**Seminarski rad predmeta Verifikacija softvera**

**Apstraktna interpretacija (analiza intervala poverenja uz podrsku alata llvm i clang)**

12.07.2019.

Izvorni kod programa sastoji se od 600 linija napisanih u programskom jeziku CPP rasporedjenih u okviru jedne izvorne datoteke pod nazivom “ai\_intervali.cpp” koja se nalazi u folderu “./src”.

Folder “./test” sadrzi odgovarajuce programe pisane u programskom jeziku CPP, na koje se primenjuje apstraktna interpretacija. Svaki .cpp fajl se uz pomoc alata **clang** (verizija 6.0) prevodi u odgovarajuci .bc format, na koji se nakon toga primenjuje apstrakta interpretacija koristeci alat **opt** (verzija 6.0).

Skripta “ai\_live.py” pisana u programskom jeziku Python omogucava interaktivnu upotrebu alata.

Projekat koristi alate **make** odnosno **cmake**.

Komande za pokretanje programa:

# pozicioniranje u forkovani direktorijum projekta

cd test

emacs test.cpp

# pisanje programa za test u jeziku cpp

clang++-6.0 -O0 -emit-llvm test.cpp -c

cd ../

opt-6.0 -instnamer -load src/AI\_INTERVALI.so -AI-PROLAZ test/test.bc

**Pokrivene funkcionalnosti:**

Trenutna verzija programa pretpostavlja kompaktnost intervala poverenja i podrzava rad iskljucivo sa celobrojnim oznacenim brojevima.

Podrzane operacije su operacije sabiranja i oduzimanja.

Analiza intervala pokriva naredbu assert(\*) biblioteke <assert.h>. Podrzana su nejednakosna ogranicenja (<, <=, >, >=) u obliku:

P ogr K,

K ogr P, gde je K celobrojna konstantna vrednost, P odgovarajuca promenljiva programa odnosno llvm-ove interne (dodatno temporarne) medjureprezentacije a ogr jedno od pomenuta cetiri ogranicenja.

Dodatno, program podrzava kompoziciju funkcija jedne celobrojne promenljive vrseci analizu svih funkcija programa i parsirajuci intervale poverenja koji svaka od njih definise; zatim koristi iste u pozivajucim funkcijama.

Program implementira i naivnu obradu naredbe grananja. Obrada naredbe grananja odnosi se na pojedinacnu analizu odgovarajucih blokova i markiranje eventualno dostiznih i odnosno nedostiznih blokova programa koji su rezultat prethodne naredbe.

**Funkcionalnosti koje trebaju biti pokrivene u buducnosti:**

1. Podrska realnim tipovima.

2. Dodavanje ostalih binarnih operacija: /, \*, mod, itd.

3. Dodavanje ogranicenja oblika P ogr P, odnosno K ogr K, gde je K konstanta odgovarajuceg tipa, P odgovarajuca promenljiva programskog jezika odnosno llvm-ove interne medjureprezentacije a ogr ogranicenje. Dodavanje jednakosnih ogranicenja.

4. Obrada funkcija vise argumenata.

5. Unapredjenje naredbe grananja u slucaju kada nije moguce odrediti kojom granom program nastavlja izvrsavanje integrisati odgovarajuce intervale poverenja u jedan I njih koristiti u daljoj analizi programa (trenuta verzija projekta analizira pomenute blokove nakon cega pretpostavlja izvrsavanje else grane I sa time nastavlja dalje apstraktno izvrsavanje programa). Ovo zahteva izmenu osnovnog tipa intervala (za koji se trenurno pretpostavalja da je kompaktan skup) u skup koji apstrahuje konacnu uniju intervala.

6. Obrada while instrukcije.

**Primer upotrebe:**

Za ulaz:

*int main(){*

*int a;*

*a = 7;*

*return a;*

*}*

Program generise sledeci izlaz:

*Functions argument initialization:*

Funkcija main nema argumente te je analiza argumenata funckije vratila prazan skup.

