UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO - CTC
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

INE5415 - Teoria da Computação
Professora: Jerusa Marchi

Fernanda Larissa Müller[1] - 21202109
Augusto de Hollanda Vieira Guerner[1] - 22102192

[1] Programa de Graduação em Ciência da Computação, Departamento de Informática e Estatística, Universidade Federal de Santa Catarina.

E-mail: mllernanda@gmail.com e guhvguerner@gmail.com.

Trabalho I: Máquinas de Turing

Florianópolis, 5 de novembro de 2023.

Questões:

1. Implemente Máquinas de Turing Determinística com fita única para computar as seguintes linguagens:

(a) (1,0pt) $L = \{a^i b^j c^k \mid i,\ j,\ k \in N\ e\ i \times j = k\}$

(b) (1.0pt) $L = \{0^{2^n} \mid n \geq 0\}$

2. Implemente Máquinas de Turing Determinística Multifitas para computar as seguintes linguagens:

(a) (1,5pt) $L = \{xyx^R y^R \mid x,\ y \in \{0,\ 1\}^*\}$ (xR é o reverso da cadeia x, yR é o reverso da cadeia y)

(b) (1,5pt) $L = \{a^i b^j c^k \mid i,\ j,\ k \in N\ e\ i^j = k\}$

A entrada deve ser fornecida integralmente na primeira fita.

3. Implemente Máquinas de Turing Determinística Multifitas para computar as seguintes funções:

(a) (2,5pt) Série de Fibonacci. A máquina recebe como entrada uma sequência de símbolos que representa n (representação unária - como X ou a). Ao término, deve constar na fita uma sequência de símbolos que indica o valor do n-ésimo termo, ou seja Fibonacci(n). Faça com que a máquina, em seus primeiros passos de computação grave as sementes fibonacci(0) = 0 e Fibonacci(1) = 1 e proceda o cálculo iterativo da série até n.

(b) (2,5pt) Algoritmo de Euclides para o Máximo Divisor Comum. A máquina recebe como entrada uma sequência de símbolos representando n e m em representação unária. Ao término, a fita deve conter o MDC(n, m).

Respostas:

Questão 1)

a)

Para solução desta primeira questão, a ideia é muito simples. Para cada a de entrada, percorremos todos b's, sendo que, para cada b, marcamos um c diferente. Se no final deste processo, ao término dos a's, tivermos apenas c's marcados, então a cadeia é aceita; do contrário, a cadeia é rejeitada. O algoritmo 1 descreve em alto nível seguindo uma lógica de pseudocódigo a mesma lógica descrita.

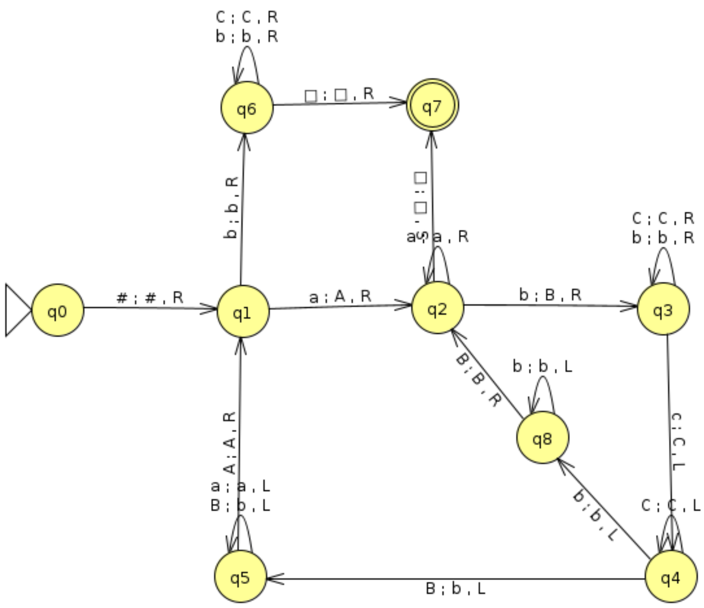
Algoritmo 1: Algoritmo para conferir o resultado de uma multiplicação entre dois números

1. *Para cada a faça*
    a. *Para cada b faça*
        i. *Se existir c, marca-o*
        ii. *Senão, rejeita a cadeia*
2. *Se ao final do processo tiver apenas c's marcados, então aceita*
3. *Do contrário, isto é, sobrar c's rejeita*

Fonte: Acervo dos autores.

