Máquina de Turing

Prof. Yandre Maldonado e Gomes da Costa
yandre@din.uem.br

Teoria da Computação

- Ciência da Computação
  - **Ênfase teórica:** idéias fundamentais e modelos computacionais;
  - **Ênfase prática:** projeto de sistemas computacionais;

"As tecnologias computacionais são construídas a partir de fundamentos da computação. Aquelas são passageiras, enquanto estes estão por trás da tecnologia em qualquer tempo."

Teoria da Computação

- Histórico da Computação:
  - Computar: do latim “*computare*”, que significa calcular, avaliar, contar;

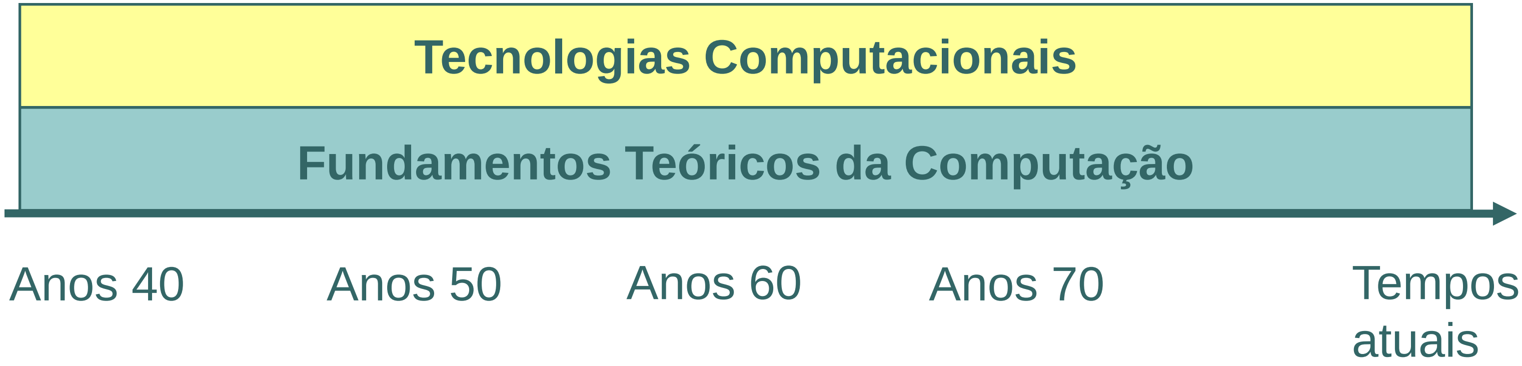
Ábaco – China, aprox. 3500 a.c.

Máquina de Babbage – R.U., 1823.

ENIAC – EUA, 1946.

Teoria da Computação

- Os fundamentos estão por trás da tecnologia em qualquer tempo.



Máquina de Turing

- Introduzida por Alan Turing em 1936;
- Ferramenta para estudar a capacidade dos processos algorítmicos;
- Modelo abstrato, concebido antes mesmo de uma implementação tecnológica;

Máquina de Turing

- Formaliza a idéia de uma pessoa que realiza cálculos;
  - Simulação de uma situação na qual uma pessoa, equipada com um instrumento de escrita e um apagador, realiza cálculos numa folha de papel;
  - "A Máquina de Turing ainda hoje é aceita como a formalização de um procedimento efetivo, ou seja, uma seqüência finita de instruções, as quais podem ser realizadas mecanicamente, em tempo finito." (Menezes, 1998).

Máquina de Turing

- MT pode ser vista como a mais simples "máquina de computação", servindo para determinar quais funções são computáveis e quais não são (Delamaro, 1998).

Máquina de Turing

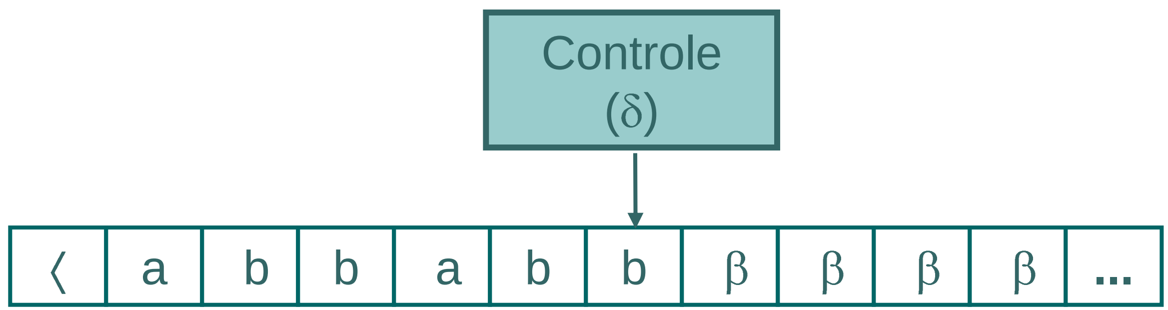
- Outros modelos foram propostos, mas todos mostraram ter, no máximo, poder computacional equivalente ao da MT;
- Estas são chamadas Máquinas Universais:
  - Máquinas capazes de expressar a solução para qualquer problema algorítmico.

Máquina de Turing

- A Máquina de Turing consiste de:
  - Uma Unidade de Controle
    - Que pode ler e escrever símbolos em uma fita por meio de um cabeçote de leitura e gravação e pode se deslocar para a esquerda ou direita;
  - A fita estende-se infinitamente em uma das extremidades e é dividida em células. Utilizada como dispositivo de entrada, saída e memória de trabalho;
    - Estas células podem armazenar qualquer elemento de um conjunto finito de símbolos, um alfabeto.

Máquina de Turing

- Programa ou Função de Transição: função que comanda as leituras e gravações, o sentido de movimento da cabeça e define o estado da máquina que será ativado.



* Como o símbolo “〈” estabelece o limite da extremidade esquerda da fita, não se pode apagá-lo e nem realizar movimentos para a esquerda deste símbolo.

Máquina de Turing

- Funcionamento da Máquina de Turing
  - A MT deve assumir sempre em um estado, pertencente à um conjunto finito de estados;
  - O processamento de uma MT começa sempre em um estado especial, chamado estado inicial;
  - Inicialmente a primeira célula da fita é preenchida com "〈 ", que indica o início da mesma;
  - A cabeça de leitura é posicionada inicialmente na segunda célula da fita, a célula seguinte a "〈";
  - As células em branco, que não fazem parte da palavra a ser processada, são preenchidas com o símbolo "β";

Máquina de Turing

- Funcionamento da Máquina de Turing
  - O processamento em uma MT consiste em uma seqüência de passos que consistem em:
    - Observar o estado e o símbolo corrente da fita (aquele em que o cabeçote está posicionado);
    - Escrever um símbolo nesta célula da fita;
    - Mover o cabeçote para a esquerda ou direita;
    - Mudar o estado corrente;
  - A ação exata a ser executada é determinada por um programa (função de transição) que comunica à unidade de controle o que deve ser feito com base na configuração (estado + símbolo corrente da fita) em que a MT se encontra.

Máquina de Turing

- Funcionamento da Máquina de Turing
  - O processamento cessa quando a máquina atinge uma configuração para a qual não existe função prevista, neste caso:
    - Se a máquina estiver com um estado final ativo, a palavra de entrada é aceita;
    - Se o estado ativo não for final, a palavra de entrada não é aceita;
    - Se a máquina não parar (*looping*), a entrada de entrada não é aceita.

Máquina de Turing

- Definição da MT, uma octupla:
  - Σ : alfabeto de símbolos de entrada;
  - Q : conjunto de estados possíveis, o qual é finito;
  - δ : programa ou função de transição

    $\forall\ \delta: Q \times (\Sigma \cup V \cup \{\beta, \langle\}) \rightarrow Q \times (\Sigma \cup V \cup \{\beta, \langle\}) \times \{E, D\}$

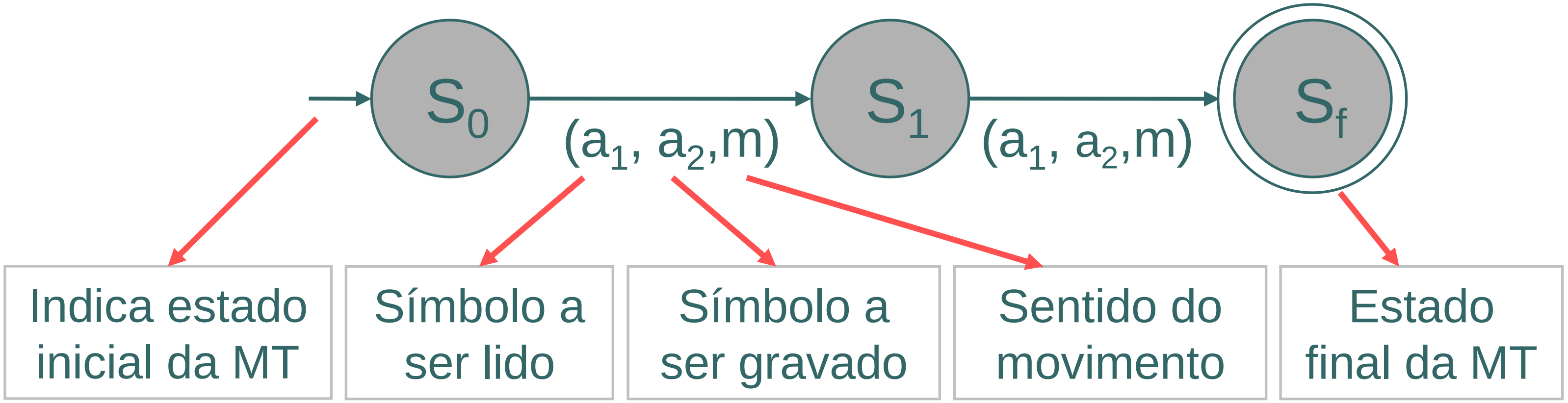
    a qual é uma função parcial;
  - q_0 : estado inicial da máquina, $q_0 \in Q$;
  - F : conjunto de estados finais, $F \subset Q$;
  - V : alfabeto auxiliar (pode ser vazio);
  - β : símbolo especial para células em branco;
  - $\langle$: símbolo especial marcador de início da fita.

Máquina de Turing

- A Máquina de Turing pode ser empregada como modelo de natureza reconhecedora ou transdutora:
  - Reconhecedora: processa a palavra de entrada aceitando-a ou rejeitando-a. Neste caso, o conjunto de palavras aceitas corresponde à linguagem descrita pela MT;
  - Transdutora: máquina para computar uma função. Aplica uma função sobre o conteúdo inicial da fita e o resultado produzido é lançado na própria fita.

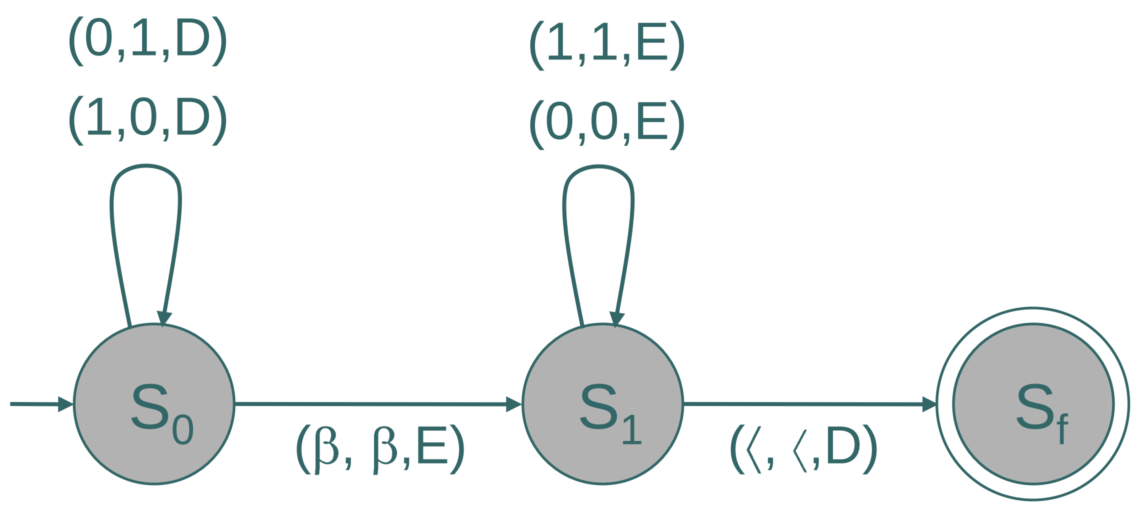
Máquina de Turing

- Representação de MT por diagrama:
  - Semelhante à representação de AFD;
  - Os rótulos das setas, que representam as funções de transição são de acordo com a seguinte legenda:



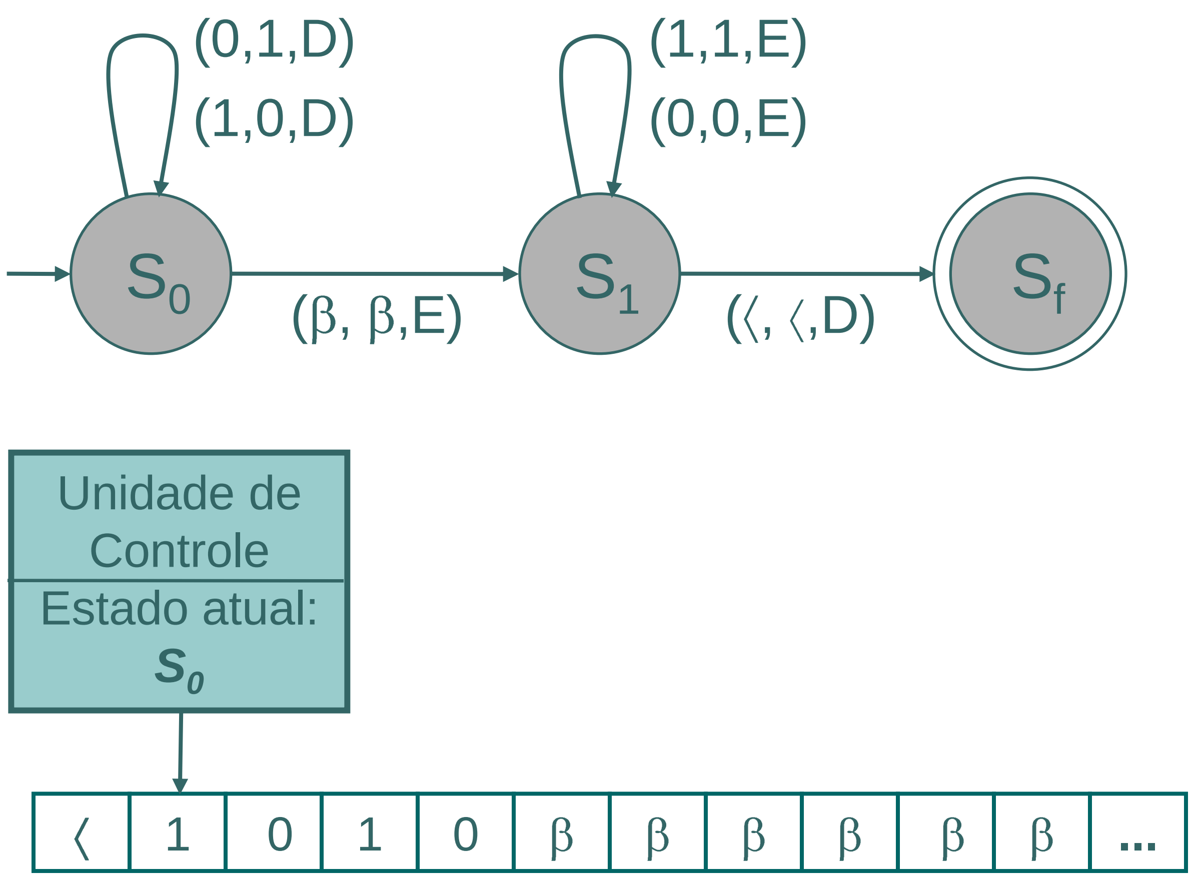
Máquina de Turing

- Exemplo 1: uma MT transdutora que devolve o complemento de uma entrada binária.