*Initialization:*

*++++++++++++++++++++++++++++++++++*

*Sintaksa ispisa po baznim blokovima.*

*Basic block name: bb*

*Obrada baznog bloka sa imenom bb (osnovni bazni blok).*

*[I]: %tmp = alloca i32, align 4*

*[opcode]: 29*

*WORKING ON: alloca...*

*DONE: alloca.*

*tmp in [-inf, +inf]*

*Obrada instrukcije alokacije. Alocira se ceo skup (promenljiva koja se alocira moze uzeti proizvoljnu celobrojnu vrednost).*

*[I]: %tmp1 = alloca i32, align 4*

*[opcode]: 29*

*WORKING ON: alloca...*

*DONE: alloca.*

*tmp1 in [-inf, +inf]*

*[I]: store i32 0, i32\* %tmp, align 4*

*[opcode]: 31*

*WORKING ON: store...*

*DONE: store.*

*tmp in [0, 0]*

*Vrednost 0 se dodeljuje promenljivoj tmp. Odgovarajuci interval poverenja jeste jedna tacka, predstavljena intervalom poverenja kod koga se poklapaju pocetna i krajnja tacka.*

*[I]: store i32 7, i32\* %tmp1, align 4*

*[opcode]: 31*

*WORKING ON: store...*

*DONE: store*

*tmp1 in [7, 7]*

*[I]: %tmp2 = load i32, i32\* %tmp1, align 4*

*[opcode]: 30*

*WORKING ON: load...*

*Searching for: tmp1*

*DONE: load.*

*tmp2 in [7, 7]*

*Instrukcijom load se ucitava odgovarajuca vrednost. Kako se ucitava vrednost registra tmp1, a interval poverenja te promenljive jeste [7, 7] isti interval poverenja nasledjuje i promenljiva tmp2.*

*[I]: ret i32 %tmp2*

*[opcode]: 1*

*WORKING ON: ret...*

*Searching for: tmp2*

*DONE: ret.*

*RETURN\_VALUE in [7, 7]*

*Instrukcija vracanja vrednosti iz funkcije.*

*++++++++++++++++++++++++++++++++++*

*End of initialization.*

*Prethodni intervali su intervali koji se generisu online u toku izvrsavanja apstraktne interpretacije. Nakon tih informacija, za svaku instrukciju ispisuju se finalni intervali poverenja u sledecem obliku:*

*Function name: main*

*Number of parsed function instruction: 6*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Intervals of confidence:*

*%tmp = alloca i32, align 4*

*Searching for: tmp*

*tmp in [0, 0]*

*Finalni interval poverenja nakon apstraktne interpretacije celog programa promenljive tmp jeste jednoclan interval [0, 0] (iako se samom instrukcijom alokacije definise ceo interval kao interval poverenja, isti kasnije tokom izvrsavanja programa bude suzen).*

*%tmp1 = alloca i32, align 4*

*Searching for: tmp1*

*tmp1 in [7, 7]*

*store i32 0, i32\* %tmp, align 4*

*Searching for: tmp*

*tmp in [0, 0]*

*store i32 7, i32\* %tmp1, align 4*

*Searching for: tmp1*

*tmp1 in [7, 7]*

*%tmp2 = load i32, i32\* %tmp1, align 4*

*Searching for: tmp2*

*tmp2 in [7, 7]*

*ret i32 %tmp2*

*Searching for: RETURN\_VALUE*

*RETURN\_VALUE in [7, 7]*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Searching for: RETURN\_VALUE*

*RETURN\_VALUE in [7, 7]*

*Current parsed functions:*

*main*

*RETURN\_VALUE in [7, 7]*

*Parsirana samo jedna funkcija (main). Njena povratna vrednosti ima interval poverenja koji predstavlja jedan broj, konkretno 7.*

***Spisak test primera:***

*10.cpp: bazni test primer; vracanje konstante, jednoclan interval*

*9.cpp: provera integracije konstantne vrednosti sa konacnim intervalom poverenja, konkretno 5 u (3, 7)*

*12.cpp: apstraktna interpretacija programa koji ne inicijalizuje celobrojnu promenljivu, a istu koristi.*

*11.cpp: bazna provera operacije sabiranja*

*13.cpp: bazna provera operacije oduzimanja*

*7.cpp: agregiranje promenljivih raznih intervala poverenja operacijom sabiranja. Konkretno, a iz [3, 6], b iz [4, 7], rezultuju sa a+b iz [7, 13]*

*6.cpp: agregiranje promenljivih raznih intervala poverenja operacijom oduzimanja. Konkretno, a iz [3, 6], b iz [4, 7], rezultuju sa a-b iz [-2, 4]. Obratiti paznju na nacin integracije intervala poverenja u slucaju operacije oduzimanja.*

*5.cpp: agregiranje eventualno beskonacnih intervala poverenja operacijom sabiranja. Konkretno a iz [10, +inf], b iz [3, 5], a+b iz [13, +inf]*

*4.cpp: agregiranje eventualno beskonacnih intervala poverenja operacijom oduzimanja. Konkretno a iz [3, 5] b iz [10, +inf], a-b iz [-inf, -5]*

*14.cpp: bazna provera operacije assert; ogranicenje tipa <*

*15.cpp: bazna provera operacije assert; ogranicenje tipa >*

*16.cpp: iterativno smanjivanje intervala operaciom assert primenom ogranicenja tipa <*

