

ANALIZA TIJEKA IZVOĐENJA PROGRAMA

- ispituje upravljačku strukturu međukoda definiranu naredbama petlji, grananja,...
- 2 osnovna pristupa:
 - ANALIZA DOMINACIJE
 - ANALIZA STRUKTURE
- obje **grade graf tijeka izvođenja programa**
- Na temelju grafa tijeka izvođenja
 - **analiza dominacije:**
 - određuje strukturu petlji
 - prethodi analizi toka podataka zasnovanoj na iterativnim postupcima
 - **analiza strukture:**
 - više različitih podgrafova
 - slijedni blokovi
 - blokovi s petljom
 - if-then-else blokovi
 - prethodi analizi toka podataka zasnovanoj na eliminacijskim postupcima

GRAF TIJEKA IZVOĐENJA PROGRAMA

- naredbe međukoda u **osnovne blokove** -> niz naredbi koji se slijedno izvode
- izvođenje
 - započinje isključivo **početnom naredbom**
 - osim završne naredbe ni jedna druga naredba ne upravlja tijekom izvođenja programa
- čvorovi
 - osnovni blokovi
- usmjerene grane
 - tijek izvođenja programa
 - povezuje čvorove ako i samo ako neposredno nakon izvedene završne naredbe jednog bloka započinje izvođenje početne naredbe drugog bloka
- **ALGORITAM:**
 - **Početne naredbe:**
 - početna naredba grananja
 - naredba na koju se usmjeruje tijek izvođenja programa
 - naredba koja neposredno slijedi naredbu grananja
 - **Osnovni blok** čini početna naredba i sve naredbe između te i sljedeće početne naredbe
 - **Blokovi A i B povezuju se usmjerenom granom** ako i samo ako vrijedi:
 - Završna naredba A usmjeruje tijek izvođenja na početnu naredbu bloka B
 - Blok B je neposredno iza bloka A koji ne završava bezuvjetnim preusmjeravanjem tijeka izvođenja programa

ANALIZA DOMINACIJE

- Za potrebe optimiranja potrebno je **odrediti strukturu petlji**
- Potrebno je odrediti na primjer:
 - u postupku koji iz petlje izlučuje višekratno računanje izraza nepromjenjive vrijednosti
 - optimiranja spregnutih varijabli
 - paralelnog izvođenja unutar zadane petlje
- Struktura petlje definira se **podgrafom tijeka izvođenja**
 - Čvorovi i grane određuju se na temelju **relacije dominacije**
- Sada prvo moramo definirati relaciju dominacije:
 - Ako čvor **d** prethodi čvoru **a** **na bilo kojem putu** (ovo za bilo kojem putu će nam biti prilično važno) koji z Početni a **započinje početnim čvorom a završava čvorom a** , onda **čvor d DOMINIRA čvorom a**.
 - Definiramo skup **dom(a)** koji je **skup svih dominatora čvora a**
 - Nadalje definiramo da je **blok sam sebi dominator** (definira se zbog načina na koji se metoda dominacije koristi u ostalim metodama optimiranja)

- Nadalje definiramo neposredno dominiranje:
 - Čvor **d je neposredni dominator čvora a** ako i samo ako ni za koji čvor c ($c \neq d$, $c \neq a \rightarrow$ isključujem dio definicije u kojoj kažemo da je čvor sam sebi dominator) ne vrijedi da je c je dom(a) i d je dom(c)
- Relacija dominacije najčešće se definira **stablom dominacije**:
 - Ako je čvor **d neposredni dominator čvora a onda je u grafu dominacije d je neposredni prethodnik čvora a.**
 - Čvor **d striktno dominira čvorom a** ako i samo ako je d u dom(a) i $d \neq a$ (time isključujem dio definicije da je čvor sam sebi dominator)
 - Analiza dominacije koristi i relaciju **postdominacije**:
 - Ako čvor p slijedi čvor a na bilo kojem putu koji započinje sa a i završava sa zadnjim čvorom Završni , onda čvor p postdominira čvorom a.
- **Ako odredišni čvor o dominira nad izvorišnim čvorom i, onda je grana i->o povratna.**
- Ako je r->g povratna grana , onda se čvorovi koji čine prirodnu petlju određuju:
 - čvor g -> glava petlje
 - čvor r -> rep petlje
 - čvorovi petlje su oni čvorovi na putu do r , uz uvjet da na putu nije čvor g (ovo nam je ograničenje da se ne vrtimo u krug)
- Budući da glava g dominira repom r povratne glave r->g, svi prethodnici repa r koji nisu prethodnici g su čvorovi petlje.
- Određivanje čvorova petlje započinje repom r. U skup se stave svi neposredni prethodnici repa r. Postupak se nastavlja za sve čvorove koji su dodani u skup čvorova petlje. Traže se njihovi neposredni prethodnici, ali uz ograničenjeda oni nisu prethodnici glave g. Ako skup čvorova nije moguće proširiti algoritam se zaustavlja.
- **NEDOSTATAK:**
 - **nemogućnost učinkovitog ponavljanja postupka tijekom optimiranja**
 - postupci optimiranja mijenjaju strukturu programa. -> potrebno ponovo ponoviti cijelu analizu dominacije, ako promijenimo ista moramo ponovo analizirati što je vremenski vrlo zahtjevno.