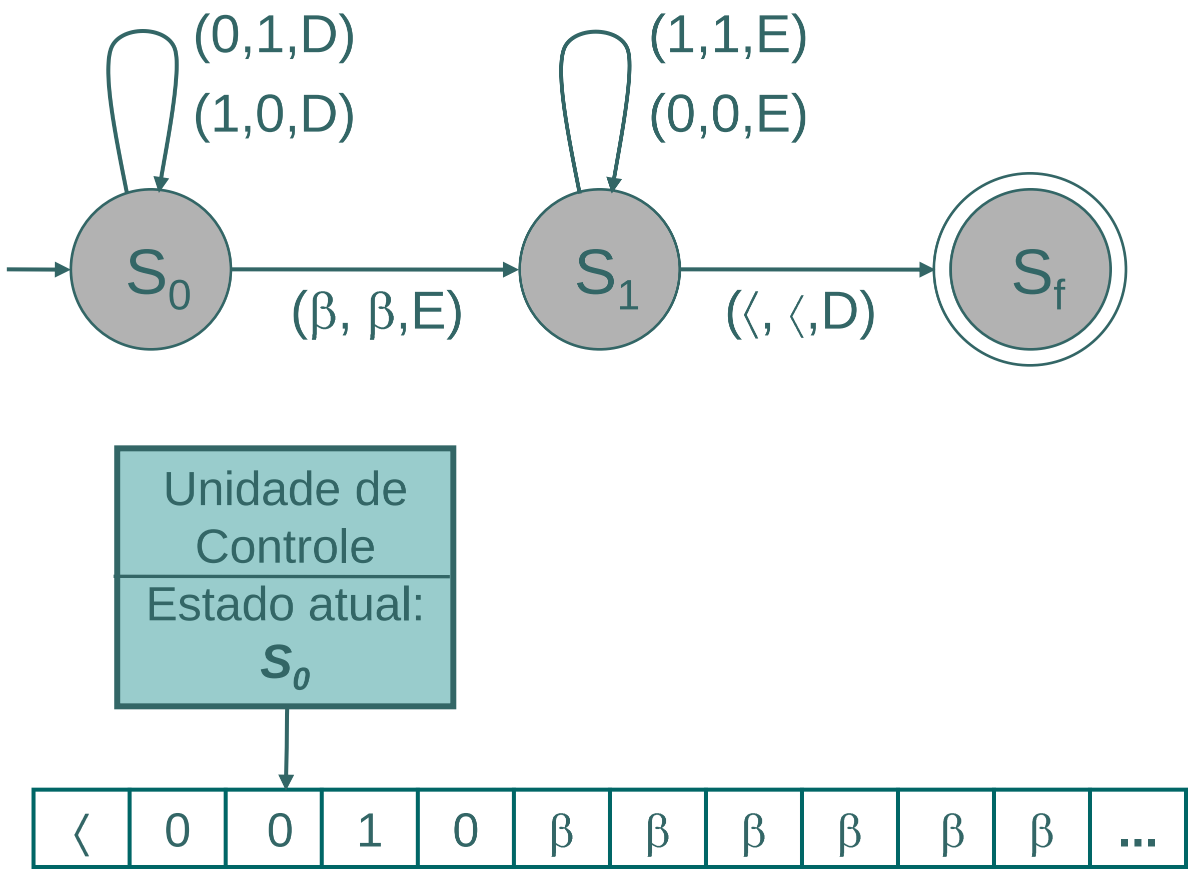
Máquina de Turing

- Processamento de “1010”:



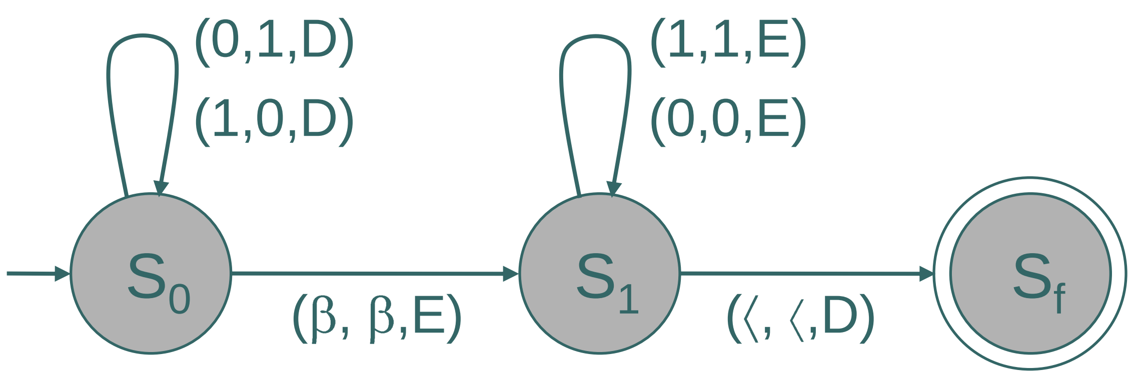
Máquina de Turing

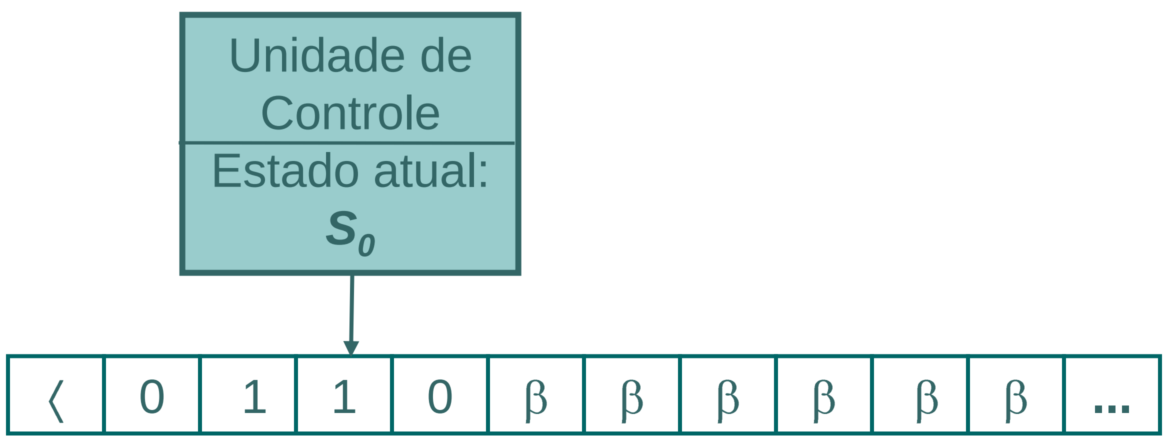
- Processamento de “1010”:



Máquina de Turing

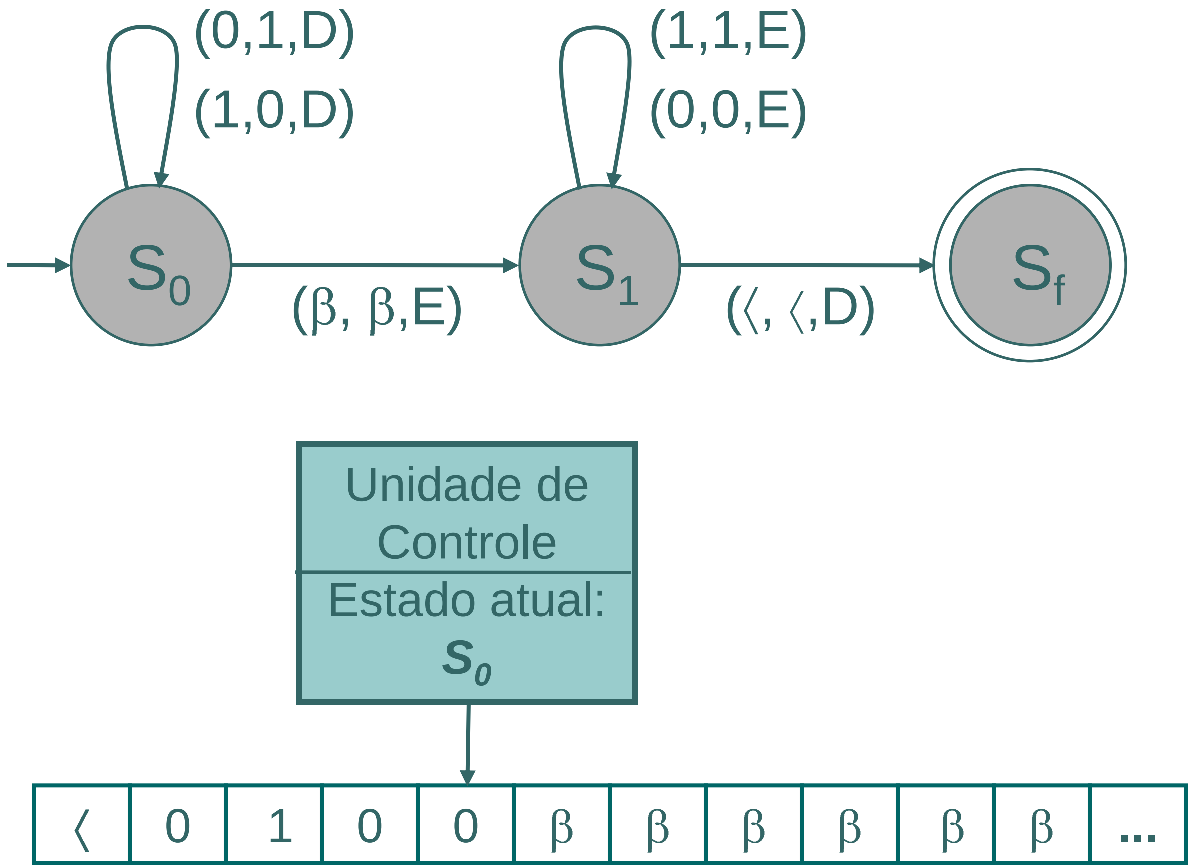
- Processamento de “1010”:





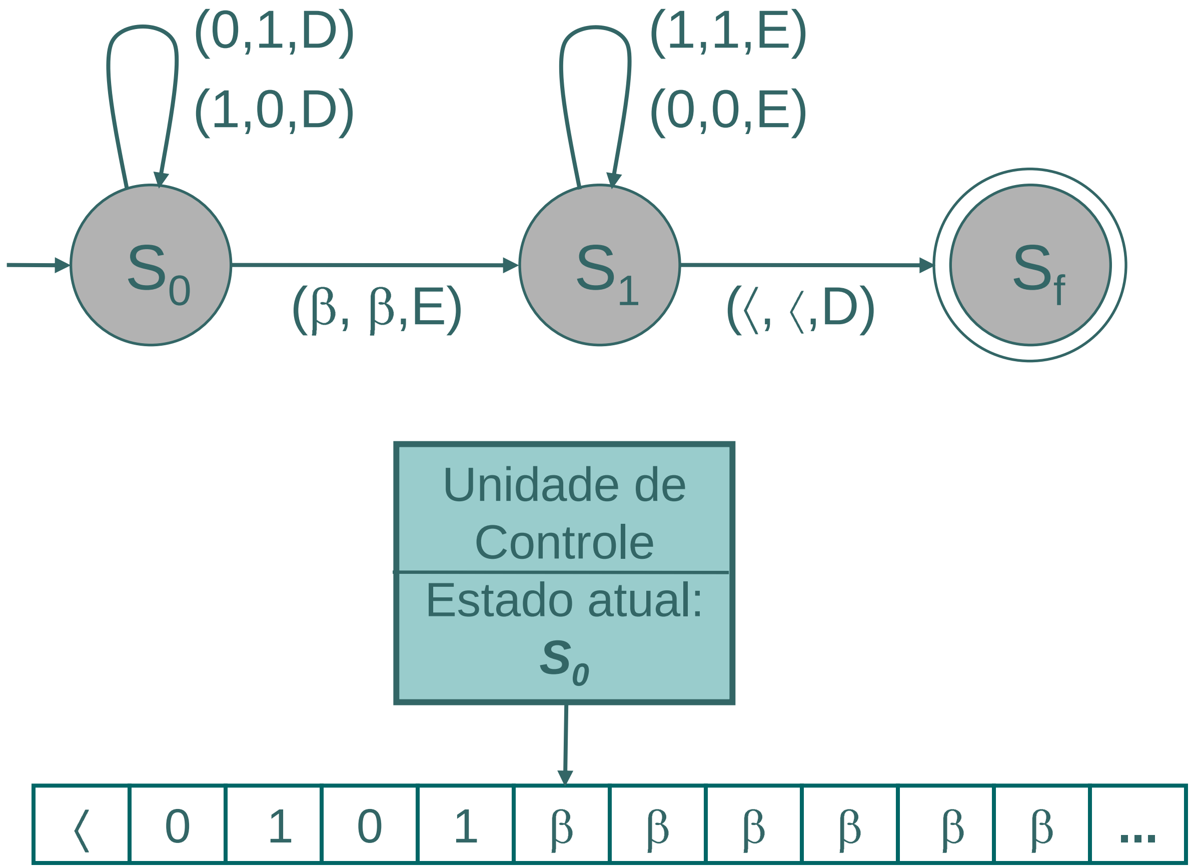
Máquina de Turing

- Processamento de “1010”:



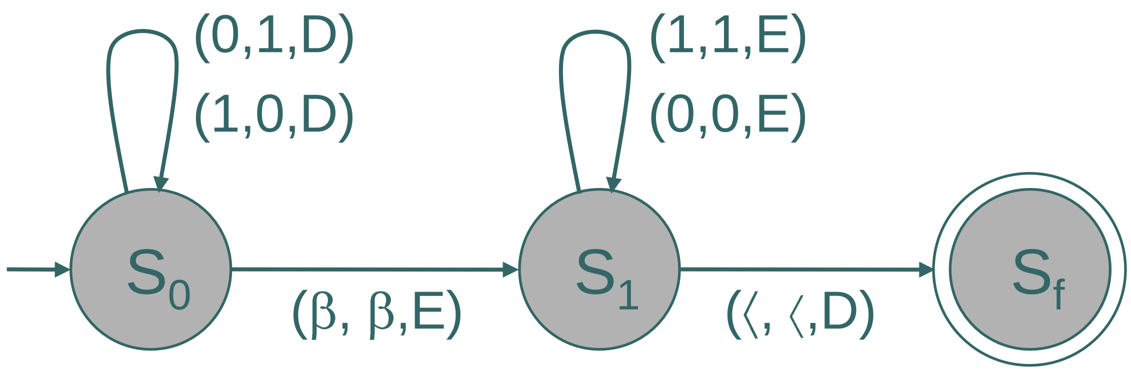
Máquina de Turing

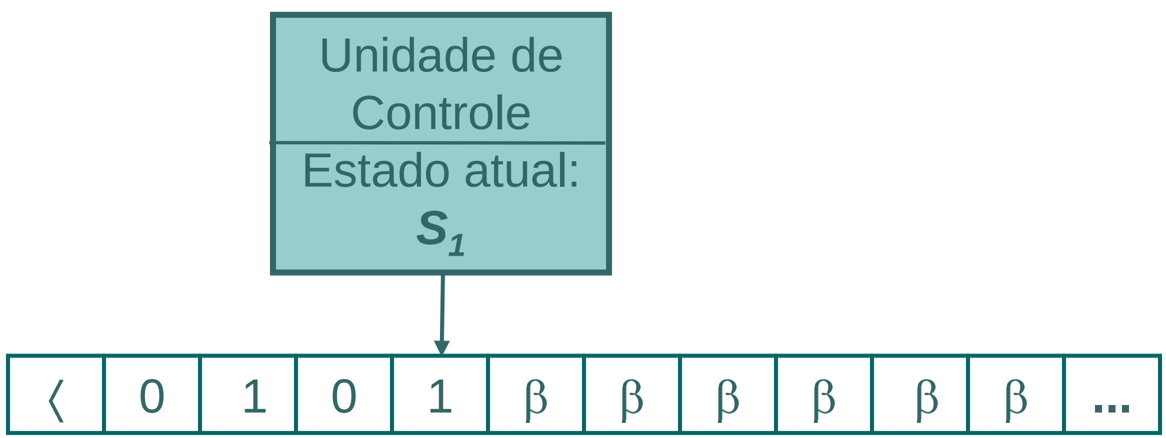
- Processamento de “1010”:



Máquina de Turing

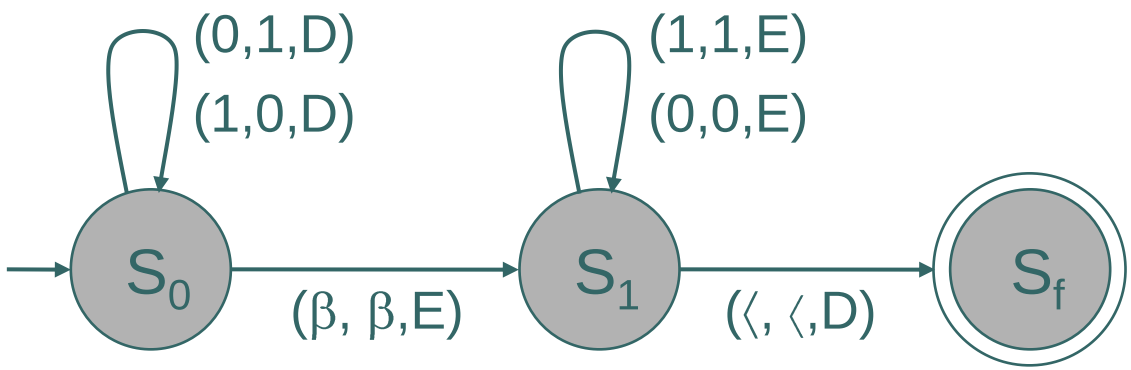
- Processamento de “1010”:

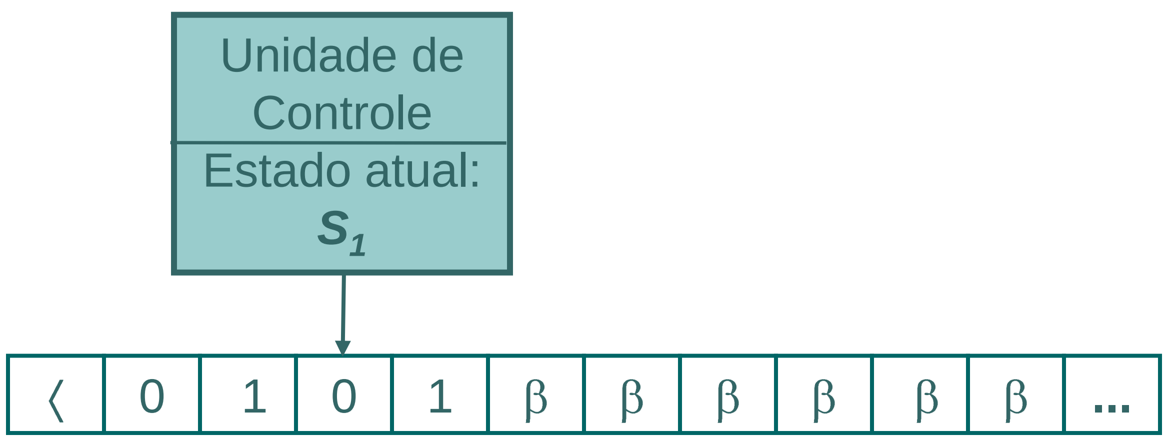




Máquina de Turing

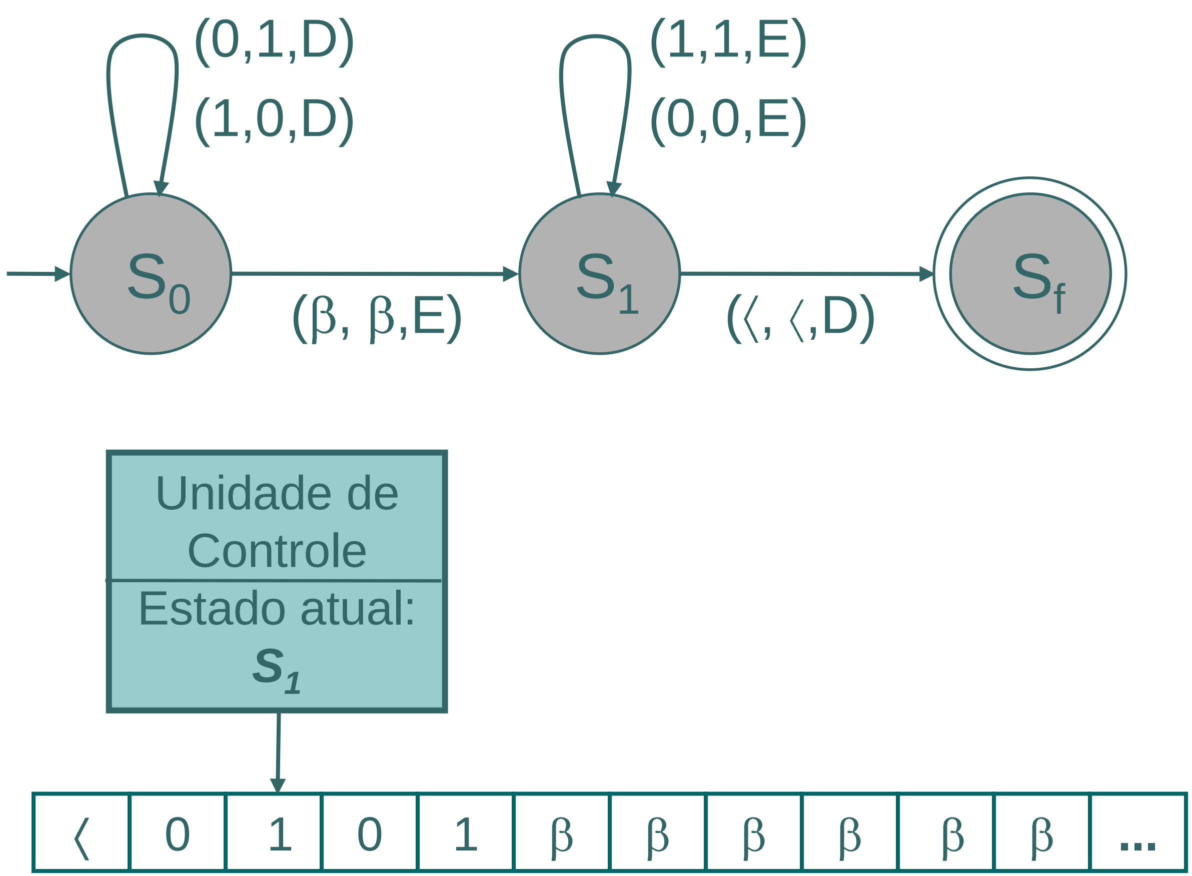
- Processamento de “1010”:





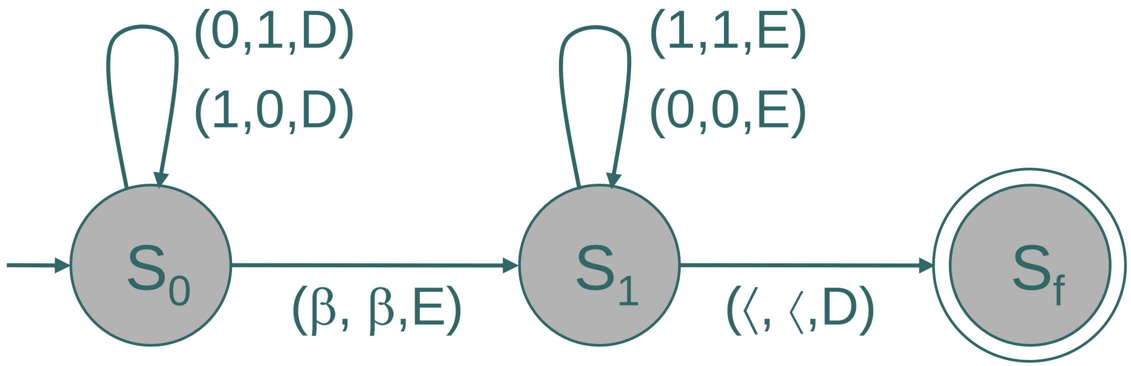
Máquina de Turing

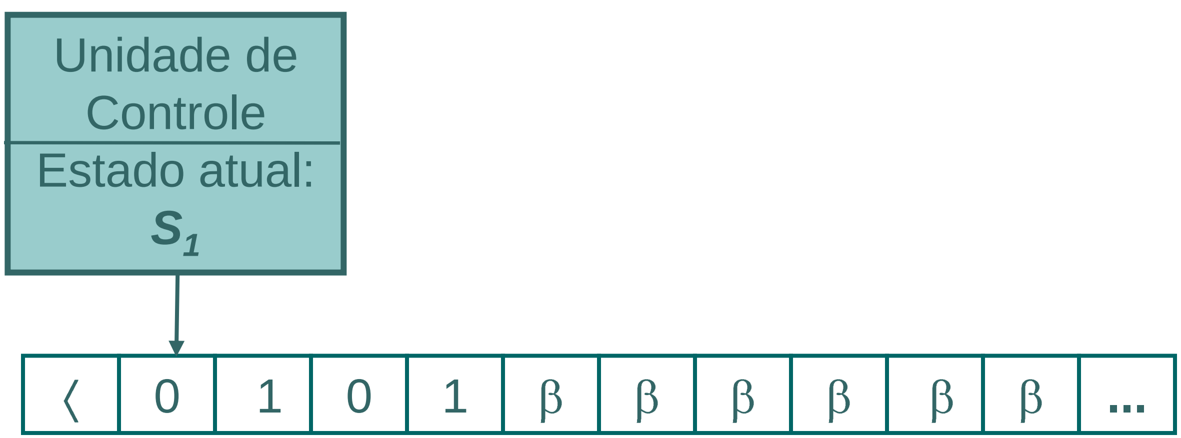
- Processamento de “1010”:



Máquina de Turing

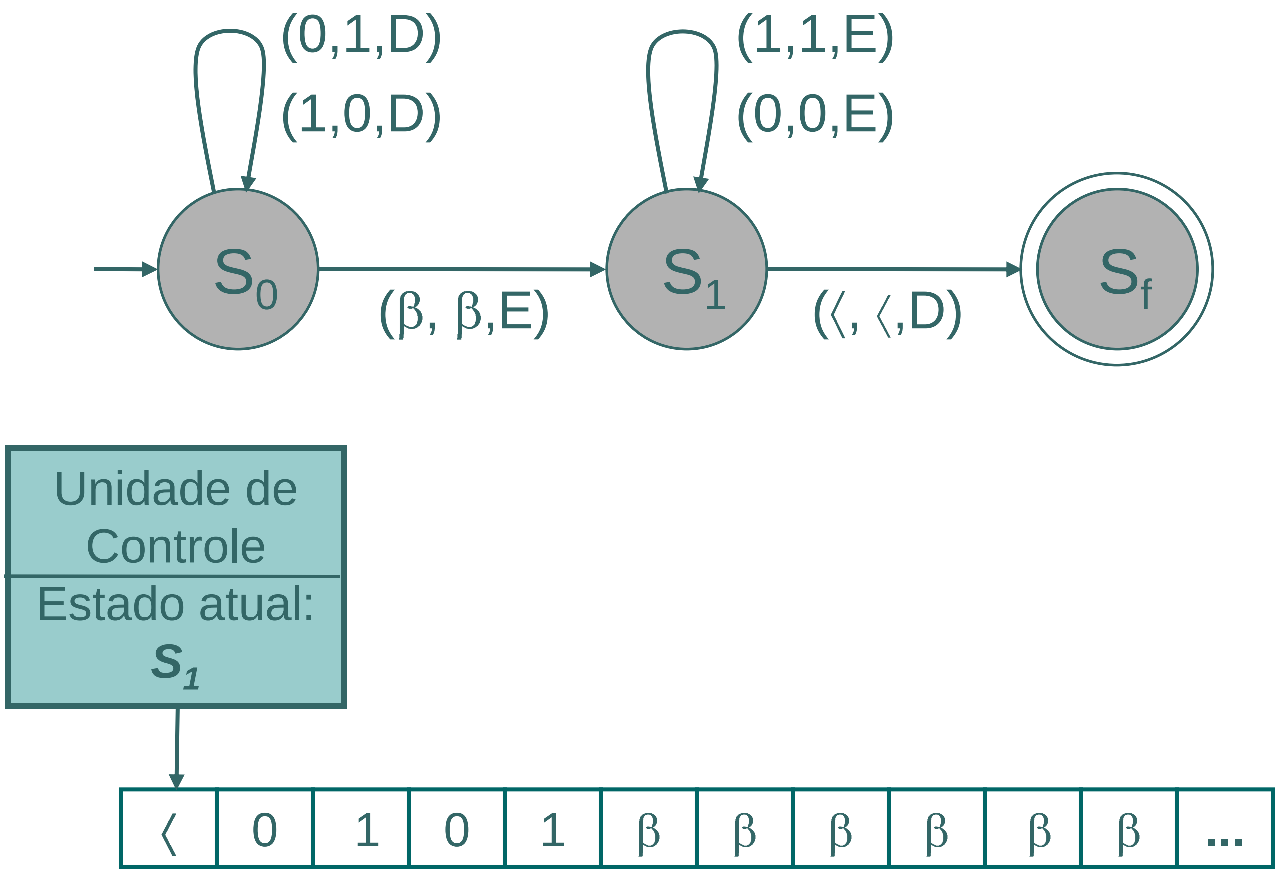
- Processamento de “1010”:





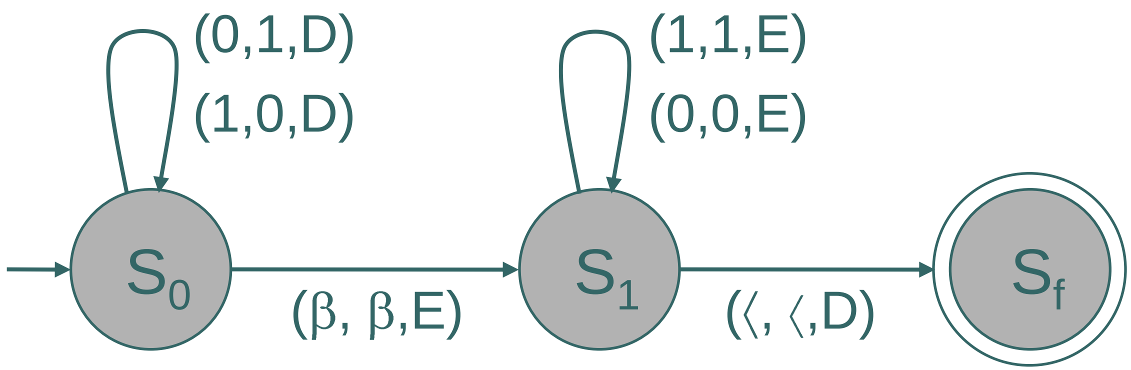
Máquina de Turing

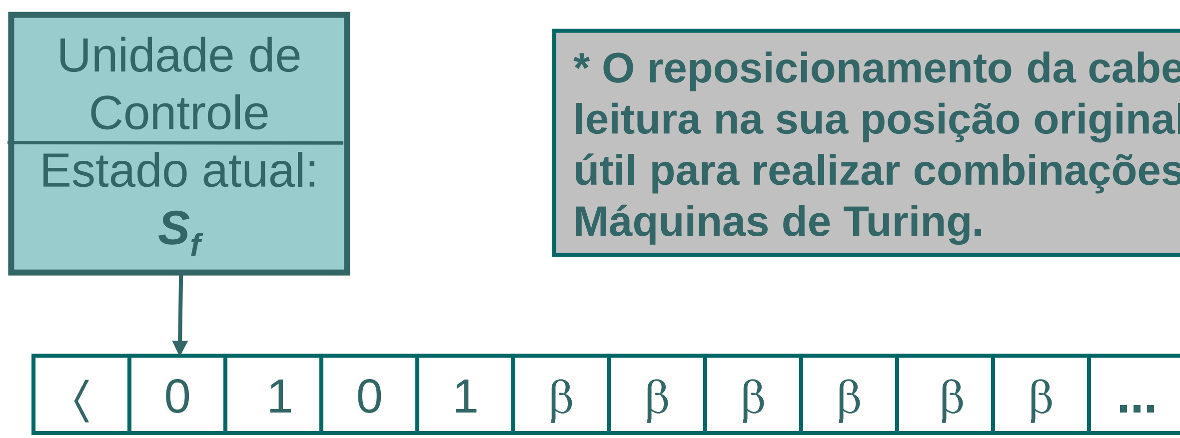
- Processamento de “1010”:



Máquina de Turing

- Processamento de “1010”:

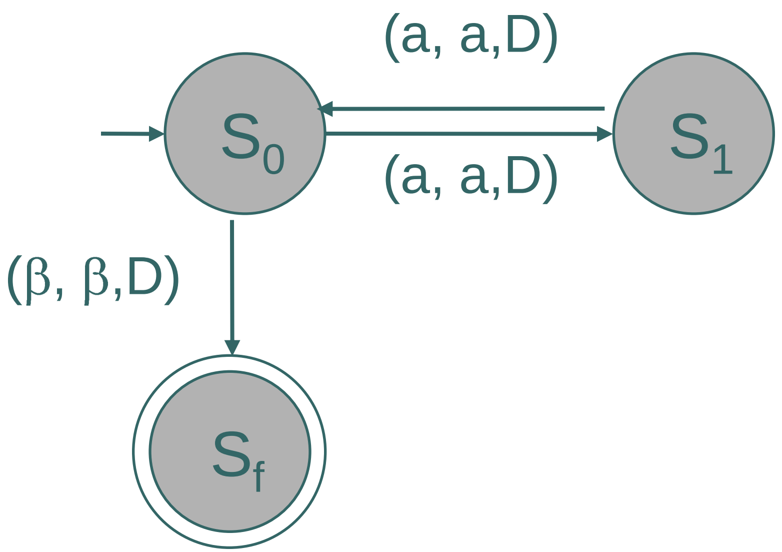




*** O reposicionamento da cabeça de leitura na sua posição original pode ser útil para realizar combinações de Máquinas de Turing.**

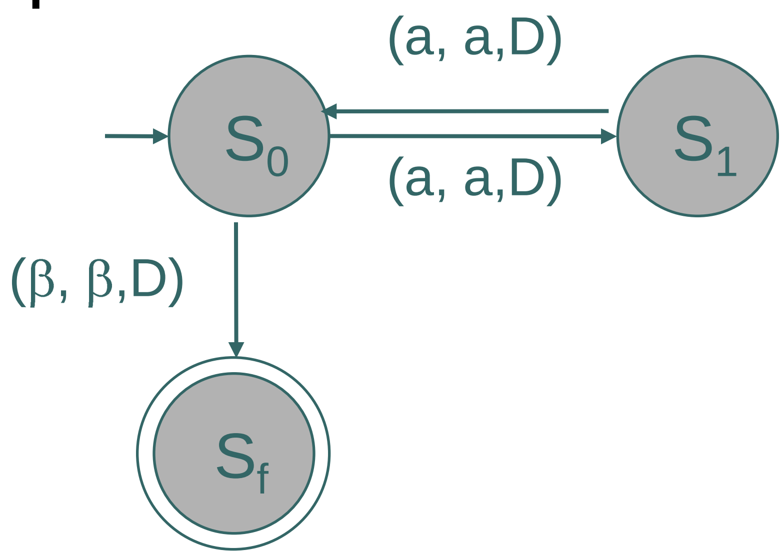
Máquina de Turing

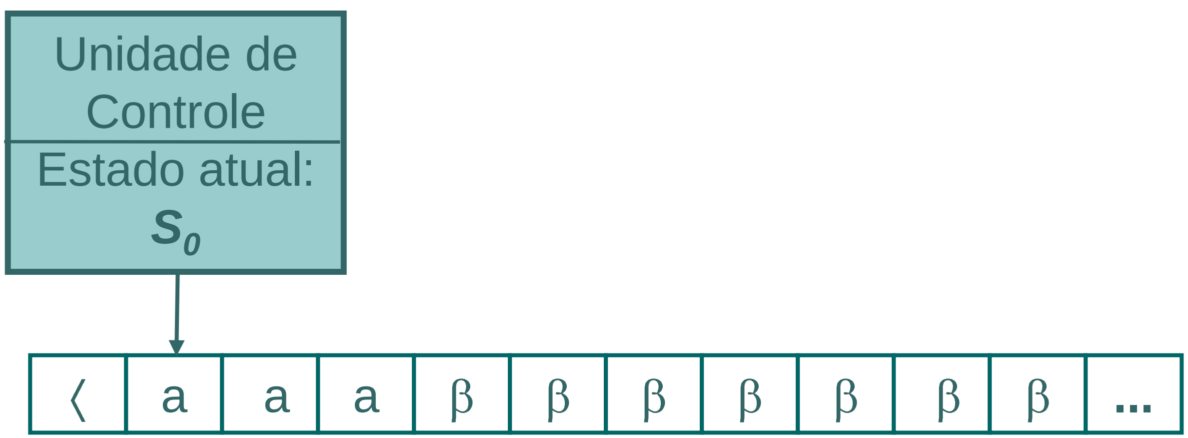
- Exemplo : uma MT com alfabeto de entrada {a} que páre se, e somente se, a palavra de entrada for da forma a^{2n} para n>=0



Máquina de Turing

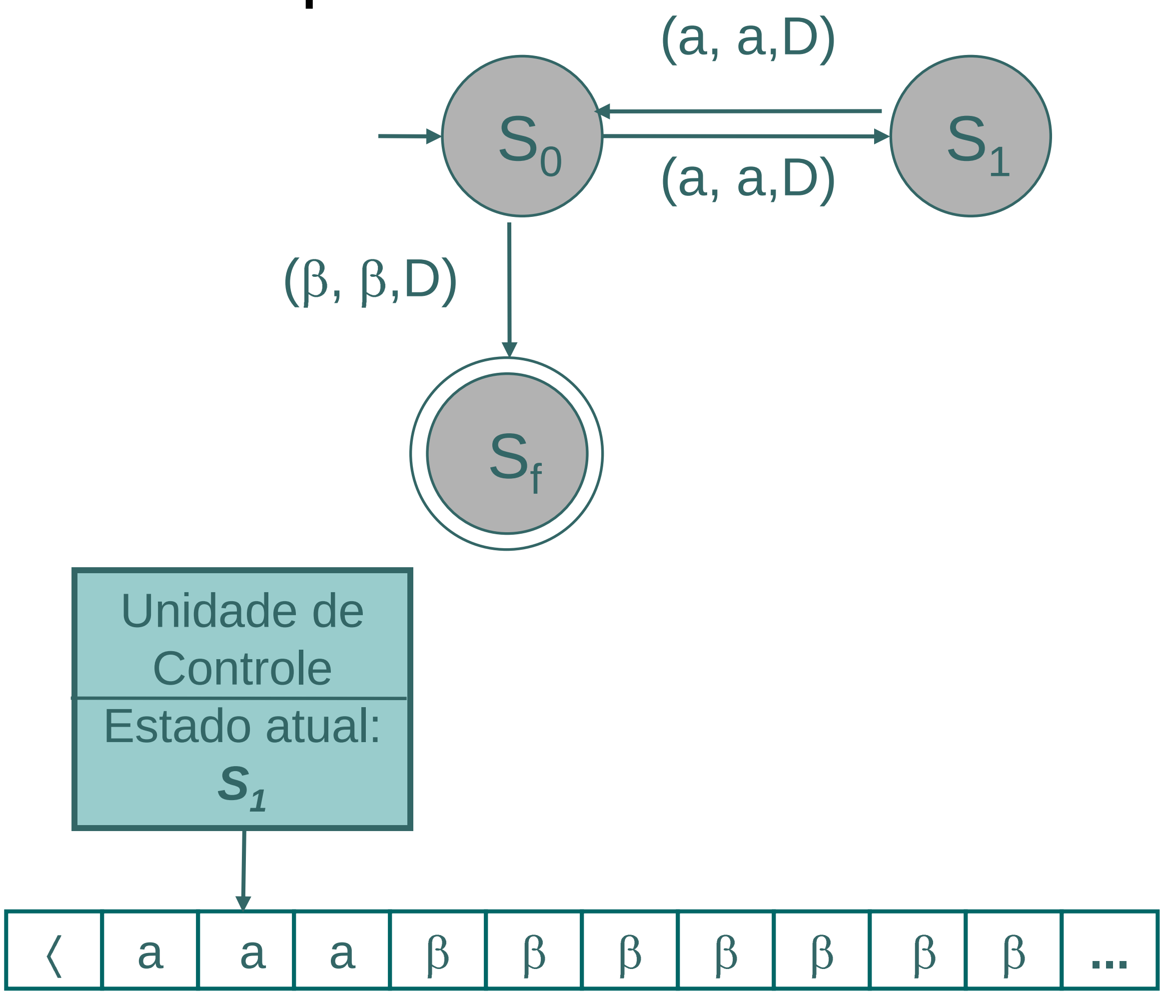
- Exemplo : Processamento de “aaa”





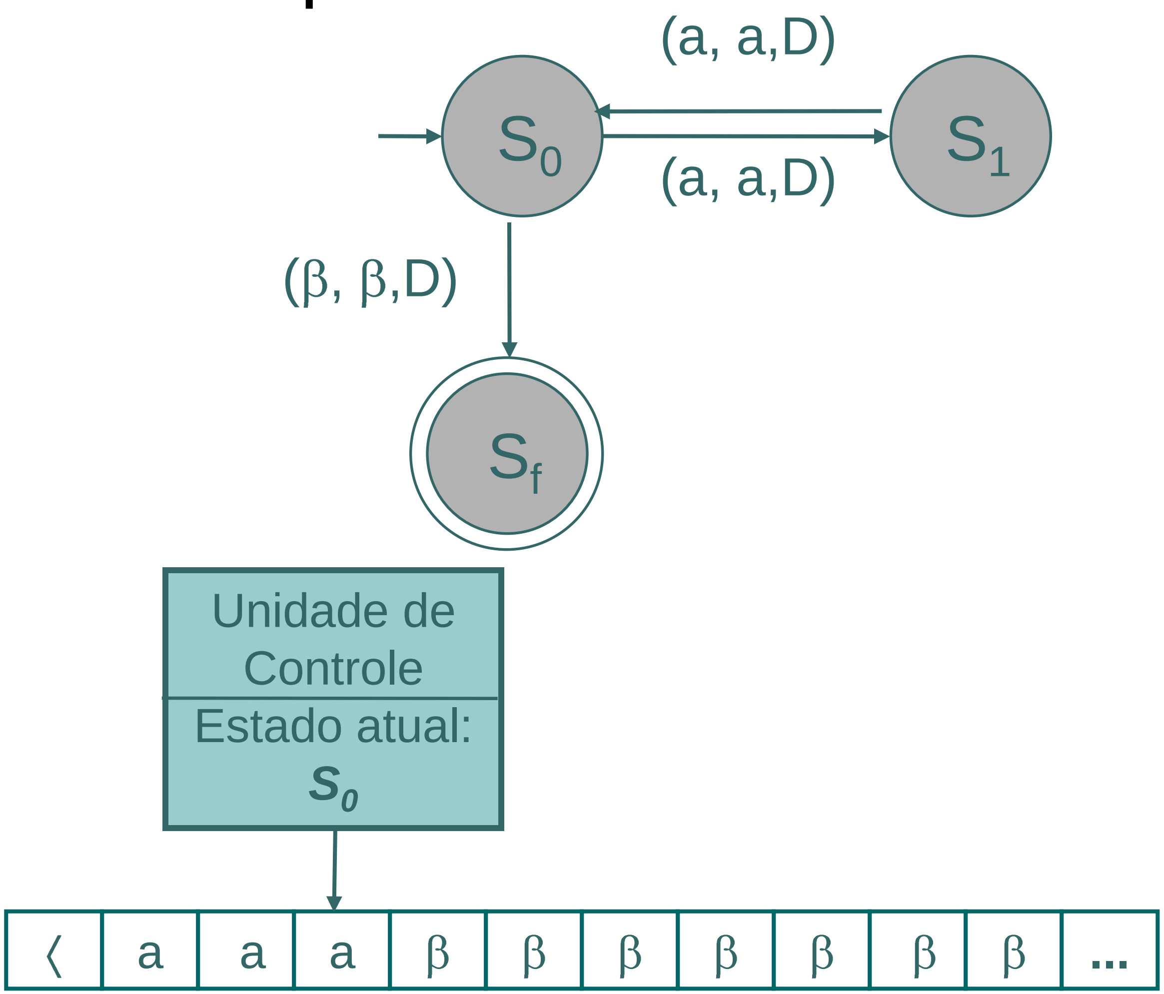
Máquina de Turing

- Exemplo : Processamento de “aaa”



