ALGORITEM ZA PRETVORBO ZAPISA REGULARNEGA IZRAZA V DEFINICIJO DETERMINISTIČNEGA KONČNEGA AVTOMATA[1]

Boštjan Vilfan
Univerza v Ljubljani

14. marec 1997

Regularni izrazi Definicijo regularnih izrazov poznamo. Kot primer regularnega izraza bomo obravnavali:

$$(a|b)^*abb$$

Regularni izraz znamo spremeniti tudi v nedeterministični avtomat z ε-prehodi (gl. sliko 2). V sliki ne prikazujemo nekaterih ε-prehodov, ki nastanejo kot posledica uporabe znanega algoritma, vendar to ne spremeni zadeve bistveno.

Avtomat na sliki 2 lahko spremenimo najprej v nedeterministični avtomat brez ε-prehodov z uporabo funkcije ε-CL in nato v deterministični avtomat z znano konstrukcijo, ki uporablja podmnožice množice stanj.

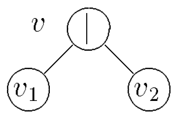
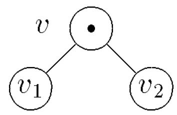
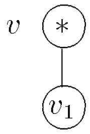
V tem sestavku bomo prikazali, kako zapis regularnega izraza neposredno spremenimo v opis determinističnega končnega avtomata.

Regularni izrazi z zaključnim simbolom Najprej ugotovimo, da se pri prehodu od nedeterminističnega avtomata z ε-prehodi do nedeterminističnega avtomata brez ε-prehodov ohranijo le stanja, pri katerih izvirajo neprazni prehodi, kajti ostala stanja “pogoltne” funkcija ε-CL. Taka stanja (pri katerih izvirajo neprazni prehodi) poimenujemo *bistvena*. Da bi zagotovili, da je končno stanje bistveno, dodamo na koncu regularnega izraza poseben simbol #, ki označuje konec besede. Tako dobimo *regularni izrazi z zaključnim simbolom*:

$$(a|b)^*abb\#$$

Nedeterministični avtomat z ε-prehodi, ki ustreza temu izrazu je prikazan na sliki 3. Bistvena stanja so oštevilčena, nebistvena pa označena s črkami.

[1] Povzeto iz Aho, Sethi, Ullman: Compilers — Principles, Techniques, and Toolš Addison–Wesley, 1986; ISBN 0-201-10088-6; str. 135-141.

Vozlišče v	nullable(v)	firstpos(v)
v je končno vozlišče ε	**true**	$\emptyset$
v je končno vozlišče z oznako i	**false**	$\{i\}$
	$\mathrm{nullable}(v_1) \vee \mathrm{nullable}(v_2)$	$\mathrm{firstpos}(v_1) \cup \mathrm{firstpos}(v_2)$
	$\mathrm{nullable}(v_1) \wedge \mathrm{nullable}(v_2)$	**if** nullable(v_1) **then** $\mathrm{firstpos}(v_1) \cup \mathrm{firstpos}(v_2)$ **else** firstpos(v_1)
	**true**	firstpos(v_1)

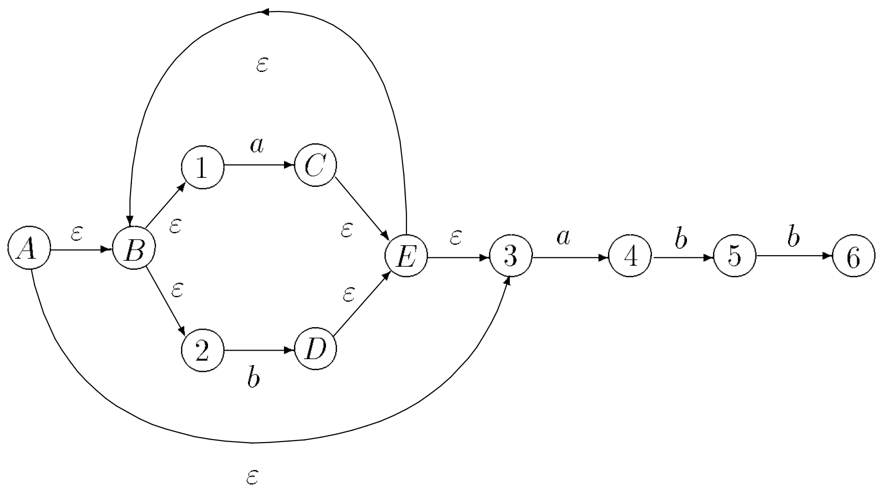
Slika 1 Definicija funkcij *nullable* in *firstpos*

Strukturno drevo Kot osnovno predstavitev pri uporabi algoritma uporabljamo *strukturno drevo*, ki ga dobimo neposredno iz regularnega izraza z zaključnim simbolom (slika 4). Neprazna končna vozlišča so oštevilčena tako, da indeks ustreza vrstnemu redu simbola v regularnem izrazu. Indeks nekega simbola v strukturnem drevesu ustreza stanju, iz katerega izhaja prehod, ki je zaznamovan s simbolom.

