Ekvivalencija determinističkog i nedeterminističkog Turingovog odlučivača

Nora Berdalović

26. lipnja 2024.

Teorem 1. Nedeterministički Turingov odlučivač sa svojstvom da svako njegovo izračunavanje stane u $q_{\checkmark}$ ili q_X ekvivalentan je determinističkom.

Dokaz. Za dani nedeterministički Turingov odlučivač $\mathcal{N}$, simuliramo njegov rad na determinističkom. Koristimo četiri glavne trake i eventualne pomoćne: kao što znamo, svaki TO s proizvoljnim konačnim brojem traka je ekvivalentan jednotračnom.

Rad TO ilustriramo sa stablom, gdje svaki čvor predstavlja jednu konfiguraciju. Djeca nekog čvora one su konfiguracije u koje je iz njega moguće prijeći. Korijen je početna konfiguracija $(q_0, 0, w\llcorner\!\lrcorner\llcorner\!\lrcorner\ldots)$, za ulaznu riječ w.

Zbog toga što je broj stanja te broj znakova u abecedi konačan, znamo da svaki čvor ima gornju ogradu na broj djece. Taj indeks grananja označavamo s b. Dakle, imamo konačan broj prijelaza između konfiguracija nad kojima napravimo neki uređaj (npr. leksikografski) te ih označavamo redom s δ_1, ... δ_b. Adresa nekog čvora bit će niz prijelaza koje je potrebno izvršiti iz korijena nadalje kako bi se došlo do čvora.

Za razliku od DTO, koji će u svom radu prolaziti samo jednom granom stabla, NTO može prolazi svim granama simultano. Kako bi ga simulirali, DTO će također morati proći cijelim stablom.

Iz pretpostavke teorema znamo da će $\mathcal{N}$ završiti izračun u jednom od dva stanja $q_{\checkmark}$ ili q_X. Ako u simulaciji njegovog rada dođemo do $\mathcal{N}.q_{\checkmark}$, prihvaćamo riječ. Međutim, ako dođemo do $\mathcal{N}.q_X$, nemamo dovoljno informacija da znamo znači li to konačno odbijanje riječi, ili samo trenutne grane – moguće je da postoji druga grana gdje ćemo doći do $\mathcal{N}.q_{\checkmark}$. Zbog toga je potrebno pratiti dosad obiđene čvorove i njihova stanja, jer tek kad dođemo do stanja $\mathcal{N}.q_X$ u svim granama stabla, znamo da DTO mora odbiti riječ.

Na prvu traku DTO zapisujemo ulaznu riječ i ostavljamo ju nepromjenjenu.

Drugu traku koristimo za simulaciju rada NTO u određenoj grani.

Na trećoj traci, koristeći adrese, pratimo čvorove koje smo dosad obišli te jesu li u stanju $\mathcal{N}.q_X$, tj. jesu li odbijeni ili ne. Odbijene adrese označavamo sa znakom X poslije njihovog zapisa. Između različitih adresa pišemo delimiter #. Stablo obilazimo sa breadth first search, pa će zapis na traci biti oblika $\delta_1\#\delta_2\#.....\delta_b\#\delta_1\delta_1X\#....\delta_1\delta_b\#\delta_2\delta_1...$

Na četvrtoj traci nalazi se trenutna adresa, koja je na početku δ_1 (prvo dijete korijena).

Algoritam, zapisan niže u pseudokodu, funkcionira na sljedeći način:

Nakon zapisivanja ulazne riječi na traku t_1, stroj provjerava poseban slučaj kad je izračun gotov već u korijenu stabla, odnosno početno stanje jednako je nekom završnom. Ako je to $\mathcal{N}.q_{\checkmark}$, prihvaća riječ, a ako je $\mathcal{N}.q_X$, odbija ju – znamo da stroj završava rad odmah na početku, pa nema daljnjih konfiguracija za provjeriti.

