Aplicação de técnica de aprendizagem por reforço para solução do Jogo Resta Um com tabuleiro triangular

Prof. Jorge Dantas

Alunos:

- Alexandre Gomes de Lima
- Francisco Sales de L. Filho

Agenda

- Definição do Problema
- Objetivo do trabalho
- Algoritmo Escolhido
- Modelagem do Problema
- Implementação
- Resultados Obtidos
- Informações Adicionais
- Conclusões

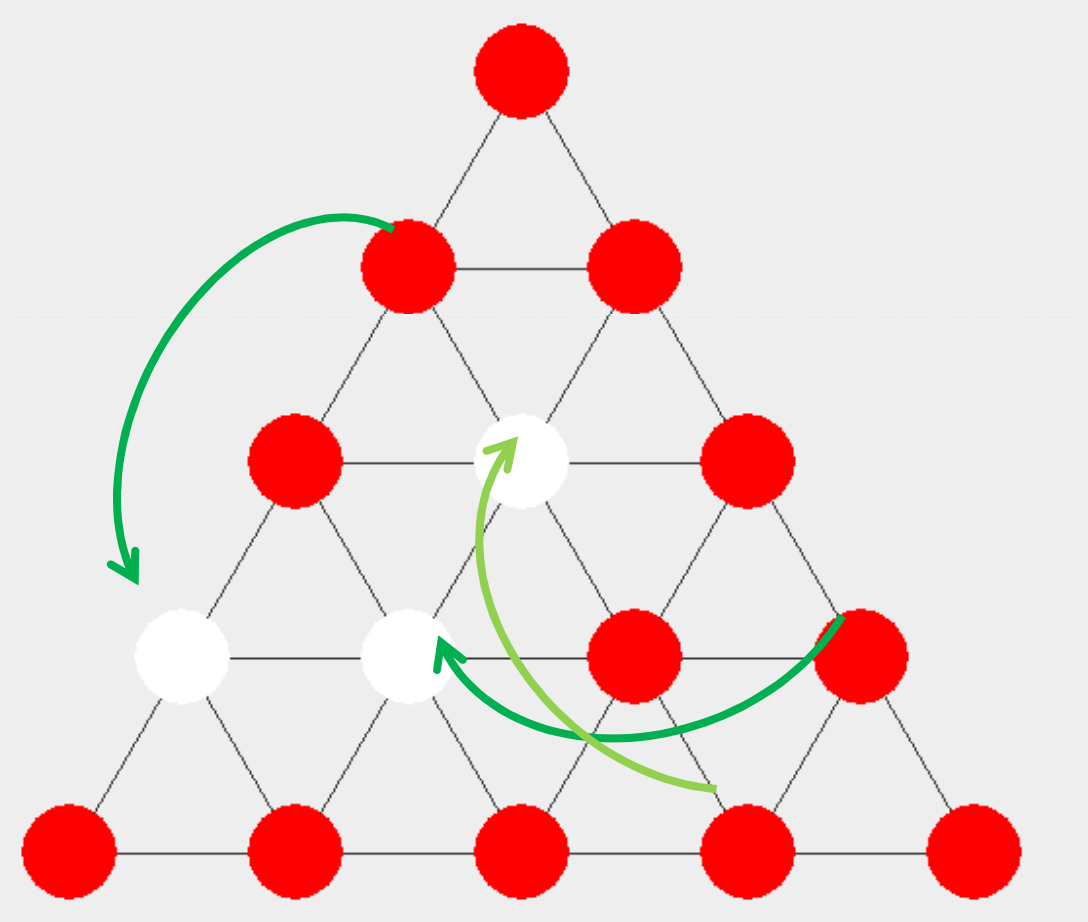
Definição do Problema

- Resta Um é jogo de tabuleiro formado por casas que podem estar ocupadas ou não
- Existem vário formatos de tabuleiro
  - Retangular
  - Triangular
  - Formato de cruz
  - …

Definição do Problema

- Regras
  - O jogo inicia com uma casa vazia
  - Uma peça apenas pode ser movida para as casas vizinhas dos vizinhos da sua casa atual e mesmo assim somente se a casa intermediária estiver ocupada e a casa de destino estiver vazia
  - Ao movimentar uma peça, a peça da casa intermediária é retirada do tabuleiro
  - O jogo termina quando não houverem mais movimentos possíveis
  - O objetivo é terminar o jogo restando apenas uma peça
- Alteração da regra para o trabalho
  - O tabuleiro inicia com uma quantidade arbitrária de casas vazias posicionadas aleatoriamente
  - O objetivo é terminar o jogo com a menor quantidade peças possíveis