Como é possível notar na **imagem 1**, vê-se claramente os dois loops criados no pseudocódigo, sendo o primeiro determinado pelos estados q1, q2, q3, q8, q4 e q5 e o segundo pelos estados q2, q3, q4 e q8. Este último é o laço mais interno, ou seja, o laço que percorre os b's e o outro o laço mais externo, isto é, o que percorre os a's. Essa estrutura condiz com a lógica citada antes: para cada a, percorre-se todos b's e, para cada b, marca-se um c diferente. Além disso, é possível ainda comentar que a transição de q2 para q7 verifica o caso de existir a, mas não existir b's e nem c's. Já a transição de q1 para q6 compreende para o caso de não existir a's e nem c's, mas existir b's; ou ainda para o caso de já se ter percorrido os a's e se ter que verificar a existência de somentes c's marcados.

Imagem 1: Máquina de Turing que verifica a corretude de uma multiplicação.



Fonte: Acervo dos autores.

b)

Nesta questão, a ideia do algoritmo foi retirada do livro do Sipser, referência da disciplina. Essencialmente, é ir dividindo por dois a quantidade de zeros até que não exista mais zeros. Se ao final de cada divisão sobrar um zero, a cadeia é rejeitada automaticamente, senão, prossegue para a próxima iteração. A seguir está o algoritmo 2 descrito mais precisamente.

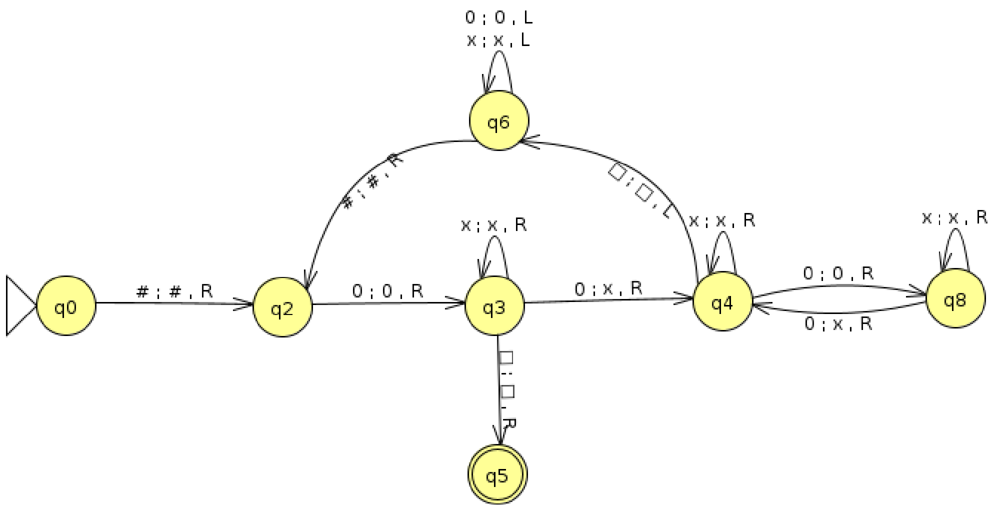
Algoritmo 2: Algoritmo para conferir se uma cadeia tem número de zeros igual a uma potência de dois.

1. *Se tem apenas um zero, o primeiro, então a cadeia é aceita*
2. *Caso contrário, segue para a próxima etapa*
3. *Marca os zeros alternadamente desde o início da cadeia de entrada até o fim dela (um não, um sim, um não, ... Note que o primeiro zero nunca vai ser marcado)*
    a. *Se ao final deste processo a máquina estiver procurando um zero para marcar e não encontrar, então a cadeia é rejeitada (observe que neste caso o número é ímpar)*
    b. *Senão, isto é, a máquina estava procurando um zero para pular e não encontra, a máquina posiciona o cabeçote para inicio novamente e prossegue com o próximo passo*
4. *Volta para o passo 1*

Fonte: Acervo dos autores.

