Teoria da Computação

Aula Teórica 15:
Minimização de AFDs

António Ravara

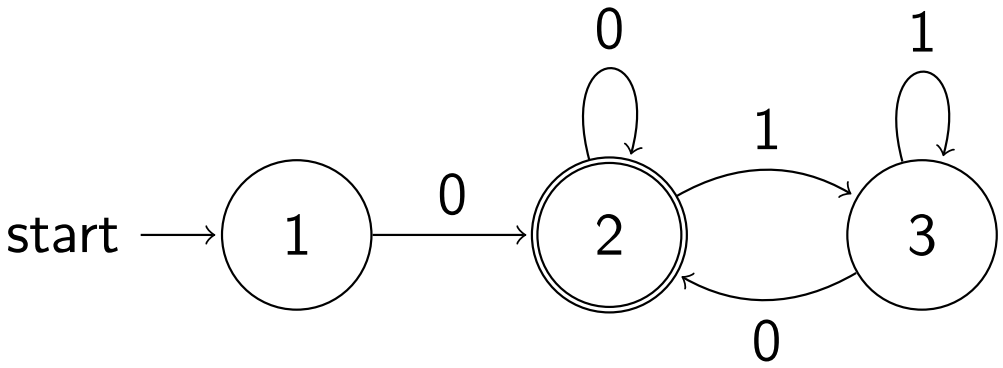
Departamento de Informática

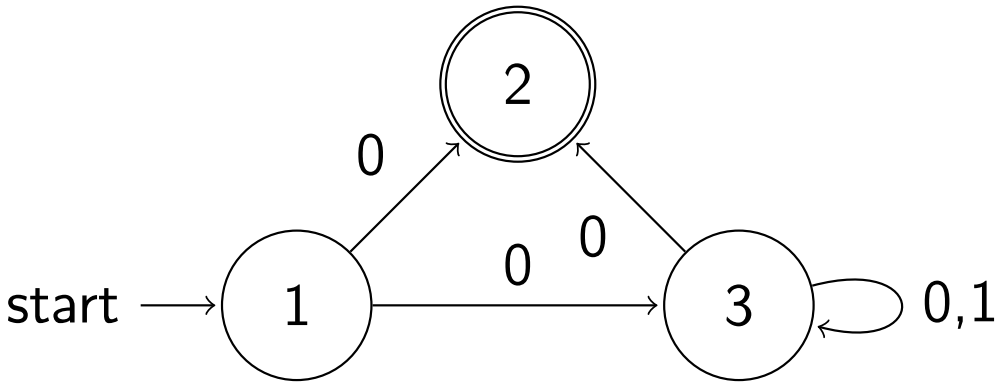
6 de Maio de 2019

Autómatos diferentes reconhecem a mesma linguagem

Exemplo: palavras sobre $\{0,1\}$ que começam e acabam em 0

$$\mathcal{L} \stackrel{\text{def}}{=} \{u \in \{0,1\}^* \mid u = 0 \vee (\exists v.v \in \{0,1\}^* \wedge u = 0v0)\}$$





Autómatos Equivalentes

- O que pode fazer um autómato?
  O conjunto das suas computações (*palavras aceites*) é a sua linguagem (reconhecida).
- Dois autómatos que reconhecem a mesma linguagem dizem-se *equivalentes*.
- Para dada linguagem regular existe uma infinidade de autómatos que a reconhecem a mesma linguagem.
- Dados 2 autómatos sintaticamente diferentes, como ver que são equivalentes?

Autómatos Especiais

autómato vacuoso: tem linguagem vazia

O autómato vacuoso canónico sobre o alfabeto Σ é $\langle \{1\}, \Sigma, \emptyset, 1, \emptyset \rangle$.

autómato trivial

- Um autómato sobre o alfabeto Σ cuja linguagem seja Σ^* diz-se *trivial*.
- O autómato trivial canónico sobre o alfabeto Σ é $\langle \{1\}, \Sigma, \{(1, a) \mapsto 1 \mid a \in \Sigma\}, 1, \{1\} \rangle$.
- Para dado Σ, nem o autómato vacuoso nem o trivial são únicos.
- Os canónicos são os "optimizados" - têm menos estados e/ou menos transições.