Definição do problema



Objetivo do trabalho

- Implementar um algoritmo de aprendizagem por reforço para obter o melhor resultado final do jogo Resta Um
- Integrar o algoritmo implementado com algum simulador do jogo com tabuleiro triangular

Algoritmo Escolhido

- Métodos de Monte Carlo
  - Não exigem conhecimento completo do modelo
  - Requerem experiência (seqüência de estados, ações e retornos a partir da interação com o modelo)
  - Necessitam de tarefas episódicas (possuem estado terminal)
  - São baseados na média dos retornos obtidos a partir dos episódios
- Aplicável ao problema do Resta Um

Algoritmo Escolhido

- Monte Carlo ES (***E**xploring **S**tarts*)
  - Utilizado para encontrar a política ótima com base no processo de avaliação e melhoria da política
  - Inicia com uma política arbitrária
  - A cada episódio realiza a melhoria da política com base na média dos retornos obtidos
  - Partidas exploratórias e um número infinito de episódios garantem a convergência para a política ótima

Algoritmo Escolhido

- Monte Carlo ES

Initialize, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
$Q(s,a) \leftarrow$ arbitrary
$\pi(s) \leftarrow$ arbitrary
$Returns(s,a) \leftarrow$ empty list

Repeat forever:
(a) Generate an episode using exploring starts and π
(b) For each pair s,a appearing in the episode:
$R \leftarrow$ return following the first occurrence of s,a
Append R to $Returns(s,a)$
$Q(s,a) \leftarrow \text{average}(Returns(s,a))$
(c) For each s in the episode:
$\pi(s) \leftarrow \arg\max_a Q(s,a)$

