

Sixteen Stone

Relatório Intercalar



Mestrado Integrado em Engenharia Informática e Computação

Programação em Lógica

Grupo Sixteen_Stone_3:
Diogo Filipe Costa - ei11014
Maria Teresa Chaves - up201306842

Faculdade de Engenharia da Universidade do Porto
Rua Roberto Frias, sn, 4200-465 Porto, Portugal

11 de Outubro de 2015

Conteúdo

1 O Jogo Sixteen Stone	3
1.1 Tabuleiro de jogo	3
1.2 Regras	4
1.2.1 Jogadas possíveis	4
1.2.2 <i>Push</i>	5
1.2.3 <i>Move</i>	7
1.2.4 <i>Sacrifice</i>	8
2 Representação do Estado do Jogo	9
3 Visualização do Tabuleiro	11
4 Movimentos	11

1 O Jogo Sixteen Stone

Sixteen Stone é o nome de um jogo de tabuleiro abstrato¹² de forma quadrada, jogado normalmente num tabuleiro de 5x5, para dois jogadores. Inicialmente, cada jogador tem 8 pedras vermelhas ou azuis, colocando-as de forma alternada em qualquer uma das células livres, sendo que o jogador vermelho é o primeiro a jogar. Assim que todas as pedras estejam no tabuleiro, o jogo começa. O jogo termina quando o jogador vencedor consegue reduzir para um o número de pedras em jogo do adversário.

1.1 Tabuleiro de jogo

Esta secção tem como objetivo introduzir o tabuleiro de jogo e os elementos que dele fazem parte.

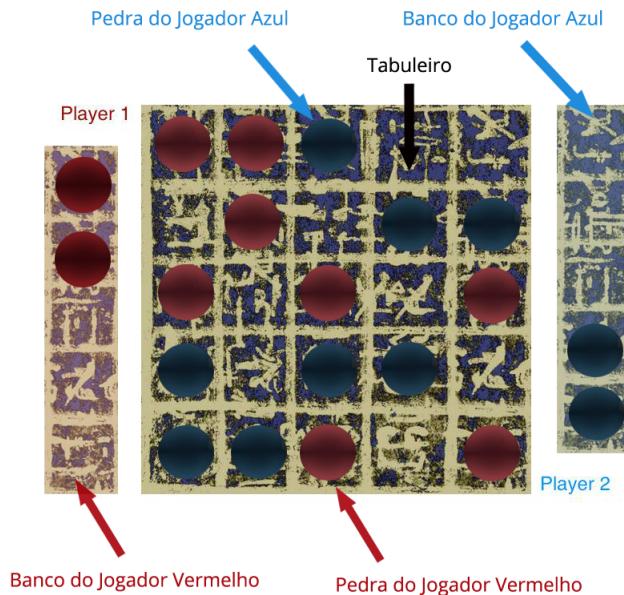


Figura 1: Explicação do tabuleiro.

Elementos do tabuleiro:

- **Pedras** - cada jogador possui um certo número de pedras, sendo que o jogador 1 possui as pedras vermelhas e o 2 as pedras azuis.
- **Banco** - cada jogador possui um banco onde estão guardadas as pedras que não estão em jogo.
- **Tabuleiro** - onde os jogadores colocam as suas pedras de jogo e podem efetuar vários movimentos.

¹Jogo abstrato - jogo de estratégia que tenta minimizar a sorte e não possui tema.

²Informação obtida em https://pt.wikipedia.org/wiki/Jogo_de_estrat%C3%A9gia_abstrato

1.2 Regras

No início do jogo distribui-se 8 pedras (vermelhas ou azuis) para cada jogador. De forma alternada os jogadores devem colocá-las em qualquer uma das células livres, começando as pedras vermelhas.

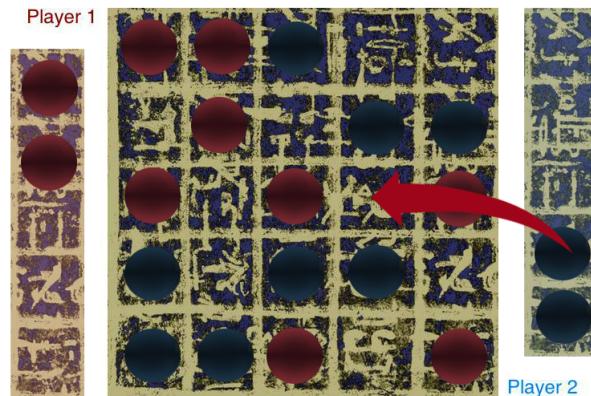


Figura 2: Preparação do tabuleiro.