Funkcije *nullable*, *firstpos*, *lastpos*, *followpos* Nedeterministični avtomat brez ε-prehodov in kasneje deterministični avtomat konstruiramo uporabljajoč funckije *nullable*, *firstpos*, *lastpos*, *followpos*. nullable(v) je resnično, ko jezik podizraza, ki ustreza vozlišču v, vsebuje prazno besedo; firstpos(v) vsebuje oznake končnih vozlišč, ki predstavljajo prve simbole besed, ki jih predstavlja regularen izraz v; lastpos(v) na podoben način vsebuje oznake končnih vozlišč, ki predstavljajo zadnje simbole besed, ki jih predstavlja v, medtem ko followpos(i) predstavlja oznake končnih vozlišč, ki predstavljajo simbole v besedah, ki se lahko pojavljajo za simbolom, ki ga predstavlja oznaka i.

Definiciji funkcij *nullable* in *firstpos* sta prikazani na sliki 1, medtem ko funkcijo *lastpos* izračunamo podobno kot *firstpos* le, da zamenjamo v_1 in v_2 ter *firstpos* in *lastpos*. Definicija funkcije *followpos* pa je naslednja:

1. Naj bo v vozlišče, ki predstavlja stik z naslednikoma v_1 in v_2. Naj bo i oznaka, ki je vsebovana v lastpos(v_1). V tem primeru so vsi elementi firstpos(v_2) vsebovani v followpos(i).



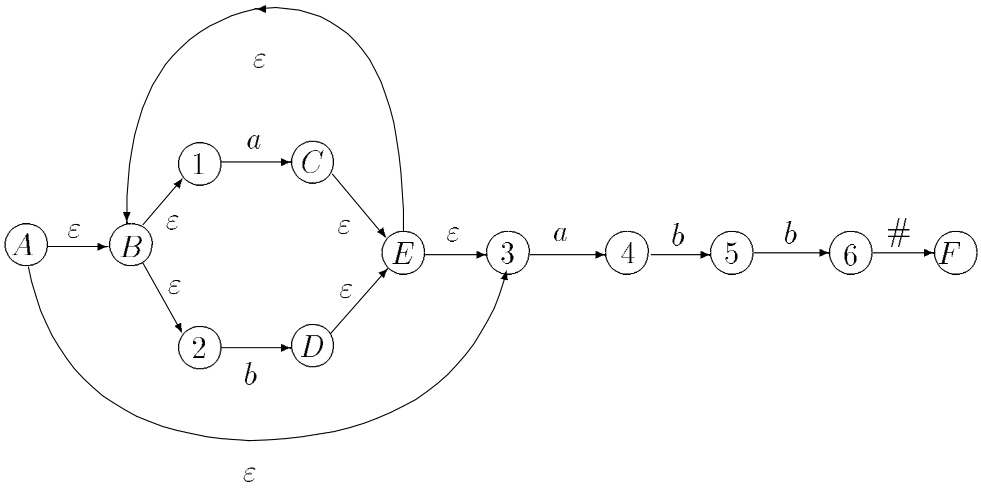
Slika 2 Nedeterministični avtomat z ε-prehodi za jezik izraza $(a|b)^*abb$

2. Naj v predstavlja iteracijo in naj bo i vsebovano v lastpos(v). V tem primeru so vsi elementi firstpos(v) vsebovani v followpos(i).

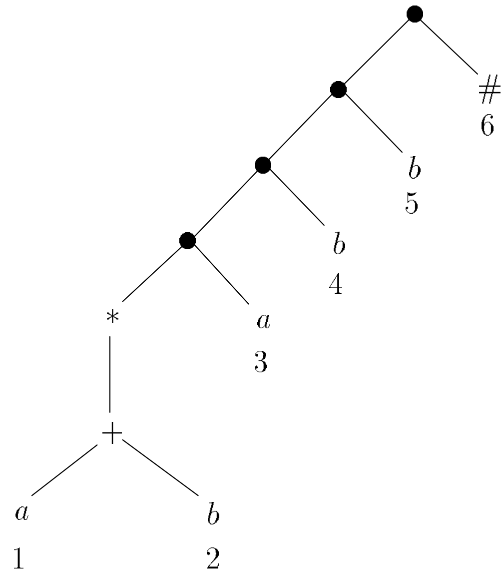
Primer izračuna funkcij *firstpos* in *lastpos* je prikazan na sliki 5.