*19.cpp: iterativno smanjivanje intervala operaciom assert primenom ogranicenja tipa <*

*17.cpp: slaganje operacije sabiranja sa intervalima poverenja definisanim operacijom assert*

*18.cpp: slaganje operacije assert sa intervalima poverenja definisanim operacijom sabiranja*

*20.cpp: suzavanje intervala poverenja generisanog operacijom sabiranja operacijom assert; konkretno interval poverenja se sa [33, 55] suzava na [40, 44]*

*21.cpp: nevalidna operacija assert; program vraca interval poverenja [10, 5] sto je po konvenciji prazan skup*

*22.cpp: nevalidna operacija assert; vracanje praznog intervala*

*3.cpp: funkcije; funkcija vraca interval poverenja [-1995, 1995]; poziv iste funkcije*

*8.cpp: analiza tri funkcije u okviru istog fajla*

*2.cpp: funkcija koja vraca ceo interval [-inf, +inf]*

*1.cpp: ilustracija poziva funkcije neinicijalizovanog argumenta*

*23.cpp: jednostavna instrukcija grananja; uslov se validira na netacno; odbacivanje bloka*

*24.cpp: instrukcija grananja, uslov se validira na tacno; analiza oba bloka; pretpostavka i nastavak dalje analize sa drugim blokom*

*25.cpp: nepoznat rezultat naredbe grananja operacije if; analiza oba bloka; pretpostavka izvrsavanja drugog bloka; nastavak dalje interpretacije sa istom pretpostavkom*

***Dodatak: Ilustracija izvrsavanja nekih test primera sa objasnjenjima:***

Za ulaz:

*int main(){*

*int x, y;*

*assert(x>10); // [10, +inf]*

*assert(5>y && 3<y); // [3, 5]*

*return y - x; // [-inf, -5]*

*}*

Program generise sledeci izlaz:

*Functions argument initialization:*

Main funkcija nema argumenata

*Initialization:*

Iteriranje kroz odgovarajuce bazne blokove odnosno njihove instrukcije.

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb*

Osnovni bazni blok.

*[I]: %tmp = alloca i32, align 4*

*[opcode]: 29*

*WORKING ON: alloca...*

*DONE: alloca.*

*tmp in [-inf, +inf]*

Alokacija promenljive. Interval poverenja inicijalno beskonacan.

*[I]: %tmp1 = alloca i32, align 4*

*[opcode]: 29*

*WORKING ON: alloca...*

*DONE: alloca.*

*tmp1 in [-inf, +inf]*

*[I]: %tmp2 = alloca i32, align 4*

*[opcode]: 29*

*WORKING ON: alloca...*

*DONE: alloca.*

*tmp2 in [-inf, +inf]*

*[I]: store i32 0, i32\* %tmp, align 4*

*[opcode]: 31*

*WORKING ON: store...*

*DONE: store.*

*tmp in [0, 0]*

Ucitavanje konkretne vrednosti 0 u promenljivu ciji je interval poverenja beskonacan. Suzavanje i konkretizacija istog.

*[I]: %tmp3 = load i32, i32\* %tmp1, align 4*

*[opcode]: 30*

*WORKING ON: load...*

*Searching for: tmp1*

*DONE: load.*

*tmp3 in [-inf, +inf]*

Nasledjivanje beskonacnog intervala poverenja instrukcijom load.

*[I]: %tmp4 = icmp sgt i32 %tmp3, 10*

*[opcode]: 51*

*WORKING ON: icmp...*

*Searching for: tmp3*

*Searching for: tmp1*

*Assertion could be satisfied.*

*DONE: icmp.*

*tmp4 in [0, 1]*

Instrukcija poredjenja. Poredi se vrednost registra %tmp3 (beskonacan interval poverenja) sa konkretnom konstantom 10. Interval poverenja promenljive %tmp3 postavlja se na [10, +inf]; interval poverenja povratne vrednosti instrukcije postavlja se na [0, 1] koji reprezentuje moguce zadovoljiv uslov u vreme izvrsavanja programa (konvencija).

*[I]: br i1 %tmp4, label %bb5, label %bb6*

*[opcode]: 2*

*WORKING ON: br...*

*Searching for: tmp4*

*Basich block: "bb5" and "bb6" both still reachable.*

*DONE: br.*

Instrukcija grananja. Kako je interval poverenja postavljen instrukcijom poredjenja postavljen na [0, 1] odnosno konstantu true, tako su oba bloka instrukcije grananja moguce dostizna zavisno od konkretnog izvrsavanja. Stoga se odgovarajuci bazni blokovi oznacavaju kao moguce dostizni.