Ako to nije slučaj, stroj ulazi u petlju. Za trenutnu adresu, provjerava nalazi li se u grani koja je već odbijena. To postiže čitanjem zapisa na t_3, gdje se nalaze sve dosad prijeđene adrese. TO čita prvi prijelaz u trenutnoj adresi sa t_4. Uzima taj prijelaz te čita t_3 dok ne naiđe na njega. Ako je idući pročitani simbol nakon tog njega X, znači da je granu u kojoj se adresa nalazi već odbio, kod nekog pretka čvora na trenutnoj adresi, pa ju nije potrebno dalje analizirati. Isto radi za sve početne komade adrese (početna dva prijelaza u adresi, početna tri itd.). Ako nijedan komad na t_3 ne slijedi X, znamo da grana u kojoj je trenutna adresa nije ranije bila odbijena.

Npr. za adresu $\delta_3\delta_1\delta_2\delta_1$, čitat će t_3 dok ne naiđe na adresu δ_3. Ako nije praćena sa X, nastavlja čitanje i isto radi za adresu $\delta_3\delta_1$, i zatim $\delta_3\delta_1\delta_2$. Cijelu adresu (sva četiri prijelaza) nije potrebno provjeravati, jer se sigurno ne nalazi na traci: svaki čvor stabla posjećujemo samo jednom.

Ako trenutna adresa nije odbijena, DTO na t_2 simulira rad $\mathcal{N}$ od početne konfiguracije do one koja je na toj adresi. To znači da će, s inicijalnom pozicijom glave na početku t_2, redom izvršavati prijelaze koji čine adresu, a koje može čitati s t_4. Adresu zatim zapisuje na t_3, tako što čita simbole na traci dok ne dođe do prve praznine, zapiše adresu i znak # koji će ju odvajati od iduće, u slučaju da na kraju simulacije nismo u stanju $\mathcal{N}.q_X$. Ako jesmo, tad zapisuje $X\#$, označavajući da je ta grana odbijena. Naravno, ako čvor na toj adresi ne postoji (čvorovi mogu imati manje od b djece, ali stroj uvijek provjerava svih b mogućnosti), također uzima da je grana odbijena. S druge strane, ako smo na kraju u stanju $\mathcal{N}.q_{\checkmark}$, tad stroj odmah prihvaća riječ i završava s radom.

Nakon toga, stroj provjerava je li u situaciji da je svaka grana odbijena, pa zna da mora odbiti riječ. To može pamtiti u varijabli `odbijeno`, koju na početku petlje stavlja na 1, te mijenja na 0 čim naiđe na granu koja nije još odbijena. Provjera se obavlja počevši od čvorova koji su djeca korijena, tj. na adresama $\delta_1, \delta_2, ...\delta_b$, te s glavom čitača na početnom mjestu trake t_3. Stroj čita traku dok ne naiđe na danu adresu. U slučaju da ne naiđe na nju prije prvog praznog znaka, postavlja varijablu `odbijeno` na 0 – znači da granu nije još ni obišao pa ju sigurno nije ni odbio. Ako je adresa na traci i idući pročitani znak iza nje je X, odmah zna da je grana odbijena, pa opet postavlja `odbijeno` $= 0$ i završava provjeru. Ako nije, pročitan znak će biti #, a pozicija glave iza tog #. Sad je potrebno provjeriti jesu li sva djeca čvora na danoj adresi bila odbijena, što bi značilo da je i on sam, tj. njegova cijela grana, odbijena. Stroj dakle u petlji za $i = 1, ...b$ rekurzivno provjerava je li čvor na adresi oblika `trenutna_adresa`δ_i odbijen, počevši čitanje od trenutne pozicije glave nadesno. Čim nađe jedno dijete za koje se nije ušlo u stanje $\mathcal{N}.q_X$, zna da nije cijela grana odbijena, pa postavlja varijablu `odbijeno` na 0 i završava provjeru.

Npr. za provjeru je li odbijena grana δ_1, ako na traci nije zapisano $\delta_1 X$, stroj provjerava grane $\delta_1\delta_1$, $\delta_1\delta_2$, ... $\delta_1\delta_b$. Ako nije naišao ni na $\delta_1\delta_1 X$, provjerava $\delta_1\delta_1\delta_1$, ... $\delta_1\delta_1\delta_b$ itd. Zbog načina na koji smo zapisivali adrese na traku, djeca nekog čvora (ako su uopće zapisana) uvijek će biti zapisana desno od njega, pa je npr. nakon provjere za $\delta_1\delta_1$ dovoljno čitati od $\delta_1\delta_1\#$ nadesno.

