FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO
Mestrado Integrado em Engenharia Informática e Computação

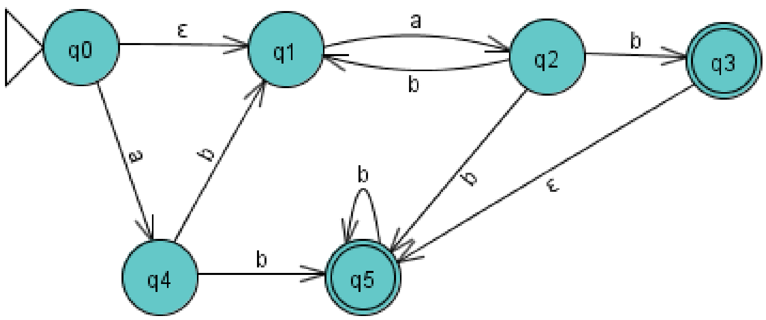
Teoria da Computação

Exame, 11 de Janeiro de 2010

DURAÇÃO MÁXIMA: 2 horas e 30 minutos

Problema 1: Autómatos Finitos e Expressões Regulares (5 valores)

Considere o seguinte ε-NFA sobre o alfabeto Σ={a,b}:



1.a) Indique o Fecho-ε de cada um dos estados no ε-NFA.

Fecho-ε(q0)={q0,q1}, Fecho-ε(q1)={q1}, Fecho-ε(q2)={q2}, Fecho-ε(q3)={q3,q5}, Fecho-ε(q4)={q4}, Fecho-ε(q5)={q5}

1.b) Converta o autómato para um DFA que aceite a mesma linguagem. Desenhe o diagrama de transição de estados do DFA completo resultante.

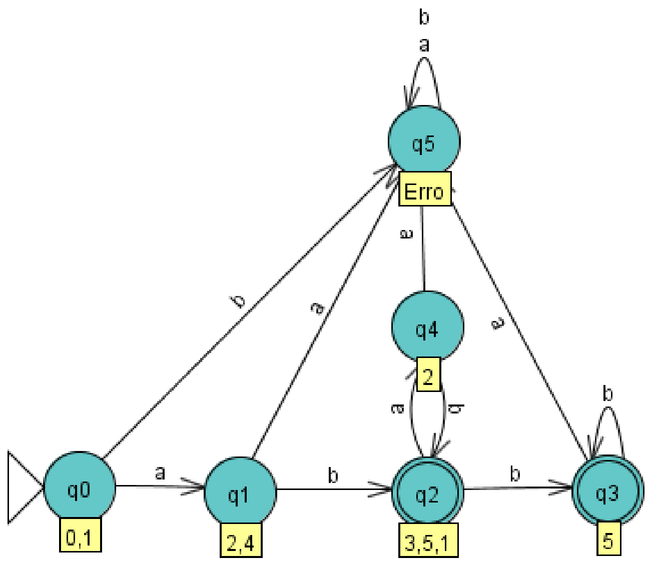
Tabela de transições do ε-NFA:

	ε	a	b
->q0	{q1}	{q4}	∅
q1	∅	{q2}	∅
q2	∅	∅	{q1,q3,q5}
*q3	{q5}	∅	∅
q4	∅	∅	{q1,q5}
*q5	∅	∅	{q5}

Tabela de transições do DFA após conversão:

	a	b
->{q0,q1}	{q2,q4}	∅
{q2,q4}	∅	{q1,q3,q5}
*{q1,q3,q5}	{q2}	{q5}
{q2}	∅	{q1,q3,q5}
*{q5}	∅	{q5}
∅	∅	∅

Diagrama de transição de estados do DFA resultante:

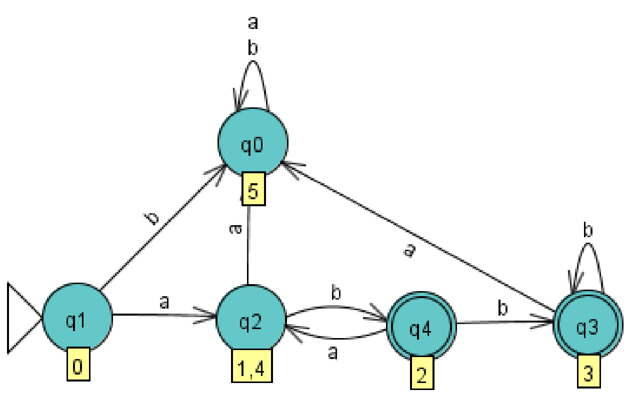


1.c) Usando o método de minimização de estados apresente o DFA resultante minimizado.