*++++++++++++++++++++++++++++++++++*

Kraj izvrsavanja prvog baznog bloka i pocetak analize drugog.

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb5*

*[I]: br label %bb8*

*[opcode]: 2*

*WORKING ON: br...*

*DONE: br.*

Bazni blok sastoji se od jedne instrukcije bezuslovnog skoka.

*++++++++++++++++++++++++++++++++++*

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb6*

*[I]: call void @\_\_assert\_fail(i8\* getelementptr inbounds ([5 x i8], [5 x i8]\* @.str, i32 0, i32 0), i8\* getelementptr inbounds ([6 x i8], [6 x i8]\* @.str.1, i32 0, i32 0), i32 4, i8\* getelementptr inbounds ([11 x i8], [11 x i8]\* @\_\_PRETTY\_FUNCTION\_\_.main, i32 0, i32 0)) #2*

*[opcode]: 54*

*WORKING ON: call...*

*Parse call:*

*inst->getName():*

*inst->getOperand(0)->getName():*

*Real name:*

*Function name: \_\_assert\_fail*

*Argument:*

*Interval of confidence: Searching for:*

*Search unsuccessfull*

*nullptr*

*Function: \_\_assert\_fail*

*Interval of confidence: nullptr*

*DONE: call.*

*in [-inf, +inf]*

Assert poziv.

*[I]: unreachable*

*[opcode]: 7*

*-- not supported --*

*++++++++++++++++++++++++++++++++++*

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb7*

*[I]: br label %bb8*

*[opcode]: 2*

*WORKING ON: br...*

*DONE: br.*

*++++++++++++++++++++++++++++++++++*

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb8*

*[I]: %tmp9 = load i32, i32\* %tmp2, align 4*

*[opcode]: 30*

*WORKING ON: load...*

*Searching for: tmp2*

*DONE: load.*

*tmp9 in [-inf, +inf]*

*[I]: %tmp10 = icmp sgt i32 5, %tmp9*

*[opcode]: 51*

*WORKING ON: icmp...*

*Searching for: tmp9*

*Searching for: tmp2*

*Assertion could be satisfied.*

*DONE: icmp.*

*tmp10 in [0, 1]*

Eventualno zadovoljivo ogranicenje. Suzavanje intervala poverenja promenljive %tmp9.

*[I]: br i1 %tmp10, label %bb11, label %bb15*

*[opcode]: 2*

*WORKING ON: br...*

*Searching for: tmp10*

*Basich block: "bb11" and "bb15" both still reachable.*

*DONE: br.*

*++++++++++++++++++++++++++++++++++*

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb11*

*[I]: %tmp12 = load i32, i32\* %tmp2, align 4*

*[opcode]: 30*

*WORKING ON: load...*

*Searching for: tmp2*

*DONE: load.*

*tmp12 in [-inf, 5]*

*[I]: %tmp13 = icmp slt i32 3, %tmp12*

*[opcode]: 51*

*WORKING ON: icmp...*

*Searching for: tmp12*

*Searching for: tmp2*

*Assertion could be satisfied.*

*DONE: icmp.*

*tmp13 in [0, 1]*

*[I]: br i1 %tmp13, label %bb14, label %bb15*

*[opcode]: 2*

*WORKING ON: br...*

*Searching for: tmp13*

*Basich block: "bb14" and "bb15" both still reachable.*

*DONE: br.*

*++++++++++++++++++++++++++++++++++*

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb14*

*[I]: br label %bb17*

*[opcode]: 2*

*WORKING ON: br...*

*DONE: br.*

*++++++++++++++++++++++++++++++++++*

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb15*

*[I]: call void @\_\_assert\_fail(i8\* getelementptr inbounds ([11 x i8], [11 x i8]\* @.str.2, i32 0, i32 0), i8\* getelementptr inbounds ([6 x i8], [6 x i8]\* @.str.1, i32 0, i32 0), i32 5, i8\* getelementptr inbounds ([11 x i8], [11 x i8]\* @\_\_PRETTY\_FUNCTION\_\_.main, i32 0, i32 0)) #2*

*[opcode]: 54*

*WORKING ON: call...*

*Parse call:*

*inst->getName():*

*inst->getOperand(0)->getName():*

*Real name:*

*Function name: \_\_assert\_fail*

*Argument:*

*Interval of confidence: Searching for:*

*in [-inf, +inf]*

*Function: \_\_assert\_fail*

*Interval of confidence: nullptr*

*DONE: call.*

*in [-inf, +inf]*

Poziv assert. Povratna vrednost poziva postavlja se na [-inf, +inf] jer se isti ne tretira kao korisnicki definisana funkcija, vec se isti tretira manuelno kroz parsiranje odgovarajucih intervala poverenja definisanih ogranicenjima.