Vrednost funkcije *followpos* računamo tako, da pri izračunu obiščemo najprej končna vozlišča in se pomikamo proti korenu. Na primer, na sliki 5 pri vozlišču, ki predstavlja iteracijo najprej ugotovimo, da sta elementa 1 in 2 v množicah followpos(1) in followpos(1). Nato pri naslednjem vozlišču na poti proti korenu, ki predstavlja stik, ugotovimo, da je element 3 v množicah followpos(1) in followpos(2). Na podoben način lahko določimo vse množice followpos(i), ki so grafično prikazane na sliki 6.

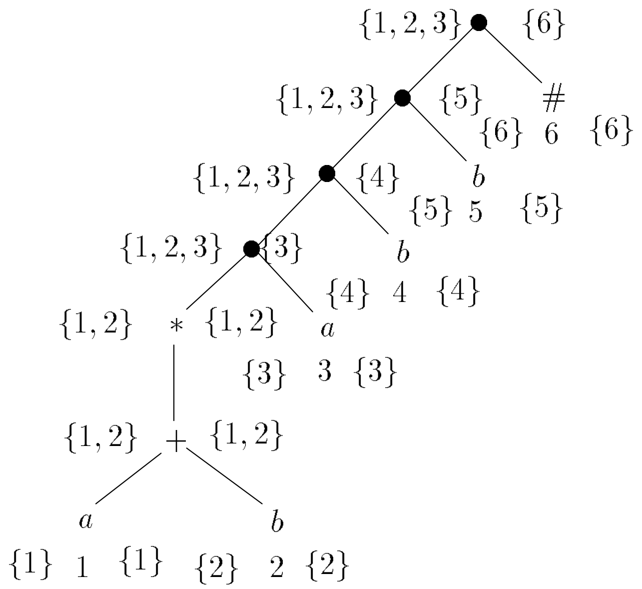
DKA sedaj lahko določimo na podlagi *followpos*. Nedeterministični avtomat brez ε-prehodov je možno dobiti iz grafične predstavitve funkcije *followpos*, tako da (a) povezavo $\langle i, j\rangle$ zaznamujemo z simbolom, ki ga predstavlja vozlišče i; (b) kot začetna stanja izberemo vsa stanja, ki pripadajo množici firstpos(e), kjer je e celoten izraz in (c) kot edino končno stanje izberemo stanje, ki ustreza zaključnemu simbolu #. Iz nedeterminističnega avtomata pa ni težko dobiti deterministični, tako da upoštevamo le tiste podmnožice, ki so dosegljive iz množice začetnih stanj. Primer, ki ga obravnavamo, je prikazan na sliki 7.



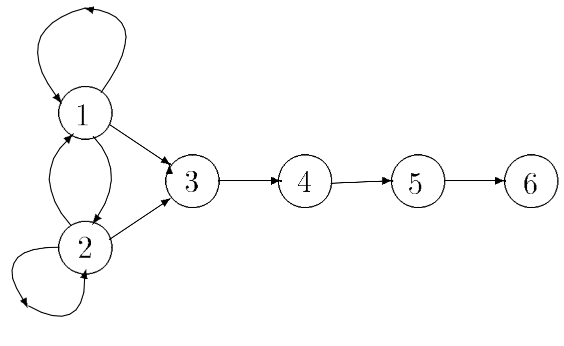
Slika 3 Nedeterministični avtomat z ε-prehodi za jezik izraza $(a|b)^*abb\#$



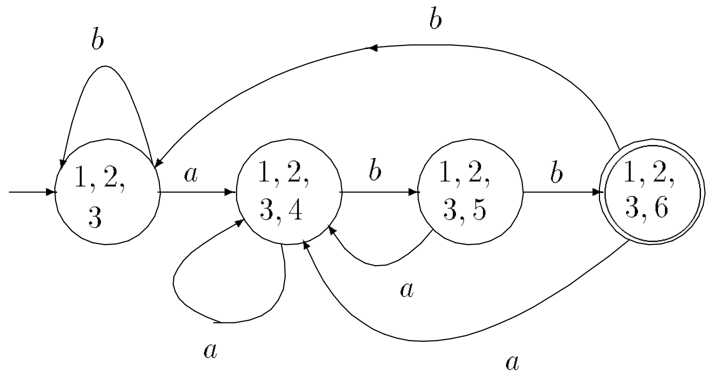
Slika 4 Strukturno drevo izraza $(a|b)^*abb\#$



Slika 5 Vrednosti funkcije *firstpos* (levo) in *lastpos* (desno) pri posameznih vozliščih



Slika 6 Vrednosti funkcije *followpos* pri posameznih vozliščih



Slika 7 Deterministični končni avtomat za jezik izraza $(a|b)^*abb$