Na **imagem 2**, nota-se a existência de um ciclo definido pelos estados q2, q3 q4 e q6. Este ciclo é responsável por fazer três coisas: a verificação da existência de apenas um zero, o primeiro, pelo estado q3; o processo de marcação alternada de zeros, feito pelos estados q4 e q8; e pelo processo de retrocesso ao início do cabeçote executado pelo estado q6.

Imagem 2: Máquina de Turing que verifica se a cadeia tem número de zeros igual a uma potência de dois.



Fonte: Acervo dos autores.

Questão 2)

a)

Esta questão gerou a máquina mais extensa do trabalho em questão de quantidade de estados. No que se refere as fitas, foram utilizadas 3: a primeira é a de entrada; a segunda fita tem 2 funções diferentes, na primeira fase serve como marcador da metade do tamanho de entrada e na segunda fase serve como marcador para sabermos onde iniciar a comparação na fita de entrada; e por fim, a terceira fita contém a metade reversa de xy.

A ideia dessa solução é: percorrer toda a fita de entrada marcando x na segunda fita a cada 2 números de entrada, assim, teremos na segunda fita a metade do tamanho de entrada e saberemos onde começa os reversos. Após esse pré-processamento, voltamos até a metade da entrada, apagando os x's da segunda fita e quando chegar a metade copiamos $x^R y^R$ para a terceira fita apagando da fita de entrada. Após isso, vamos comparando número por número da fita de entrada (da direita para esquerda) e da terceira fita (da esquerda para direita), caso os números sejam diferentes marcamos x na segunda fita e voltamos ao final da fita de entrada e ao início da terceira fita. Para as próximas iterações começaremos do final da fita de entrada pulando o número de x's marcados na segunda fita e na terceira fita começando do início normalmente. Se após isso chegar ao início de fita na fita de entrada, significa que encontramos o x e x^R, agora voltamos ao final da fita de entrada e comparamos com o que sobrou a direita da terceira fita para ver se y e y^R estão corretos.

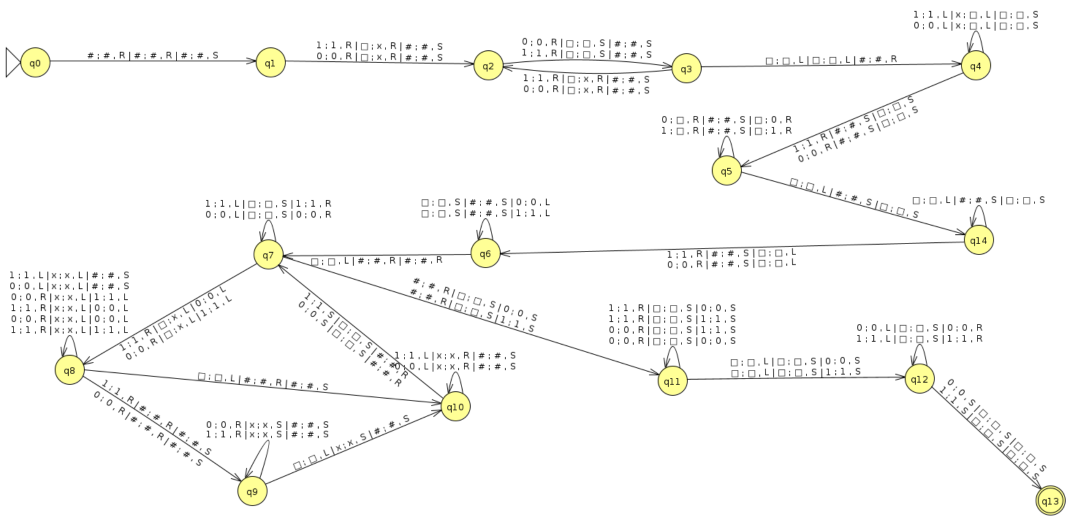
Algoritmo 3: Algoritmo que verifica a corretude da operação de potenciação.

