Generisanje test primera za programe u C-u korišćenjem Fuzz tehnike

Seminarski rad u okviru kursa
Metodologija stručnog i naučnog rada
Matematički fakultet

Ana Đorđević, Mladen Lazić, Aleksandra Đurić
djordjevicana93@gmail.com, mladen.lazic93@hotmail.com,
aleksandradjuric@mail.ru

9. april 2015.

Sažetak

Sadržaj

1 Uvod

2 Testiranje

Testiranje predstavlja važan deo životnog ciklusa razvoja softvera. Testiranjem se lakše uočavaju greške i propusti nastali prilikom implementacije. Pored toga ono predstavlja i jedan od načina specifikacije problema. Zbog prednosti koje se dobijaju čestim i redovnim testiranjem najzastupljenije i trenutno najpopularnije metodologije razvoja promovišu paralelno pisanje testova i implementacije softvera. Ekstremno programiranje kao jedan od predstavnika agilnih metodologija posebno ističe razvoj vođen testovima i pisanje testova prihvatljivosti [1].

2.1 Vrste testiranja

2.2 Fuzz testiranje

2.2.1 Uloga

2.2.2 Vrste

2.2.3 Prednosti

3 Implementacija Fuzz testera za C programe

Softver koji smo napravili predstavlja Fuzz tester za programe napisane u C-u. Proverava se njihova robusnost. Programski jezik u kome je implementiran tester je C++. Neophodno je da C program pored izvornog koda ima i Makefile pomoću kog se dobija izvršni kod. U cilju pojednostavljivanja primene testera napravljen je jednostavan korisnički interfejs korišćenjem Qt biblioteka.

3.1 Zahtevi

Prilikom izučavanja teme seminarskog rada i iz ličnog iskustva prilikom pisanja koda u C-u zaključili smo da se dosta vremena gubi na zadavanje ulaza kojim se posmatra dotadašnje stanje napisanog programa i ispituje njegova ispravnost. Automatizacija ovog koraka donosi veliku uštedu vremena. Uz to, ovako se najlakše uočavaju greške i propusti nastali zbog neadekvatnog rukovanja unetim vrednostima. Na ovaj način se dodatno simuliraju i greške koje korisnici programa napisanih u programskom jeziku C mogu zadati kao input. Ako ulaz ne zadaje korisnik već neki drugi program tada ispravnost ulaza zavisi od ispravnosti programa koji generise ulaz i provere su takođe neophodne.

Neophodno je obezbediti dovoljno veliku količinu test primera koji će omogućiti što obuhvatniju proveru ulaznih podataka. Ove provere se obavljaju u cilju praćenja ponašanja napisanog programa. Svaki od različitih vrsta podataka koji se unose zahteva testiranje kako regularnima vrednostima tako i onim koje bi mogle predstavljati problem ukoliko rukovanje podacima nije dovoljno dobro implementirano. Ovo sve zahteva detaljnu analizu specifičnosti svakog tipa podataka koji je podržan testovima.

3.2 Rešenja

Tester ima grupu fajlova koji pokrivaju najčešće vrste ulaza koje C programi očekuju kao input. Svaki fajl pokriva po jedan tip inputa. Potrebno je napraviti listu tipova podataka koje program koji se testira očekuje na ulazu. Neophodno je voditi računa o ispravnom redosledu. Tester na osnovu napravljene liste popunjava novu listu. Svaki element te liste se dobija slučajnim izborom vrednosti iz fajla za zahtevani tip. Nova lista sa konkretnim vrednostima predstavlja ulazne podatke za izvršni program koji se testira.

Podržani su osnovni tipovi podataka: int, float, double i char. U svakom fajlu su sadržane neke od regularnih vrednosti, vrednosti koje su kritične za pojedine tipove i neke od potpuno pogrešnih vrednosti. U kritične vrednosti spadaju minimalne i maksimalne vrednosti za taj tip. Potpuno pogrešne vrednosti su one koje nisu primenjive na dati tip.

Omogućena je i provera nizovima čiji su elementi osnovni tipovi podataka. Moguće je zadati maksimalan ili tačan broj elemenata. Izbor ovog broja se može prepustiti i slučajnom odabiru. Glavni razlog dodavanja ove mogućnosti je provera ispravnosti rada programa u zavisnosti od toga da li se koriste dinamički ili statički alocirani nizovi.