Quando as pedras estiverem todas colocadas no tabuleiro, o jogador vermelho começa a jogar e alterna de turno com o jogador adversário.

1.2.1 Jogadas possíveis

Um jogador pode no seu respetivo turno realizar cada uma das seguintes jogadas: *Push*, *Move* e *Sacrifice*. No primeiro turno de cada jogador, deve ser feito um *Push* ou um *Move*, mas nunca ambos.

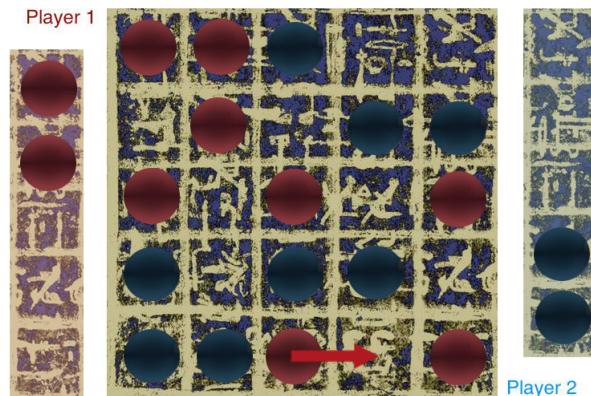


Figura 3: Jogador vermelho realiza um *Move* no seu primeiro turno.

1.2.2 Push

Para realizar um *Push* é necessário que o jogador tenha mais pedras nessa linha que o adversário. Assim, se após um *Push*, a pedra do adversário é "empurrada" para fora do tabuleiro, então vai para o "banco" do respetivo jogador. As pedras que são "empurradas" apenas se movem uma célula na direção do *Push*, que pode ser realizado em qualquer direção.

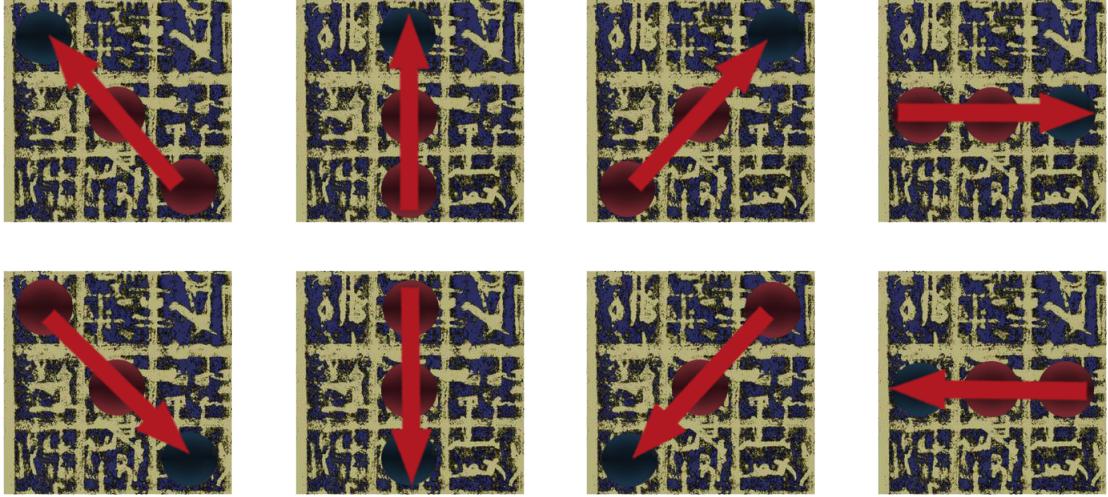


Figura 4: Direções possíveis de um *Push* e situações em que a pedra do adversário é "empurrada" para fora do tabuleiro.

