Autômatos Finitos e Linguagens Regulares

Prof. Hamilton José Brumatto

CIC-UESC

21 de dezembro de 2024

Retomando o tópico anterior

- Um automato finito é definido por uma 5-upla: $\{Q, \Sigma, \delta, q_0, F\}$, onde:
  - Q é o conjunto de estados do autômato.
  - Σ é o alfabeto que define os elementos de uma cadeia computável
  - δ é a função de relação $\delta : Q \times \Sigma \rightarrow Q$ que define a computação passo a passo.
  - q_0 é o estado inicial do autômato.
  - F é o conjunto de estados de aceitação.

Definição Formal de Computação

Nós dissemos informalmente que um autômato aceita uma cadeia se a partir do estado inicial a cadeia for computada terminando em um estado final. Na matemática esta informação é muito imprecisa. Precisamos definir a computação formalmente:

Definição Formal de Computação

Seja $M = \{Q, \Sigma, \delta, q_0, F\}$ um autômato finito e suponha que $w = w_1, w_2, \ldots, w_n$ seja uma cadeia de tamanho n onde cada w_i é um membro do alfabeto Σ. Então M **aceita** w se existe uma sequência de estados $r_0, r_1, \ldots, r_n$ em Q com três condições:

1. $r_0 = q_0$,
2. $\delta(r_i, w_{i+1}) = r_{i+1}$, para $i = 0, \ldots, n-1$, e
3. $r_n \in F$.

Entendendo as condições

$r_0 = q_0$

A primeira condição diz que o autômato inicia em seu estado inicial.

$\delta(r_i, w_{i+1}) = r_{i+1}$, para $i = 0, \ldots, n-1$

A segunda condição diz que a máquina vai de estado para estado conforme a função de transição.

$r_n \in F$

A terceira condição diz que a máquina aceita a cadeia de entrada se ela termina em um estado de aceitação.

Linguagem Regular

Dizemos que M **reconhece a linguagem** A se:
$A = \{w | M \text{ aceita } w\}$

Uma linguagem é chamada de uma LINGUAGEM REGULAR se algum autômato finito a reconhece.

Então para demonstrar que uma linguagem é regular, precisamos construir um autômato finito que reconheça a linguagem.

Por onde começar?

A única informação que temos é o alfabeto e a linguagem (para definir uma linguagem é necessário especificar um alfabeto). Como projetar um autômato a partir daí?
Se fosse uma única cadeia seria simples, mas uma linguagem é o conjunto de todas as cadeias possíveis, mesmo que a cadeia seja finita, ela pode ser tão grande quando imaginarmos.
Veja um exemplo?

Seja $\Sigma = \{a, u, v\}$, considere a linguagem: $L = \{w | w$ termina com a cadeia $uva\}$.

Como projetar um autômato para esta linguagem? Lembre-se que poderíamos ter 500 entradas antes da cadeia final.

Definindo os estados

- A primeira tarefa a fazer é: Definir quais estados o autômato possui!
- Os estados representam a memória do autômato, portanto, o que será necessário lembrar?
- Para que reconheçamos que a cadeia *uva* tenha entrado, precisamos lembrar das três entradas na sequência:
  1. u, vamos marcar isto no estado. q_u.
  2. v, seguido de u, vamos marcar isto no estado q_{uv}, observe que a única transição possível para o estado q_{uv} seria a entrada $\delta(q_u, v)$.
  3. O estado final de aceitação, quando teríamos a entrada a após as entradas v e u, ou seja, do estado q_{uv}, somente a entrada a o levaria ao estado q_{uva}.
  4. Precisamos começar de algo: vamos colocar um estado inicial s.

Definindo os estados

- A primeira tarefa a fazer é: Definir quais estados o autômato possui!
- Os estados representam a memória do autômato, portanto, o que será necessário lembrar?
- Para que reconheçamos que a cadeia *uva* tenha entrado, precisamos lembrar das três entradas na sequência:
  1. u, vamos marcar isto no estado. q_u.
  2. v, seguido de u, vamos marcar isto no estado q_{uv}, observe que a única transição possível para o estado q_{uv} seria a entrada $\delta(q_u, v)$.
  3. O estado final de aceitação, quando teríamos a entrada a após as entradas v e u, ou seja, do estado q_{uv}, somente a entrada a o levaria ao estado q_{uva}.
  4. Precisamos começar de algo: vamos colocar um estado inicial s.