Questões relevantes

- Como verificar se dois autómatos sobre dado Σ são equivalentes.
  É necessário ver se aceitam a mesma linguagem – é um problema algorítmico (veremos uma solução).
- Dado um autómato:
  - existe sempre um autómato "mínimo" equivalente ao original?
  - se sim, existe algoritmo para obter esse autómato "mínimo"?
  - e o autómato "mínimo" é único?

  Estas 3 últimas questões têm todas resposta positiva!

Minimização de um autómato requer:

- eliminar estados inúteis;
- "fundir" estados equivalentes.

Estados Notáveis

Considere um autómato $D = \langle S, \Sigma, \delta, s, F \rangle$.

Estados produtivos, acessíveis e úteis

- Um estado $t \in S$ diz-se *acessível*, se existe $w \in \Sigma^*$ tal que $\delta^*(s, w) = t$
  Um estado é acessível se é alcançável do estado inicial.
- Um estado $t \in S$ diz-se *produtivo*, se existe $w \in \Sigma^*$ tal que $\delta^*(t, w) \in F$
  Um estado é produtivo se dele se alcança um estado final.
- Um estado $t \in S$ diz-se *útil*, se é acessível e produtivo.

Estados Equivalentes

Minimização de um autómato

- Eliminar estados inúteis:
  Retirando a um autómato todos os estados inúteis e as transições de e a partir deles, obtém-se um autómato equivalente (se o autómato é vacuoso, deve manter-se o estado inicial).
- "Fundir" estados equivalentes: como determinar que dois estados são equivalentes?
  Intuitivamente, se permitem "executar" as mesmas palavras.

Considere um autómato $D = \langle S, \Sigma, \delta, s, F \rangle$.

- Os estados u e v dizem-se *equivalentes*, se para qualquer $w \in \Sigma^*$ se tem $\delta^*(u, w) \in F$ se e só se $\delta^*(v, w) \in F$
- Estados não equivalentes dizem-se *distinguíveis*.

Eliminação de Estados Equivalentes

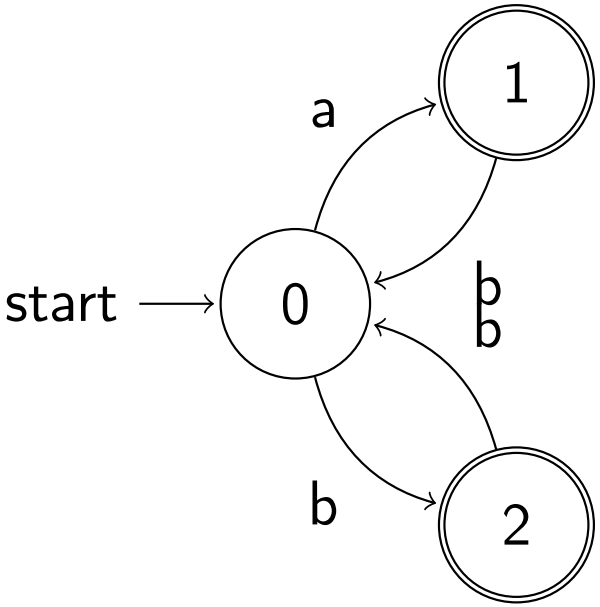
Considere um autómato $D = \langle S, \Sigma, \delta, s, F \rangle$ e dois estados u e v distintos mas equivalentes, sendo $u \neq s$.

O autómato $D' = \langle S', \Sigma, \delta', s, F' \rangle$ é equivalente a D, se:

- $S' = S \setminus \{u\}$
- $\delta' \subseteq S' \times \Sigma \to S'$, com
  - $\delta'(r, a) = v$, se $\delta(r, a) = u$
  - $\delta'(r, a) = \delta(r, a)$, se $\delta(r, a) \in S'$
  - $\delta'(r, a) = \perp$, se $\delta(r, a) = \perp$
- $F' = F \setminus \{u\}$

Exemplo de Eliminação de Estados Equivalentes

Considere o seguinte autómato em que 1 e 2 são equivalentes.



Palavras aceites pelo estado 1:

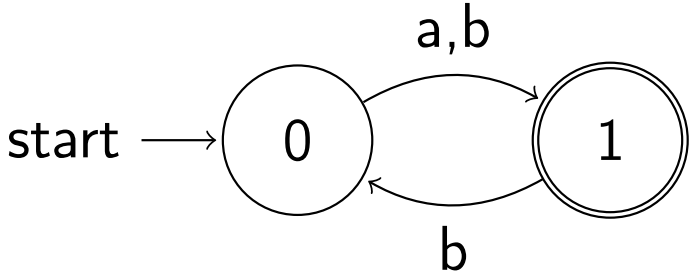
$$(b(a+b))^*$$

Palavras aceites pelo estado 2:

$$(b(b+a))^*$$

São equivalentes!