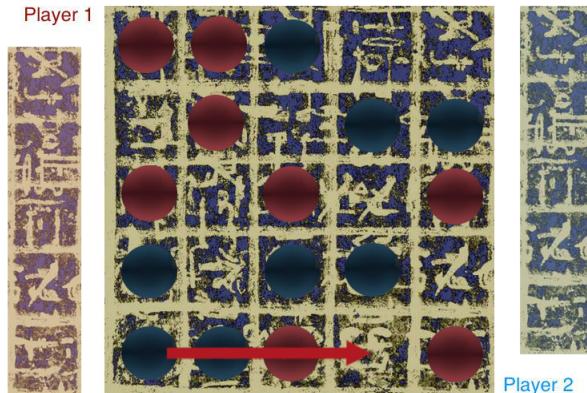


Figura 5: *Push* sem que a pedra adversária seja "empurrada" para fora do tabuleiro.

O número de pedras máximo (PE) que um jogador pode "empurrar" é igual a:

$$PE = P - 1 \quad (\text{sendo } P \text{ o número das suas pedras nessa linha}) \quad (1)$$

Isto é, por exemplo, caso numa linha o jogador azul tenha três pedras, então pode realizar um *Push* a uma pedra ou a duas do adversário. Mas caso tenha duas apenas pode realizar um *Push* a uma pedra vermelha.

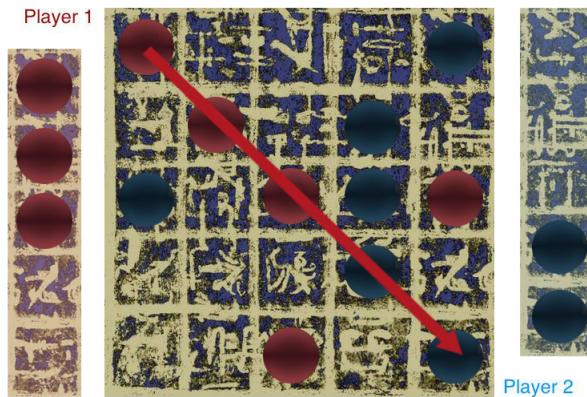


Figura 6: Push com três pedras.

Além das regras referidas anteriormente, um jogador não pode realizar um *Push* nas suas próprias pedras. Por fim, para realizar um *Push* é necessário que exista uma pedra para ser "empurrada". Por exemplo, caso numa coluna o jogador vermelho tenha duas pedras, mas nessa coluna o jogador vermelho não tem pedras, então realizar um *Push* é uma jogada inválida.

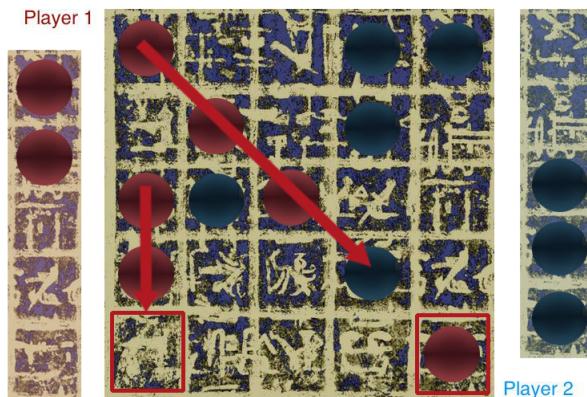


Figura 7: Push inválido: nas suas próprias pedras (a); numa célula vazia (b).

1.2.3 Move

É possível realizar um *Move* para uma célula vazia em qualquer direção. Se o jogador tentar realizar um *Move* para uma célula ocupada, então é considerado uma jogada inválida.

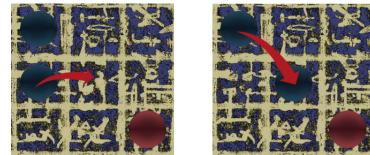


Figura 8: *Move* válido (a) e um *Move* inválido (b).

Após um *Move* se alguma das pedras adversárias ficar cercada³, então esta é capturada e substituída por uma pedra do "banco" do jogador que a capturou.