Definindo os estados

- A primeira tarefa a fazer é: Definir quais estados o autômato possui!
- Os estados representam a memória do autômato, portanto, o que será necessário lembrar?
- Para que reconheçamos que a cadeia *uva* tenha entrado, precisamos lembrar das três entradas na sequência:
  1. u, vamos marcar isto no estado. q_u.
  2. v, seguido de u, vamos marcar isto no estado q_{uv}, observe que a única transição possível para o estado q_{uv} seria a entrada $\delta(q_u, v)$.
  3. O estado final de aceitação, quando teríamos a entrada a após as entradas v e u, ou seja, do estado q_{uv}, somente a entrada a o levaria ao estado q_{uva}.
  4. Precisamos começar de algo: vamos colocar um estado inicial s.

Definindo os estados

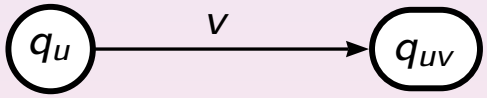
- A primeira tarefa a fazer é: Definir quais estados o autômato possui!
- Os estados representam a memória do autômato, portanto, o que será necessário lembrar?
- Para que reconheçamos que a cadeia *uva* tenha entrado, precisamos lembrar das três entradas na sequência:
  1. u, vamos marcar isto no estado. q_u.
  2. v, seguido de u, vamos marcar isto no estado q_{uv}, observe que a única transição possível para o estado q_{uv} seria a entrada $\delta(q_u, v)$.
  3. O estado final de aceitação, quando teríamos a entrada a após as entradas v e u, ou seja, do estado q_{uv}, somente a entrada a o levaria ao estado q_{uva}.
  4. Precisamos começar de algo: vamos colocar um estado inicial s.

Autômato UVA

Primeiro o estado q_u:

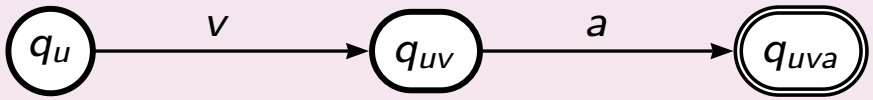
Autômato UVA

Primeiro o estado q_u:
O estado q_{uv} sendo que somente de q_u com v se chega a q_{uv}



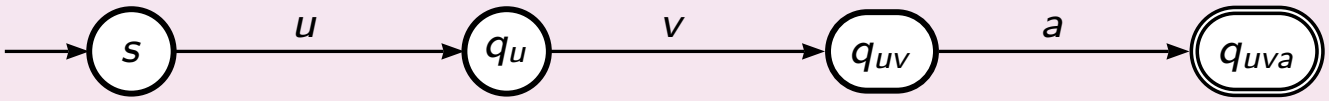
Autômato UVA

Primeiro o estado q_u:
O estado q_{uv} sendo que somente de q_u com v se chega a q_{uv}
O estado q_{uva}, somente de q_{uv} com a se chega a q_{uva}



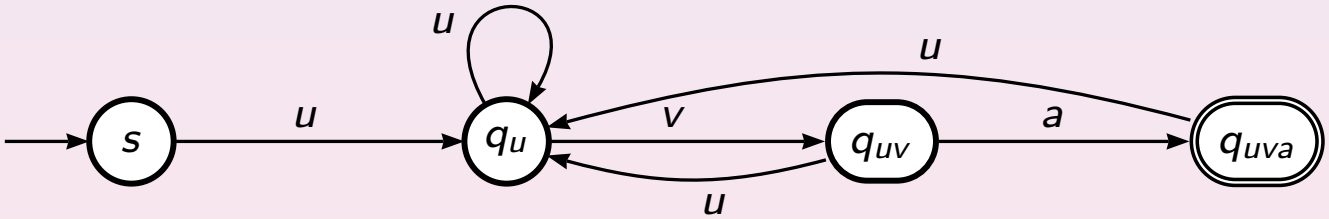
Autômato UVA

Primeiro o estado q_u:
O estado q_{uv} sendo que somente de q_u com v se chega a q_{uv}
O estado q_{uva}, somente de q_{uv} com a se chega a q_{uva}
O estado s inicial que permite chegar a q_u a partir do u



