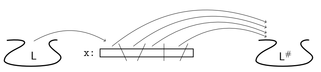
Pedaços, prefixos e etc

Agora imagine que L é a linguagem reconhecida pelo autômato M.

Imagine que nós pegamos as palavras de L e quebramos em pedaços de todas as maneiras possíveis.

E imagine que nós formamos uma nova linguagem $\mathsf{L}^{\#}$ com todos esses pedaços



A questão é

- *Será que nós conseguimos construir um autômato que reconhece a linguagem* $\mathsf{L}^{\#}$ *?*

Hmm ...

A intuição parece ser a seguinte

- $\mathsf{L}^{\#}$ foi construída com os pedaços das palavras de L
  logo, $\mathsf{M}^{\#}$ deve ser construído com os pedaços de M

Hmm ...

Mas ainda não é isso.

Quer dizer, a ideia é a seguinte: *os caminhos do autômato* $\mathsf{M}^{\#}$ *devem ser construídos com pedaços dos caminhos de* M.

O que nós temos em mente aqui é que toda palavra de L corresponde a um caminho de aceitação no autômato M

Logo, um pedaço dessa palavra corresponde a ume pedaço do caminho

O que nós queremos é que esse pedaço de caminho seja um caminho de aceitação em $\mathsf{M}^{\#}$

— (*porque o pedaço da palavra está em* $\mathsf{L}^{\#}$)

Bom, para isso basta fazer com que $\mathsf{q_i}$ seja o estado inicial e que $\mathsf{q_j}$ seja um estado final de $\mathsf{M}^{\#}$.

Mas, $\mathsf{q_i}$ e $\mathsf{q_j}$ são dois estados quaisquer no meio do caminho.

Logo, o que nós precisamos é que

→ *todos os estados de* M *sejam estados iniciais de* $\mathsf{M}^{\#}$
e todos os estados de M *sejam estados finais de* $\mathsf{M}^{\#}$

Quer dizer,

Na verdade, não são realmente todos os estados de M que viram estados iniciais e finais de $\mathsf{M}^{\#}$.

Quer dizer, são apenas os estados que aparecem em algum caminho de aceitação de M.

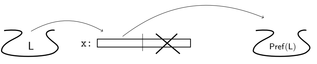
Por exemplo, esse não é o caso do estado de rejeição.

Vejamos outros exemplos.

Exemplos

a. Prefixos e sufixos

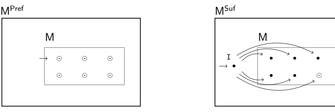
Imagine outra vez que L é a linguagem reconhecida pelo autômato M.
Imagine que nós pegamos as palavras de L e quebramos em um ponto qualquer.
E imagine que nós formamos a linguagem $\mathsf{Pref(L)}$ com a primeira parte de todas esses cortes



Quer dizer, $\mathsf{Pref}(\mathsf{L})$ é a linguagem formada pelos prefixos das palavras da linguagem L.
E nós podemos definir a linguagem dos sufixos $\mathsf{Suf}(\mathsf{L})$ de maneira análoga.
O problema consiste em

→ *Construir autômatos que reconhecem as linguagens* $\mathsf{Pref}(\mathsf{L})$ *e* $\mathsf{Suf}(\mathsf{L})$

Mas, agora isso já não é difícil.
Quer dizer, basta usar os truques que a gente acabou de ver — (*um para cada autômato*)
A coisa fica assim



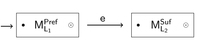
◇

b. Pref-Suf

Agora imagine que L_1, L_2 são as linguagens reconhecidas pelos autômatos M_1, M_2.
E imagine que nós queremos construir um autômato para a seguinte linguagem

$$\mathsf{Pref}(\mathsf{L}_1) \cdot \mathsf{Suf}(\mathsf{L}_2)$$

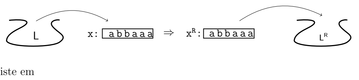
Quer dizer, essas são as palavras formadas pela concatenação do prefixo de uma palavra de L_1 com o sufixo de uma palavra de L_2.
Ora, mas agora isso é bem fácil.
Quer dizer, basta fazer o seguinte



◇

c. Reverso

Imagine mais uma vez que L é a linguagem reconhecida pelo autômato M.
E imagine que nós formamos a linguagem L^{R} virando as palavras de L ao contrário



O problema consiste em

→ *Construir um autômato que reconhece a linguagem* L^{R}

E aqui é a hora para uma pequena esperteza.
Quer dizer, a ideia é que basta virar uma expressão regular ao contrário, para obter a expressão regular do reverso da linguagem original.
Por exemplo,

$$\mathsf{L} \;=\; \mathsf{a(ab \cup bab)^*ab^*} \qquad \Rightarrow \qquad \mathsf{L^R} \;=\; \mathsf{b^*a(bab \cup ba)^*a}$$

Mas, nós também podemos fazer o mesmo truque com os autômatos.

Quer dizer, para obter o autômato M^{R}

- nós invertemos a direção das transições
- transformamos o estado inicial em estado final
- e transformamos os estados finais em estados iniciais

Legal, não é?

◇