Tabela de estados distinguíveis para o DFA anterior[1]:

	q0	q1	q2	q3	q4
q1	X				
q2	X	X			
q3	X	X	X		
q4	X		X	X	
q5	X	X	X	X	X

Estados equivalente q1=q4 e o DFA resultante minimizado é:



[1] Notem que não é necessário incluir o estado de erro na tabela de estados distinguíveis pois este estado é distinguível de todos os outros estados do DFA.

1.d) Determine utilizando o método de eliminação de estados uma expressão regular que represente a linguagem que o autómato aceita. Apresente os passos que efectuar indicando a ordem de eliminação de estados que considerar. [Nota: nesta resolução não são apresentados todos os passos efectuados]

Directamente do DFA minimizado, $RE = RE_{q3} + RE_{q4}$

RE_{q3}: ab(ab)*bb*

RE_{q4}: ab(ab)*

RE = ab(ab)*bb* + ab(ab)* = ab(ab)*(ε+ bb*) = ab(ab)*b*

1.e) Prove usando o método de indução que o ε-NFA da figura aceita cadeias: **abb***.

Hipótese: S(n): abb^n

Base: ab que é aceite pelo autómato: δ^(q0,ab)={q5,q1,q3} e q3 e q5 são estados de aceitação

Passo indutivo: S(n+1): abb^{n+1}

$abb^{n+1} = abb^n b$

considerando a hipótese, S(n), verdadeira, ao concatenarmos mais um 'b' a strings da forma abb^n continuamos a obter strings aceites pelo autómato.

Como se pode comprovar pelas seguintes transições que culminam com o DFA no estado de aceitação q5:

$\delta\hat{}(q0,abb^n) = \{q5\}$ e

$\delta\hat{}(q0,abb^n b) = \delta(\delta\hat{}(q0,abb^n), b) = \delta(q5, b) = \{q5\}$

Problema 2: Linguagens (3 valores)

Considere a linguagem $L = \{a^m b^n \mid m,n \geq 0 \text{ e } m \neq n\}$.

2.a) L é uma linguagem regular ou não regular?

L é não regular.

2.b) Caso L seja uma linguagem não regular, recorra ao Lema da Bombagem para provar que L é uma linguagem não regular. Caso L seja uma linguagem regular, mostre que L satisfaz o Lema da Bombagem.

Solution 1:

We will use the pumping lemma to show B is non-regular. Assume B is regular.

Let $w = 0^n 1^{n+n!}$, $|w| = 2n + n! > n$. Since $n + n! \neq n$, w is in B. Then consider the decomposition of w as described in the pumping lemma: y is a nonempty susbstring of the first n characters of w. Since w starts with n zeroes, we are guaranteed that $y = 0^k$ for some $1 \leq k \leq n$. Then $xy^{\frac{n!}{k}+1}z = 0^{n+n!}1^{n+n!}$ must be in A. However, this string contains the same number of 0's and 1's, so it's not in the language.

Contradiction. B is not regular.

Solution 2:

Assume B is regular.
Since regular languages are closed under complement, $\overline{B}$ is regular.
Consider the regular language $L = 0^*1^*$ (it has a regular expression).
Consider the language $L_{nn} = \overline{B} \cap L = \{0^n 1^n \;:\; n \geq 0\}$. Since regular languages are closed under intersection, it should also be regular.
However, in class it was shown that L_{nn} is not regular, due to the pumping lemma. Contradiction. B is not a regular language.