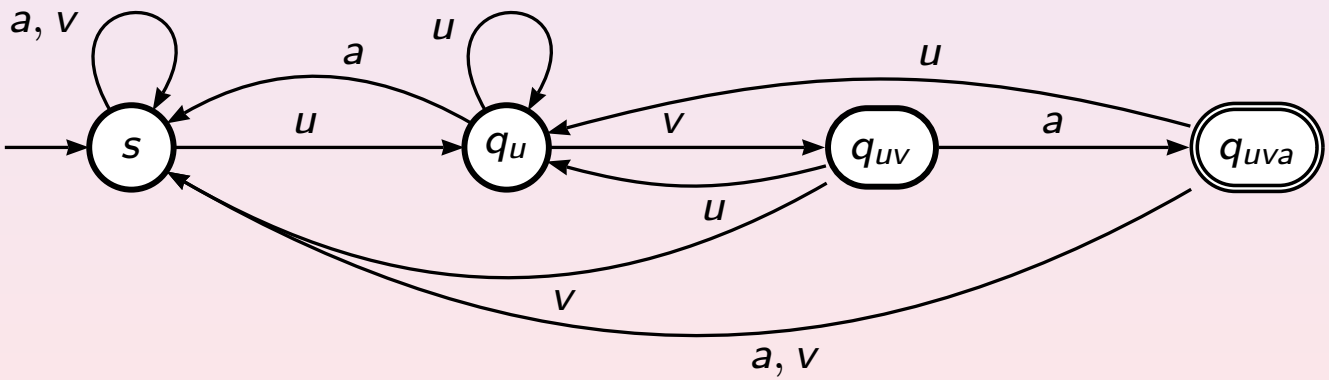
Autômato UVA

Primeiro o estado q_u:
O estado q_{uv} sendo que somente de q_u com v se chega a q_{uv}
O estado q_{uva}, somente de q_{uv} com a se chega a q_{uva}
O estado s inicial que permite chegar a q_u a partir do u
De qualquer estado se chega a q_u com um u



Autômato UVA

Primeiro o estado q_u:
O estado q_{uv} sendo que somente de q_u com v se chega a q_{uv}
O estado q_{uva}, somente de q_{uv} com a se chega a q_{uva}
O estado s inicial que permite chegar a q_u a partir do u
De qualquer estado se chega a q_u com um u
Qualquer outra coisa vai para s



Autômato 001

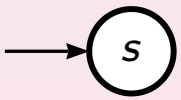
Vamos considerar a seguinte linguagem dada pelo alfabeto $\Sigma = \{0, 1\}$
$L = \{w | w \text{ contém a subcadeia } 001\}$.

Este é muito semelhante ao anterior, exceto que o 0 se repete e esta não é uma subcadeia final.

Quais são os estados que precisamos nos lembrar?

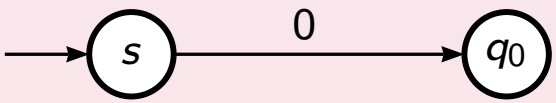
Autômato 001

1. Temos um estado inicial s
2. Um 0 foi inserido após um 1 ou o estado inicial $\rightarrow q_0$.
3. Um 0 foi inserido após outro 0 $\rightarrow q_{00}$
4. Um 1 foi inserido após o estado $q_{00} \rightarrow q_{001}$, que é o estado de aceitação.
5. Um 1 foi inserido após qualquer outro estado $\rightarrow s$,
6. Qualquer valor foi inserido após o estado de aceitação $\rightarrow q_{001}$.



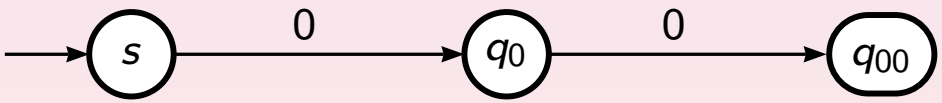
Autômato 001

1. Temos um estado inicial s
2. Um 0 foi inserido após um 1 ou o estado inicial $\rightarrow q_0$.
3. Um 0 foi inserido após outro 0 $\rightarrow q_{00}$
4. Um 1 foi inserido após o estado $q_{00} \rightarrow q_{001}$, que é o estado de aceitação.
5. Um 1 foi inserido após qualquer outro estado $\rightarrow s$,
6. Qualquer valor foi inserido após o estado de aceitação $\rightarrow q_{001}$.