ANALIZA STRUKTURE

- gradi **upravljačko stablo na temelju raščlambe grafa tijekom izvođenja progama**
- **upravljačko stablo** -> hijerarhijski zapis strukture grafa tijekom izvođenja
- **podgrafovi uzorci** (slični su nam ko uzorci za zamijenu kod parsiranja od dna prema vrhu, a i tu gradimo upravljačko stablo od dna prema vrhu):
 - **slijedni čvorovi**
 - **čvorovi petlje**
 - **if-then-else čvorovi**
- Analiza obilazi graf tijekom izvođenja, **traži podgrafove uzorke** i kad ih nađe onda ih **zamijeni jednim zamijenskim čvorom** (kao da smo napravili redukciju) i **gradi upravljačko stablo**
- **Listovi upravljačkog stabla su čvorovi grafa izvođenja programa**
- Zamijenom podgrafova uzorka zamjenskim čvorom **smanji se broj čvorova** grafa tijekom izvođenja programa, a u upravljačko stablo doda se zamijenski čvor koji je neposredni prethodnik čvorova podgrafova uzorka
- sada tu objasnim na našem grafu tijekom izvođenja programa
- neka koristimo 3 podgrafova uzorka ona gore,... i igram se s tim.
- Gradnja upravljačkog stabla ide od dna prema vrhu.

ANALIZA TOKA PODATAKA

- **izravan nastavak tijekom izvođenja programa**
- na temelju grafa izvođenja kojeg smo sagradili prije - analizira način uporabe podataka primjenom ITERATIVNOG ili ELIMINACIJSKOG postupka
- **ITERATIVNI** -> analizu dominacije
- **ELIMINACIJSKI** -> analizu strukture
- **analiza toka podataka:**
 - **obilazi stablo tijekom izvođenja** programa različitim smjerovima

- unaprijedna analiza
- unazadna analiza
- dvosmjerna analiza
- **analiziraju se:**
 - doseg definicije varijable
 - važeće definicije varijable
 - dostupni izrazi
 - žive varijable
 - prenošenje naredbe preslikavanja
 - prenošenje konstante
 - višekratno računanje izraza
- Prije nego što objasnimo ove analize potrebno je **definirati još nekoliko pojmova:**
 - **ako je u bloku (čvoru) zadana naredba pridruživanja onda blok (čvor) definira (mijenja) vrijednost one varijable koja je određište naredbe pridruživanja**
 - ako je varijable **operand onda blok (čvor) koristi njenu vrijednost**
 - ako ni jedan čvor puta ne mijenja vrijednost varijable, onda se kaže da je ni zadani put ne mijenja
 - ako barem jedan čvor koristi vrijednost varijable, onda se se kaže da je koristi i zadani put

• **DOSEG DEFINICIJE VARIJABLE:**

- **unaprijedna analiza** kojom se određuje **skup čvorova koji koriste zadanu definiciju varijable** (logično je da je to unaprijedna analiza jer ako u nekom gornjem čvoru definiramo neku varijablu onda moram gledati do koje dubine se neće mijenjati definicija neke varijable)
- Doseg definicije varijable definira se na sljedeći način :
 - neka čvor **d** definira vijednost varijable i neka je čvor u koristi. Ako postoji bar jedan put od čvora **d** do čvora **u** i ako ni jedan drugi čvor nanovo ne definira vrijednost te varijable onda je čvor **u** u dosegu definivije čvora **d**
 - dodeg definicije moguće je zapisati **u bit vektor**. **U bit vektoru zasebni bitovi dodjeljuju se različitm definicijama varijable.**
 - Vrijednost vektora se racuna na svim **ulazima i izlazima čvorova**
 - Ako je **ulaz ili izlaz u dosegu definicije varijable onda je vrijednost bita 1**
 - analizira se **iterativnim postupkom**

