

## Sinteza programa

Anja Ivanišević  
Ivan Ristović  
Milana Kovačević  
Vesna Katanić

maj 2018.

# Šta je sinteza programa?

- ▶ Oblast koja se bavi automatskim generisanjem programa

- ▶ Vizija:

*Nema više programiranja, sve se generiše automatski!*

- ▶ Pričaćemo o:
  - ▶ primenama
  - ▶ izazovima
  - ▶ tehnikama

# Primene

- ▶ Priprema podataka
- ▶ Popravka koda
- ▶ Sugestije prilikom kodiranja
- ▶ Grafika
- ▶ Superoptimizacija
- ▶ Konkurentno programiranje

# Primene - Priprema podataka

- ▶ Proces pripreme često obuhvata sledeće operacije nad podacima:
  - ▶ izvlačenje
  - ▶ transformacija
  - ▶ formatiranje
- ▶ Manipulisanje niskama ili izmene samih tipova podataka
- ▶ Korisnici se zamaraju pisanjem skriptova ili makroa
- ▶ PBE je idealan za ovakav posao

## Primene - Popravka koda

- ▶ Računanje modifikacija programa  $P$  koje stvaraju nov program  $P'$  takav da zadovoljava specifikaciju  $\phi$
- ▶ Ubacuju se alternativni izbori za izraze u programu
- ▶ Tehnikama programske sinteze izraza pronađu izrazi koji program dovode u oblik koji zadovoljava  $\phi$

# Primene - Popravka koda - Primer

inb	Ulaz		Izlaz	
	usep	dsep	expected	actual
1	0	100	0	0
1	11	110	1	0
0	100	50	1	1
1	-20	60	1	0
0	0	10	0	0

```
int buggy(int inb, int usep, int dsep)
{
    int bias;
    if (inb)
        bias = dsep; //fix: bias = usep+100
    else
        bias = usep;
    if (bias > dsep)
        return 1;
    else
        return 0;
}
```

**Slika:** Primer koda sinteziranog od strane programa *SemFix* koristeći skup ulaznih i izlaznih test primera.

# Primene - Sugestije prilikom kodiranja

- ▶ Podrške okruženja za rad:
  - ▶ *IntelliSense* za *MS Visual Studio*
  - ▶ *Content Assist* za *Eclipse*
- ▶ Tehnike za generisanje čitavih jedinica koda:
  - ▶ *statistički modeli*
  - ▶ *dopuna usmerena tipovima* (eng. *type-directed completion*)
  - ▶ ostale tehnike (koriste ih *InSynth* i *Bing Developer Assistant*)

# Primene - Grafika

- ▶ Konstrukcija ponovljenih objekata
- ▶ Korišćenjem PBE, korisnik prikaže par primera a sintezer predviđa naredne objekte u nizu
- ▶ Interaktivno postavljanje grafičkih objekata preko grafičkog interfejsa
- ▶ Efikasnija geometrijska izračunavanja, brže animacije



# Primene - Superoptimizacija

- ▶ Kreiranje optimalnog poretka instrukcija mašinskog koda zarad dobijanja na performansama
- ▶ Primer

originalni kod:  $prosek = \frac{x+y}{2}$

optimizovani kod:  $prosek = (x \mid y) - ((x \oplus y) \gg 1)$

- ▶ Jedan od načina da se kod automatski optimizuje je korišćenje *enumerativne pretrage*

# Primene - Konkurentno programiranje

- ▶ Pomoć pri pisanju bezbednog kompleksnog višenitnog koda
- ▶ *Sinteza vođena apstrakcijom:*
  - ▶ Pravi se apstraktna reprezentacija programa u apstraktnom domenu
  - ▶ Proverava se da li postoji kršenje postavljene specifikacije programa (obično trka za podacima)
  - ▶ Ukoliko postoji prekršenje, menjamo ili apstrakciju (npr. sužavanjem domena) ili sam program dodajući sinhronizacione konstrukte
  - ▶ Ovaj postupak se ponavlja sve dok se ne nađe program koji može biti verifikovan apstrakcijom

# Izazovi

- ▶ Sa visokog nivoa, problem sinteze se može razložiti na dva potproblema:
  - ▶ Definisanje specifikacija željenog programa
  - ▶ Pretraživanje prostora mogućih programa u potrazi za onim koji zadovoljava definisane specifikacije
- ▶ Prostor programa se povećava eksponencijalno brzo u odnosu na veličinu željenog programa

# Izazovi - Definisanje specifikacija

- ▶ Većina programa koji se danas koriste su previše komplikovani da bi se u potpunosti opisali bilo formalnim bilo neformalnim metodama
- ▶ Potrebno je omogućiti korisniku da definiše željeni program do neke tačke, a da kasnije tokom sinteze, interaktivno sa računarom, postepeno dolazi do rešenja
- ▶ *FlashFill*

# Izazovi - Pretraživanje prostora programa

- ▶ Prostor programa - skup koji sadrži sve moguće programe koji se mogu napisati
- ▶ Pretraga ovog skupa znači nalaženje programa koji zadovoljava specifikacije
- ▶ Tehnike pretrage se mogu zasnivati na:
  - ▶ Enumerativnoj pretrazi
  - ▶ Dedukciji
  - ▶ Tehnikama sa ograničenjima
  - ▶ Induktivnim i statističkim metodama

# Izazovi - Pretraživanje prostora programa - Enumerativna pretraga

- ▶ Jedna od najefikasnijih tehnika za generisanje malih programa
- ▶ Tehnike *čišćenja*
- ▶ Prvo se na neki način opiše prostor pretrage u kome se nalazi željeni program
- ▶ Kada se mogući programi numerišu po osobinama, mogu da se odmah odbace oni koji ne zadovoljavaju specifikaciju
- ▶ Enumerativna tehnika je poluodlučiva