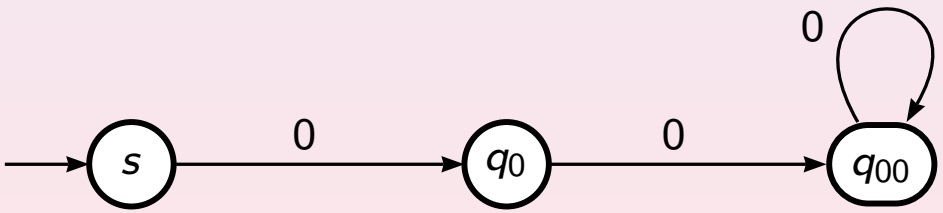
Autômato 001

1. Temos um estado inicial s
2. Um 0 foi inserido após um 1 ou o estado inicial $\rightarrow q_0$.
3. Um 0 foi inserido após outro 0 $\rightarrow q_{00}$
4. Um 1 foi inserido após o estado $q_{00} \rightarrow q_{001}$, que é o estado de aceitação.
5. Um 1 foi inserido após qualquer outro estado $\rightarrow s$,
6. Qualquer valor foi inserido após o estado de aceitação $\rightarrow q_{001}$.



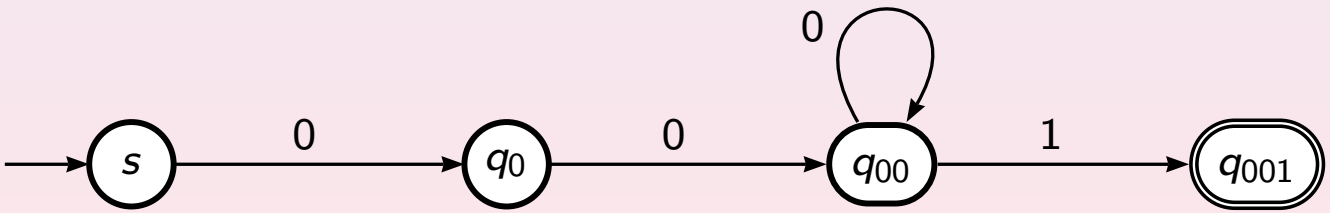
Autômato 001

1. Temos um estado inicial s
2. Um 0 foi inserido após um 1 ou o estado inicial $\rightarrow q_0$.
3. Um 0 foi inserido após outro 0 $\rightarrow q_{00}$
4. Um 1 foi inserido após o estado $q_{00} \rightarrow q_{001}$, que é o estado de aceitação.
5. Um 1 foi inserido após qualquer outro estado $\rightarrow s$,
6. Qualquer valor foi inserido após o estado de aceitação $\rightarrow q_{001}$.



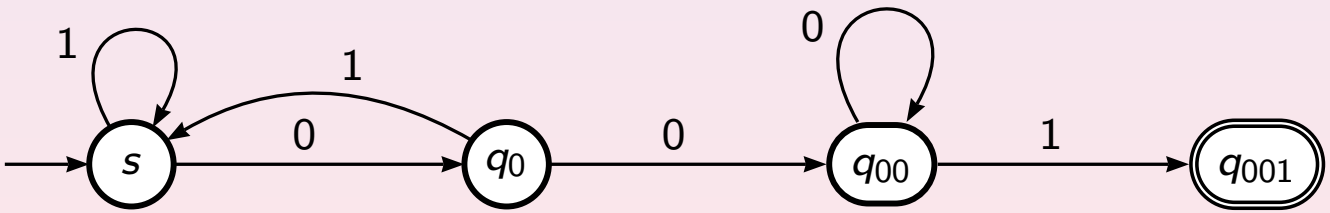
Autômato 001

1. Temos um estado inicial s
2. Um 0 foi inserido após um 1 ou o estado inicial $\rightarrow q_0$.
3. Um 0 foi inserido após outro 0 $\rightarrow q_{00}$
4. Um 1 foi inserido após o estado $q_{00} \rightarrow q_{001}$, que é o estado de aceitação.
5. Um 1 foi inserido após qualquer outro estado $\rightarrow s$,
6. Qualquer valor foi inserido após o estado de aceitação $\rightarrow q_{001}$.



