Visual Analysis

Praktični seminarski rad u okviru kursa
Verifikacija softvera
Matematički fakultet

Matija Lojović, 1017/2023

7. januar 2024.

Sažetak

U ovom izveštaju biće detaljno opisani alati korišćeni za analizu projekta Visual, kao i rezultati dobijeni njihovim pokretanjem. Takođe, biće ukazano na potenicjalne probleme i predložene određene izmene izvornog projekta tamo gde je to moguće. Najzad, biće dat opšti utisak o projektu i zaključci.

Sadržaj

1 Clang-tidy

Clang-tidy je alat za analizu C++ koda baziran na Clang kompajleru. Svrha mu je pružanje proširivog okvira za dijagnostikovanje i ispravljanje uobičajenih programerskih grešaka, poput kršenja stila, nepravilne upotrebe interfejsa ili bagova koji se mogu dedukovati putem statičke analize.

U ovoj analizi, clang-tidy je pokrenut nad svim *.cpp* i *.h* datotekama u glavnom direktorijumu projekta. Pronađene su dve nedoslednosti u kodu, konkretno u datoteci *graphwindow.cpp*, koje se tiču potencijalnog pristupanja polju objekta koji ima vrednost *null*. Takođe, dat je niz grananja kojim se može doći do takvog ishoda. Do problema dolazi jer se u linijama 853 i 863 nakon provere da li su promenljive *node1* i *node2* jednake *null* nastavlja sa izvršavanjem nakon upozorenja korisniku. Potrebno je ovde prekinuti dalje izvršavanje algoritma i zatražiti ponovni unos.

2 Memcheck

Memcheck je jedan od alata iz skupa alata za dinamičku analizu Valgrind. Osnovna svrha mu je pronalaženje grešaka u memoriji, pa se tako može koristiti za detektovanje pristupa nedozvoljenoj memoriji, korišćenja nedefinsanih vrednosti, nekorektno oslobađanje memorije itd..

U ovoj analizi memcheck je pokrenut tri puta i sačuvani su izveštaji koje je generisao. Ispostavlja se da je broj memorijskih blokova koji su izgubljeni direktno relativno mali, u proseku samo desetak, ali je veliki broj indirektno izgubljenih blokova. Uzrok za ovo leži u klasi *Graph* gde postoji vektor ivica pod nazivom *edgeSet* koji se ni u jednom trenutku ne oslobađa. Samim tim, ako se u toku izvršavnja izgubi pokazivač na ovaj vektor (recimo, pravljenjem novog grafa), dolazi do curenja memorije svih ivica koje su se sadržale u grafu, pa odatle veliki broj indirektno izgubljenih blokova. Potrebno je adekvatno osloboditi ovaj vektor pri brisanju kako bi se rešio ovaj problem.

3 Perf i FlameGraph

Perf je alat za analizu performansi i profajliranje na Linux sistemima. Pokreće se iz komandne linije i širok spektar prikaza performansi programa. FlameGraph je alat za pravljenje vizuelizacija izlaza iz profajlera u obliku grafika sa naslaganim kutijama, pri čemu svaka kutija prikazuje svojom širinom broj uzoraka u kojima se javlja određena funkcija.

U ovoj analizi, perf je korišćen da isprati izvršavanje programa u kojem su pokrenuta tri algoritma: DFS, BFS i MST. Izlaz iz alata perf prosleđen je zatim alatu FlameGraph koji je kreirao adekvatan grafik. Na grafiku se može primetiti da je vreme izvršavanja algoritama DFS i BFS slično, dok je MST donekle brži. Ovakva analiza može se sprovesti za različite ulazne grafove i algoritme pokretanjem odgovarajuće skripte. Ovim procesom se može utvrditi koji algoritmi su potencijalno sporiji za određene grafove i samim tim podložniji optimizaciji.

3.1 Flawfinder

Flawfinder je alat koji služi za otkrivanje bezbednosnih propusta u C i C++ kodu. Funkcioniše tako što pretražuje izvorni kod kako bi našao potencijalno nebezbedna mesta koristeći predefinisan skup pravila. Svakom

propustu dodeljuje određeni nivo značajnosti i upisuje ga u sveobuhvatni izveštaj.

Pokretanjem flawfinder alata na ovom projektu dobija se 5 upozorenja srednjeg značaja (nivo 3). Svih 5 se odnose na način otvaranja datoteka u datoteci *graphwindow.cpp*. Naime, datoteka se bira kroz klasu *QFileDialog* bez ikakvih bezbednosnih provera, pa je moguće potencijalno preusmeravanje od strane napadača na neželjeni sadržaj (recimo, sistemske ili poverljive korisničke datoteke). Potrebno je uvesti određene bezbednosne provere unosa kako bi se utvrdilo da li izabrane datoteke zaista smeju biti otvarane kako bi se izbegli ovakvi problemi.

3.2 Cachegrind

Kao i memcheck, cachegrind je jedan od alata u sastavu Valgrinda. On služi za profajliranje keš memorije i pojedinačno praćenje instrukcija programa. Na svom izlazu daje izveštaj sa brojem keš pogodaka i upisa po nivoima koji može biti koristan za pronalaženje mesta za optimizaciju.

U ovoj analizi, pokretanjem cachegrind alata dobili smo izveštaj koji sadrži ukupan broj instrukcija (oko 3 milijarde), kao i podatke o broju keš promašaja za različite nivoe. Vidimo da je broj promašaja na nivou I1, odnosno prvom nivou keša instrukcija, oko 50 miliona, što iznosi oko 1.8% ukupnog referisanja keša instrukcija. Takođe, vidimo da je procenat promašaja na poslednjem nivou keša približno 0, te je velika većina instrukcija nađena na nekom od nivoa keš memorije. Što se tiče keša podataka, bilo je oko 1.1 milijardu pristupa (oko 700 miliona pisanja i 400 miliona čitanja). Procenat promašaja na prvom nivou je oko 3.8%, a na poslednjem oko 0.1%. Generalni utisak je da, pošto su oba prva nivoa keš memorije pogođena u preko 95% slučajeva, ovaj projekat prilično dobro barata keš memorijom.

4 Zaključak

Sveukupni utisak dobijen analizom ovog projekta ukazuje da je on prilično kvalitetan. Zamerke koje su nađene tiču se jednog propusta u proveri vrednosti promenljivih koje mogu biti null, neoslobađanja memorije kod jednog vektora objekata, kao i potencijalnih bezbednosnih rizika pri otvaranju datoteka. Proverene su i performanse kroz profajliranje keš memorije i oslikavanje stanja steka tokom izvršavanja i došlo se do zaključka da su algoritmi napisani tako da program radi prilično efikasno. Ostaje da se potencijalno produbi analiza pojedinačnih algoritama za različite ulaze, kako bi se našli dalji smerovi za optimizaciju.