Tenpair

Reinforcement learning

João Carmona
Jorge Couto
Nuno Minhoto

Especificação do problema

- Tabuleiro com números de 1 a 9
- Combinar dois números elimina-os
  - Números iguais
    OU
  - Soma dos números é 10
  - Adjacentes
    - Vertical
    - Esquerda -> Direita (Cima -> Baixo)
    - Células vazias não contam
- Termina com tabuleiro vazio

- Simplificações
  - Tabuleiro mais pequeno
  - Remover “deal”
  - Remover “compact”

Modelo do jogo

- Episódico
- Estado - Tabuleiro atual
  - Matriz 2x4
  - Números de 0-9
- Ações

Operador	Condições	Efeitos
Step(action)	(n[i]==n[j] || n[i]+n[j]==10) && (i%4==j%4 || i==j+1 || j==i+1)	n[i]==0, n[j]==0

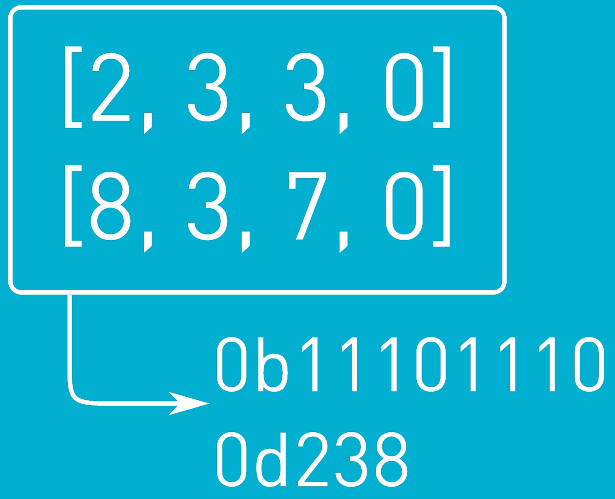
- Se não houver mais ações possíveis então faz-se reset ao tabuleiro

Policies e Recompensas

- Q-learning
- ε-greedy adaptativo
- Recompensa -1 por ação
- Nenhuma ação possível, recompensa de -2

Implementação

- Representação do estado
  - Simplificação (explosão combinatória)
  - Conversão binária do estado (esquerda -> direita)
    - Espaço vazio - 0
    - Else - 1
- 11 ações em cada estado
- Q-table de dimensão 256x11



Q-learning

- Em cada passo
  - Escolhida uma ação
  - Atualizar a Q-table: $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t) \right]$
  - Em estados terminais, novo episódio
- ε variável
  - Fase inicial de grande exploração
  - Exploração passa a ação com #passos e #episódios

Resultados

- Tabuleiro possível -> Solução ótima

```
[[3. 1. 8. 9.]
 [3. 9. 8. 9.]]
action to perform:  7
possible:  True
[[0. 1. 8. 9.]
 [0. 9. 8. 9.]]
action to perform:  9
possible:  True
[[0. 1. 0. 9.]
 [0. 9. 0. 9.]]
action to perform:  1
possible:  True
[[0. 0. 0. 0.]
 [0. 9. 0. 9.]]
action to perform:  5
possible:  True
[[0. 0. 0. 0.]
 [0. 0. 0. 0.]]
```

```
[[1. 8. 9. 1.]
 [9. 8. 9. 1.]]
action to perform:  7
possible:  True
[[0. 8. 9. 1.]
 [0. 8. 9. 1.]]
action to perform:  2
possible:  True
[[0. 8. 0. 0.]
 [0. 8. 9. 1.]]
action to perform:  8
possible:  True
[[0. 0. 0. 0.]
 [0. 0. 9. 1.]]
action to perform:  6
possible:  True
[[0. 0. 0. 0.]
 [0. 0. 0. 0.]]
```

```
[[8 4 6 2]
 [7 3 6 4]]
action to perform:  1
possible:  True
[[8 0 0 2]
 [7 3 6 4]]
action to perform:  4
possible:  True
[[8 0 0 2]
 [0 0 6 4]]
action to perform:  0
possible:  True
[[0 0 0 0]
 [0 0 6 4]]
action to perform:  6
possible:  True
[[0 0 0 0]
 [0 0 0 0]]
```

```
[[8 2 9 1]
 [7 4 6 3]]
action to perform:  5
possible:  True
[[8 2 9 1]
 [7 0 0 3]]
action to perform:  2
possible:  True
[[8 2 0 0]
 [7 0 0 3]]
action to perform:  4
possible:  True
[[8 2 0 0]
 [0 0 0 0]]
action to perform:  0
possible:  True
[[0 0 0 0]
 [0 0 0 0]]
```

Referências

Dandurand, F., Cousineau, D., Shultz, T. R. *Solving nonogram puzzles by reinforcement learning* [online] escholarship.org

https://towardsdatascience.com/simple-reinforcement-learning-q-learning-fcddc4b6fe56

https://www.freecodecamp.org/news/diving-deeper-into-reinforcement-learning-with-q-learning-c18d0db58efe/