

SPACE BLOCK

APRESENTAÇÃO

Afonso Santos | Carolina Jesus | Rafael Santos || EIACD || 2024/25



Space Block: Roll The Block

O objetivo do jogo é levar um **bloco** retangular até um ponto específico (**meta**) no tabuleiro, utilizando o menor número de movimentos possível.

No percurso também há obstáculos como **buracos**, **chãos de vidro** e obrigatoriedade de apanhar **gemas**.

WASD: Mover | Q: Sair
Movimentos:

Fig.1 – Comandos

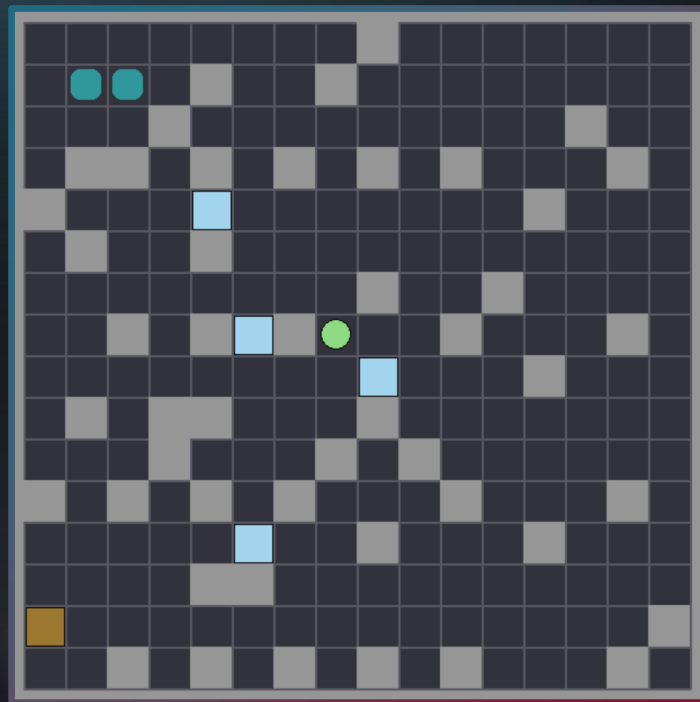


Fig.2 – Tabuleiro do jogo



Fig.3
Bloco



Fig.4
Objetivo

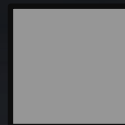


Fig.5
Buracos



Fig.6
Caminho

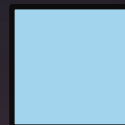


Fig.7
Vidro



Fig.8
Gema



Fig.9 – Barra de progresso

O jogo como um problema de pesquisa

Estado inicial: começa sempre *standing*.

Estado objetivo: chegar ao quadrado dourado e acabar nele *standing*.

Operadores: o *horizontal_x*, o *horizontal_y* e o *standing*.

Precondições: as precondições para todos os movimentos é a necessidade de os quadrados para onde o bloco se vai mover terem de existir.

Custo: 1 (que é o movimento do bloco).



Fig.10 – *horizontal_y*



Fig.11 – *standing*



Fig.12 – *horizontal_x*



Fig.13 – objetivo

Níveis, dificuldades e jogabilidade

À medida que os níveis avançam, a dificuldade e a complexidade dos níveis aumenta.

Nível 1 → Nível 6: níveis que podem ser resolvidos pelo computador com diferentes algoritmos ou com o modo Humano.

Nível 7 → Nível 9: níveis que só têm o modo Humano.

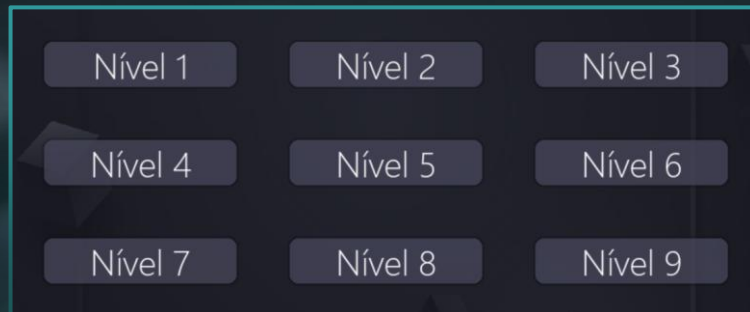


Fig.14 – níveis

Interface Gráfica

Elementos da interface gráfica foram criados com IA, como a música de fundo do jogo através da plataforma Soundraw.



Fig.15 – Soundraw

Código Implementado

Implementação do código através de **classes**,
dividindo o jogo por ficheiros, facilitando assim a
separação dos componentes

Implementação de todos os níveis em **pygame**.

Usamos ainda algumas bibliotecas fundamentais,
como a biblioteca `copy(deepcopy)`, `pygame`,
`collections` e `sys`.