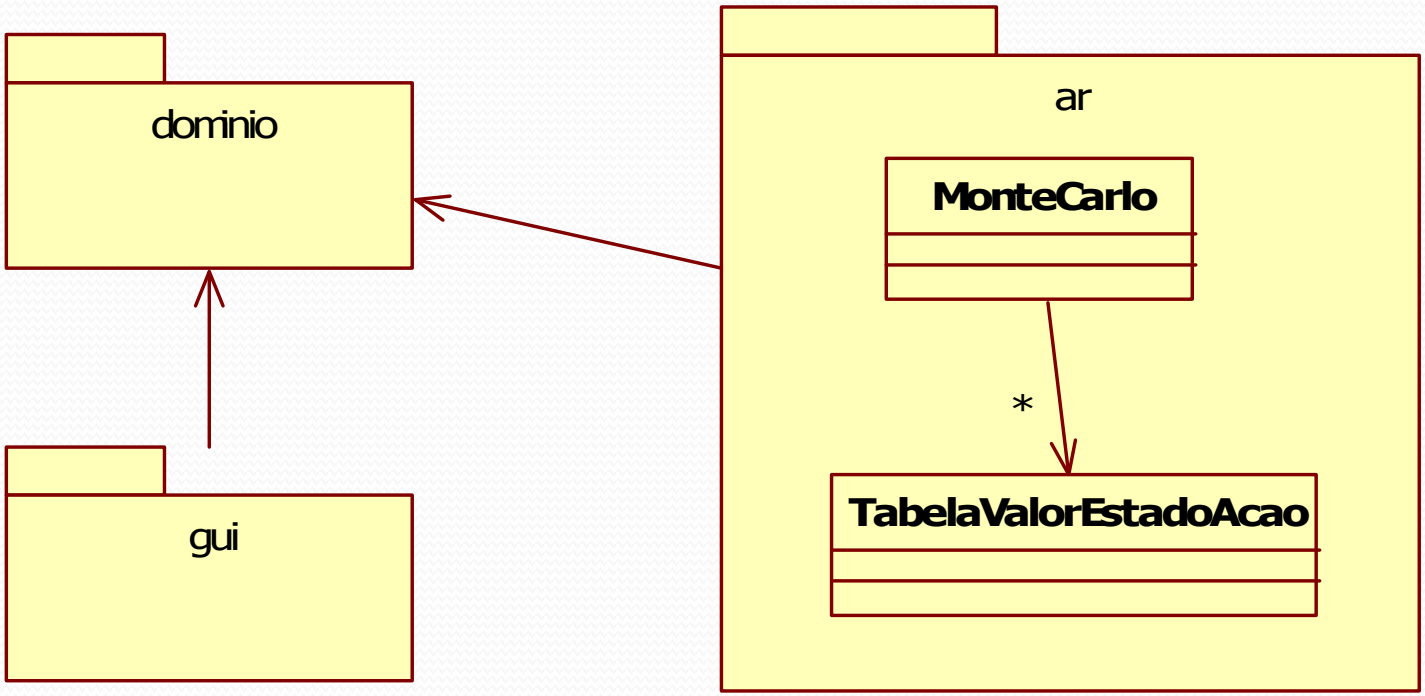
Modelagem do Problema

- Estados: configurações do tabuleiro
- Ações: movimentos possíveis
- Retornos r(s, a):

```
Se (quantidade de casas ocupadas de s') = 1
   R ← 100
Senão
   R ← (quantidade de casas vazias de s') * 2
```

Implementação

- Linguagem de programação Java



Experimentos

- Para o tabuleiro de 15 peças foi gerada uma política a partir de um treinamento com 1.000.000 de episódios
- Para o tabuleiro de 21 peças foi gerada uma política a partir de um treinamento com 400.000 episódios