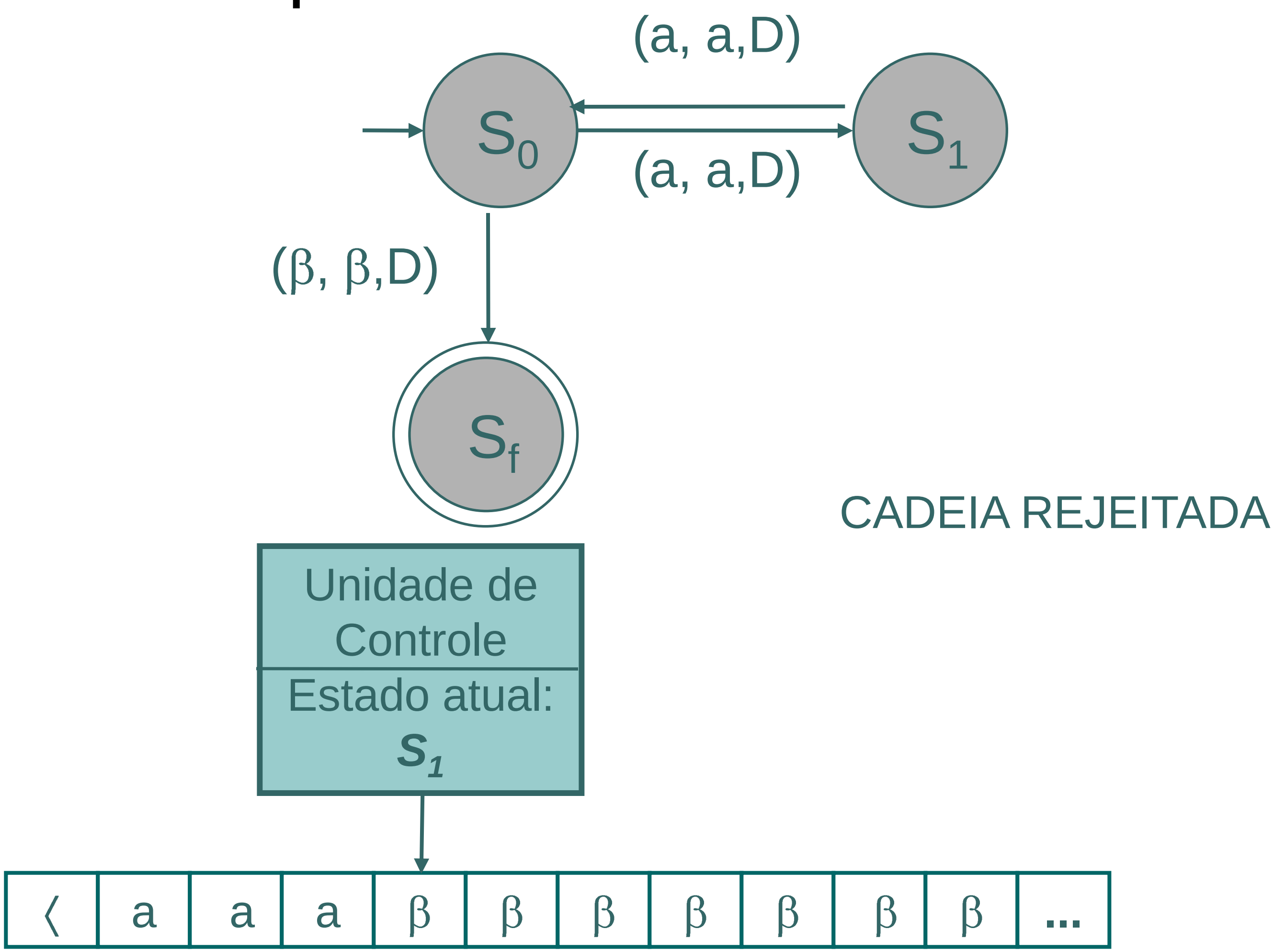
Máquina de Turing

- Exemplo : Processamento de “aaa”



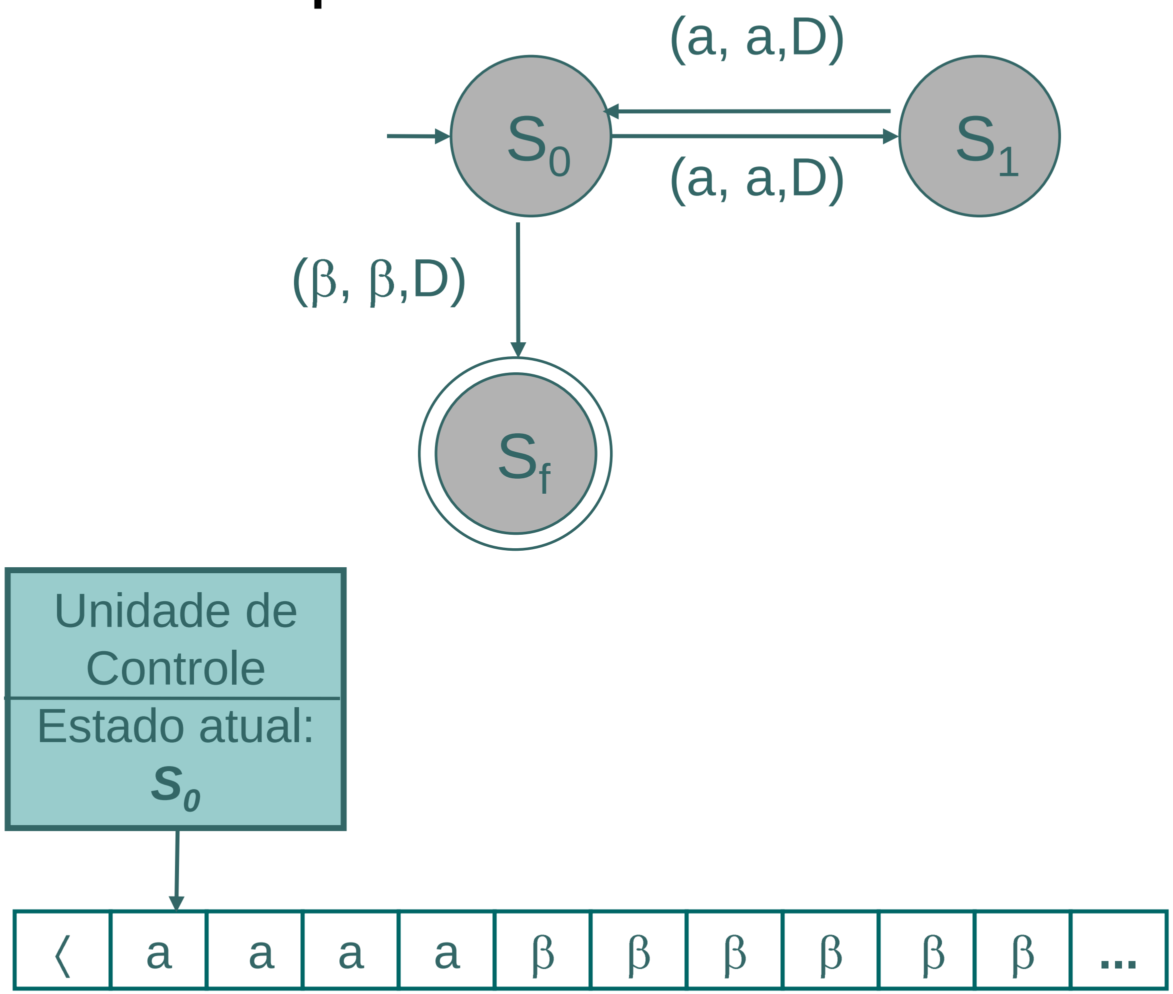
Máquina de Turing

- Exemplo : Processamento de “aaa”



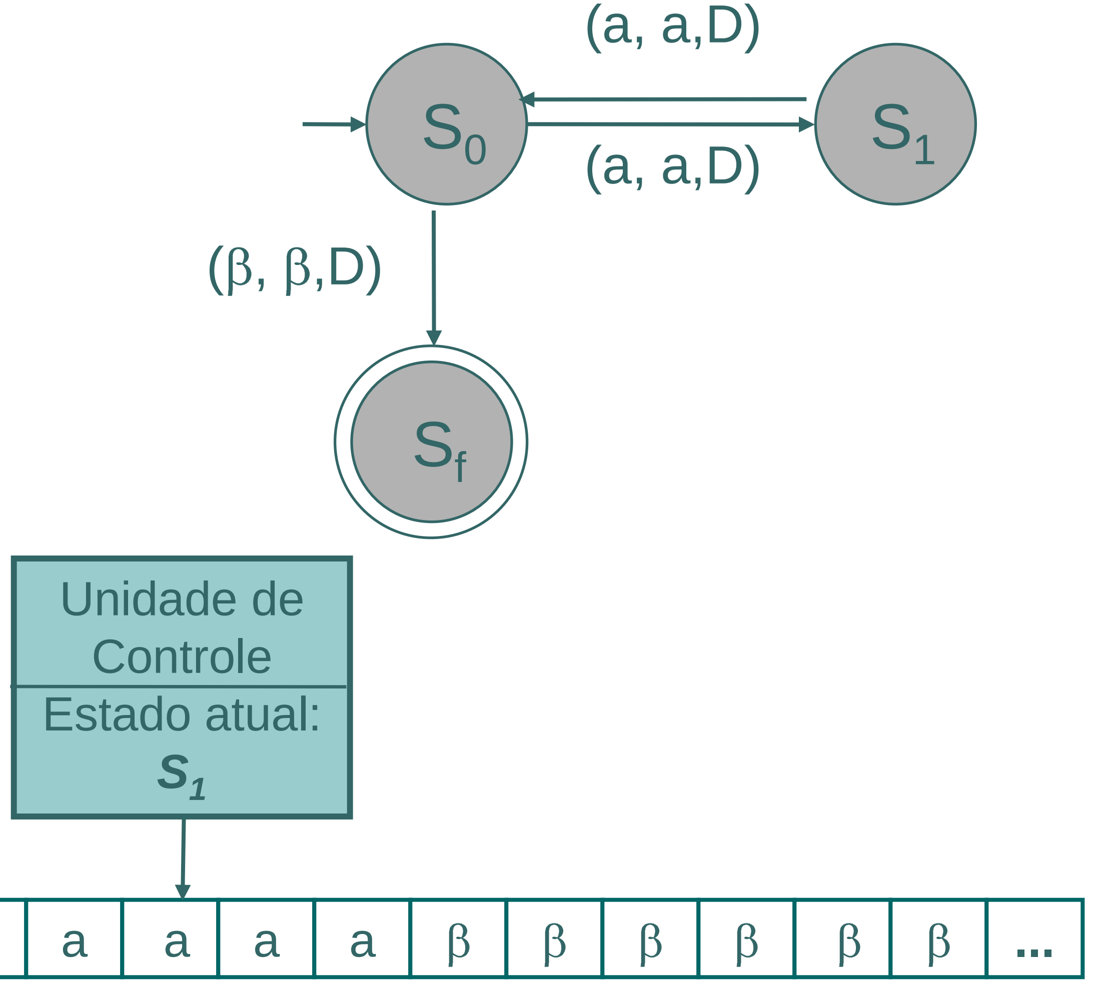
Máquina de Turing

- Exemplo : Processamento de “aaaa”



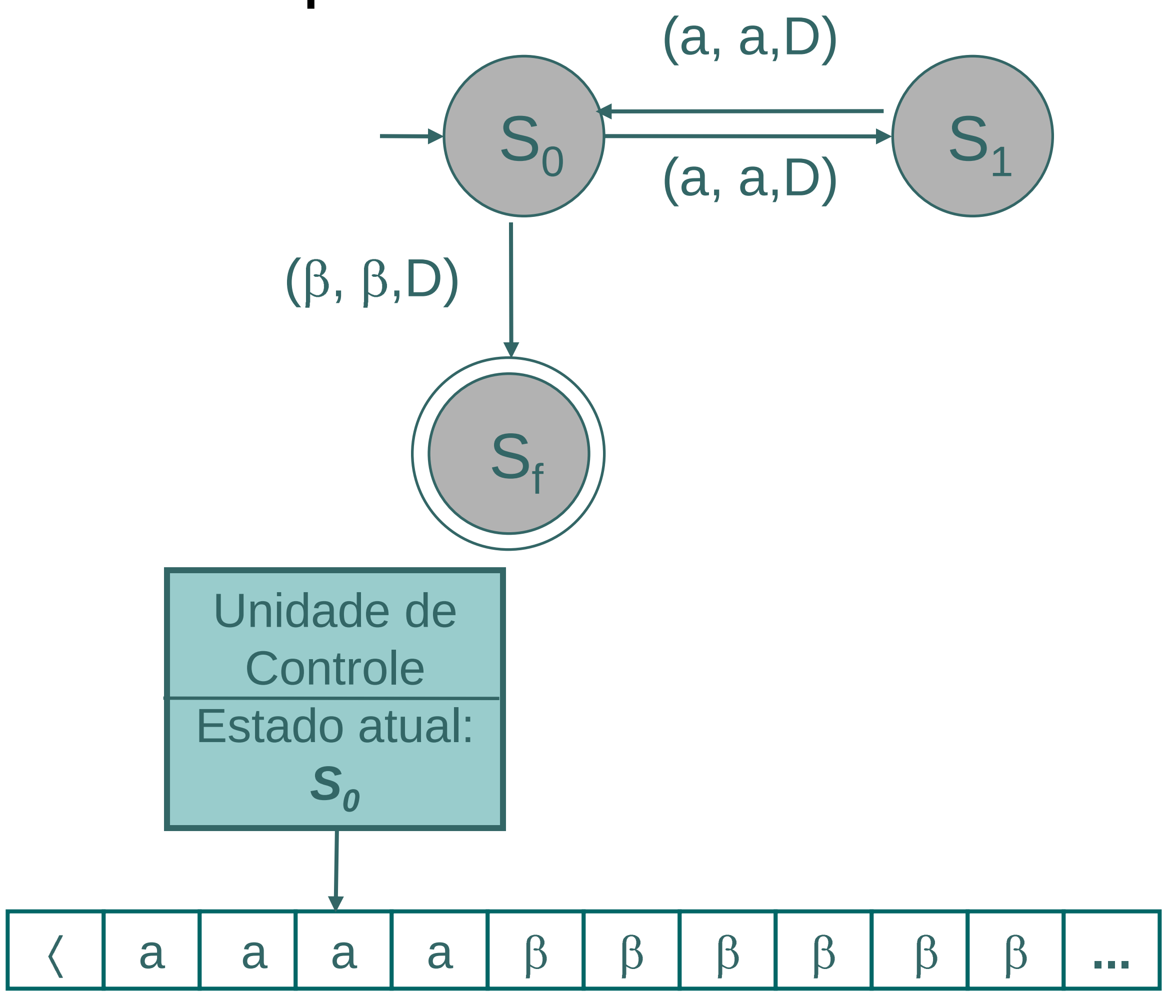
Máquina de Turing

- Exemplo : Processamento de "aaaa"