A eliminação do estado 2 gera o seguinte autómato equivalente:



Apaga-se a transição que sai de 2 e redirecciona-se para 1 a que entra em 2.

Estados Distinguíveis

Propriedades

Considere um autómato $D = \langle S, \Sigma, \delta, s, F \rangle$ e dois estados u e v de D. Estes estados são distinguíveis, se:

- $u \in F$ e $v \notin F$ — a palavra vazia basta para ver que não são equivalentes
- u é produtivo mas v não — existe uma palavra que leva de u a um estado final mas não leva de v a um estado final
- existe um $a \in \Sigma$ tal que $\delta(u, a) = r$ e r é produtivo, mas $\delta(v, a)$ não está definido — u é produtivo mas v não
- existe um $a \in \Sigma$ tal que $\delta(u, a) = r$ e $\delta(v, a) = t$, sendo r e t distinguíveis — se existe $w \in \Sigma^*$ tal que $\delta^*(r, w) \in F$ mas $\delta^*(t, w) \notin F$ (ou vice-versa) então $\delta^*(u, aw) \in F$ mas $\delta^*(v, aw) \notin F$ (ou vice-versa)

Procura de Estados Distinguíveis

- O algoritmo anterior (de eliminação de estados equivalentes) usa-se quando já se identificaram estados equivalentes.
- Mas dado um autómato $D = \langle S, \Sigma, \delta, s, F \rangle$, como identificar que estados são equivalentes?
- Como a definição de estados equivalentes usa uma quantificação universal, é mais fácil identificar estados distinguíveis (basta encontrar uma palavra que os distinga).
- Procede-se então em duas fases:
  - Primeiro encontra-se o conjunto $Dist \subseteq S \times S$ dos pares de estados distinguíveis;
  - Para cada $(u, v) \in (S \times S) \setminus Dist$, elimina-se v.

Este algoritmo produz um autómato equivalente ao dado, que é mínimo (os conjuntos de estados e de transições são mínimos).

Algoritmo de Procura de Estados Distinguíveis

Dado um autómato $D = \langle S, \Sigma, \delta, s, F \rangle$, pretende-se encontrar o conjunto $Dist \subseteq S \times S$ dos pares de estados distinguíveis.

Constroi-se $Dist \subseteq S \times S$ iterativamente

- Primeiro colocam-se em $Dist$ os pares que têm um estado final e outro não final: $Dist \stackrel{\text{def}}{=} \{(u, v) \in S \times S \mid u \in F \wedge v \notin F\}$
- Colocam-se depois os pares que têm um estado com uma transição por dada acção e outro que não tem transição por essa acção:
  $Dist \stackrel{\text{def}}{=} Dist \cup \{(u, v) \in S \times S \mid \exists a.\delta(u, a) \in S \wedge \delta(v, a) = \bot\}$

Algoritmo de Procura de Estados Distinguíveis, *APED*

Passo iterativo

Para cada par $(u, v) \in \mathit{Dist}$, colocam-se os pares que levam a (u, v) por uma transição por dada acção a: seja $\delta(r, a) = u$ e $\delta(t, a) = v$.

$$\mathit{Dist} \cup \{(r, t) \in S \times S \mid (r, t) \notin \mathit{Dist} \wedge \exists a.(\delta(r, a), \delta(t, a)) \in \mathit{Dist}\}$$

O conjunto $\mathit{Dist} \subseteq S \times S$ dos pares de estados distinguíveis é calculado pelo algoritmo *APED*:

$$\begin{aligned} I \stackrel{\text{def}}{=} & \quad \{(u, v) \in S \times S \mid u \in F \wedge v \notin F\} \\ \cup & \quad \{(u, v) \in S \times S \mid \exists a.\delta(u, a) \in S \wedge \delta(v, a) = \perp\} \end{aligned}$$

$$A \stackrel{\text{def}}{=} \{(u, v) \in S \times S \mid (u, v) \notin I \wedge \exists a.(\delta(u, a), \delta(v, a)) \in I\}$$

$$\mathit{Dist} \stackrel{\text{def}}{=} \mathit{APED}(D, I \cup A)$$

Algoritmo de Procura de Estados Distinguíveis, *APED*

Define-se então o algoritmo *APED* de cálculo de *Dist* como uma função recursiva.