1. *Percorre a fita de entrada e marca x na segunda fita a cada 2 números da entrada (ex.: para a entrada '0110' marcamos x p/ o primeiro 0, pulamos o 1, marcamos x p/ segundo 1 e pulamos o segundo 0).*
2. *Volta até a metade da fita de entrada apagando os x's da segunda fita.*
3. *Copia a metade à direita da fita de entrada para a terceira fita e vai apagando esses valores da fita de entrada.*
4. *Percorre a fita de entrada da direita para esquerda e para cada valor compara com a terceira fita que corre normalmente da esquerda para direita.*
    a. *Se os valores forem iguais:*
        i. *Passa para o próximo valor da terceira fita;*
        ii. *e volta um valor da fita de entrada.*
    b. *Se os valores forem diferentes:*
        i. *Marca x na segunda fita;*
        ii. *Volta para o início da segunda fita;*
        iii. *e volta ao final da fita de entrada e:*
            1. *para cada x na segunda fita:*
                a. *pula um valor da fita de entrada (da direita p/ esquerda).*
5. *Se encontrar início de fita na fita de entrada, significa que encontramos o x e x^R. Então, volta ao final da fita de entrada e compara valor a valor com o que sobrou na terceira fita.*
    a. *Se todos os valores forem iguais:*
        i. *Aceita.*
    b. *Se algum valor for diferente:*
        i. *Rejeita.*

Fonte: Acervo dos autores.

A **imagem 3** ilustra o resultado final da máquina. Os estados q1, q2 e q3 marcam a metade da fita e garantem que a entrada tem tamanho par. O estado q4 é responsável por voltar até a metade da fita de entrada e o estado q5 é responsável por escrever $x^R y^R$ na terceira fita. O estado q14 volta até o último elemento da fita de entrada e o estado q6 volta até o início da terceira fita. Os estados q7, q8, q9 e q10 são os responsáveis por encontrar x e x^R. Quando isso ocorre o estado q11 vai até o final da fita de entrada novamente para o estado q12 poder comparar cada valor com o que sobrou na terceira fita. Se tudo for igual até o final de fita da terceira fita achamos y e y^R e o estado q13 aceita a entrada.

Imagem 3: Máquina de Turing para xyx^Ry^R.



Fonte: Acervo dos autores.

b)

Primeiramente, no que se refere as fitas, foram utilizadas 5: a primeira é a de entrada e que contém o valor de k; a segunda contém o valor de i; a terceira o valor j; a 4 o resultado da potenciação; e a quinta um buffer.

Já no que tange ao processamento, ele ocorre da seguinte maneira: primeiro é feito o setup da máquina definindo cada valor na sua respectiva fita; em seguida é percorrido cada j e, para cada um deles, é percorrido todos i's sendo que, para cada i, é incrementado a fita 5 (resultado) pela fita 4 (inicialmente 1, mas ao longo do processo assume o valor anterior da fita 5, isso se depois que se percorreu todos os i's); depois de percorrer todos os j's é comparada os k's da entrada com o resultado presente na fita 5.