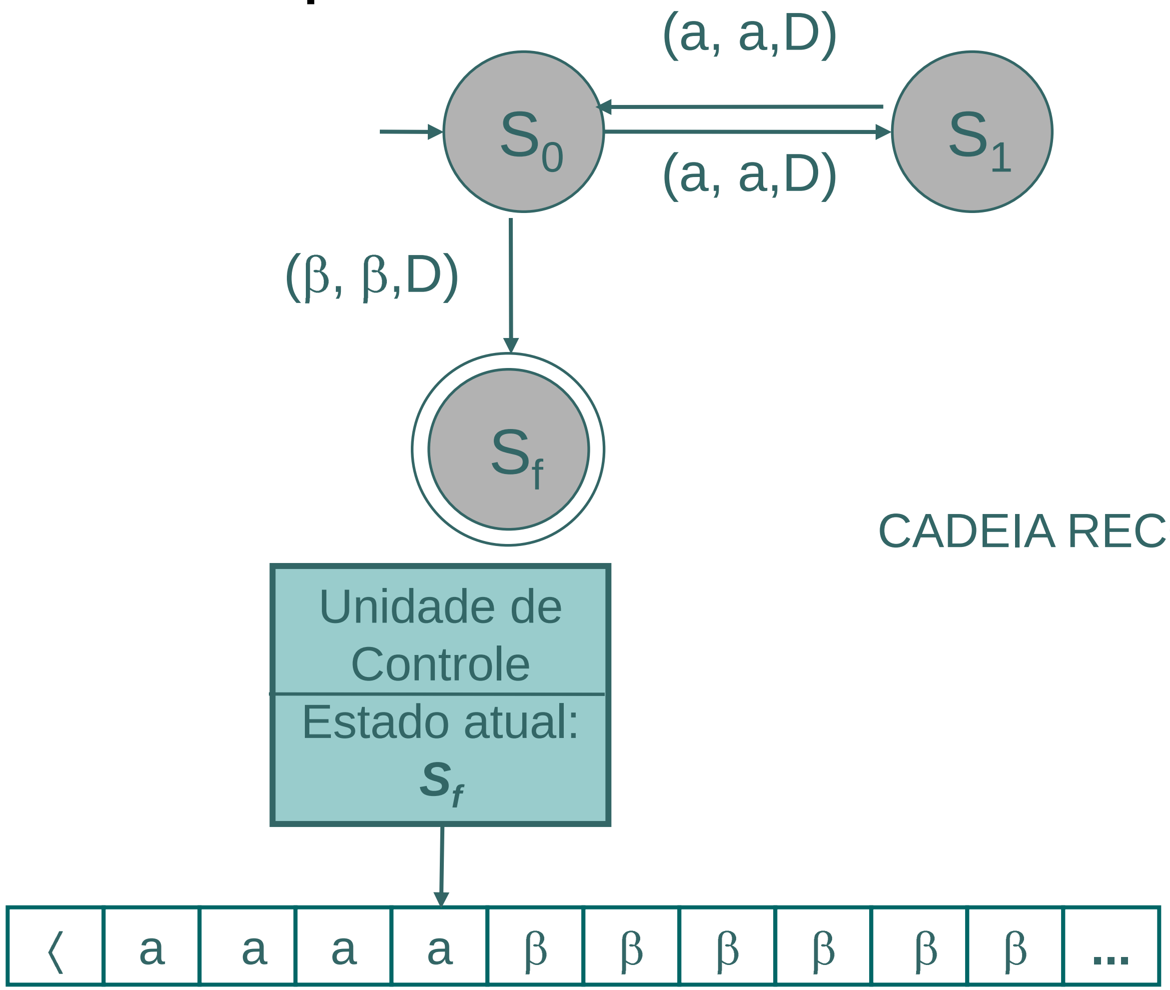
Máquina de Turing

- Exemplo : Processamento de “aaaa”



Máquina de Turing

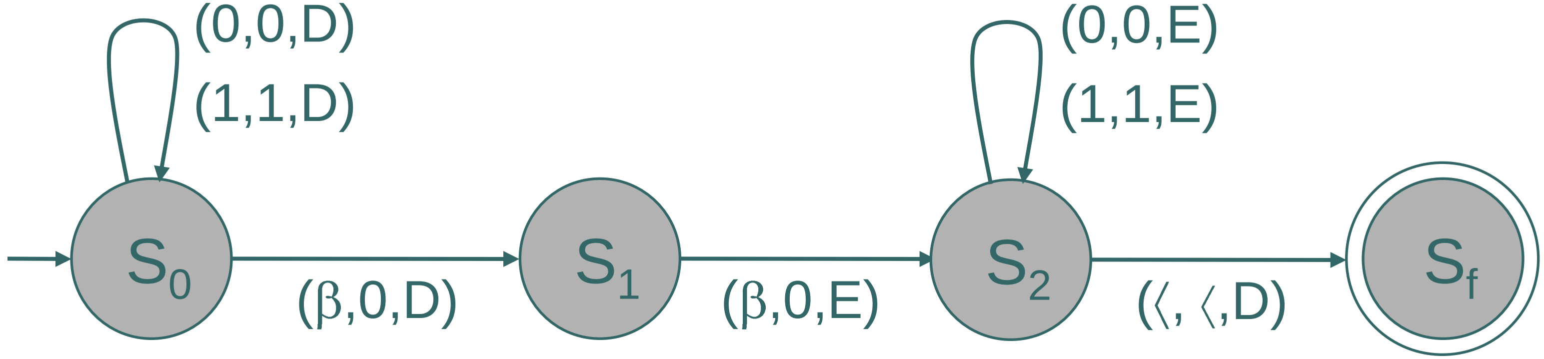
- Exemplo : Processamento de “aaaa”



CADEIA RECONHECIDA

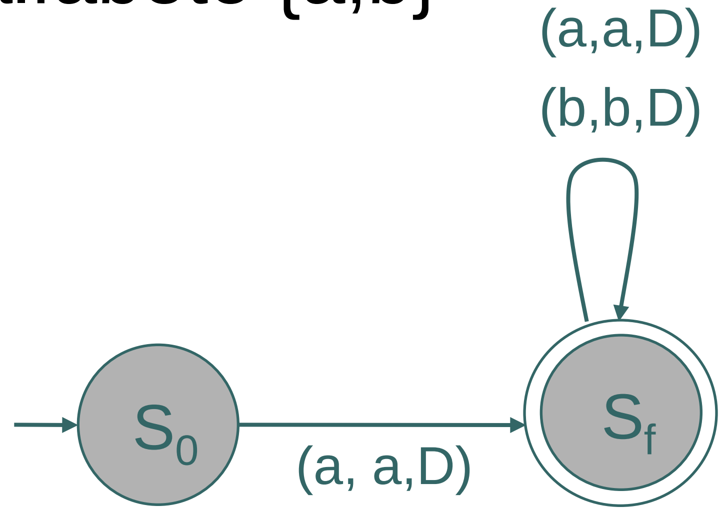
Máquina de Turing

- Exercício 1 (lista Yandre 1a):
  - Construa uma MT transdutora que receba como entrada um número binário e devolva o quádruplo do mesmo.



Máquina de Turing

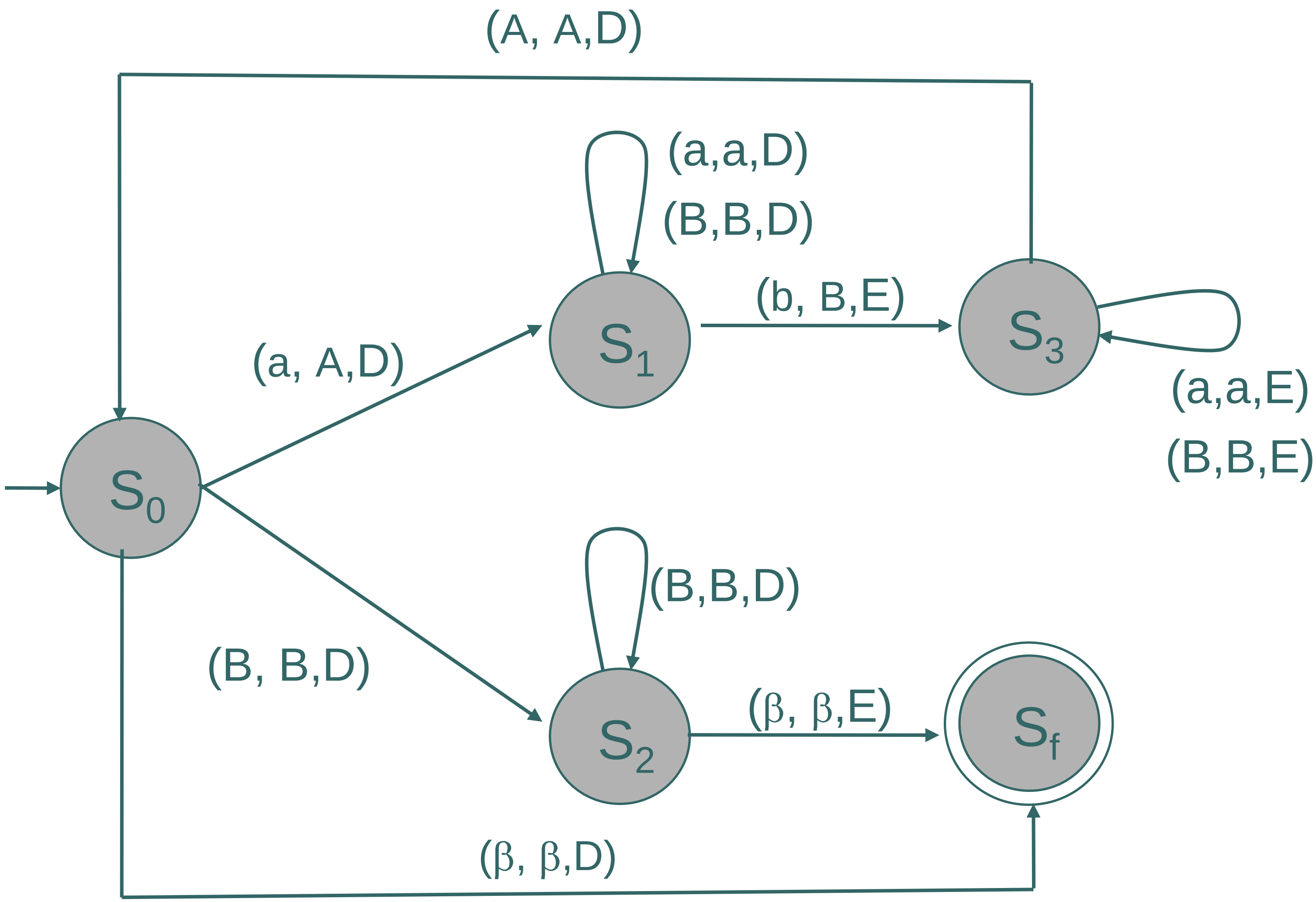
- Exercício 2: Construa uma MT com menor número de estados possível que reconheça a linguagem denotada pela ER a(a+b)*, assumindo o alfabeto {a,b}



Máquina de Turing

- Exercício 3: Construa uma MT com menor número de estados possível que reconheça a linguagem denotada $\{a^n b^n | n \geq 0\}$ – (lista Yandre 3c)

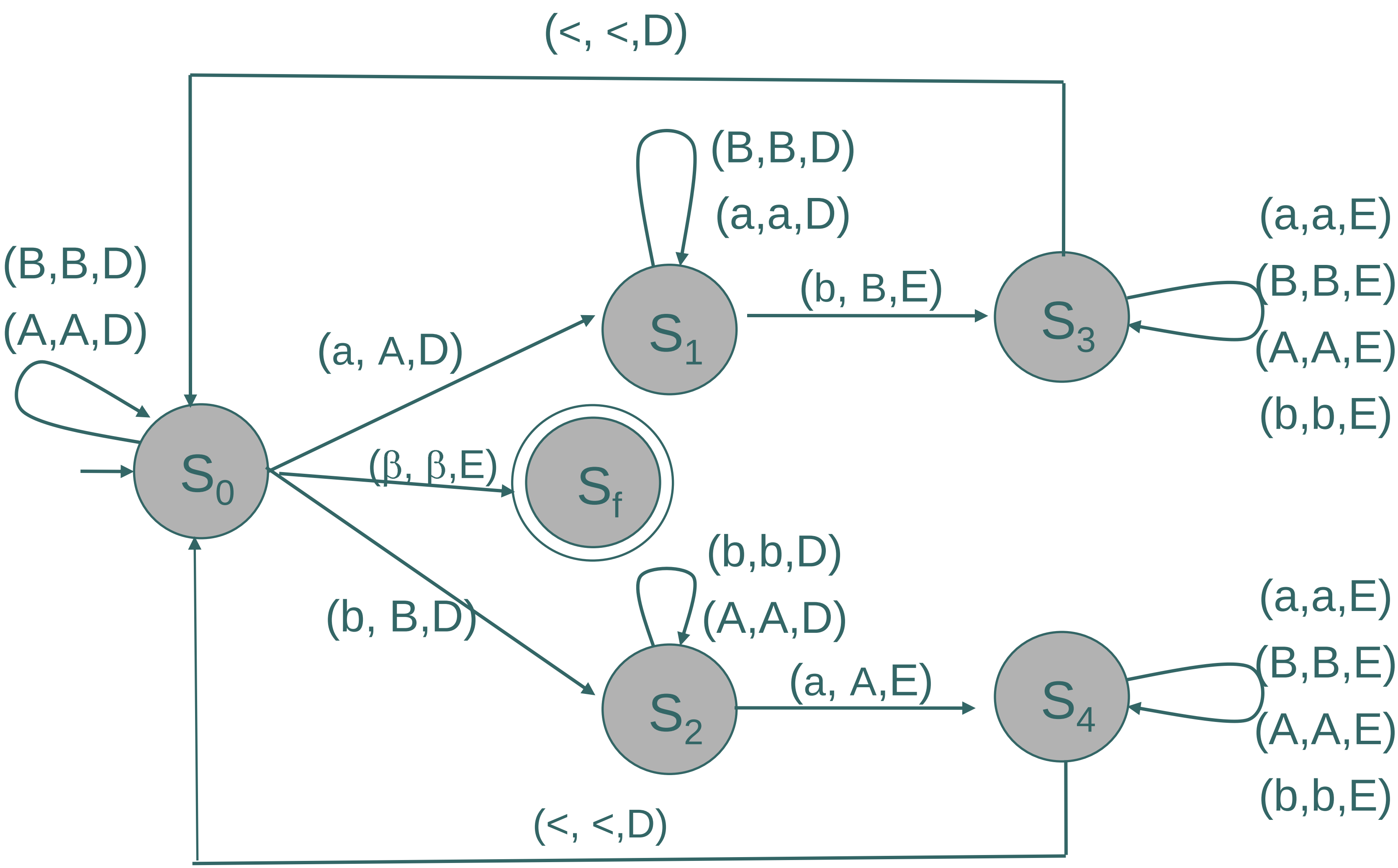
Exercício 3



Máquina de Turing

- Exercício 4: Construa uma MT com menor número de estados possível que reconheça a linguagem denotada {W pertence {a,b}| o número de a s em W é igual ao número de b s}.