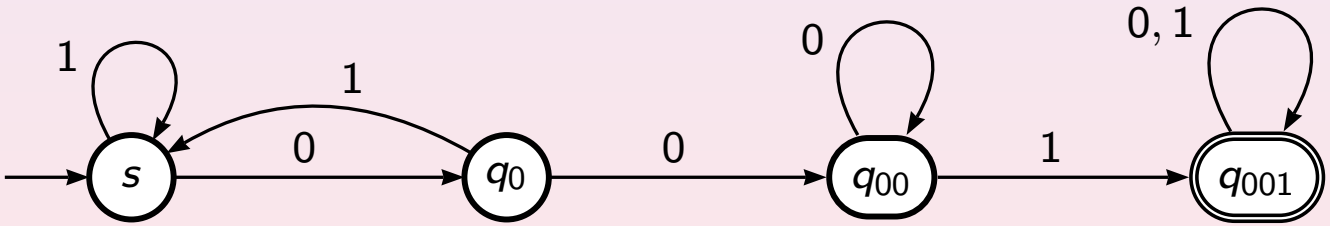
Autômato 001

1. Temos um estado inicial s
2. Um 0 foi inserido após um 1 ou o estado inicial $\rightarrow q_0$.
3. Um 0 foi inserido após outro 0 $\rightarrow q_{00}$
4. Um 1 foi inserido após o estado $q_{00} \rightarrow q_{001}$, que é o estado de aceitação.
5. Um 1 foi inserido após qualquer outro estado $\rightarrow s$,
6. Qualquer valor foi inserido após o estado de aceitação $\rightarrow q_{001}$.



Autômato 001

1. Temos um estado inicial s
2. Um 0 foi inserido após um 1 ou o estado inicial $\rightarrow q_0$.
3. Um 0 foi inserido após outro 0 $\rightarrow q_{00}$
4. Um 1 foi inserido após o estado $q_{00} \rightarrow q_{001}$, que é o estado de aceitação.
5. Um 1 foi inserido após qualquer outro estado $\rightarrow s$,
6. Qualquer valor foi inserido após o estado de aceitação $\rightarrow q_{001}$.



Autômato *Ímpar*

Vamos considerar a seguinte linguagem dada pelo alfabeto $\Sigma = \{0, 1\}$
$L = \{w | w \text{ possui um número ímpar de } 1\}$.

Quais são os estados que precisamos nos lembrar?

Autômato *Ímpar*

Vamos considerar a seguinte linguagem dada pelo alfabeto $\Sigma = \{0, 1\}$
$L = \{w | w \text{ possui um número ímpar de } 1\}$.

Quais são os estados que precisamos nos lembrar?

Somente é possível acontecer duas coisas: termos um número *Ímpar* de 1s ou um número *Par* de 1s.

Autômato *Ímpar*

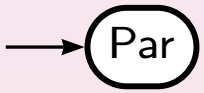
Vamos considerar a seguinte linguagem dada pelo alfabeto $\Sigma = \{0, 1\}$
$L = \{w | w \text{ possui um número ímpar de } 1\}$.

Quais são os estados que precisamos nos lembrar?

Somente é possível acontecer duas coisas: termos um número *Ímpar* de 1s ou um número *Par* de 1s. Em especial, o estado de entrada possui nenhum 1 e isto é um estado que possui um número par de 1s.

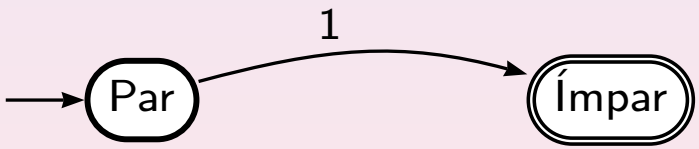
Autômato *Ímpar*

1. Temos um estado inicial *Par*
2. Um 1 foi inserido neste estado, ele vai para o estado *Ímpar*, que é o estado de aceitação.
3. Um 1 foi inserido no estado *Ímpar* ele vai para o estado *Par*.
4. Um 0 não provoca mudança de estado.



