

Projekat: Simulacija DFA (Deterministic Finite Automata)

Aid Hodžić, TKN, Odsjek za matematiku, PMF Sarajevo

Abstract – U tekstu je opisan način na koji je izvršena simulacija DFA. Simulacija je isprogramirana u C++ programskom jeziku. Glavni zadatak je da saznamo da li se ulazni string prihvata ili ne.

1. UVOD

Deterministički konačni automat (DFA) je konačni automat u kojem za svaki par stanja i ulaznog znaka postoji jedan i samo jedan prijelaz u sljedeće stanje. Deterministički konačni automati prepoznaju skup regularnih jezika.

DFA prima niz ulaznih znakova, i za svaki ulazni znak obavlja prijelaz u stanje koje određuje funkcija prijelaza. Kada je pročitani cijeli ulazni niz, prihvatit će ili odbiti niz znakova ovisno o tome je li DFA u prihvatljivom ili neprihvatljivom stanju.

Problem

Ima su dvije informacije. Jedna se nalazi smještena u fajlu i sadrži informacije o konačnom automatu. Druga informacija je ulazni string koji korisnik unosi sa tastature prilikom samog izvođenja simulacije. Simulacija treba da izbacuje da li je uneseni string prihvaćen ili ne.

2. OPIS RJEŠENJA

U ovom dijelu cilj je objasniti šta sadrži opisni fajl i kako je iskodirana simulacija DFA.

Fajl iz kojeg se učitava konfiguracija DFA sadrži po redovima sljedeće:

- broj ukupnih stanja automata
- početno stanje
- završno stanje
- sva stanja koja od kojih se sastoji DFA
- ponašanje svih stanja u tri kolone odnosu na ulazni string (prva kolona: trenutno stanje; druga kolona: ukoliko se na stanje dovodi nula koje je sljedeće; treća kolona: ukoliko se na stanje dovodi jedinica koje je sljedeće).

Nakon čitanja konfiguracijskog fajla, smještanja informacija u varijable te unosa željenog stringa izvršava se funkcija koja nam daje informaciju da li je string prihvaćen ili ne. Princip rada te funkcije je opisan u sljedećem paragrafu.

Funkciji proslijeđujemo sve informacije o DFA koje smo pročitali iz fajla. Pretvaramo uneseni string i vektor cijelih brojeva. Nakon toga prolazimo kroz svaki znak unesenog stringa (vektora) for-petljom. Ispituje se da li je znak na koji smo naišli 0 ili 1. Ukoliko je znak 1 uz pomoć proslijeđenih varijabli se ispituje koje je trenutno stanje i koje je sljedeće stanje za to stanje. Trenutnom stanju se dodjeljuje to stanje i ispisuje se string sa novim stanjem. Ukoliko naiđemo na znak 0 analogno se ispituje samo se iz varijabli čita koje je sljedeće stanje trenutnog stanja, ali kad naiđe znak 0. Na kraju

funkcije ispitujemo da li je trenutno stanje jednako završnom i ukoliko jeste string je prihvaćen, a ukoliko nije string nije prihvaćen.

FUNKCIJA (Početno stanje, Završno stanje, Ukupan broj stanja, Unešeni string, Prelaz stanja):

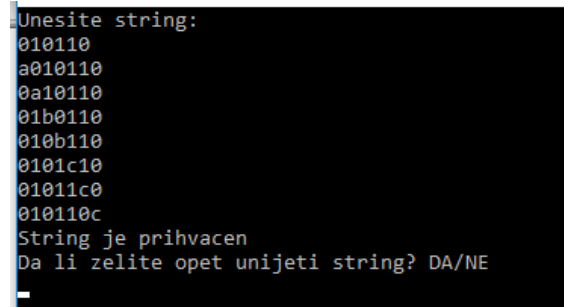
1. Inicijaliziraj vektor;
2. Pretvoriti unešeni string u vektor;
3. Stavi početno stanje kao trenutno;
4. **for** $i=1$ do dužina stringa radi
5. **if** i -ti član vektora jednak 1
 - a. if trenutno stanje jednako traženom stanju
 - b. $\text{trenutno} \leftarrow \text{ново stanje}$ u odnosu na traženo
 - c. Ispiši vektor
6. **if** i -ti član vektora jednak 0
 - a. **if** trenutno stanje jednako traženom stanju
 - b. $\text{trenutno} \leftarrow \text{ново stanje}$ u odnosu na traženo
 - c. Ispiši vektor
7. **If** trenutno stanje jednako traženom vrati **TRUE**
8. Vрати **FALSE**

3. PRIMJER

Ovdje ćemo navesti primjer izvršavanja date simulacije. DFA koji je konfigurisan fajlom vrši sljedeće. Ukoliko se u stringu nalaze dvije ili više jedinica na bilo kojim mjestima string se prihvata u suprotnom string se odbija.

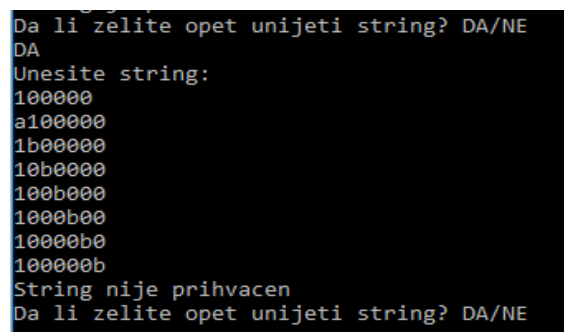
Imamo tri stanja. Prvo stanje je početno, a treće je završno. Ukoliko na bilo koje od stanja dovedemo znak 1 ono mijenja stanje na sljedeće, a u suprotnom ostaje isto.

Imamo mogućnost da nakon unešenog stringa ponovo unosimo string ili da prekinemo program.



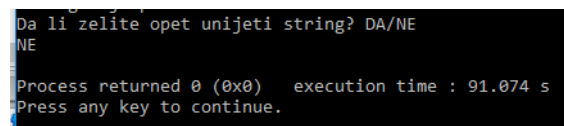
```
Unesite string:
010110
a010110
0a10110
01b0110
010b110
0101c10
01011c0
010110c
String je prihvacen
Da li zelite opet unijeti string? DA/NE
DA
```

Slika 1. Primjer prihvaćenog stringa



```
Da li zelite opet unijeti string? DA/NE
DA
Unesite string:
100000
a100000
1b00000
10b0000
100b000
1000b00
10000b0
100000b
String nije prihvacen
Da li zelite opet unijeti string? DA/NE
DA
```

Slika 2. Primjer unosa novog stringa i odbijanje istog



```
Da li zelite opet unijeti string? DA/NE
NE
Process returned 0 (0x0) execution time : 91.074 s
Press any key to continue.
```

Slika 3. Prekid rada programa prilikom unosa NE

4. ZAKLJUČAK

Iznad je opisana simulacija i način kodiranja simulacije DFA. Promjenom konfiguarcijskog fajla možemo mijenjati i različite uslove prihvatanja unešenog stringa. Uspješno testiranje je izvršenog na velikom broju primjera, ali svi nisu prikazani.