Exercício 4

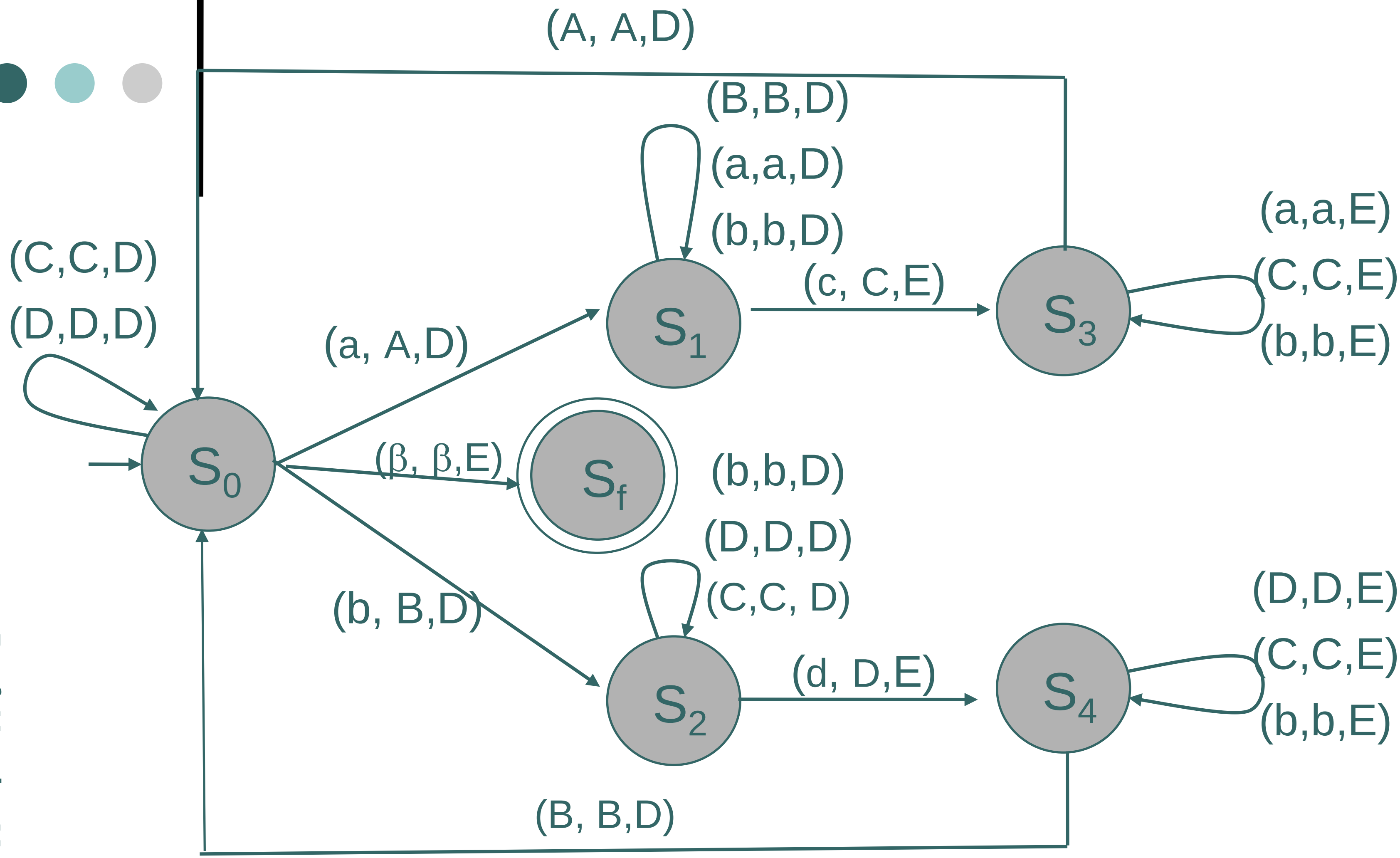


(<, <,D)
(B,B,D)
(a,a,D)
(a,a,E)
(B,B,D)
(A,A,D)
(b, B,E)
(B,B,E)
(A,A,E)
(b,b,E)
(a, A,D)
S1
S3
S0
(β, β,E)
Sf
(b,b,D)
(A,A,D)
(a,a,E)
(B,B,E)
(A,A,E)
(b,b,E)
(b, B,D)
(a, A,E)
S2
S4
(<, <,D)

Máquina de Turing

- Exercício 8: Construa uma MT com menor número de estados possível que reconheça por estado final e com cabeçote imóvel a seguinte linguagem de alfabeto {a,b,c,d}

  - $\{a^m b^n c^m d^n \mid m, n \text{ pertence a } N\}$


(A, A,D)
(B,B,D)
(a,a,D)
(b,b,D)
(a,a,E)
(C,C,E)
(b,b,E)
(C,C,D)
(D,D,D)
(c, C,E)
(a, A,D)
S1
S3
S0
(β, β,E)
Sf
(b,b,D)
(D,D,D)
(C,C, D)
(D,D,E)
(C,C,E)
(b,b,E)
(b, B,D)
(d, D,E)
S2
S4
(B, B,D)

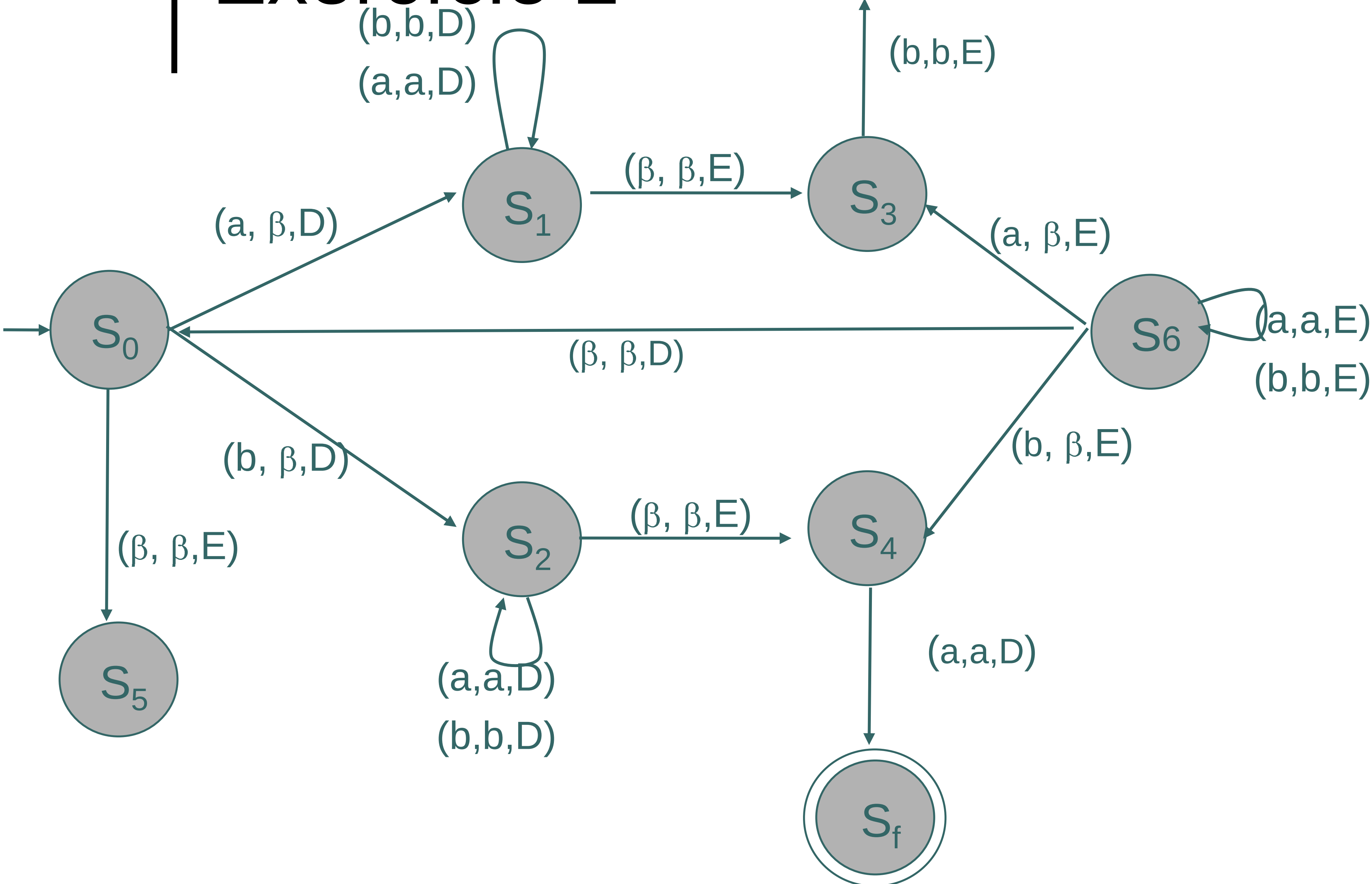
Exercícios para casa:

1. Construa uma Máquina de Turing para a seguinte linguagem (com o mínimo de estados):
   $\{ a^n b^n c^n \mid n \geq 0 \}$
   (Yandre 3d)

2. Construa uma MT que reconheça a linguagem { w pertence $\{a,b\}^* \mid w \neq w^R\}$, ou seja, o conjunto das palavras sobre {a,b} que não são palíndromos.

3. Construa uma MT com fita bidirecional e alfabeto {a} que pare se a fita contiver célula em branco.

Exercício 2



(b,b,D)
(a,a,D)
S_f
(b,b,E)
(β, β,E)
S_1
S_3
(a, β,D)
(a, β,E)
S_0
S_6
(a,a,E)
(b,b,E)
(β, β,D)
(b, β,E)
(b, β,D)
(β, β,E)
S_2
S_4
(β, β,E)
S_5
(a,a,D)
(a,a,D)
(b,b,D)
S_f

Máquina de Turing

- Máquina de Turing
  - Dada a sua natureza conceitual, a MT pode ser implementada de diversas formas;
  - Os computadores modernos são MT (exceto pelo fato de terem memória finita)
    - O processador corresponde à unidade de controle, cujos estados podem ser definidos pelos padrões de bits que podem ser associados aos registradores;
    - A memória da máquina corresponde ao sistema de armazenamento em fita;
    - Os padrões de bits (0 e 1) correspondem ao alfabeto da fita.

Máquina de Turing

- Importância da MT para a Ciência da Computação:
  - A potência computacional da MT é tão grande quanto a de qualquer sistema algorítmico;
  - Se um problema não puder ser resolvido por uma MT, não poderá ser resolvido por qualquer sistema algorítmico;
  - MT representa a fronteira teórica da capacidade computacional para as máquinas modernas reais.

MÁQUINA UNIVERSAL

- Se for possível representar qualquer algoritmo como um programa em uma máquina simples e poderosa então tal máquina é de fato uma **MÁQUINA UNIVERSAL**

Máquina deveria ser

- **Simples**
  - para permitir estudos de propriedades sem a necessidade de considerar características não-relevantes, bem como permitir estabelecer conclusões gerais sobre a classe de funções computáveis

- Poderosa
  - capaz de simular qualquer característica de máquinas reais ou teóricas, de tal forma que os resultados provados fossem válidos para modelos aparentemente com mais recursos e para que qualquer função computável possa ser nela representada.

Máquinas Universais

- As evidências de que máquina é de fato universal, podem se classificadas como:
  - **EVIDÊNCIA INTERNA**

    Demonstração de que qqr extensão das capacidades da máquina proposta computa, no máximo a mesma classe de funções, ou seja, não aumenta o poder computacional
  - **EVIDÊNCIA EXTERNA**

    Consiste no exame de outros modelos que definem a noção de algoritmo, juntamente com a prova de que são, no máximo, computacionalmente equivalentes.

Máquinas Universais

O modelo de máquina universal mais utilizado é a Máquina de TURING

- A MT possui tantas células de armazenamento de dados quanto necessário

- Em 1936, Alonzo Church apresentou a
  - **Hipótese de Church**

Máquinas Universais

Hipótese de Church:

- Qualquer função computável pode ser processada por uma MT, ou seja,
- Existe um algoritmo expresso na forma de MT capaz de processar a função.

- *Mas essa noção intuitiva de algoritmo não é matematicamente precisa.*

Máquinas Universais

- Modelos equivalentes à MT:
  - Máquina Norma
    - Máquina de registradores onde o conjunto de registradores é infinito
  - Máquina de Post
    - Baseada na estrutura de dados do tipo fila (o primeiro dado armazenado é o primeiro a ser recuperado)
  - Máquinas com Pilhas
    - São necessárias pelo menos duas pilhas para simular o mesmo poder computacional de um fita ou fila

Extensões da MT não aumentam seu poder computacional

- Extensões:
  - Não determinismo:
    - Permite que a máquina possa tentar diversos caminhos alternativos para uma mesma situação
  - Múltiplas fitas
  - Múltiplas unidades de controle
  - Fitas infinitas nas duas extremidades

Máquinas de Turing

- Existem 3 maneiras de abordar o estudo das Máquinas de Turing e de seus modelos equivalentes:
  - Reconhecimento de Linguagens
    - Linguagens que podem ser reconhecidas e suas propriedades
  - Processamento de Funções
    - Funções computáveis e suas propriedades
  - Solucionabilidade de Problemas: problemas solucionáveis e não-solucionáveis, problemas parcialmente solucionáveis (computáveis) e completamente insolúveis (não computáveis) e suas propriedades.

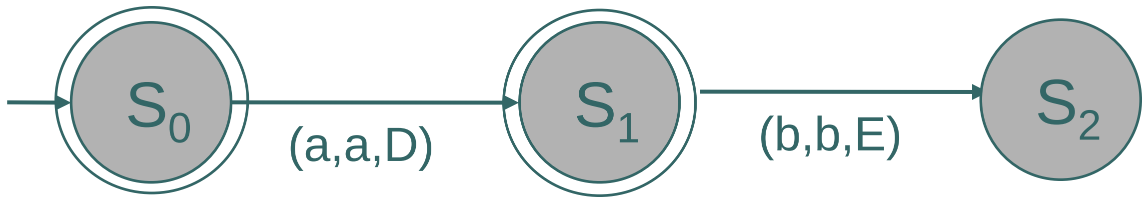
Máquina de Turing

Reconhecimento de Linguagens

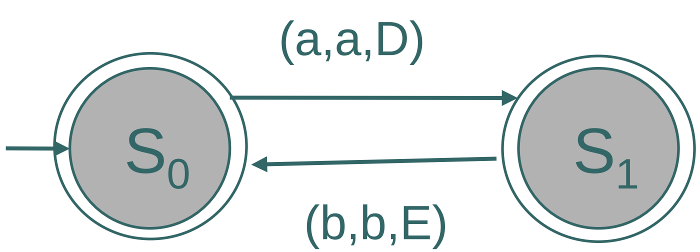
Máquinas que param sempre e que param se aceitam.

- MT que aceitam a linguagem das palavras no alfabeto {a,b,c} que não têm ab como prefixo, ou seja, {a,b,c}*-({ab}{a,b,c}*)

Máquinas que param sempre e que param se aceitam.