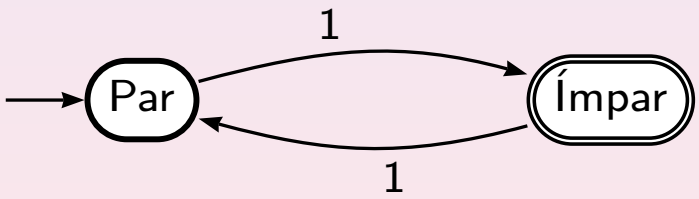
Autômato *Ímpar*

1. Temos um estado inicial *Par*
2. Um 1 foi inserido neste estado, ele vai para o estado *Ímpar*, que é o estado de aceitação.
3. Um 1 foi inserido no estado *Ímpar* ele vai para o estado *Par*.
4. Um 0 não provoca mudança de estado.



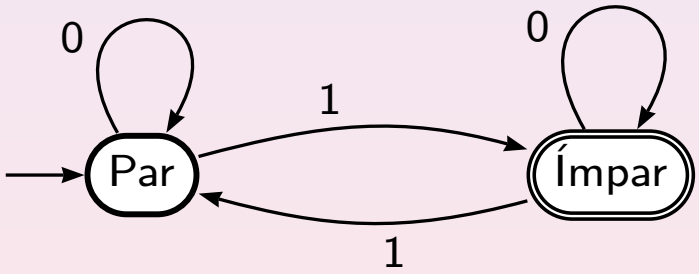
Autômato *Ímpar*

1. Temos um estado inicial *Par*
2. Um 1 foi inserido neste estado, ele vai para o estado *Ímpar*, que é o estado de aceitação.
3. Um 1 foi inserido no estado *Ímpar* ele vai para o estado *Par*.
4. Um 0 não provoca mudança de estado.



Autômato *Ímpar*

1. Temos um estado inicial *Par*
2. Um 1 foi inserido neste estado, ele vai para o estado *Ímpar*, que é o estado de aceitação.
3. Um 1 foi inserido no estado *Ímpar* ele vai para o estado *Par*.
4. Um 0 não provoca mudança de estado.



Exercício

Seja o alfabeto $\Sigma = \{⇳, \square\}$ e a linguagem: $L = \{w | w = \varepsilon \vee w$ possui uma quantidade par de ⇳ e termina com $\square\}$. Mostre que L é regular.

Linguagens e Operações Regulares

No Projeto de autômatos, é possível manipular as linguagens através de operações. Sejam A e B linguagens, tais que: $A = \{w | w \in A\}$ e $B = \{w | w \in B\}$. A seguinte operações são definidas:

- **União**: $A \cup B = \{w | w \in A \vee w \in B\}$
- **Concatenação**: $A \circ B = \{wz | w \in A \wedge z \in B\}$
- **Estrela**: $A^* = \{w_1 w_2 \ldots w_k | k \geqslant 0, \forall w_i \in A\}$

Na operação unária *Estrela* sendo w_k uma coleção qualquer, ela pode ser vazia, então ε é sempre membro de A^*

As operações binárias: *União* e *Concatenação* precisam que as linguagens regulares a que se aplicam sejam baseadas no mesmo alfabeto.

Exemplos de Operações Regulares

Seja o alfabeto Σ padrão de 26 símbolos: $\{a, b, c, d, \ldots, z\}$ e A e B as linguagens: $A = \{luar, sol\}$, $B = \{brilhante, distante$
As operações são:

- $A \cup B$: $\{luar, brilhante, distante, sol\}$
- $A \circ B$: $\{luarbrilhante, luardistante, solbrilhante, soldistante\}$
- $A^\star$: $\{\varepsilon, luar, sol, luarsol, solluar, luarluar, solsol, luarluarsol, luarsolluar, \ldots\}$

Fecho de Operações

Definição

Uma operação é *fechada* sobre um conjunto quando a operação aplicada sobre elementos do conjunto sempre resulta em um elemento do próprio conjunto.

Por exemplo:

- A operação de multiplicação "$\times$" é fechada sobre o conjunto dos inteiros $\mathcal{Z}$.
- A operação de divisão "$\div$" **não** é fechada sobre o conjunto dos inteiros $\mathcal{Z}$.

A operação de União é regular

Teorema

A classe de linguagens regulares é fechada sobre a operação de união. Ou, se A_1 e A_2 são linguagens regulares, o mesmo acontece com $A_1 \cup A_2$.