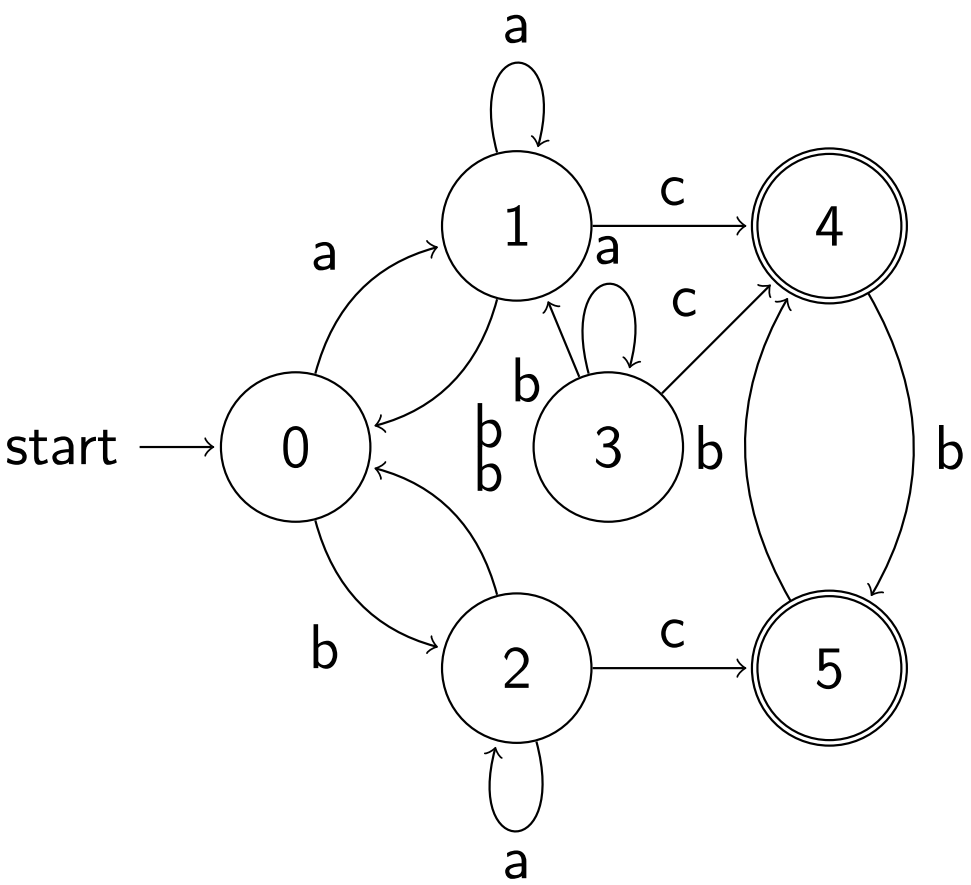
$$APED \subseteq \mathcal{DFA} \times \wp(S \times S) \rightarrow \wp(S \times S)$$

$$\begin{aligned} APED(D, P \cup Q) &= P \cup Q \text{ , se } P \cup Q = S \times S \ \vee \ Q = \emptyset \\ APED(D, P \cup Q) &= APED(D, P \cup Q \cup R) \text{ , caso contrário} \end{aligned}$$

sendo

$$R \stackrel{\text{def}}{=} \{(u, v) \in S \times S \mid (u, v) \notin P \ \wedge \ \exists a.(\delta(u, a), \delta(v, a)) \in Q\}$$

Exemplo: minimização do autómato D pelo APED



1. Retira-se primeiro o estado 3, que é inútil.
2. Define-se depois o conjunto dos pares de estados final / não final.
3. Termina-se a inicialização com o conjunto dos pares de estados em que um faz dada acção mas o outro não.