Ako je poslije ovog računa ostalo `odbijeno` $= 1$, znači da stroj nije naišao ni na jednu granu NTO u kojoj riječ nije odbijena. Došao je do stanja $\mathcal{N}.q_X$ u svim granama – može odbiti riječ.

Na kraju, ako još nije ni prihvatio ni odbio riječ, stroj prije ponovnog ulaska u petlju računa sljedeću adresu. Prvo čita trenutnu adresu sa trake t_4 sve dok ne dođe do praznog znaka, pa se vraća unazad jedno mjesto i čita zadnji prijelaz u adresi, koji pohranjuje u varijablu `zadnji`. Taj prijelaz zatim može obrisati s kraja adrese (tj. napisati prazne znakove preko njega). Ako je `zadnji` različit od δ_b, potrebno ga je samo povećati za 1 te zapisati nazad na pomoćnu traku, od prve praznine nadalje. Inače se pomiče na početak trake, čita adresu koja je zapisana (to je trenutna adresa bez svog zadnjeg prijelaza), te rekurzivno na isti način računa i zapisuje njezinu sljedeću adresu te na kraj prijelaz δ_1.

Npr. za $b = 3$, sljedbenik adrese $\delta_3\delta_1\delta_2$ bit će $\delta_3\delta_1\delta_3$ (zadnji dio uvećan za 1), dok će za $\delta_1\delta_3\delta_3$ sljedbenik biti $\delta_2\delta_1\delta_1$.

Pseudokod:

```
write($t_1, w$);
if ($q_0$==$\mathcal{N}.q_{\checkmark}$) prihvati;
else if ($q_0$==$\mathcal{N}.q_X$) odbij;
write ($\delta_1$,t$_4$);
adresa=read($t_4$);
while True:
   odbijeno=1;
   if (not gotovo(adresa)):
      $s$=$\mathcal{N}.q_0$;
      while(read($t_4$)==($p, \alpha, q, \beta, d$))$\neq$ ␣:
         if (read($t_2$)=$\alpha$ and $s$=$p$):
            $s$=$q$;
            write($t_2, \beta$); move($t_2, d$); move($t_4, 1$);
   zapisi(adresa,s);
   if($s$==$\mathcal{N}.q_{\checkmark}$) prihvati;
   for (i=1, i<=b, i++):
      // provjera za svaku početnu granu je li odbijena
      if (not kraj($\delta_i$,0)):
         odbijeno=0;
         break;
   if (odbijeno==1) odbij; //znači da su sve grane odbijene
   adresa=sljedeca(adresa);
```

Korištene funkcije:

```
gotovo(adresa):
   dio="";
   while True:
      $\delta_j$=read($t_4$);
      dio+=$\delta_j$;
      if (dio==adresa) break;
      do (read($t_3$))
         until (read($t_3$)==dio);
```

```
    if (read(t_3, X)):
        return True;
    return False;


zapisi(adresa,s):
    do (read(t_3))
        until (read(t_3)==␣);
    move(t_3, −1);
    write(adresa,t_3);
    if (s==𝒩.q_X):
        write(X#, t_3);
    else:
        write(#, t_3);
    return;


sljedeca(adresa):
    premotaj t_4 na početak;
    do(read(t_4))
        until(read(t_4)==␣);
    move(t_4, −2);
    δ_zadnji=read(t_4);
    move(t_4, −1);
    write(t_4,␣);
    premotaj t_4 na početak;
    nova_adresa=
        do(read(t_4))
            until(read(t_4)==␣); //adresa bez zadnjeg člana
    if (δ_zadnji ≠ δ_b):
        δ_zadnji=δ_zadnji+1;
        write(δ_zadnji+1,t_4);
    else:
        premotaj t_4 na početak;
        write(sljedeca(nova_adresa,t_4);
        write (δ_1,t_4);
    return;
```

```
kraj(adresa,mjesto_na_traci):
   gotovo=0;
   move(t_3,mjesto_na_traci);
   do(b=read(t_3))
      until (b==adresa or b==␣)
   if (b==adresa):
      if (read(t_3)==X):
         return 1;
      else:
         read(t_3); //pročitamo #
         novo_mjesto=trenutna pozicija glave;
         for (i=1, i<b, i++):
            gotovo=return kraj(adresaδ_i,novo_mjesto);
            if (gotovo==0) break;
   return gotovo;
```

□