Analiza projekta korišćenjem alata za verifikaciju softvera

verifikaciju softvera Seminarski rad u okviru kursa Verifikacija softvera Matematički fakultet

$\begin{array}{c} {\rm Tamara~ Duki\acute{c}} \\ {\rm tamarazdjukic@gmail.com} \end{array}$

22. april 2024.

Sadržaj

1	Valgrind															2	,							
	1.1	Memcheck.																					2)
	1.2	Callgrind .																					2)

1 Valgrind

1.1 Memcheck

Memcheck je najpoznatiji Vagrind-ov alat i on se podrazumevano poziva ako ne stavimo dodatnu opciju koji alat pozivamo. Memcheck detektuje memorijske greške korisničkog programa kao što su:

- Curenje memorije
- Neispravno oslobađanje memorije na hipu
- Čitanje ili pisanje u nedopuštenu memoriju na hipu, steku
- Korišćenje nedefinisanih vrednosti, vrednosti koje nisu inicijalizovane ili koje su izvedene od drugih nedefinisanih vrednosti

Pokrećemo skriptu $run_memcheck.sh$ koja sadrži poziv memcheck-a. Rezultat izvršavanja se nalazi u memcheck result.txt.

```
==13690==
5 ==13690== LEAK SUMMARY:
5 ==13690== definitely lost: 19,960 bytes in 29 blocks
7 ==13690== indirectly lost: 191,046 bytes in 294 blocks
5 ==13690== possibly lost: 10,465,824 bytes in 87 blocks
6 ==13690== still reachable: 24,602,834 bytes in 62,488 blocks
7 ==13690== of which reachable via heuristic:
1 ==13690== multipleinheritance: 3,440 bytes in 10 blocks
7 ==13690== suppressed: 0 bytes in 0 blocks
8 ==13690== For lists of detected and suppressed errors, rerun with: -s
8 ==13690== ERROR SUMMARY: 11110 errors from 73 contexts (suppressed: 0 from 0)
```

Slika 1: Rezultat memcheck-a

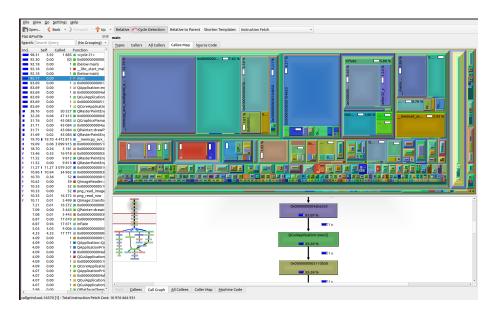
Primetimo da ima dosta memorije koja je definitivno i indirektno izgubljena. Takođe imamo dosta memorije kojoj možemo da pristupimo i da je oslobodimo.

1.2 Callgrind

Callgrind je alat koji u vidu grafa generiše listu poziva funkcija korisničkog programa. Podrazumevano, prikupljeni podaci se sastoje od broja izvršenih instrukcija, njihovog odnosa prema izvornim linijama, odnosa između pozivajućih i pozvanih funkcija, i broja takvih poziva.

Pokrećemo preko skripte run_callgrind.sh.

Kada se završi rad Callgrind-a, rezultati analize su zapisani u fajlu *call-grind.out.16370*. Otvorićemo taj fajl pomoću KCachegrind-a koji će nam grafički pokazati rezultate.



Slika 2: Rezultat callgrind-a