• **ITERATIVNI POSTUPAK RAČUNANJA DOSEGA DEFINICIJE VARIJABLE**

- to je iterativni postupak zato se postupak računanja Ulaza i Izlaza ponavlja **dok ne uđemo u stacionarno stanje.**
- za potrebe iterativnog postupka analize toka podataka koristi se **bit vektor veličine n gdje n broj različitih definicija varijabli** u grafu tijeka izvođenja programa
- Da bi to odredili prvo pogledamo kako izgleda izgrađeni graf tijeka izvođenja programa i pogledamo u kojim se blokovima definiraju vrijednosti varijable te svugdje gdje se definiraju vrijednosti varijable nanovo zapišemo u bitvektor.
- Vekrori čije nam vrijednosti pomažu da izračunamo vrijednosti bit vektora na ulazu i izlazu iz čvora su :
 - **gen(a)** -> jedinicama se označe sve definicije koje su zadane u bloku a
 - **čuva(a)** -> jedinicama se označe sve definicije koje blok a ne mijenja.
- **Ulaz(a) i Izlaz(a)**
 - **Izlaz(a) = Gen(a) V (Ulaz(a) i Čuva(a))**
 - prvo pogledam što mi se definira u zadanom čvoru i pogledam ono što sam naslijedila od ulaza, dakle sve ono što je došlo na ulaz a nije se promijenilo u zadanom čvoru. Da bi odredili koje se varijable nisu mijenjale u zadanom čvoru radimo operaciju logičko i s bit vektorom Čuva (a) -> on nam ustvari služi kao maska onoga što smijemo proslijediti na izlaz)
 - **Ulaz (a) = Vi Izlaz (i)** -> ulaz je logičko ili svega onoga što sam dobila od neposrdnih prethodnika čvora a na ulaz
 - **Početne vrijednosti** bit vektora ulaza i izlaza su sve nule
- **Postupak iteracije započinje računanjem vrijednosti vektora Ulaz() i Izlaz()** početnim vrijednostima. Analiza je unaprijedna pa krenemo od početak. i završavamo sa završnim...
- U nastavku iteracije koriste se novo izračunate vrijednosti bit vektora. **Ako se vrijednosti bit vektora ne promijene, onda se ostupak iteracije zaustavlja.**

- Kao što možemo vidjeti u prvoj iteraciji u primjeru nemamo izračunatu vrijednost Izlaza(B6) koja je ulaz(B4). U drugoj iteraciji je ta vrijednost dostupna iz prošlog koraka pa je Ulaz(B4) je logičko ili izlaza B6 i B3.
- Ako pogledamo vrijednosti vektora onda **možemo vidjeti dokle doseže vrijednost neke varijable definirane u odrđenom bloku**. Tako definicija varijable a0 u bloku B1 doseže do bloka B6

• ODREĐIVANJE VAŽEĆIH DEFINICIJA VARIJABLE

- samo obrnuto od dosega varijable
- **unazadna analiza**
- određuje se **skup čvorova koji definiraju vrijednost varijable**
- Definira se na sljedeći način:
 - Neka čvor **d** **definira vrijednost varijable** i neka je čvor **u** **koristi**. Ako postoji **barem jedan put od čvora d do čvora u** i ako **ni jedan čvor na tom putu ne definira nanovo vrijednost varijable onda je definicija čvora d važeća za čvor u**.
- u bit vektor zapisuju se različite uporabe varijable

• ODREĐIVANJE DOSTUPNIH IZRAZA

- **unaprijedna analiza**
- Definiiramo je na sljedeći način:
 - Ako se **zadani izraz računa na svim putevima od ulaznog čvora do čvora a** i ako ni jedan put od mjesta računanja tog izraza do čvora a ne mijenja vrijednosti varijabli koje su dio zadanog izraza, onda je zadani izraz dostupan u čvoru a.
 - Dakle ako se neki izraz računa u više dijelova oni moraju kroz nepromijenjeni put doći bez ikakvih izmjena.
- U bit vektor idu različite uporabe varijabli