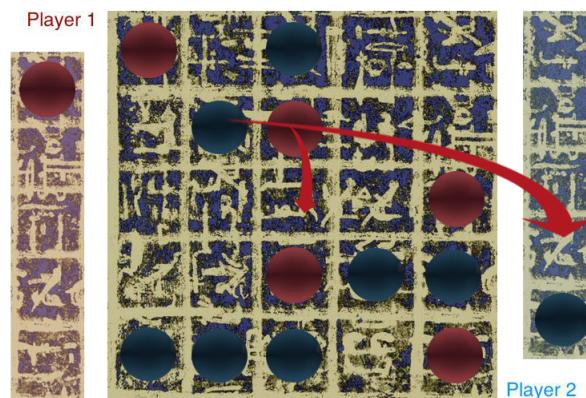


Figura 9: Pedra azul capturada após um *Move* de uma pedra vermelha.

Se um jogador realizar um *Move* e a sua pedra ficar voluntariamente cercada, esta não é capturada.

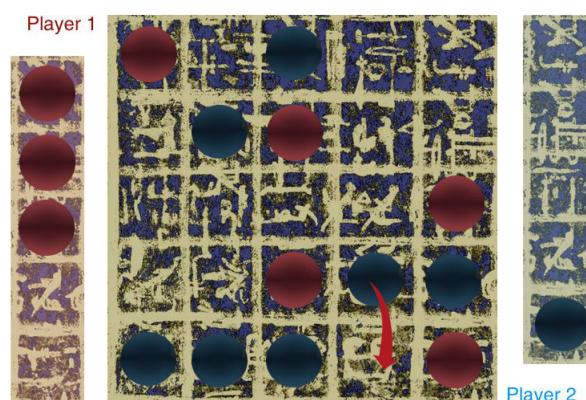


Figura 10: Pedra que após um *Move* não é capturada.

³Uma pedra fica cercada quando existem pelo menos em duas direções opostas pedras adversárias.

1.2.4 *Sacrifice*

Um jogador pode sacrificar uma pedra do seu "banco", permanentemente, para poder realizar um *Push* ou um *Move* adicional.



Figura 11: Jogador azul realiza um *Move* (I), um *Sacrifice* (II) e depois um *Push* (III) e ganha o jogo.

2 Representação do Estado do Jogo

O tabuleiro é representado por uma lista de listas $L = \{L_1, L_2, \dots, L_n\}$, $n > 0 \wedge n$ é múltiplo de 5, em que n é a dimensão do tabuleiro.

Com o predicado `make_board` é possível criar um tabuleiro (*Board*) com um tamanho *Size*. Para isso o `make_board` utiliza o predicado `make_line` para criar uma linha e depois usa o `make_board` recursivamente. De forma a tornar mais clara a visualização do tabuleiro de jogo, após criá-lo com o `make_board` utiliza-se o predicado `print_board` que o imprime no ecrã.

```
make_board(5, Board), print_board(Board).
[| - | - | - | - | - |
 | - | - | - | - | - |
 | - | - | - | - | - |
 | - | - | - | - | - |
 | - | - | - | - | - |
Board = [[0,0,0,0,0],[0,0,0,0,0],[0,0,0,0,0],[0,0,0,0,0],[0,0,0,0,0]]
```

Figura 12: Estado inicial do jogo.

A figura acima demonstra o estado do tabuleiro na sua fase inicial, sem nenhuma célula preenchida. De uma forma mais visual, segue-se abaixo uma figura demonstrativa do tabuleiro.



Figura 13: Tabuleiro de jogo.

O jogador pode realizar cada uma das três jogadas possíveis: *Push*, *Move* e *Sacrifice*. Para o *Push* é usado o predicado `push(Stone_src, Stone_dst, Dir, Board)`, onde `Stone_src` e `Stone_dst` são as coordenadas x-y (separadas por um traço) das pedras inicial e destino, `Dir` é a direção do *Push* e pode ter os valores: 0 (cima), 1 (direita), 2 (baixo) ou 3 (esquerda) e `Board` é o tabuleiro que é retornado após o *Push*.



Figura 14: Exemplo de um *Push*.

Para o *Move* é usado o predicado `move(Stone_src, Cell_dst, Board)`, onde `Stone_src` é a pedra que se pretende mover, `Cell_dst` é a célula de destino e o `Board` é o tabuleiro que é retornado.