*[I]: unreachable*

*[opcode]: 7*

*-- not supported --*

*++++++++++++++++++++++++++++++++++*

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb16*

*[I]: br label %bb17*

*[opcode]: 2*

*WORKING ON: br...*

*DONE: br.*

*++++++++++++++++++++++++++++++++++*

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb17*

*[I]: %tmp18 = load i32, i32\* %tmp2, align 4*

*[opcode]: 30*

*WORKING ON: load...*

*Searching for: tmp2*

*DONE: load.*

*tmp18 in [3, 5]*

*[I]: %tmp19 = load i32, i32\* %tmp1, align 4*

*[opcode]: 30*

*WORKING ON: load...*

*Searching for: tmp1*

*DONE: load.*

*tmp19 in [10, +inf]*

Instrukcija load. Nasledjivanje poluzatvorenog intervala poverenja.

*[I]: %tmp20 = sub nsw i32 %tmp18, %tmp19*

*[opcode]: 13*

*WORKING ON: sub...*

*Searching for: tmp18*

*Searching for: tmp19*

*DONE: sub.*

*tmp20 in [-inf, -5]*

Operacija oduzimanja. Iz dinamicke istorije ucitavaju se intervali poverenja odgovarajucih promenljivih koje ucestvuju u operaciji. Isti se integrisu; rezultat integracije predstavlja interval poverenja koji se dodeljuje povratnoj vrednosti operacije.

*[I]: ret i32 %tmp20*

*[opcode]: 1*

*WORKING ON: ret...*

*Searching for: tmp20*

*DONE: ret.*

*RETURN\_VALUE in [-inf, -5]*

*++++++++++++++++++++++++++++++++++*

*End of initialization.*

*Function name: main*

*Number of parsed function instruction: 25*

Kraj inicijalizacije. Funkcija main sastoji se od 25 parsiranih instrukcija. Za svaku od njih se u drugom delu ispisuju finalni intervali poverenja.

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Intervals of confidence:*

*%tmp = alloca i32, align 4*

*Searching for: tmp*

*tmp in [0, 0]*

*%tmp1 = alloca i32, align 4*

*Searching for: tmp1*

*tmp1 in [10, +inf]*

*%tmp2 = alloca i32, align 4*

*Searching for: tmp2*

*tmp2 in [3, 5]*

*store i32 0, i32\* %tmp, align 4*

*Searching for: tmp*

*tmp in [0, 0]*

*%tmp3 = load i32, i32\* %tmp1, align 4*

*Searching for: tmp3*

*tmp3 in [10, +inf]*

*%tmp4 = icmp sgt i32 %tmp3, 10*

*Searching for: tmp4*

*tmp4 in [0, 1]*

*br i1 %tmp4, label %bb5, label %bb6*

*nullptr*

*br label %bb8*

*nullptr*

*call void @\_\_assert\_fail(i8\* getelementptr inbounds ([5 x i8], [5 x i8]\* @.str, i32 0, i32 0), i8\* getelementptr inbounds ([6 x i8], [6 x i8]\* @.str.1, i32 0, i32 0), i32 4, i8\* getelementptr inbounds ([11 x i8], [11 x i8]\* @\_\_PRETTY\_FUNCTION\_\_.main, i32 0, i32 0)) #2*

*Searching for:*

*in [-inf, +inf]*

*unreachable*

*nullptr*

*br label %bb8*

*nullptr*

*%tmp9 = load i32, i32\* %tmp2, align 4*

*Searching for: tmp9*

*tmp9 in [-inf, 5]*

*%tmp10 = icmp sgt i32 5, %tmp9*

*Searching for: tmp10*

*tmp10 in [0, 1]*

*br i1 %tmp10, label %bb11, label %bb15*

*nullptr*

*%tmp12 = load i32, i32\* %tmp2, align 4*

*Searching for: tmp12*

*tmp12 in [3, 5]*

*%tmp13 = icmp slt i32 3, %tmp12*

*Searching for: tmp13*

*tmp13 in [0, 1]*

*br i1 %tmp13, label %bb14, label %bb15*

*nullptr*

*br label %bb17*

*nullptr*

*call void @\_\_assert\_fail(i8\* getelementptr inbounds ([11 x i8], [11 x i8]\* @.str.2, i32 0, i32 0), i8\* getelementptr inbounds ([6 x i8], [6 x i8]\* @.str.1, i32 0, i32 0), i32 5, i8\* getelementptr inbounds ([11 x i8], [11 x i8]\* @\_\_PRETTY\_FUNCTION\_\_.main, i32 0, i32 0)) #2*