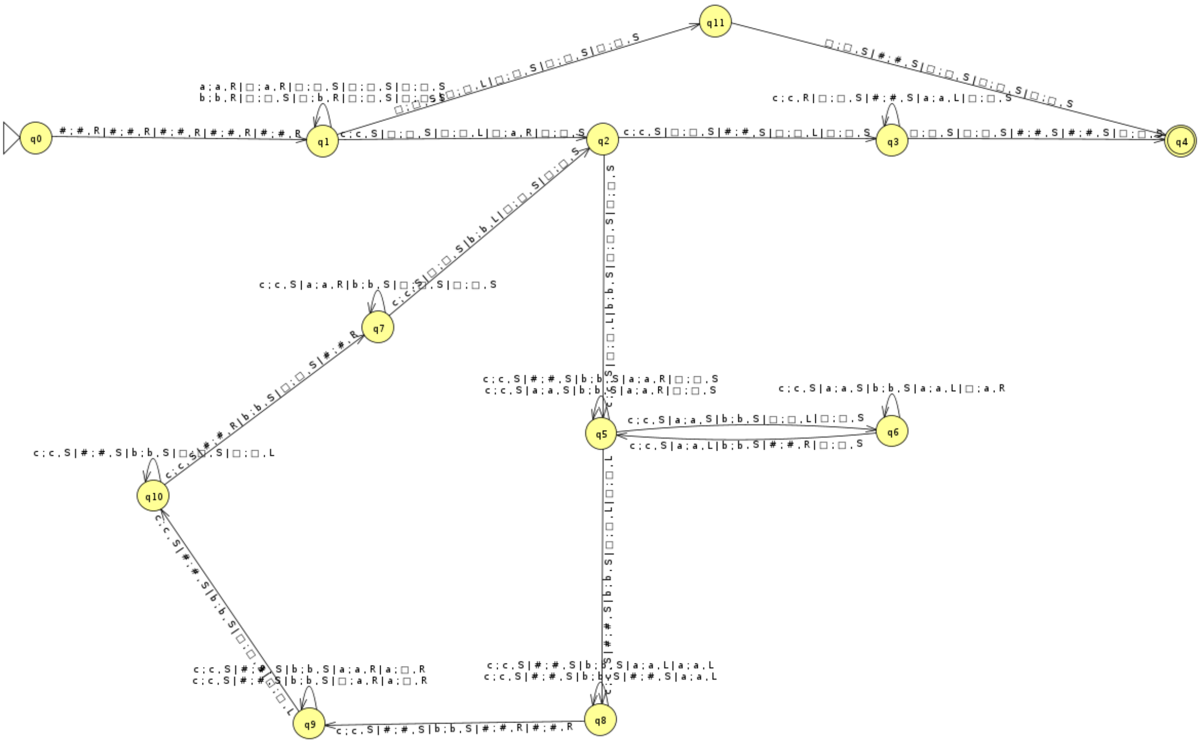
Algoritmo 4: Algoritmo que verifica a corretude da operação de potenciação.

6. *Define i, j e k nas fitas 2, 3 e 1 (fita de entrada) respectivamente*
7. *Atribui a fita 4 o valor 1 e a fita 5 0*
8. *Para cada j da fita 3 faça*
    a. *Para cada i da fita 2 faça*
        i. *Incrementa a fita 5 com o valor da fita 4*
    b. *Copia a fita 5 na fita 4*
9. *Compara a fita 4 com a fita 1 em que está o valor de k*
    a. *Se ambas as fitas tiverem o mesmo número de caracteres, a máquina aceita*
    b. *Senão, a máquina rejeita*

Fonte: Acervo dos autores.

A **imagem 4** ilustra o resultado final da máquina. Os estados q2, q5, q8, q9, q10 e q7 define o loop que percorre cada j. Os estados q5 e q6 determina o loop que percorre cada i e incrementa o valor da fita 4 na fita 5. O estado q3 é responsável pela comparação do resultado obtido na fita 5 e do número de k's da entrada. Por último, o q1 faz o setup da máquina.

Imagem 4: Máquina de Turing que verifica a corretude da operação de potenciação.



Fonte: Acervo dos autores.

Questão 3)

a)

A máquina desta questão no geral é muito simples, sobretudo com o uso de uma multifita. Para ela, usou-se 4: a de entrada que indica o termo a ser calculado da sequência; a segunda que é o termo n - 2 dá sequência; a terceira que é o n - 1; e a quarta que é o n. Com elas, é feito o seguinte processamento: para cada a da fita de entrada, soma-se a fita 2 com a 3 colocando o resultado na 4; depois é deslocado o conteúdo da fita 3 para a 2 e o da 4 para 3; por fim é zerada a fita 4 para a próxima iteração caso ainda haja a na entrada. Caso não, a máquina termina aceitando com o resultado armazenado na fita 3. A seguir está o **algoritmo 5** com uma descrição mais exata. Observe a condicional antes de iniciar as sucessivas somas da fita 3 com a 2 para cada a da fita 1.

Algoritmo 5: Algoritmo iterativo para calcular o n-ésimo termo da sequência de fibonacci.