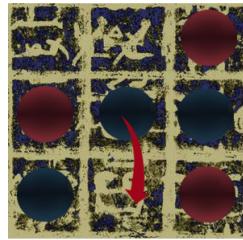


Figura 15: Exemplo de um *Move*.

Para o *Sacrifice* é usado o predicado `sacrifice(Stone, Board)`, onde `Stone` é a pedra que se pretende sacrificar e o `Board` é o tabuleiro que é retornado.



Figura 16: Exemplo de um *Sacrifice* (II).

Existe ainda um predicado `moveToPool(Stone, Player, Pool)` que move uma pedra (`Stone`) para o "banco" de um dado jogador (`Player`) retorna o "banco" do jogador (`Pool`) após a pedra ter sido movida.

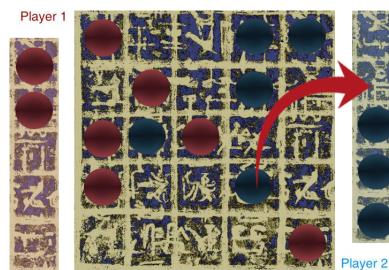


Figura 17: Exemplo de uma pedra a ser colocada no "banco" de um jogador.

3 Visualização do Tabuleiro

A cor das peças dos jogadores são representadas pelos caracteres x para as peças azuis e o para as peças vermelhas, sendo que o carácter _ é usado para representar casas vazias.

Os predicados utilizados para construir e visualizar o tabuleiro foram os seguintes:

- `make_line/2` - Gera uma linha do tabuleiro
- `make_board/2` - Gera um tabuleiro de jogo com o tamanho passado
- `print_line/1` - Predicado para imprimir uma linha do tabuleiro
- `print_board/1` - Predicado que imprime o tabuleiro, passado sob a forma de lista de listas
- `draw_board/2` - Predicado que chama os predicados anteriores para gerar um tabuleiro com tamanho passado e o imprime no ecrã

```
| ?- draw_board(5,L).
|   _ | _ | _ | _ | _
|   _ | _ | _ | _ | _
|   _ | _ | _ | _ | _
|   _ | _ | _ | _ | _
|   _ | _ | _ | _ | _
L = [[0,0,0,0,0],[0,0,0,0,0],[0,0,0,0,0],[0,0,0,0,0],[0,0,0,0,0]] ? yes
yes
| ?- draw_board(10,L).
|   _ | _ | _ | _ | _ | _ | _ | _ | _ | _
|   _ | _ | _ | _ | _ | _ | _ | _ | _ | _
|   _ | _ | _ | _ | _ | _ | _ | _ | _ | _
|   _ | _ | _ | _ | _ | _ | _ | _ | _ | _
|   _ | _ | _ | _ | _ | _ | _ | _ | _ | _
|   _ | _ | _ | _ | _ | _ | _ | _ | _ | _
|   _ | _ | _ | _ | _ | _ | _ | _ | _ | _
|   _ | _ | _ | _ | _ | _ | _ | _ | _ | _
|   _ | _ | _ | _ | _ | _ | _ | _ | _ | _
L = [[0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0],[0,0,0,0,0,0,0,0,0,0]] ? yes
yes
| ?-
```

Figura 18: Exemplo da chamada do predicado `draw_board` para um tamanho 5x5

Descrever a forma de visualização do tabuleiro em modo de texto e o(s) predicado(s) Prolog construídos para o efeito. Deve ser incluída pelo menos uma imagem correspondente ao output produzido pelo predicado de visualização.

4 Movimentos

Elencar os movimentos (tipos de jogadas) possíveis e definir os cabeçalhos dos predicados que serão utilizados (ainda não precisam de estar implementados).

Bibliografia

”Jogo de estratégia abstrato”, https://pt.wikipedia.org/wiki/Jogo_de_estrat%C3%A9gia_abstrato (acedido em 7 de Outubro de 2015)

”Sixteen Stone”, <http://www.boardgamegeek.com/boardgame/173193/sixteen-stone> (acedido em 7 de Outubro de 2015)

”Sixteen Stone - rules v.1.0”, <http://www.boardgamegeek.com/filepage/117073/sixteen-stone-rules-v10> (acedido em 7 de Outubro de 2015)