## Izazovi - Pretraživanje prostora programa - Deduktivna pretraga

- ▶ Pretpostavka da postoji celokupna formalna specifikacija željenog programa
- ▶ Rešenje se sintetiše postupkom dokazivanja teorema, logičkim zaključivanjem i razrešavanjem ograničenja
- ▶ Deduktivna pretraga je pretraga odozgo nadole
- ▶ Koristi tehniku podeli-pa-vladaj
- ▶ Deljenje problema na potprobleme nije moguće u opštem slučaju
- ▶ Kombinovanje deduktivne pretrage sa enumerativnom

# Izazovi - Pretraživanje prostora programa - Tehnike sa ograničenjima

- ▶ Tehnike prilagođavanja datim ograničenjima
- ▶ Dva velika koraka:
  - ▶ Generisanje ograničenja
  - ▶ Razrešavanje ograničenja



# Izazovi - Pretraživanje prostora programa - Statistička pretraga

- ▶ Koriste neku vrstu statistike kako bi došle do rešenja
- ▶ *Mašinsko učenje*
- ▶ *Genetsko programiranje*
- ▶ *Probabilističko zaključivanje*

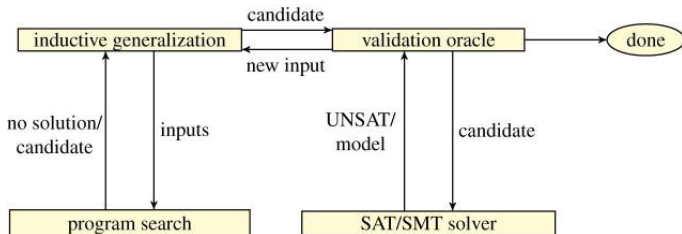
# Izazovi - Pretraživanje prostora programa - Induktivna pretraga

- ▶ Može se smatrati kao nadogradnja tehnike pretrage sa ograničenjima
- ▶ Prilikom svake iteracije se generišu ograničenja
- ▶ Rešavačem se dođe do mogućeg rešenja a zatim se ispita da li je ono zadovoljavajuće kao opšte rešenje
- ▶ Može da koristi tehnike mašinskog učenja - *Aktivno učenje*
- ▶ *CEGIS*

- ▶ Program se može sintetisati tako što se definiše specifikacija i zapiše u vidu formule koja se prosledi SMT rešavačima
- ▶ Koji je najmanji podskup ulaza koji je potrebno razmatrati da bi se sintetisao program koji zadovoljava date specifikacije?
- ▶ Korišćenjem SMT rešavača dolazimo do svih mogućih implementacija željenog programa koristeći sve ulaze koji su razmatrani do tog trenutka
- ▶ Paralelno, drugi SMT rešavač pronalazi kontraprimer koji pokazuje da poslednji sintetisani program nije rešenje

# CEGIS - Arhitektura

- ▶ Pretraga vođena kontraprimerima (eng. *counterexample-guided*)
- ▶ CEGIS se sastoji iz dve faze: *induktivne sinteze* i *verifikacije*
- ▶ U fazi sinteze se pronalazi program kandidat koji može da zadovolji specifikaciju
- ▶ U fazi verifikacije se proverava da li taj kandidat zaista zadovoljava specifikaciju



Slika: CEGIS petlja

- ▶ Da bismo u potpunosti definisali CEGIS sintezu programa, potrebno je odgovoriti na sledeća pitanja:
  - ▶ Kako treba da izgleda specifikacija traženog programa?
  - ▶ Kako ćemo vršiti sintezu programa kandidata?
  - ▶ Kako da proverimo da li program kandidat zadovoljava specifikacije?
  - ▶ Kako da prosledimo povratne informacije za buduće kandidate?

# CEGIS - Sinteza vodjena uzorom

- ▶ *Oracle-guided synthesis*
- ▶ Pretpostavlja da imamo implementaciju programa koji želimo da sintetišemo - uzor
- ▶ Uzor se tretira kao crna kutija
- ▶ Novi test primeri se kreiraju generišući proizvoljne ulaze a od uzora se dobijaju odgovarajući izlazi za svaki prosleđeni ulaz

# CEGIS - Stohastička superoptimizacija

- ▶ Pretražuje se prostor programa i traži se brži ili efikasniji ekvivalent polaznog programa
- ▶ Takođe se pretpostavlja da imamo implementaciju programa kao specifikaciju
- ▶ Koristi se *MCMC* (eng. *Markov-chain Monte Carlo sampling*) kao mera udaljenosti

## CEGIS - Enumerativna pretraga

- ▶ Za specifikaciju se koristi konačan skup test primera
- ▶ Pretpostavlja se da je dostupna gramatika koja opisuje ciljani jezik (`add(x, sub(x,y))`)
- ▶ Programi se dele prema dubini
- ▶ Sinteza kreće od dubine 0 i numerišu se svi programi na toj dubini
- ▶ Na dubini  $k$ , ispituju se svi programe koji imaju oblik `operacija(a,b)`, gde su  $a$  i  $b$  bilo koji izrazi dubine  $k - 1$



# Zaključak

- ▶ Uspešno sintetisanje manjih programa može značajno da olakša rad programerima
- ▶ Da li će programeri moći da prestanu da govore računarima **kako** da rade, već da se fokusiraju na to da im kažu **šta** treba da urade
- ▶ Najveći potencijal ima induktivna sinteza programa.
- ▶ CEGIS je, kao vodeći predstavnik ove grupe, u praksi pokazao neočekivano dobre rezultate

# Pitanja

???

TODO

Hvala na pažnji!