Resultados Obtidos

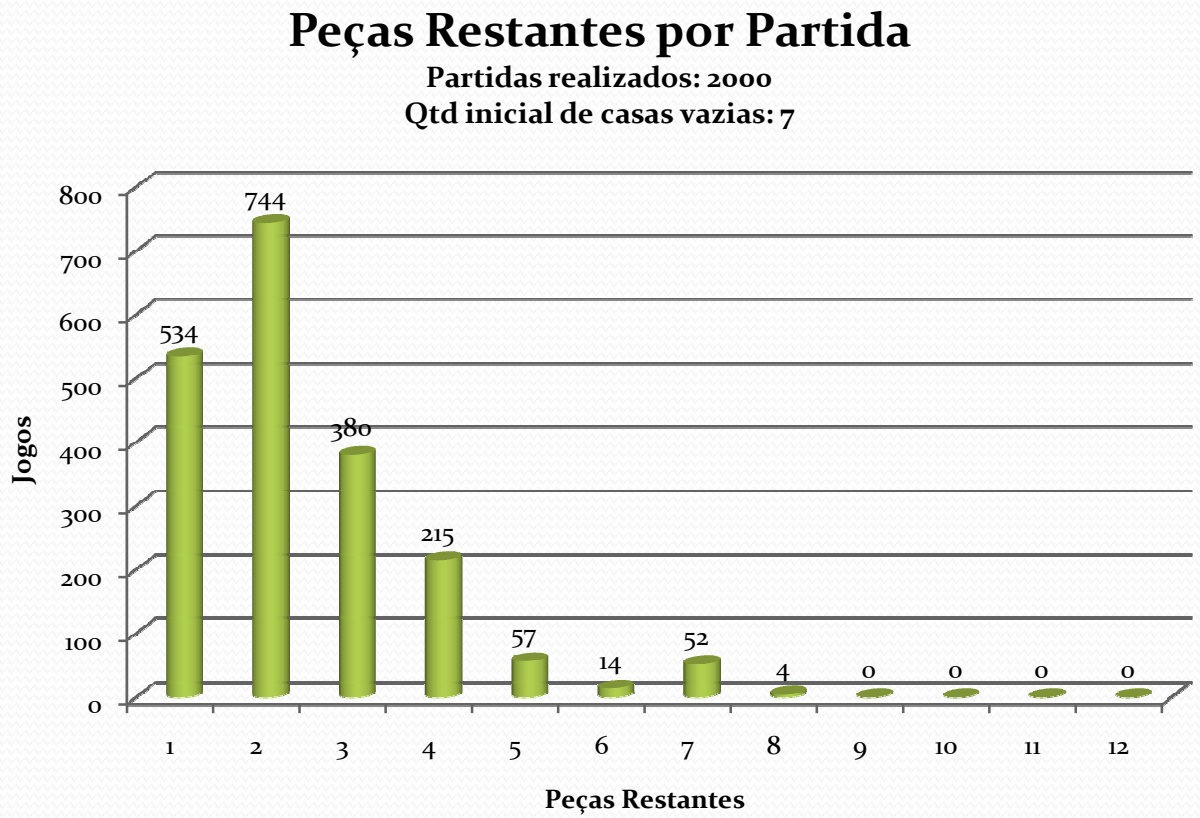
- Tabuleiro com 15 casas

Resultados Obtidos

- Tabuleiro com 15 casas

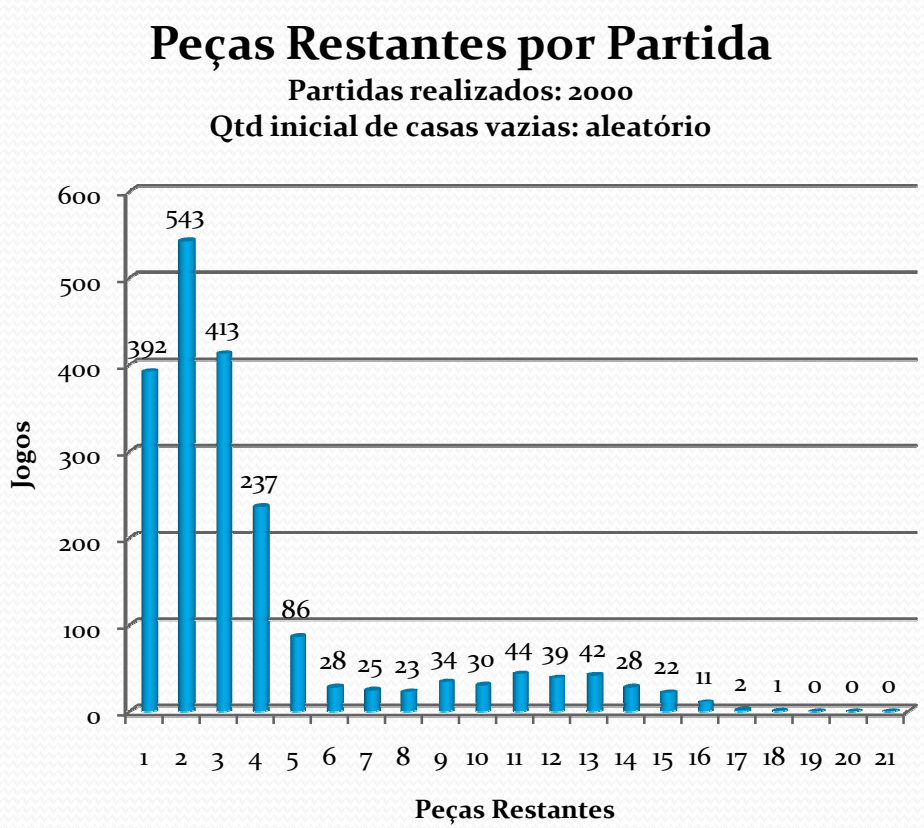
Resultados Obtidos

- Tabuleiro com 15 casas



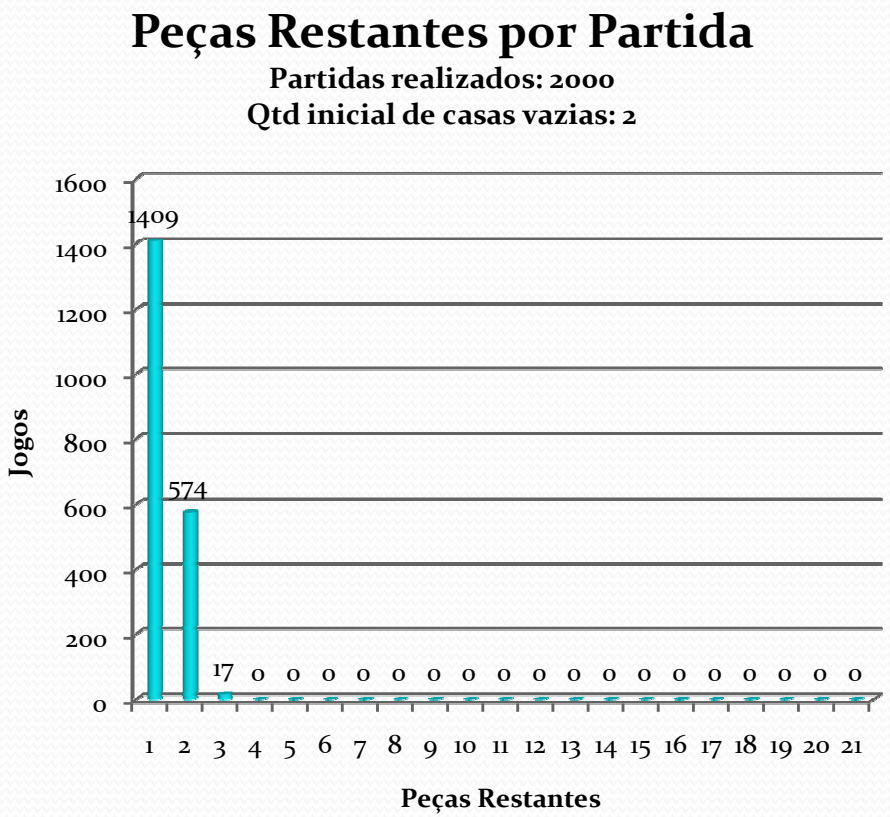
Resultados Obtidos

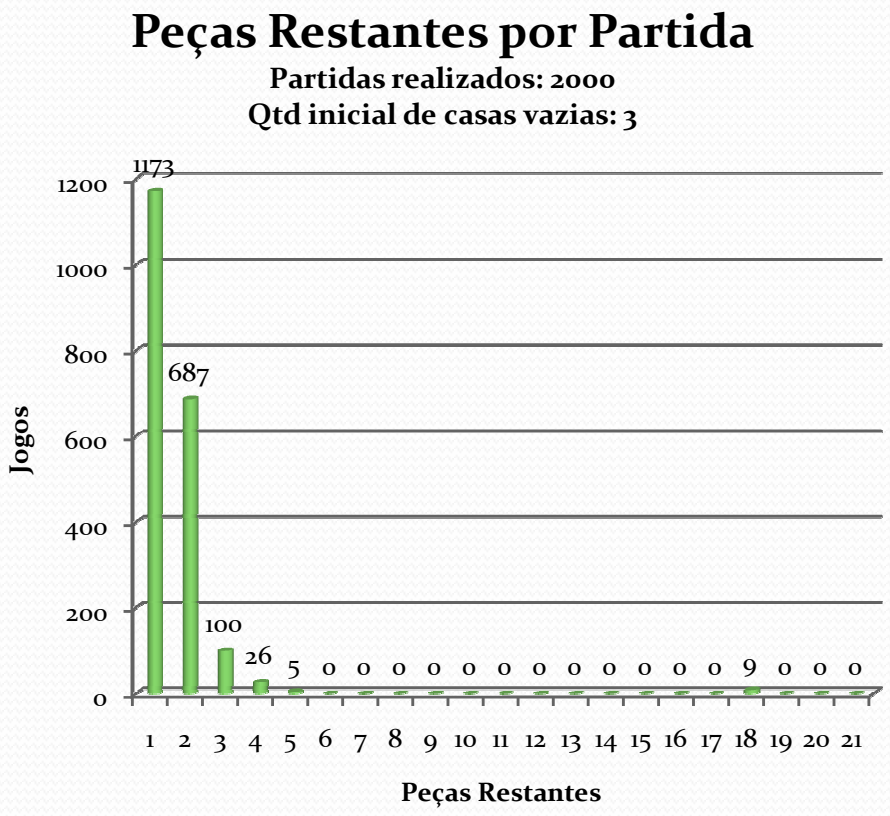
- Tabuleiro com 21 casas



Resultados Obtidos

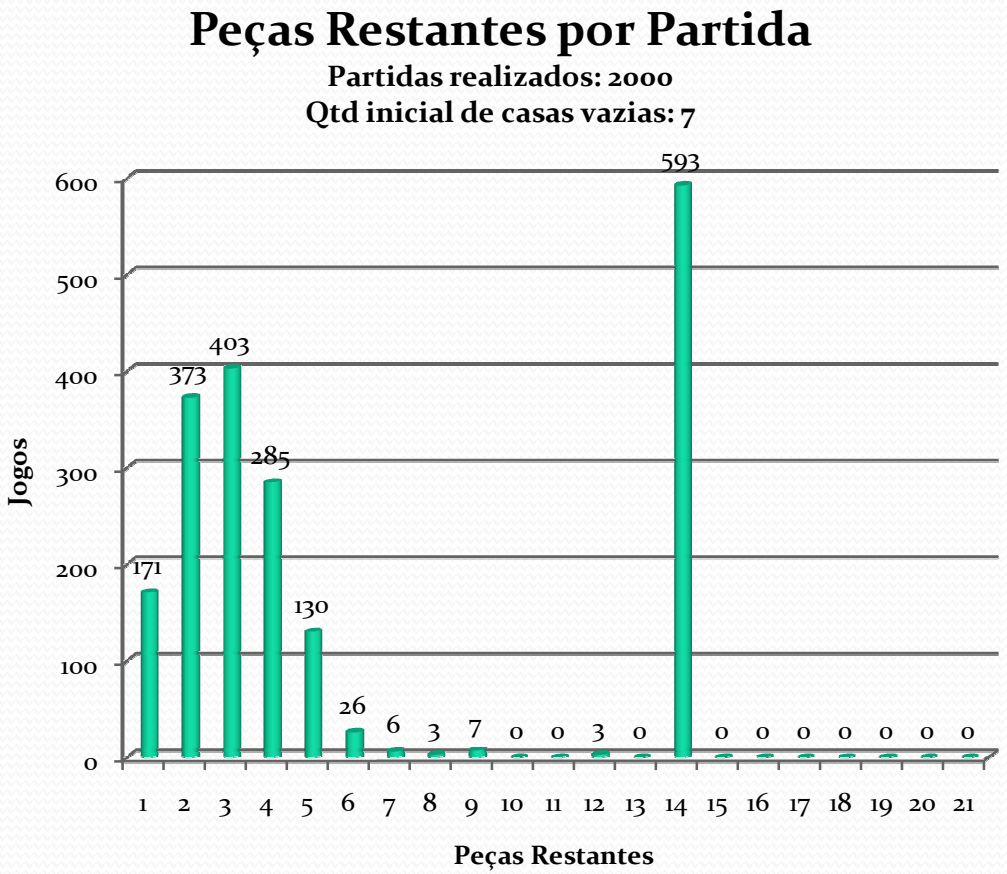
- Tabuleiro com 21 casas





Resultados Obtidos

- Tabuleiro com 21 casas



Informações Adicionais

- Problemas técnicos de implementação
  - Estouro da capacidade inicial do *heap (64 MB)*
  - Tabuleiro de 15 peças
    - *Para 500.000 episódios foram utilizados cerca de 100MB de memória*
    - *Para 1.000.000 de episódios cerca de 180MB*
