Vizuelizacja simboličkog izvršavanja

Seminarski rad u okviru kursa
Verifikacija softvera
Matematički fakultet

Aleksandra Kovačević,
Nikola Stojević,
Kristina Stanojević

aleksandrakovacevic099@gmail.com,
nikolastojevic@gmail.com,
stanojevic.kristina@gmail.com

19. april 2019

Sažetak

U ovom tekstu je ukratko objašnjeno korišćenje programa **sym_tree.py** koji pomoću alata *Klee* [2] vrši vizuelizaciju simboličkog izvršavanja nekog programa napisanog u programskom jeziku *C*. Klee služi za simboličko izvršavanje programa i za automatsko generisanje test primera. Koristeći fajlove koje izgeneriše iscrtavaćemo drvo simboličkog izvršavanja. Takodje postoji opcija za iscrtavanje stabla do zadate dubine, u slučaju da je prosledjeni program koji analiziramo previše kompleksan, ili nas zanima samo odredjeni deo instrukcija.

Sadržaj

1 Skup fajlova i način pokretanja programa

Pre korišćenja programa potrebno je instalirati odgovarajuće alate (*clang* [1], Klee, ...). Spisak tih instalacija nećemo redom navoditi u ovom radu. Osnova jesu oni alati koji se koriste u okviru kursa *Verifikacija softvera* na Matematičkom fakultetu u Beogradu.

1.1 Pokretanje

Pored osnovnih instalacija, u fajlu **requirements.txt** navedeno je šta je još neophodno imati kako bi dati program radio. To se može instalirati pokretanjem sledeće linije:

pip install -r requirements.txt

Pokretanje samog programa je olakšano skriptom **makeTree.sh**. Najpre je potrebno da se omogući izvršavanje te skripte kao i samog programa korišćenjem komande:

chmod +x && chmod +x makeTree.sh

Nakon tih koraka sledećom komandom pokrećemo naš program:

./makeTree.sh {absolute_path_to_c_file} {absolute_path_to_where_you_want_your_sym_tree_image}

Primer pokretanja:
./makeTree.sh /Downloads/07_materijali/pointer_error_sym.c /Downloads/drvo , nakon čega dobijamo fajlove *drvo.pdf* i *drvo.gv* kao rešenje. Način korišćenja se može videti i jednostavnim pokretanjem *./makeTree.sh -h*.

1.2 Ukratko o koracima

Izvršavanje alata Klee za sobom ostavlja fajlove koje smo koristili za iscrtavanje. U odgovarajućim fajlovima sa ekstenzijom **.pc** nalaze se instrukcije napisane u obliku koji nama nije odgovarajući za čitljivo ispisivanje stanja u čvorovima stabla. Zato je potrebno prilagoditi ispis korišćenjem parsera **parser.py** koji se nalazi u folderu **Parser**. Vrši se analiza, parsiranje i odgovarajuće iscrtavanje stabla, sa detaljima o stanju i putanjama u okviru čvorova, koje je dato u PDF fajlu nakon izvršavanja. Detaljno objašnjenje svake linije koda neće biti dato u tekstu.

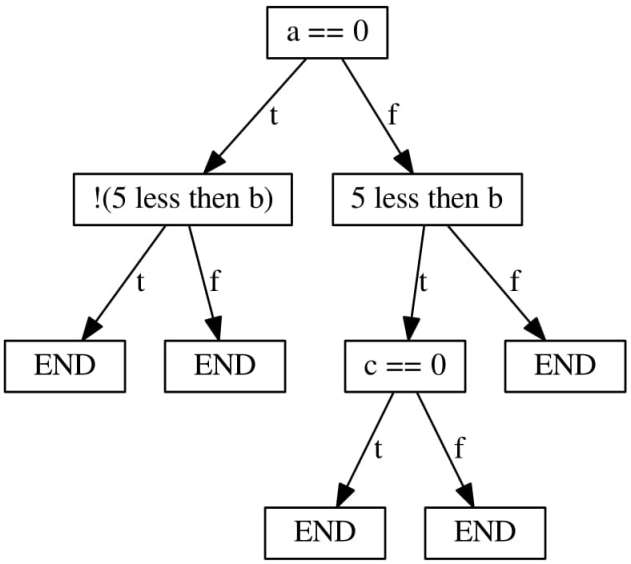
1.3 Drugi način pokretanja

Ukoliko dodje do nekih grešaka u toku pokretanja skripte *makeTree.sh*, moguće je samostalno pokretati jednu po jednu liniju u sledećem poretku kako bi se došlo do istog krajnjeg cilja:

- Generisanje llvm koda: **myclang -I {path_to_klee_include} -emit-llvm -c -g {name_of_the_file}.c**, nakon čega dobijamo {name_of_the_file}.bc fajl
- Prepuštanje dobijenog fajla Klee alatu: **myklee –write-sym-paths –write-pcs {name_of_the_file}.bc**, nakon čega dobijamo odgovarajuće foldere
- Pokretanje našeg programa: **python3 sym_tree.py -d 10**

2 Primer rezultata

Na slici 1 prikazan je jedan rezultat dobijen analizom nekog C programa. U čvorovima su ispisani uslovi koji se ispituju, grane su obeležene kao t ili f ako pretstavljaju da li je u toj putanji uslov iz čvora ispunjen ili ne. U listovima *END* predstavlja kraj jedne od putanja.



Slika 1: Primer jednog stabla simboličkog izvršavanja

Literatura

[1] Free Software Foundation. CLANG. on-line at: https://clang.llvm.org/get_started.html.

[2] Pod licencom NCSA otvorenog kooda Univerziteta Ilinois. KLEE. on-line at: https://klee.github.io/.