Exemplo: estados distinguíveis de D

Inicialização

1. Define-se o conjunto dos pares de estados final / não final:
   $I_1 \stackrel{\text{def}}{=} \{(0,4),(1,4),(2,4),(0,5),(1,5),(2,5)\}$
2. Termina-se a inicialização com o conjunto dos pares de estados em que um faz dada acção mas o outro não:
   $I_2 \stackrel{\text{def}}{=} \{(0,1),(0,2)\}$

Tem-se então $\mathit{Dist} = I_1 \cup I_2$.

Passo

Juntam-se novos pares que levem pela mesma acção a pares já em Dist — não há!

Logo, $\mathit{Dist} = \{(0,1),(0,2),(0,4),(0,5),(1,4),(1,5),(2,4),(2,5)\}$.

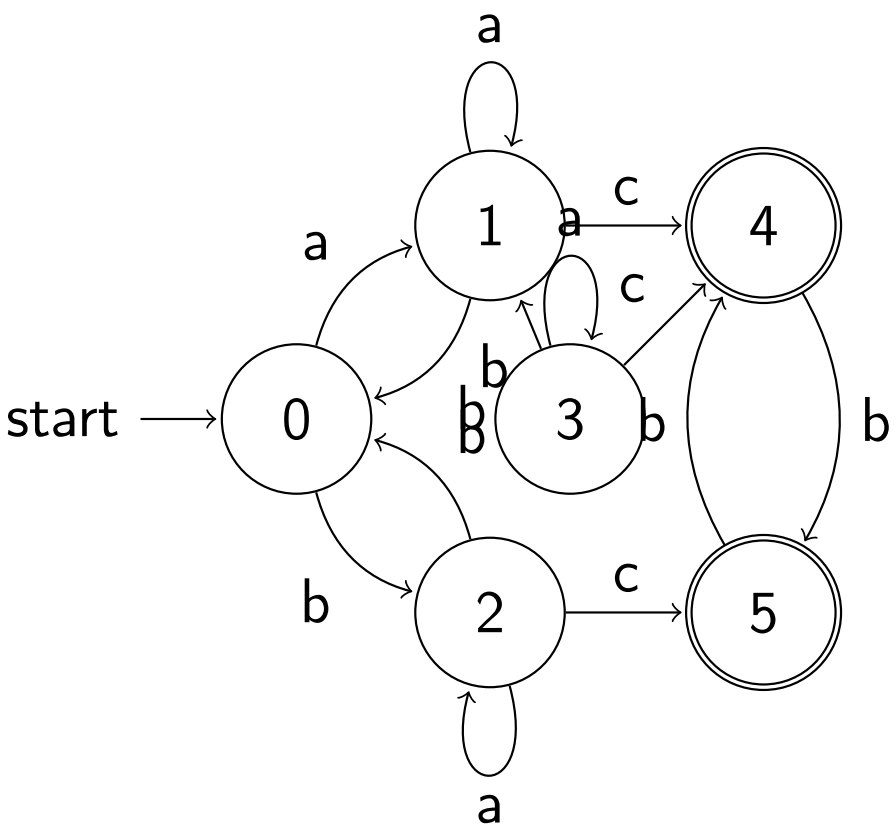
O exemplo, numa tabela

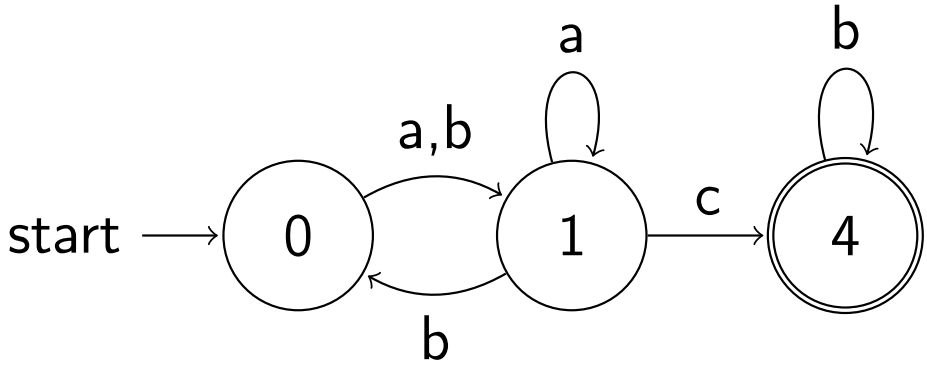
1. Desenha-se uma matriz em que as linhas e as colunas têm os nomes dos estados a determinar se são distinguíveis (note-se que qualquer estado é equivalente a sí mesmo).
2. Em cada passo do algoritmo assinalam-se na tabela os pares que se determinam distinguíveis.

