Projeto busca de palavras

Autoria: prof. Daniel Martin (UFABC)

Regras

- Não é permitida a reutilização de nenhum código. Comece seu código do zero.
- O programa deve ser implementado **individualmente**.
- Linguagens permitidas: C ou C++.
- Entregue o código-fonte em um único arquivo (`EP1.c` ou `EP1.cpp`), via Moodle.
- Seu programa será compilado num sistema linux com um dos comandos a seguir.

  ```
  gcc -ansi EP1.c
  g++ -ansi EP1.cpp
  ```

  Portanto, você deve utilisar os padrões ANSI C (C90) ou C++ 98.

- O script de teste vai rodar seu programa, por exemplo, com uma das linhas a seguir:

  ```
  java EP1 < in001.txt > out001.txt
  a.out < in001.txt > out001.txt
  ```

  Isso é apenas um exemplo pois, como serão uma centena de testes, os nomes dos arquivos de entrada e saída irão variar. Não leia a entrada de um arquivo!!! É imprescindível que você leia a entrada da entrada-padrão e escreva na saída-padrão[1].

- Seu programa deve seguir rigorosamente as especificações de entrada e a saída, pois a nota deste projeto será atribuida por um script que faz testes automatizados no programa submetido. O script de teste será disponibilizado no Moodle.

Enunciado

Dada uma palavra $w \in \{\mathtt{a}, \mathtt{b}\}^*$, você deverá implementar um DFA[2] que reconhece a linguagem $\{v \in \{\mathtt{a}, \mathtt{b}\}^* : w \text{ é subpalavra de } v\}$.

Exemplo: para $w = abab$, o DFA na Figura 1 reconhece as palavras que têm $abab$ como subpalavra.

[1] Se você não sabe o que é entrada-padrão ou saída-padrão, não tente adivinhar: pesquise no Google!
[2] DFA é uma abreviação de um termo em inglês: *deterministic finite automaton.*

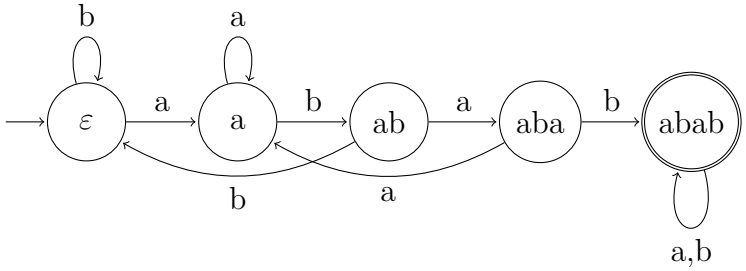

Figura 1: DFA para abab

1 Descrição geral do DFA

Dadas palavras $w, u \in \{\mathtt{a}, \mathtt{b}\}^*$, dizemos que u é um **prefixo** de w se existe $v \in \{\mathtt{a}, \mathtt{b}\}^*$ tal que $w = uv$. Para todo $w \in \{\mathtt{a}, \mathtt{b}\}^*$, seja $\mathcal{P}(w)$ o conjunto de todos os prefixos de w. Note que ε e w são sempre prefixos de w (i.e., sempre vale que $\{w, \varepsilon\} \subseteq \mathcal{P}(w)$).

Dadas palavras $w, v \in \{\mathtt{a}, \mathtt{b}\}^*$, dizemos que v é um **sufixo** de w se $w = uv$ para alguma palavra $u \in \{\mathtt{a}, \mathtt{b}\}^*$. Para todo $w \in \{\mathtt{a}, \mathtt{b}\}^*$, seja $\mathcal{S}(w)$ o conjunto de todos os sufixos de w. Note também que $\{w, \varepsilon\} \subseteq \mathcal{S}(w)$.

Para $w \in \{\mathtt{a}, \mathtt{b}\}^*$, seja $A(w) = (Q, \Sigma, \delta, s, F)$ o DFA definido por:

- $Q = \mathcal{P}(w)$,
- $\Sigma = \{\mathtt{a}, \mathtt{b}\}$,
- $s = \varepsilon$,
- $F = \{w\}$, e
- função de transição: para $v \in Q$ e $\sigma \in \Sigma$, defina $\delta(v, \sigma) = s$ tal que s é uma palavra no conjunto $\mathcal{S}(v\sigma) \cap \mathcal{P}(w)$ de **maior** comprimento possível.

2 Especificação de entrada e saída

Entrada: A entrada deve ser lida da entrada-padrão. A primeira linha da entrada é uma palavra $w \in \{\mathtt{a}, \mathtt{b}\}^*$ de comprimento pelo menos 1 e no máximo 10.

A segunda linha é um inteiro positivo $k \leq 1000$. As k linhas seguintes consistem de exatamente uma palavra $v \in \{\mathtt{a}, \mathtt{b}\}^*$ (seguida de quebra-de-linha) de comprimento pelo menos 1 e no máximo 200. Sejam $v_1, \ldots, v_k$ as k palavras dessas linhas (em ordem).

Exemplo de entrada:

```
abab
4
ab
abababab
baaabbbba
a
```

Isto é, $w = \texttt{abab}$, $k = 4$, $v_1 = \texttt{ab}$, $v_2 = \texttt{abababab}$, $v_3 = \texttt{baaabbbba}$, $v_4 = \texttt{a}$.

Código: O seu código deve implementar o DFA $A(w)$. Para cada palavra v_i, você deve processar v_i percorrendo os estados do autômato e decidir se v_i é aceita. **Atenção: Crie uma função com nome `constroi` que deve construir $A(w)$ e uma função chamada `percorre` que deve percorrer os estados do autômato dada uma palavra (naturalmente, estas funções podem fazer chamadas para outras funções).**

Saída: A saída deve ser escrita na saída-padrão. Você deve começar imprimindo $|w| + 1$ linhas seguidas de k linhas no seguinte formato.

Na i-ésima linha você deve imprimir o prefixo p de w de comprimento $i - 1$ seguido do estado $\delta(p, a)$ e do estado $\delta(p, b)$. Separe as palavras por exatamente um espaço em branco (sem espaço depois da última palavra). Use eps para denotar ε.

Para $i \in \{1, \ldots, k\}$, a $(|w| + 1 + i)$-ésima linha deve conter exatamente um número: 1 se w é subpalavra de v_i e 0 caso contrário.

Exemplo de saída:

```
eps a eps
a a ab
ab aba eps
aba a abab
abab abab abab
0
1
0
0
```