MAC–IME–USP

Carlos Eduardo Ferreira
Sala 108C Tel.: 3091 6079
e-mail cef@ime.usp.br
monitor Guilherme P. Assunção
e-mail puglia@ime.usp.br

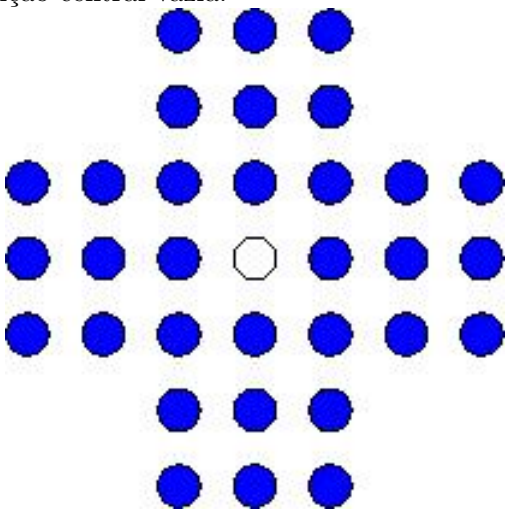
MAC 323 - Estruturas de Dados

Primeiro semestre de 2011

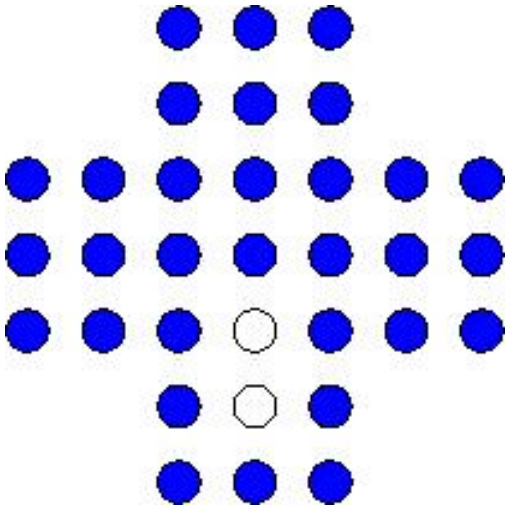
Backtracking e pilhas – **Entrega: 24/03/2011**

Este primeiro exercício-programa tem como objetivo a implementação de um algoritmo de backtracking com uso de pilhas. Assim, você deverá **necessariamente** utilizar uma pilha para armazenar as várias opções que seu programa poderá seguir.

O quebra-cabeça a ser resolvido é o **Resta um** (na verdade uma generalização dele). Você certamente já deve ter visto um desses. Trata-se de um tabuleiro em que 32 pinos são dispostos em forma de cruz, com a posição central vazia.



Em cada movimento um pino é eliminado, movendo um pino adjacente a este a uma posição vazia também adjacente. O objetivo é, após 31 movimentos, obter apenas um pino na posição central do tabuleiro (aquela que estava vazia no início). Por exemplo, no tabuleiro inicial mostrado acima temos 4 possíveis movimentos. Após realizar o movimento do pino abaixo do buraco o tabuleiro fica na seguinte posição:



Neste exercício vamos considerar uma versão mais genérica do quebra-cabeça. É dado um tabuleiro com l linhas e c colunas, em que m pinos e n buracos são espalhados (no caso do quebra-cabeça conhecido, l=c=7 e espalhamos 32 pinos e 1 buraco) e desejamos saber se após $m - n$ movimentos válidos o tabuleiro invertido pode ser obtido, ou seja, o tabuleiro em que as posições ocupadas por pinos são exatamente aquelas que no início estavam com buraco.

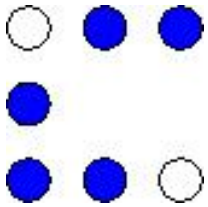
Para resolver o quebra-cabeça vamos utilizar a estratégia backtrack, em que todas as possibilidades são exploradas. Em cada instante a matriz é percorrida e todos os possíveis movimentos são analisados. Uma possibilidade é escolhida e as outras são armazenadas em uma pilha para serem exploradas mais tarde. Quando atingimos uma configuração em que nenhum movimento é possível e não atingimos a configuração desejada, voltamos à última posição na pilha, que armazena uma nova possibilidade a ser explorada em seguida.

Entrada

São dadas várias instâncias. Cada instância de entrada é dada por inteiros l e c indicando, respectivamente, o número de linhas e colunas do tabuleiro. Uma matriz com 0 linhas e colunas indica o fim dos dados. Em seguida, nas próximas $l \times c$ linhas vêm triplas de inteiros com os dados da matriz de entrada. Cada tripla traz o número da linha, o número da coluna e o que tem na posição: 1 indica um pino, -1 indica um buraco e 0 indica nada.

Exemplo: A matriz abaixo corresponde ao tabuleiro da figura a seguir:

```
3 3
0 0 -1
0 1 1
0 2 1
1 0 1
1 1 0
1 2 0
2 0 1
2 1 1
2 2 -1
```



Saída

Seu programa deverá, para cada instância de entrada, dizer se o quebra-cabeça tem ou não solução. Se existir solução seu programa deverá imprimir $m - n$ linhas, onde cada linha é composta por dois inteiros (x, y) que correspondem à posição do pino no tabuleiro (sendo que o canto superior esquerdo corresponde à posição $(0, 0)$ e um caracter que indica a direção do movimento (N [norte], S [sul], L [leste], O [oeste]).

Caso não exista solução a saída deverá ser uma única linha contendo a string `NAO EXISTE SOLUCAO`.