Para a demonstração usaremos:

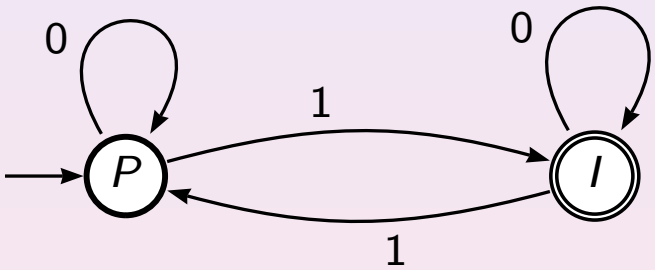
- Temos M_1 e M_2 autômatos que reconhecem A_1 e A_2. Construímos M que reconhece $A_1 \cup A_2$ a partir de M_1 e M_2 simulando ambos.
- Simulamos os estados que A_1 e A_2 estariam sob uma determinada cadeia de entrada. Cada estado de M representa uma par de estados.
- Se um dos estados simulados é de aceitação, então em M também o é.
- Se M_1 tem k_1 estados, e M_2 tem k_2 estados, então M tem $k_1 \times k_2$ estados.

Exemplo da técnica da demonstração

Linguagem A_1

Seja o alfabeto $\Sigma = \{0, 1\}$ e a linguagem $A_1 = \{w | w$ tem um número ímpar de $1\}$.

Já construímos este autômato e M_1 é dado pela seguinte máquina de estados:



$M_1 = \{Q_1, \Sigma, \delta_1, q_{0_1}, F_1\}$

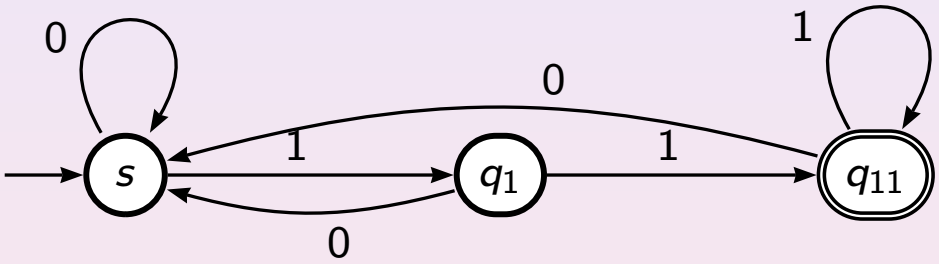
$M_1 = \{\{P, I\}, \{0, 1\}, \delta_1, P, \{I\}\}$.

δ_1	P	I
0	P	I
1	I	P

Exemplo da técnica da demonstração

Linguagem A_2

Seja o alfabeto $\Sigma = \{0, 1\}$ e a linguagem $A_2 = \{w | w$ termina com a sequencia $11\}$.



$M_2 = \{Q_2, \Sigma, \delta_2, q_{0_2}, F_2\}$

$M_2 = \{\{s, q_1, q_{11}\}, \{0, 1\}, \delta_2, s, \{q_{11}\}\}$.

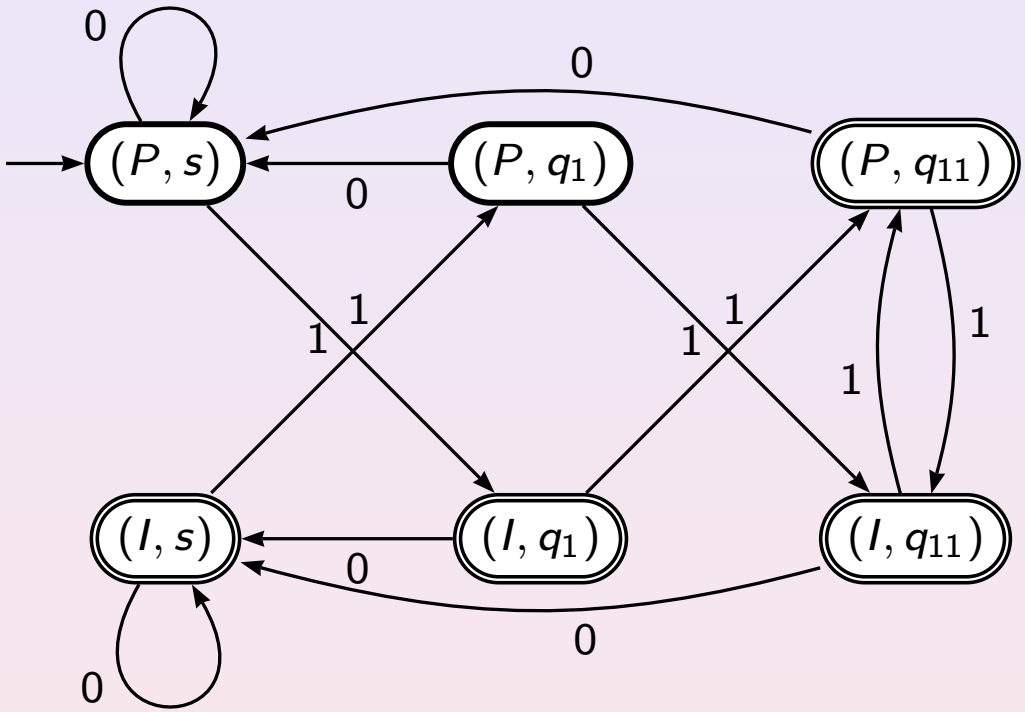
δ_2	s	q_1	q_{11}
0	s	s	s
1	q_1	q_{11}	q_{11}