  - Tabuleiro de 21 peças
    - *Para 400.000 episódios cerca de 600MB*

Informações Adicionais

- Treinamentos com tabuleiro de 15 peças
  - Com 500.000 episódios (com *profiler* ativado): 25 minutos
  - Com 300.000 episódios o programa raramente conseguia atingir o objetivo de restar um
  - Com 500.000 episódios foram visitados 25.335 estados e 68.840 pares estado-ação, sendo que o total de pares-ação inicializados arbitrariamente foi de 127.639
  - A política gerada a partir de 500.000 episódios mostrou-se ótima para partidas com uma casa vazia
  - Com 1.000.000 episódios (com *profiler* ativado): 40 minutos
- Treinamentos com tabuleiro de 21 peças
  - Com 400.000 episódios: 14 minutos
  - Com 400.000 episódios foram visitados 434.754 estados e 528.738 estados ação, sendo que o total de pares estado-ação inicializados arbitrariamente foi de 2.700.650

Conclusões

- Não foi necessário visitar todos os pares estado-ação para encontrar políticas ótimas para partidas de uma casa vazia
  - Tabuleiro de 15 peças
    - Total de estados: 32.768
    - Estados visitados: 25.335
  - Tabuleiro de 21 peças
    - Total de estados: 2.097.152
    - Estados visitados: 434.754
- Não foi possível determinar se para partidas com mais de uma casa vazia a política encontrada não era a ótima ou tratavam-se de configurações de tabuleiro sem possibilidade de restar um

Referências bibliográficas

- [1] Richard S. Sutton and Andrew G. Barto, Reinforcement Learning: An Introduction.