	0	1	2	4
1	x	·	·	·
2	x		·	·
4	x	x	x	·
5	x	x	x	

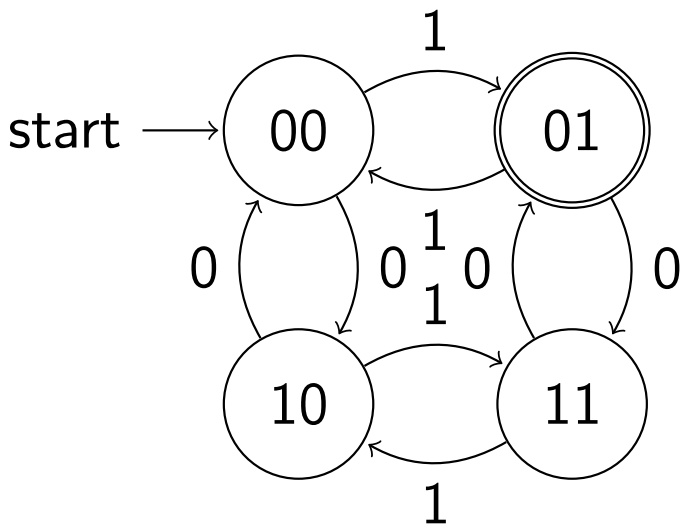
Conclui-se que os estados 1 e 2 são equivalentes e que 4 e 5 também.

O exemplo, antes e após a minimização





Outro exemplo de aplicação de APED



1. Note-se que o autómato não tem estados inúteis.
2. Determina-se pelo APED se tem estados equivalentes.

Execução do APED.

1. Inicialização:

   $I_1 = \{(01, 00), (01, 10), (01, 11)\}$

   $I_2 = \emptyset$ porque δ é uma função total.

   Faz-se então $Dist \stackrel{\text{def}}{=} I_1$.
2. Determina-se pelo APED se tem estados equivalentes.

Outro exemplo de aplicação de APED: passo do algoritmo

Da inicialização do APED obteve-se

	00	10	11
10		·	·
11			·
01	x	x	x

1. Toma-se $(00, 01) \in \mathit{Dist}$
   - 1.1 existe (x, y) tal que $\delta(x, 0) = 00$ e $\delta(y, 0) = 01$?
     Sim: $(x, y) = (10, 11)$. Logo, acrescenta-se $(10, 11)$ a *Dist*.
   - 1.2 existe (x, y) tal que $\delta(x, 1) = 00$ e $\delta(y, 1) = 01$?
     Sim: $(x, y) = (01, 00)$. Como $(01, 00) = (00, 01)$ e $(00, 01) \in \mathit{Dist}$, não se altera *Dist*.
2. Toma-se $(10, 01) \in \mathit{Dist}$
   - 2.1 existe (x, y) tal que $\delta(x, 0) = 10$ e $\delta(y, 0) = 01$?
     Sim: $(x, y) = (00, 11)$. Logo, acrescenta-se $(00, 11)$ a *Dist*.
   - 2.2 existe (x, y) tal que $\delta(x, 1) = 10$ e $\delta(y, 1) = 01$?
     Sim: $(x, y) = (11, 00)$. Como $(11, 00) = (00, 11)$ e $(00, 11) \in \mathit{Dist}$, não se altera *Dist*.

Exemplo de aplicação de APED: ainda o passo do algoritmo

Dos passos anteriores do APED obteve-se

	00	10	11
10		·	·
11	x	x	·
01	x	x	x

Toma-se $(11, 01) \in \mathit{Dist}$

1. existe (x, y) tal que $\delta(x, 0) = 11$ e $\delta(y, 0) = 01$?
   Sim: $(x, y) = (01, 11)$. Como $(01, 11) = (11, 01)$ e $(11, 01) \in \mathit{Dist}$, não se altera Dist.
2. existe (x, y) tal que $\delta(x, 1) = 11$ e $\delta(y, 1) = 01$?
   Sim: $(x, y) = (10, 00)$. Logo, acrescenta-se $(10, 00)$ a Dist.

Como $\mathit{Dist} = S^2$, termina-se a execução do APED, concluindo que todos os estados são distinguíveis – o DFA já é mínimo.