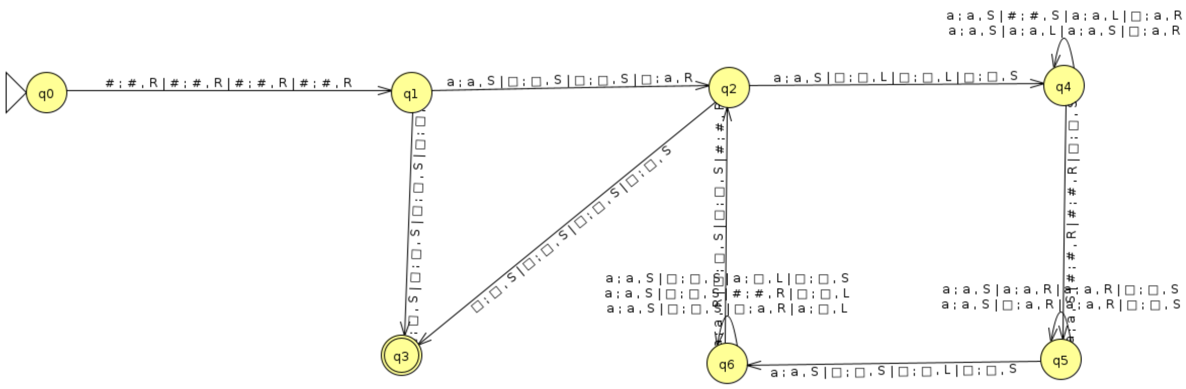
1. *Verifica se tem a na fita de entrada (fita 1)*
    a. *Se não tiver, ele aceita e o resultado fica na fita 3 (0)*
    b. *Se tiver, vai para o próximo passo.*
2. *Seta a fita 4 com 1 (insere um a)*

3. *Para cada a da fita de entrada faça*
    a. *Soma a fita 2 e 3 colocando o resultado na fita 4*
    b. *Copia o conteúdo da fita 3 na 2*
    c. *Copia o conteúdo da fita 4 na 3*
    d. *Define a fita 4 como 0*
4. *Quando não tiver mais a na entrada, a máquina termina a execução e o resultado fica na fita 3 (fibonacci(n))*

Fonte: Acervo dos autores.

Na **imagem 5**, tem-se a máquina correspondente ao algoritmo descrito acima. Observa-se claramente o loop formado pelos estado q2, q4, q5 e q6. Este loop é responsável pelas somas do termo n - 2 e do n - 1 da sequência de fibonacci: o estado q2 verifica se já terminou os a's de entrada; o q4 estado que faz propriamente a soma; o q5 copia a fita 3 para a 2; e q6, da fita 4 para a 3. Note que de q1 para q3 tem uma transição para o caso de não existir a's na fita.

Imagem 5: Máquina que calcula n-ésimo termo da sequência fibonacci.



Fonte: Acervo dos autores.

b)

Por último, tem-se o algoritmo e a máquina que calcula o MDC de dois números. O algoritmo é baseado em subtrações sucessivas de n por m até que n seja menor que m. Quando menor, troca-se n por m e m por n. Depois disso, se m for zero, então a máquina termina o processamento e o resultado fica na fita 1. Segue o **algoritmo 6** formalizando a ideia.

Algoritmo 6: Algoritmo iterativo para calcular o MDC de n e m.

1. *Define n e m respectivamente na fita 1 e 2*
2. *Verifica se a fita 2 é maior que 0*
    a. *Se não for, o programa termina e o resultado fica na fita 1*
    b. *Se for, contínua no passo seguinte*
3. *Verifica se fita 1 é menor que fita 2*
    a. *Se for, troca-se os valores da fita 2 e da fita 1, isto é, a fita 1 passa a ter o valor da fita 2 e a fita 2 passa a ter o valor da fita 1*
    b. *Se não for, segue para o próximo passo*
4. *Decrementa a fita 1 pela fita 2*
5. *volta para o passo 2*

Fonte: Acervo dos autores.

Na imagem 6, percebe-se um loop principal determinado pelos estados q2, q3, q5, q7, q9, q6, q8 e q10. Ele é responsável pela as subtrações sucessivas e eventuais trocas n e m. Note que q5 tem duas transições, isto é, duas possibilidades: uma em que m é maior que n, transição q5 para q6, e outra em que m é igual ou menor que n, transição q5 e q7. Além disso, a transição q3 para q4 caracteriza a verificação da linha 2 do algoritmo.

Imagem 6: Máquina que calcula o MDC de n e m.