*Searching for:*

*in [-inf, +inf]*

*unreachable*

*nullptr*

*br label %bb17*

*nullptr*

*%tmp18 = load i32, i32\* %tmp2, align 4*

*Searching for: tmp18*

*tmp18 in [3, 5]*

*%tmp19 = load i32, i32\* %tmp1, align 4*

*Searching for: tmp19*

*tmp19 in [10, +inf]*

*%tmp20 = sub nsw i32 %tmp18, %tmp19*

*Searching for: tmp20*

*tmp20 in [-inf, -5]*

*ret i32 %tmp20*

*tmp20 in [-inf, -5]*

*ret i32 %tmp20*

*Searching for: RETURN\_VALUE*

*RETURN\_VALUE in [-inf, -5]*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Searching for: RETURN\_VALUE*

*RETURN\_VALUE in [-inf, -5]*

*Current parsed functions:* Parsirana funkcija main i poziv assertion.

*\_\_assert\_fail*

*nullptr*

*main*

*RETURN\_VALUE in [-inf, -5]*

Za ulaz:

*#include<assert.h>*

*int funkcija(int x){*

*assert(-1995<x && x<1995);*

*return x;*

*}*

*int main(){*

*int a;*

*return funkcija(a);*

*}*

Analiza generise sledeci izlaz:

*WARNING: You're attempting to print out a bitcode file.*

*This is inadvisable as it may cause display problems. If*

*you REALLY want to taste LLVM bitcode first-hand, you*

*can force output with the `-f' option.*

*Functions argument initialization:*

*[arg]: 0x1dd5ac0*

*\*arg: i32 %arg*

*Obradjena vrednost je [argument funkcije]*

*Ime funkcije: 0x1dd0948*

Analiza funkcije jedne promenljive.

*Initialization:*

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb*

*[I]: %tmp = alloca i32, align 4*

*[opcode]: 29*

*WORKING ON: alloca...*

*DONE: alloca.*

*tmp in [-inf, +inf]*

*[I]: store i32 %arg, i32\* %tmp, align 4*

*[opcode]: 31*

*WORKING ON: store...*

*Searching for: arg*

*Search unsuccessfull*

*DONE: store.*

*tmp in [-inf, +inf]*

*[I]: %tmp1 = load i32, i32\* %tmp, align 4*

*[opcode]: 30*

*WORKING ON: load...*

*Searching for: tmp*

*DONE: load.*

*tmp1 in [-inf, +inf]*

*[I]: %tmp2 = icmp slt i32 -1995, %tmp1*

*[opcode]: 51*

*WORKING ON: icmp...*

*Searching for: tmp1*

*Searching for: tmp*

*Assertion could be satisfied.*

*DONE: icmp.*

*tmp2 in [0, 1]*

*[I]: br i1 %tmp2, label %bb3, label %bb7*

*[opcode]: 2*

*WORKING ON: br...*

*Searching for: tmp2*

*Basich block: "bb3" and "bb7" both still reachable.*

*DONE: br.*

*++++++++++++++++++++++++++++++++++*

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb3*

*[I]: %tmp4 = load i32, i32\* %tmp, align 4*

*[opcode]: 30*

*WORKING ON: load...*

*Searching for: tmp*

*DONE: load.*

*tmp4 in [-1995, +inf]*

*[I]: %tmp5 = icmp slt i32 %tmp4, 1995*

*[opcode]: 51*

*WORKING ON: icmp...*

*Searching for: tmp4*

*Searching for: tmp*

*Assertion could be satisfied.*

*DONE: icmp.*

*tmp5 in [0, 1]*

*[I]: br i1 %tmp5, label %bb6, label %bb7*

*[opcode]: 2*

*WORKING ON: br...*

*Searching for: tmp5*

*Basich block: "bb6" and "bb7" both still reachable.*

*DONE: br.*

*++++++++++++++++++++++++++++++++++*

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb6*

*[I]: br label %bb9*

*[opcode]: 2*

*WORKING ON: br...*

*DONE: br.*

*++++++++++++++++++++++++++++++++++*

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb7*

*[I]: call void @\_\_assert\_fail(i8\* getelementptr inbounds ([18 x i8], [18 x i8]\* @.str, i32 0, i32 0), i8\* getelementptr inbounds ([6 x i8], [6 x i8]\* @.str.1, i32 0, i32 0), i32 3, i8\* getelementptr inbounds ([18 x i8], [18 x i8]\* @\_\_PRETTY\_FUNCTION\_\_.\_Z8funkcijai, i32 0, i32 0)) #3*