Problema 3: Gramáticas e Autómatos de Pilha (5 valores)

Considere a linguagem L definida no Problema 2.

3.a) Essa linguagem pode ser especificada com uma gramática livre de contexto (CFG)? Justifique a resposta dada e caso tenha respondido afirmativamente apresente uma possível CFG para L.

Sim, a linguagem L pode ser especificada com uma gramática livre de contexto (CFG). Sendo os PDAs autómatos que definem CFLs poderemos justificar esta resposta analisando se um PDA poderia implementar L. Como com um PDA conseguimos ter uma forma de memorizar os a's consecutivos (colocando-os na pilha), podemos desenhar um PDA que aceita strings que após a sequência de a's têm um número de b's consecutivos em número menor ou maior do que o número de a's que ocorreram[2].

Possível CFG para L:

S → XY | YZ

X → aX | a

Z → bZ | b

Y → aYb | ε

Outra possível CFG para L:

S → X | Y

X → aXb | aX | a

Y → aYb | bY | b

[2] Uma justificação mais completa poderia incluir um PDA para a linguagem L.

Seja G = (V, Σ, R, S) a seguinte CFG. V={S, I, E}; Σ={i, f, e, l, s}; e R o conjunto de regras:

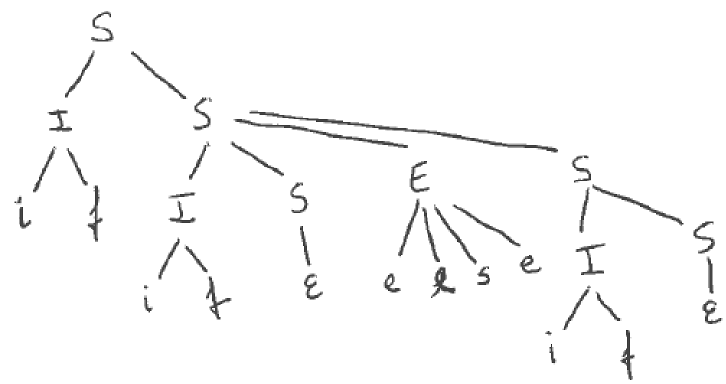
S → ε | SS | IS | ISES

I → if

E → else

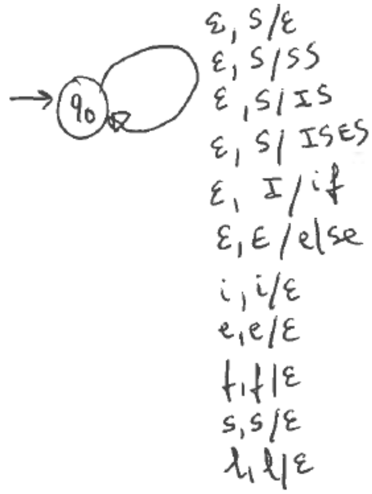
3.b) Desenhe uma árvore de análise para a cadeia **ififelseif**.

Possível árvore de análise para a cadeia **ififelseif**:



3.c) Converta a gramática para um PDA que aceita por pilha vazia e desenhe o PDA resultante. Mostre uma sequência de descrições instantâneas que conduzem à aceitação quando o PDA obtido processa a string **ififelseif**.

PDA P = ({q0}, {i, f, e, l, s}, {i, f, e, l, s, S, I, E}, δ, q0, S)

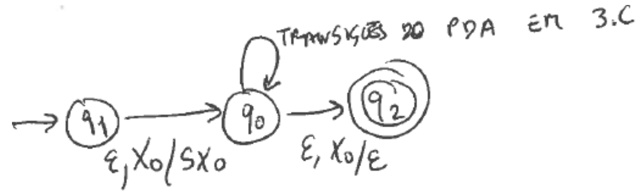


Uma das sequência de descrições instantâneas que conduzem à aceitação quando o PDA obtido processa a string **ififelseif**:

(q0, **ififelseif**,S) ├ (q0, **ififelseif**,IS) ├ (q0, **ififelseif**,ifS) ├ (q0, **fifelseif**,fS) ├ (q0, **ifelseif**,S) ├ (q0, **ifelseif**,ISES) ├ (q0, **ifelseif**,ifSES) ├ (q0, **felseif**,fSES) ├ (q0, **elseif**,SES) ├ (q0, **elseif**,ES) ├ (q0, **elseif**,elseS) ├ (q0, **lseif**,lseS) ├ (q0, **seif**,seS) ├ (q0, **eif**,eS) ├ (q0, **if**,S) ├ (q0, **if**,IS) ├ (q0, **if**,ifS) ├ (q0, **f**,fS) ├ (q0, **ε**,S) ├ (q0, **ε**,**ε**)

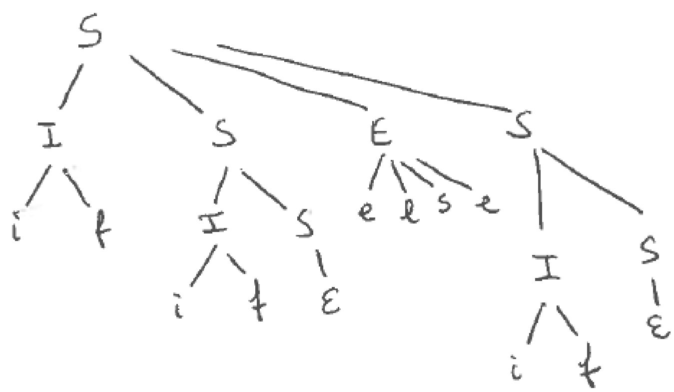
3.d) Converta o PDA anterior para um PDA que aceita por estado de aceitação.

PDA $P = (\{q_0, q_1, q_2\}, \{i, f, e, l, s\}, \{i, f, e, l, s, S, I, E, X_0\}, \delta, q_0, X_0, \{q_2\})$



3.e) A gramática é ambígua? Justifique a resposta dada. Caso a gramática seja ambígua apresente as modificações que terá de realizar para retirar a ambiguidade.

A gramática é ambígua pois existe pelo menos uma string aceite pela gramática para a qual podemos formar pelo menos duas árvores sintácticas diferentes. Por exemplo, para a string **ififelseif**, para além da árvore representada em 3a) existe pelo menos mais uma árvore de análise:



Gramática modificada[3] não-ambígua:

S → IS | ITES | ε

T → ITET | ε

I → if

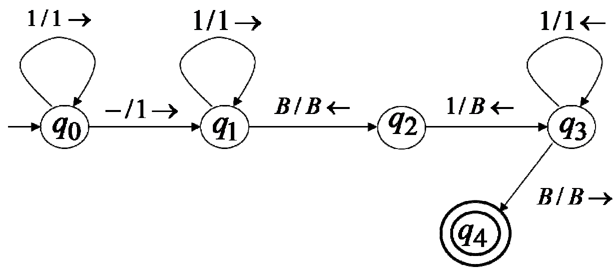
E → else

Problema 4: Máquina de Turing (4 valores)

Pretende-se realizar com uma máquina de Turing funções aritméticas sobre números inteiros positivos representados unitariamente (com 1's). As funções com mais do que um operando consideram que os operandos estão separados pelo símbolo '-'.

4.a) Apresente o traço de computação da Máquina de Turing apresentada a seguir quando a entrada na fita é 1-11.

[3] A gramática modificada teria de aceitar a mesma linguagem da gramática original.



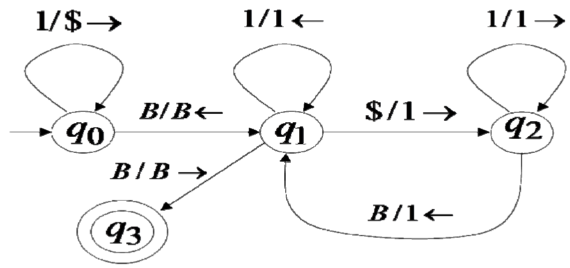
$q_0 1\text{-}11 \vdash 1q_0\text{-}11 \vdash 11q_1 11 \vdash 111q_1 1 \vdash 1111q_1 B \vdash 111q_2 1 \vdash 11q_3 1B \vdash 1q_3 11 \vdash q_3 111 \vdash q_3 B111 \vdash Bq_4 111$

4.b) Indique a função realizada pela Máquina de Turing da alínea anterior.