Algoritmos e Heurística

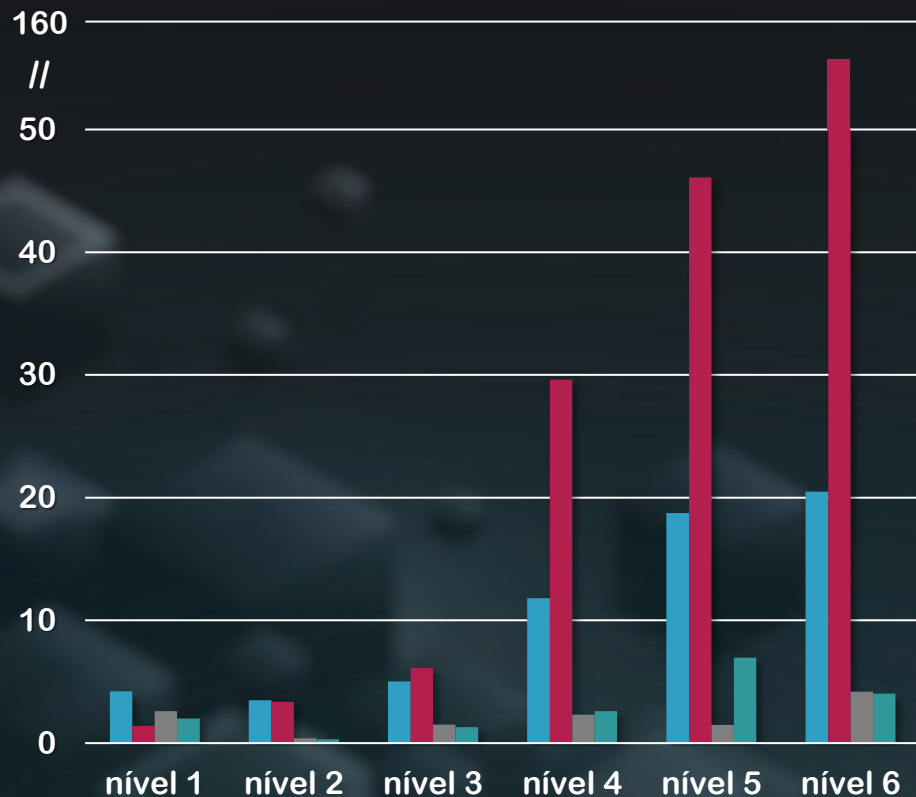
Implementação de 4 algoritmos de pesquisa dos níveis $1 \rightarrow 6$:

- o **BFS** (Breadth First Search);
- o **DFS** (Depth First Search);
- o **Greedy** search;
- o **A*** search.

Em relação ao Greedy search e ao A* search, usamos uma heurística, a **distância de Manhattan**, que é calculada através da soma das distâncias das coordenadas x e y entre a posição atual do bloco e a meta.

Comparação dos resultados

Algoritmos (tempo de execução/ ms)



Algoritmos (nós)

	BFS	DFS	Greedy	A*
nível 1	57	21	36	27
nível 2	7	242	4	4
nível 3	120	409	28	25
nível 4	292	1834	28	42
nível 5	387	3184	24	113
nível 6	399	8868	72	72

Discussão dos resultados

Obtenção dos resultados experimentais: média de 5 testes para cada algoritmo de pesquisa em cada nível.

Elaboração da tabela (células a verde → solução ótima de cada nível) e gráficos.

Considerações:

- **DFS:** algoritmo de pesquisa menos eficiente para este jogo (visita bastantes nós antes de chegar à solução ótima).
- **Greedy:** não é muito eficiente neste jogo (encontrou apenas a solução ótima 2 vezes).
- **BFS:** demora um pouco mais que o A* (porque os níveis não são muito complexos), mas encontrou bastante rápido a solução → é uma opção viável para este jogo.
- **A*:** não é tão rápido como o Greedy, no geral, mas encontra sempre a solução ótima.

Algoritmos (tempo de execução/ ms)

	BFS	DFS	Greedy	A*
nível 1	4,23	1,417	2,601	2,007
nível 2	3,505	3,357	0,401	0,301
nível 3	5,025	6,117	1,499	1,305
nível 4	11,808	29,607	2,308	2,6
nível 5	18,758	46,089	1,496	6,95
nível 6	20,498	50	4,206	4,05

Conclusão

Em suma, o jogo “**Space Block: Roll The Block**” foi explorado como um problema de pesquisa e foram aplicados diferentes algoritmos de pesquisa (o BFS, o DFS, o Greedy e o A*) para encontrar soluções eficientes para o jogo.




Os resultados indicam que o algoritmo A* obteve o melhor desempenho em termos de qualidade das soluções encontradas, enquanto que o DFS teve o pior desempenho.

Além disso, este estudo também mostrou que a distância de Manhattan (para este jogo) melhora o desempenho, eficácia e rapidez do algoritmo A*.

Assim sendo, com este trabalho é possível inferir que o uso de algoritmos de pesquisa pode ser uma ferramenta valiosa para a resolução de problemas de jogos, fornecendo soluções eficientes e otimizadas, sendo, no entanto, também premente considerar as limitações de cada algoritmo e escolher o que melhor se adapta.



Pesquisa Bibliográfica

- <https://play.google.com/store/apps/details?id=com.zawor.spaceblock&hl=en&gl=US>
- https://www.mathplayground.com/logic_blockorz.htm
- Ficheiros disponibilizados no moodle
-  Copilot
-  deepseek
-  **SOUNDRAW**