*[opcode]: 54*

*WORKING ON: call...*

*Parse call:*

*inst->getName():*

*inst->getOperand(0)->getName():*

*Real name:*

*Function name: \_\_assert\_fail*

*Argument:*

*Interval of confidence: Searching for:*

*Search unsuccessfull*

*nullptr*

*Function: \_\_assert\_fail*

*Interval of confidence: nullptr*

*DONE: call.*

*in [-inf, +inf]*

*[I]: unreachable*

*[opcode]: 7*

*-- not supported --*

*++++++++++++++++++++++++++++++++++*

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb8*

*[I]: br label %bb9*

*[opcode]: 2*

*WORKING ON: br...*

*DONE: br.*

*++++++++++++++++++++++++++++++++++*

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb9*

*[I]: %tmp10 = load i32, i32\* %tmp, align 4*

*[opcode]: 30*

*WORKING ON: load...*

*Searching for: tmp*

*DONE: load.*

*tmp10 in [-1995, 1995]*

*[I]: ret i32 %tmp10*

*[opcode]: 1*

*WORKING ON: ret...*

*Searching for: tmp10*

*DONE: ret.*

*RETURN\_VALUE in [-1995, 1995]*

*++++++++++++++++++++++++++++++++++*

*End of initialization.*

*Function name: \_Z8funkcijai*

*Number of parsed function instruction: 14*

Uspesna analiza funkcije funkcija. Uspesno parsirano 14 instrukcija.

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Intervals of confidence:*

*i32 %arg*

*nullptr*

*%tmp = alloca i32, align 4*

*Searching for: tmp*

*tmp in [-1995, 1995]*

*store i32 %arg, i32\* %tmp, align 4*

*Searching for: tmp*

*tmp in [-1995, 1995]*

*%tmp1 = load i32, i32\* %tmp, align 4*

*Searching for: tmp1*

*tmp1 in [-1995, +inf]*

*%tmp2 = icmp slt i32 -1995, %tmp1*

*Searching for: tmp2*

*tmp2 in [0, 1]*

*br i1 %tmp2, label %bb3, label %bb7*

*nullptr*

*%tmp4 = load i32, i32\* %tmp, align 4*

*Searching for: tmp4*

*tmp4 in [-1995, 1995]*

*%tmp5 = icmp slt i32 %tmp4, 1995*

*Searching for: tmp5*

*tmp5 in [0, 1]*

*br i1 %tmp5, label %bb6, label %bb7*

*nullptr*

*br label %bb9*

*nullptr*

*call void @\_\_assert\_fail(i8\* getelementptr inbounds ([18 x i8], [18 x i8]\* @.str, i32 0, i32 0), i8\* getelementptr inbounds ([6 x i8], [6 x i8]\* @.str.1, i32 0, i32 0), i32 3, i8\* getelementptr inbounds ([18 x i8], [18 x i8]\* @\_\_PRETTY\_FUNCTION\_\_.\_Z8funkcijai, i32 0, i32 0)) #3*

*Searching for:*

*in [-inf, +inf]*

*unreachable*

*nullptr*

*br label %bb9*

*nullptr*

*%tmp10 = load i32, i32\* %tmp, align 4*

*Searching for: tmp10*

*tmp10 in [-1995, 1995]*

*ret i32 %tmp10*

*Searching for: RETURN\_VALUE*

*RETURN\_VALUE in [-1995, 1995]*

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Searching for: RETURN\_VALUE*

*RETURN\_VALUE in [-1995, 1995]*

*Current parsed functions:*

*\_Z8funkcijai* Pre ove funkcije obradjena je funkcija funkcija. Koristimo njen interval poverenja