Naš fuzz tester ima i test primere koji pokrivaju grupe stringova koje su specifične i često se upotrebljavaju. Obuhvaćene su sekvence karaktera koje predstavljaju datum, vreme, email adrese, putanje do fajlova. Ovo je implementirano kao demonstracija provere podacima koji nisu osnovni tipovi.

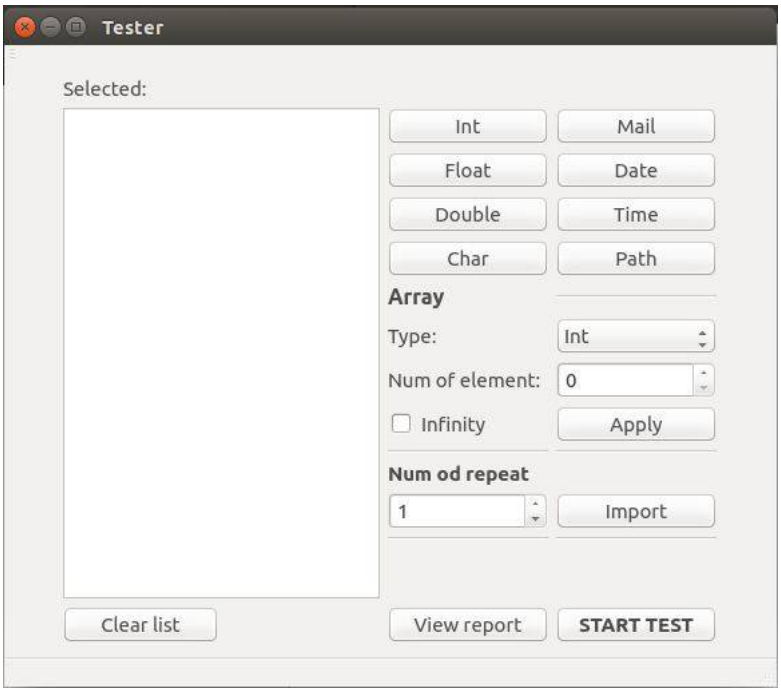
Segment koda koji vrši odabir jednog test primera:

```
/* uzima se element po element vektora koji
čuva tip ulaznih podataka u zadatom redosledu */
for(unsigned i = 0; i < testType.size(); i++)
{
    std::string choosed = testType.at(i);

    /* u zavisnosti od tipa podatka u fajl
    temp.txt se upisuje vrednost koja se dobija
    slučajnim izborom jednog od elemenata vektora */
    if(choosed.compare("int") == 0)
    {
        /* vektor int_test se popunjava podacima iz fajla
        int.txt u kome se cuvaju svi test primeri za int */
        if(int_test.empty())
            fill_vector("int");

        int rand_int = rand() % int_test.size();
        temp << QString::fromStdString(int_test.at(rand_int));
        temp << " ";
    ...
```

Vrednosti korišćenog test primera kao i exit_status programa se čuvaju u obliku izveštaja koji se potrebi može sačuvati.

3.3 Korisnički interfejs

Korisnički interfejs je napravljen da bi se korisnicima omogućila lakša upotreba testera. Pokušali smo da ga napravimo što intuitivnije i jednostavnije. Sa leve strane se nalazi lista koja predstavlja tipove podataka koji se očekuju na ulazu u odgovarajućem redosledu. Sa desne strane su dugmići kojima se dodaju elementi u listu. Za nizove se nudi i izbor broja elemenata. Klikom na dugme učitaj program se otvara prozor kojim se bira Makefile C programa koji se testira. Korisnik zadaje koliko puta će se pokrenuti program. Na slici 1 je prikazan izgled korisničkog interfejsa.



Slika 1: Korisnički interfejs

Izveštaj rada se prikazuje u posebnoj formi u kojoj se pored informacija o vrednostima koje su korićene za svaki test primer čuva i informacija o tome kako se program završio u obliku exit_statusa. Čuvanje ovog izveštaja na lokalnom disku je omogućeno klikom na dugme sačuvaj.

4 Zaključak

Literatura

[1] Saša Malkov. Agilni razvoj, testovi, 2013. on-line at: http://poincare.matf.bg.ac.rs/~smalkov/files/rs.r290.2015/public/Predavanja/Razvoj%20softvera.2015.05%20-%20Agilni%20razvoj,%20Testovi.pdf.