Máquina que pára sempre



Máquina que pára se aceita

MT como Reconhecedores de Linguagens

- A linguagem aceita por M, denotada por **ACEITA(M)** ou L(M), é o conjunto de todas as palavras pertencentes a Σ^* aceitas por M, ou seja:

  - ACEITA(M)=

  $\{w \mid M$ ao processar $w \in \Sigma^*$, pára em um estado $qf \in F\}$

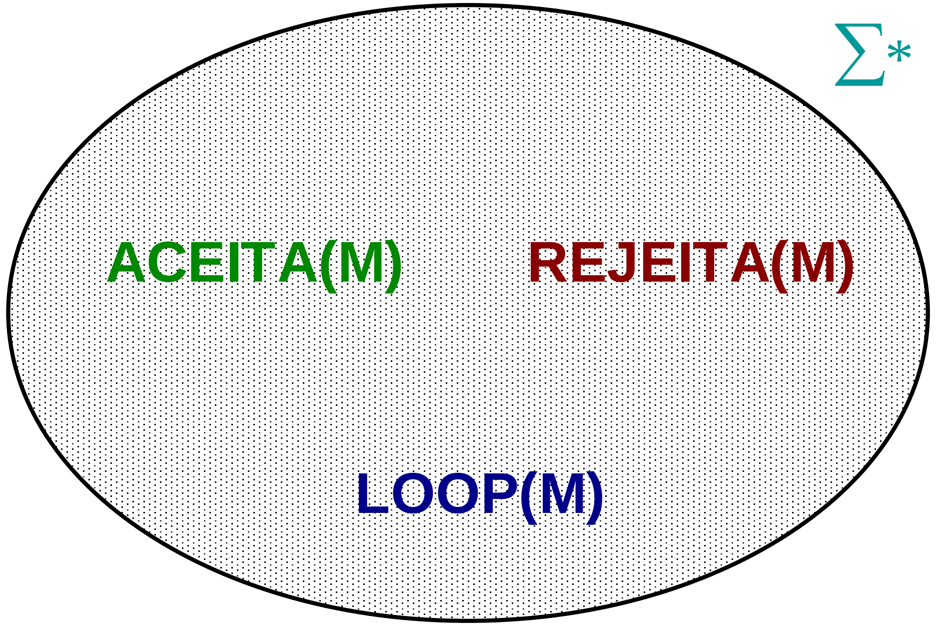
MT como Reconhecedores de Linguagens

- A linguagem rejeitada por M, denotada por REJEITA(M), é o conjunto de todas as palavras de Σ^* rejeitadas por M, ou seja:
  - REJEITA(M) = $\{w \mid$ M ao processar $w \in \Sigma^*$, pára em um estado $q \notin F\}$

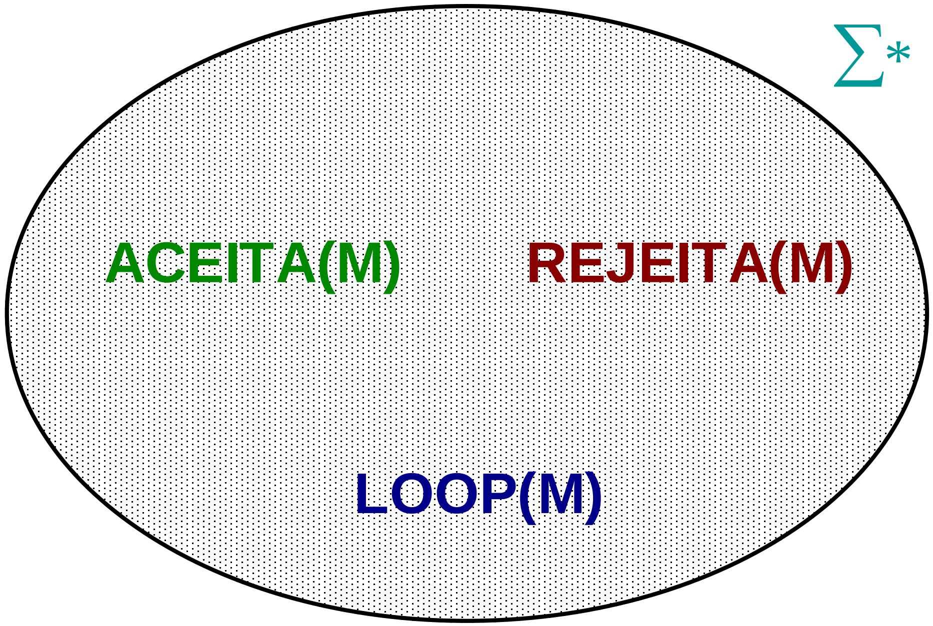
MT como Reconhecedores de Linguagens

- A linguagem para a qual M fica em *loop* infinito, denotada por LOOP(M) é conjunto de todas as palavras de $\sum^*$ para as quais M fica processando indefinidamente.

MT como Reconhecedores de Linguagens



MT como Reconhecedores de Linguagens



$$ACEITA(M) \cap REJEITA(M) = \emptyset$$
$$ACEITA(M) \cap LOOP(M) = \emptyset$$
$$REJEITA(M) \cap LOOP(M) = \emptyset$$
$$ACEITA(M) \cap REJEITA(M) \cap LOOP(M) = \emptyset$$
$$ACEITA(M) \cup REJEITA(M) \cup LOOP(M) = \Sigma^*$$

MT como Reconhecedores de Linguagens

- *Critério para o Reconhecimento de Linguagens.*

  - *Se a máquina pára para toda palavra da linguagem sobre o alfabeto de entrada, ela é reconhecida pela Máquina de Turing.*

Linguagem Enumerável Recursivamente

- Uma linguagem aceita por uma Máquina de Turing é dita *enumerável recursivamente*.

  - ***Enumerável*** deriva do fato de que as palavras de qualquer linguagem *enumerável recursivamente* podem ser enumeradas ou listadas por uma Máquina de Turing.
  - ***Recursivamente*** é um termo matemático, anterior ao computador, com significado similar ao de recursão, utilizado na computação.

Linguagem Enumerável Recursivamente

- A *classe das* ***Linguagens Enumeráveis Recursivamente*** inclui as
  - **linguagens livre do contexto** e algumas outras linguagens para as quais não se pode, mecanicamente, determinar se uma dada palavra pertence ou não à linguagem.

Linguagem Enumerável Recursivamente

- Se **L** é uma dessas linguagens, então para qualquer máquina **M** que aceita a linguagem L, existe pelo menos uma palavra w, não pertencente a L, que, ao ser processada por M, resulta que a máquina entre em *loop* infinito.

- Se w pertence a L, M ***pára e aceita*** a entrada

- Se w não pertence a L, M pode parar, rejeitando a palavra, ou permanecer processando indefinidamente (*loop*).

Exemplos Linguagem Enumerável Recursivamente

- As seguintes linguagens são exemplos de linguagens Enumeráveis Recursivamente:
  - Duplo_Bal = $\{ a^n b^n \mid n \geq 0\}$
  - Triplo_Bal = $\{ a^n b^n c^n \mid n \geq 0\}$
  - Palavra_Palavra = { ww | w é palavra sobre os símbolos a e b}
  - { w / w tem o mesmo número de símbolos a que b}
  - $\{ a^i b^j c^k \mid i=j \text{ ou } j=k\}$

Classe das Linguagens Recursivas

- Uma **sub-classe** da Classe das **Linguagens Enumerável Recursivamente**, denominada Classe das Linguagens Recursivas, é composta pelas linguagens para as quais existe pelo menos uma **Máquina de Turing** que ***pára*** para qualquer entrada, aceitando ou rejeitando.

Classe das Linguagens Recursivas

A máquina não entra em loop

- **Definição Linguagem Recursiva**
  - Uma linguagem é dita *recursiva* se existe uma Máquina de Turing tal que:
    - ACEITA(M) = L
    - REJEITA(M) = $\sum^*$ - L.
    - LOOP(M) = $\emptyset$

Classe das Linguagens Recursivas

- Pode-se afirmar que a classe das **Linguagens Recursivas** representa todas as linguagens que podem ser reconhecidas mecanicamente.

- Existem conjuntos que **não são Enumeráveis Recursivamente**, ou seja, linguagens para as quais não é possível desenvolver uma MT que as reconheça.

Propriedades das Linguagens Recursivas

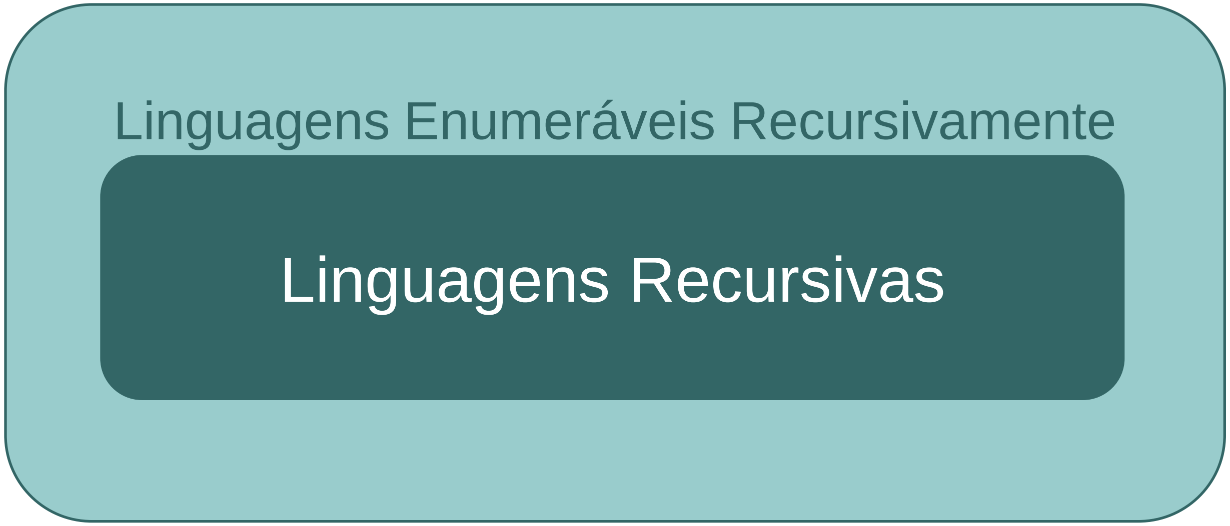
- Se uma linguagem L sobre um alfabeto ∑ qualquer é recursiva, então seu complemento, ou seja, ∑*-L, é recursivo.

Propriedades das Linguagens Recursivas

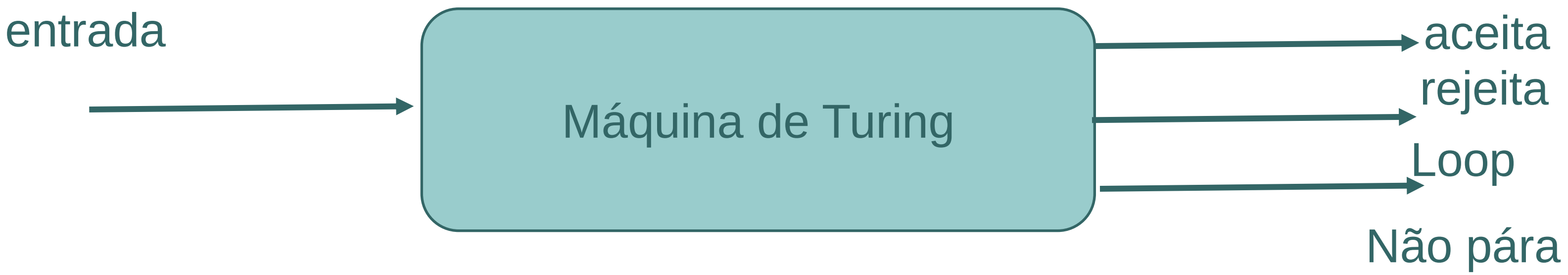
- Uma linguagem L sobre um alfabeto Σ qualquer é recursiva se, e somente se, L e seu complemento são enumeráveis recursivamente.

Propriedades das Linguagens Recursivas

- A Classe das Linguagens Recursivas está contida propriamente na Classe das Linguagens Enumeráveis Recursivamente.

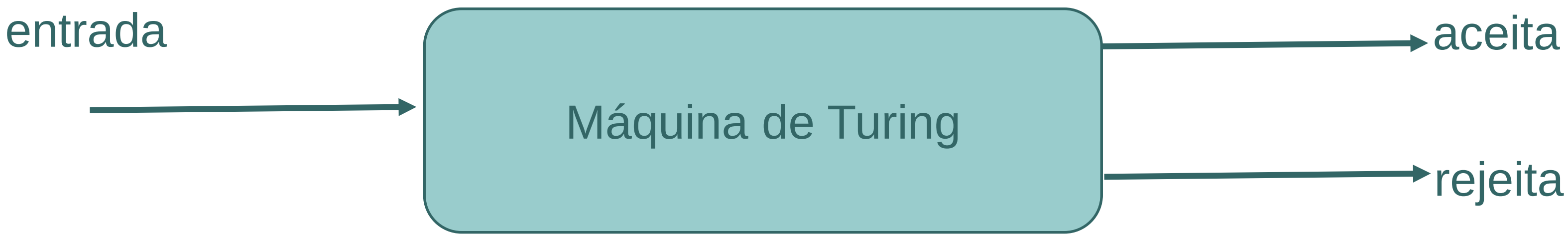


Linguagens Enumeráveis Recursivamente.



TURING RECONHECÍVEL

Linguagens Recursiva Param para toda entrada



TURING DECIDÍVEL

A MT sempre pára

Linguagens Recursiva TURING DECIDÍVEIS

- As linguagens decidíveis são fechadas sob as operações
  - União,
  - Concatenação
  - Kleene,
  - Complemento
  - Interseção

Linguagem Recursiva Enumeráveis
TURING RECONHECÍVEIS

- As linguagens RECURSIVAS ENUMERÁVEIS são fechadas sob as operações
  - União,
  - Concatenação
  - Kleene,
  - Interseção

Toda linguagem DECIDÍVEL é TURING RECONHECÍVEL

Nem toda linguagem TURING RECONHECÍVEL é TURING DECIDÍVEL

DECIDIBILIDADE

- Na teoria da computação e na teoria da complexidade computacional um problema de decisão é uma questão sobre um sistema formal com uma resposta do tipo
- Sim ou não

DECIDIBILIDADE

- Dados dois números
  - X e Y
  - Y é divisível por X?

- Dado um número inteiro x
  - X é número primo?

DECIDIBILIDADE

- Um problema de decisão também pode ser formalizado como o problema de verificar se determinada CADEIA DE CARACTERES pertence ou não a uma linguagem formal

DECIDIBILIDADE

- Um problema de decisão é chamado DECIDÍVEL ou EFETIVAMENTE SOLÚVEL se tem uma representação na qual o conjunto de strings aceito é um LINGUAGEM RECURSIVA.

DECIDIBILIDADE

- Um problema de decisão é chamado SEMI-DECIDÍVEL ou SOLÚVEL ou PROVÁVEL se tem uma representação na qual o conjunto de strings aceito é um LINGUAGEM RECURSIVAMENTE ENUMERÁVEL.

DECIDIBILIDADE

- Um problema de decisão é chamado INDECIDÍVEL ou INSOLÚVEL ou IMPROVÁVEL se
- NÃO HÁ UMA MÁQUINA DE TURING QUE O SOLUCIONE

DECIDIBILIDADE

Um exemplo clássico de um problema de decisão é o conjunto dos número primos:

É possível decidir efetivamente se um número natural é primo

DECIDIBILIDADE

Problemas de decisão INDECIDÍVEIS incluem o PROBLEMA DA PARADA

"*Dada uma MT arbitrária M e uma palavra arbitrária w, determinar se a computação de M com entrada w pára*"

Exercício

- Sejam L uma LRE e R uma linguagem recursiva, mostre:

a) L – R é uma LRE

- Tem-se que $L - R = L \cap \overline{R}$.
- Como R é uma LRec e a clase das LRecs é fechada sob complementação, tem-se que $\overline{R}$ também é uma LRec e, portanto,uma LRE. Como a classe das LREs é fechada sob interseção,$L \cap \overline{R}$ é uma LRE. Logo, L – R é uma LRE.

Exercício

- Sejam L uma LRE e R uma linguagem recursiva, mostre:

L – R pode não ser recursiva

Seja R = ø uma LR e, portanto, uma LRec.

Então, L – R = L – ø = L. Logo, se L for uma LRE não recursiva, então L-R pode não ser recursiva.

Exercício

- Sejam L uma LRE e R uma linguagem recursiva, mostre:

R - L pode não ser um LRE

Seja R = Σ^* uma LR e, portanto, uma LRec.

Então, R - L = $\Sigma^* - L = \overline{L}$. Como a classe das LRE não é fechada sob complementação, $\overline{L}$ pode não ser uma LRE.

Bibliografia

- BROOKSHEAR, J. G. Ciência da Computação. Porto Alegre: Bookman, 2000;
- DELAMARO, Márcio Eduardo. Linguagens Formais e Autômatos. Maringá: UEM, 1998;
- HOPCROFT, J. E. & ULLMAN, J. D. Formal Languages and Their Relation to Automata. Addison-Wesley, 1969;
- MENEZES, Paulo Blauth. Linguagens Formais e Autômatos. Porto Alegre: Editora Sagra-Luzzatto, 1998;
- VIEIRA, Newton José. Introdução aos Fundamentos da Computação. São Paulo: Pioneira Thomson Learning, 2006.