*RETURN\_VALUE in [-1995, 1995]*

*\_\_assert\_fail*

*nullptr*

*Functions argument initialization:*

*Initialization:*

*++++++++++++++++++++++++++++++++++*

*Basic block name: bb*

*[I]: %tmp = alloca i32, align 4*

*[opcode]: 29*

*WORKING ON: alloca...*

*DONE: alloca.*

*tmp in [-inf, +inf]*

*[I]: %tmp1 = alloca i32, align 4*

*[opcode]: 29*

*WORKING ON: alloca...*

*DONE: alloca.*

*tmp1 in [-inf, +inf]*

*[I]: store i32 0, i32\* %tmp, align 4*

*[opcode]: 31*

*WORKING ON: store...*

*DONE: store.*

*tmp in [0, 0]*

*[I]: %tmp2 = load i32, i32\* %tmp1, align 4*

*[opcode]: 30*

*WORKING ON: load...*

*Searching for: tmp1*

*DONE: load.*

*tmp2 in [-inf, +inf]*

*[I]: %tmp3 = call i32 @\_Z8funkcijai(i32 %tmp2)*

*[opcode]: 54*

*WORKING ON: call...*

*Parse call:*

*inst->getName(): tmp3*

*inst->getOperand(0)->getName(): tmp2*

*Real name: tmp3*

*Function name: \_Z8funkcijai*

*Argument: tmp2*

*Interval of confidence: Searching for: tmp2*

*tmp2 in [-inf, +inf]*

*Function: \_Z8funkcijai*

*Interval of confidence: RETURN\_VALUE in [-1995, 1995]*

*DONE: call.*

*tmp3 in [-1995, 1995]*

Obrada poziva (prethodno obradjene korisnicki definisane funkcije funkcija). Koriscenje odgovarajuceg intervala poverenja.

*[I]: ret i32 %tmp3*

*[opcode]: 1*

*WORKING ON: ret...*

*Searching for: tmp3*

*DONE: ret.*

*RETURN\_VALUE in [-1995, 1995]*

*++++++++++++++++++++++++++++++++++*

*End of initialization.*

*Function name: main*

*Number of parsed function instruction: 6*

Uspesno proparsirana funkcija main koja se sastoji od cetiri instrukcije.

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Intervals of confidence:*

*i32 %arg*

*nullptr*

*%tmp = alloca i32, align 4*

*Searching for: tmp*

*tmp in [0, 0]*

*store i32 %arg, i32\* %tmp, align 4*

*Searching for: tmp*

*tmp in [0, 0]*

*%tmp1 = load i32, i32\* %tmp, align 4*

*Searching for: tmp1*

*tmp1 in [-inf, +inf]*

*%tmp2 = icmp slt i32 -1995, %tmp1*

*Searching for: tmp2*

*tmp2 in [-inf, +inf]*

*br i1 %tmp2, label %bb3, label %bb7*

*nullptr*

*%tmp4 = load i32, i32\* %tmp, align 4*

*Searching for: tmp4*

*tmp4 in [-1995, 1995]*

*%tmp5 = icmp slt i32 %tmp4, 1995*

*Searching for: tmp5*

*tmp5 in [0, 1]*

*br i1 %tmp5, label %bb6, label %bb7*

*nullptr*

*br label %bb9*

*nullptr*

*call void @\_\_assert\_fail(i8\* getelementptr inbounds ([18 x i8], [18 x i8]\* @.str, i32 0, i32 0), i8\* getelementptr inbounds ([6 x i8], [6 x i8]\* @.str.1, i32 0, i32 0), i32 3, i8\* getelementptr inbounds ([18 x i8], [18 x i8]\* @\_\_PRETTY\_FUNCTION\_\_.\_Z8funkcijai, i32 0, i32 0)) #3*

*Searching for:*

*in [-inf, +inf]*

*unreachable*

*nullptr*

*br label %bb9*

*nullptr*

*%tmp10 = load i32, i32\* %tmp, align 4*

*Searching for: tmp10*

*tmp10 in [-1995, 1995]*

*ret i32 %tmp10*

*Searching for: RETURN\_VALUE*

*RETURN\_VALUE in [-1995, 1995]*

*%tmp = alloca i32, align 4*

*Searching for: tmp*

*tmp in [0, 0]*

*%tmp1 = alloca i32, align 4*

*Searching for: tmp1*

*tmp1 in [-inf, +inf]*

*store i32 0, i32\* %tmp, align 4*

*Searching for: tmp*

*tmp in [0, 0]*

*%tmp2 = load i32, i32\* %tmp1, align 4*

*Searching for: tmp2*

*tmp2 in [-inf, +inf]*

*%tmp3 = call i32 @\_Z8funkcijai(i32 %tmp2)*

*Searching for: tmp3*

*tmp3 in [-1995, 1995]*

*ret i32 %tmp3*

*Searching for: RETURN\_VALUE*

*RETURN\_VALUE in [-1995, 1995]*

Povratna vrednost funkcije main u odgovarajucem intervalu poverenja.

*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*Searching for: RETURN\_VALUE*

*RETURN\_VALUE in [-1995, 1995]*

*Current parsed functions:* Proparsirane funkcija main I funkcija funkcija. Dodatno I poziv assert. Odgovarajuci intervali poverenja povratnih vrednosti ovih funkcija dati su u nastavku.

*\_Z8funkcijai*

*RETURN\_VALUE in [-1995, 1995]*

*\_\_assert\_fail*

*nullptr*

*main*

*RETURN\_VALUE in [-1995, 1995]*