Construindo a máquina união

- O conjunto de estados será o produto cartesiano entre ambos conjuntos Q_1 e Q_2: $Q = Q_1 \times Q_2$.
  $Q = \{(P, s), (P, q_1), (P, q_{11}), (I, s), (I, q_1), (I, q_{11})\}$
- O estado inicial é o estado que representa ambos estados iniciais: (P, s).
- O conjunto de estados de aceitação é o conjunto que possui pelo menos um dos estados de aceitação:
  $F = \{(P, q_{11}), (I, s), (I, q_1), (I, q_{11})\}$.
- A função de transição deve considerar como seria a transição considerando os estados como se fossem independentes:

δ	(P, s)	(P, q_1)	(P, q_{11})	(I, s)	(I, q_1)	(I, q_{11})
0	(P, s)	(P, s)	(P, s)	(I, s)	(I, s)	(I, s)
1	(I, q_1)	(I, q_{11})	(I, q_{11})	(P, q_1)	(P, q_{11})	(P, q_{11})

A máquina de estados



Demonstração do Teorema

Construir M que reconheça $A = A_1 \cup A_2$; $M = \{Q, \Sigma, \delta, q_0, F\}$

- $Q = \{(r_1, r_2) | r_1 \in Q_1 e r_2 \in Q_2\}$
- $\Sigma \equiv \Sigma_1 \equiv \Sigma_2$
- δ é definida como: $\delta((r_1, r_2), a) = (\delta_1(r_1, a), \delta_2(r_2, a))$
- $q_0 = (q_{0_1}, q_{0_2})$
- $F = \{(r_1, r_2) | r_1 \in F_1 \vee r_2 \in F_2\}$, isto é o mesmo que: $F = (F_1 \times Q_2) \cup (Q_1 \times F_2)$.

Uma cadeia de M_1 seria aceita quando os estados r_{1_i} resultasse em um estado de aceitação. Em M, como δ define as transições para os estados relacionados com M_1 através da função δ_1 no primeiro par da 2-upla, então o estado final será um estado de aceitação equivalente ao da máquina M_1, e todos os estados equivalentes da máquina M_1 são estados de aceitação em M. O mesmo raciocínio aplica-se para uma cadeia aceita em M_2. Logo M aceita cadeias que M_1 aceita e cadeias que M_2 aceita, logo ele aceita o conjunto união das linguagens A_1 e A_2.

Fecho nas operações de Concatenação e Estrela

Poderíamos utilizar algo semelhante, no entanto além da questão de seguir por uma ou outra máquina, há também a questão de quebrar a cadeia concatenada para saber onde inicia outra cadeia.

O problema de quebrar a cadeia é complicado em autômatos finitos. Para tanto teremos de entender um novo conceito: Autômatos Finitos Não Determinísticos, que complementa o conceito dos Autômatos Finitos Determinísticos que vínhamos estudando.

Atividades baseadas no Sipser

- Ler as seções restantes do capítulo 1.1:
- resolver os exercícios: 1.4, 1.5, 1.6.