenja obuhvate manje od 10%. Proverom upozorenja detektuje se koja su od njih lažna. Na osnovu rezultata iz tabele, procenat bezbednih i nebezbednih vrednosti je bar 90, čime se smanjuje prostor upozorenja koji je potrebno dodatno proveriti.

Program	Vreme za analizu	Preciznost
Paparazzi	22s	99%
Gen2	1m03s	98%
FLTz	10m30s	91%
Arduplane	6m30s	94%

Tabela 1: Rezultati analize greške prekoračenja bafera

7 Slični radovi

Radovi bazirani na apstraktnoj interpretaciji koji dele najviše sličnosti sa IKOS programskim okvirom su: C Global Surveyor, CodeHawk, Astrée i PolySpace Verifier.

C Global Surveyor[3] je statički analizator, predak IKOS-a, razvijen u istraživačkom centru NASA. U proseku proizvodi 20% upozorenja, što je značajno lošiji rezultat u odnosu na IKOS koji za iste ulaze proizvodi 1 do 2% upozorenja.

CodeHawk[1] je programski okvir za razvoj analizatora baziranih na apstraktnoj interpretaciji. Od svih pobrojanih radova najbliži je IKOS-u. Nažalost CodeHawk je komercijalan, te je premalo podataka javno dostupno da bi se izvršilo poređenje.

Astrée[5] je statički analizator prilagođen za analizu specifičnih Airbus kodova. Fokusiran je na računanje u pokretnom zarezu, odliku koja još nije implementirana u IKOS-u. Zbog svoje prilagođenosti specifičnim kodovima obuhvata mnogo užu klasu C programa od IKOS programskog okvira. Nažalost trenutna verzija je komercijalna, te je poređenje vršeno samo na originalnoj verziji.

PolySpace Verifier[2] je bio prvi od pobrojanih analizatora koji je koristio apstraktnu interpretaciju. Pokazao se veoma uspešno za kodove pisane u programskom jeziku Ada, dok je kod programa pisanih u C ili C++ programskom jeziku ispoljavao probleme sa skalabilnošću i veliki broj upozorenja.

8 Zaključak

U radu je dat pregled IKOS-a, programskog okvira koji primenjuje apstraktnu interpretaciju pri statičkoj analizi. U okviru IKOS-a izvorni kod pisan u programskim jezicima C i C++ najpre se prevodi u LLVM međukod, a zatim u AR međureprezentaciju. Demonstrirana je preciznost i skalabilnost IKOS-a sa osvrtom na analizu greške prekoračenja bafera.

Literatura

- [1] Kestrel Technology: CodeHawk. on-line at: <http://www.kestreltechnology.com>.
- [2] *IKOS: a Framework for Static Analysis based on Abstract Interpretation*, 2014.
- [3] Guillaume Brat Arnaud Venet. Precise and Efficient Static Array Bound Checking for Large Embedded C Programs. *PLDI*, pages 231–242, 2004.
- [4] Patrick Cousot and Radhia Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Proceedings of the 4th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages*, POPL '77, pages 238–252, New York, NY, USA, 1977. ACM.
- [5] Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, David Monniaux, and Xavier Rival. The astrÉE analyzer. In *Programming Languages and Systems, Proceedings of the 14th European Symposium on Programming, volume 3444 of Lecture Notes in Computer Science*, pages 21–30. Springer, 2005.