Adição de dois operandos, i.e., f(a,b)=a+b, com representação unitária e separados por '-'.

4.c) Desenhe o diagrama de transições de estado de uma Máquina de Turing que calcule f(a) = 2×a. Exemplos: 11 → 1111; 1 → 11

Possível TM:



4.d) Apresente o traço de computação da sua Máquina de Turing quando a entrada na fita é **111**.

$q_0 111 \vdash \$q_0 11 \vdash \$\$q_0 1 \vdash \$\$\$q_0 B \vdash \$\$q_1\$ \vdash \$\$1q_2 B \vdash \$\$q_1 11 \vdash \$q_1\$11 \vdash \$1q_2 11 \vdash \$11q_2 1 \vdash \$111q_2 B \vdash \$11q_1 11 \vdash \$1q_1 111 \vdash \$q_1 1111 \vdash q_1\$1111 \vdash 1q_2 1111 \vdash 11q_2 111 \vdash 111q_2 11 \vdash 1111q_2 1 \vdash 11111q_2 B \vdash 1111q_1 11 \vdash 111q_1 111 \vdash 11q_1 1111 \vdash 1q_1 11111 \vdash q_1 111111 \vdash q_1 B111111 \vdash q_3 111111$

Problema 5: Afirmações sobre Linguagens (3 valores)

Para cada uma das afirmações seguintes, diga se é verdadeira ou falsa e dê uma justificação sucinta.

5.a) Existe uma linguagem não regular L1 para a qual não se consegue arranjar uma linguagem L2 cuja união L1 ∪ L2 resulte numa linguagem regular.

Falso, a linguagem constituída por todas as strings formadas no alfabeto ∑ (i.e., ∑*) é regular e a união com qualquer linguagem não-regular sobre o mesmo ∑ dá ∑* (que é regular).

5.b) Dada uma string w de uma linguagem regular, pode-se sempre encontrar uma cadeia y não vazia, próxima do início de w, que pode ser bombeada, repetida um número arbitrário de vezes ou apagada, produzindo cadeias que pertencem à linguagem.

Falso.

(a) não é possível se a linguagem for finita ou,

(b) se w não contiver pelo menos uma sub-string cuja repetição dela própria 1 ou mais vezes produza sempre strings da linguagem. E.g., seja L(ab*c), se w for "ac" o enunciado não se pode verificar.

5.c) A linguagem $L = \{a^n a^n \mid n \geq 0\}$ é uma linguagem regular.

Verdadeiro, pois trata-se da linguagem constituída por cadeias de a's com comprimento par que pode ser representada por uma expressão regular ((aa)*) e implementada por um Autómato Finito Determinista.

5.d) Uma linguagem sem contexto L diz-se ambígua se não for possível definir uma gramática sem contexto não ambígua para L.

Verdadeiro. Existem linguagens para as quais não conseguimos especificar uma gramática sem contexto não ambígua. Exemplo: $L= \{a^n b^n c^m d^m \mid n\geq 1, m\geq 1\} \cup \{a^n b^m c^m d^n \mid n\geq 1, m\geq 1\}$

5.e) A intersecção de duas linguagens sem contexto (CFLs) produz sempre uma CFL.

Falso, a intersecção das linguagens sem contexto $L1=\{a^n b^n c^m \mid n,m \geq 0\}$ e $L2=\{a^n b^m c^m \mid n,m \geq 0\}$ produz a linguagem $L=\{a^n b^n c^n \mid n,m \geq 0\}$ que não é reconhecidamente uma linguagem sem contexto (pelo lema da bombagem para linguagens sem contexto)

(Fim.)