
# Relatório do Trabalho de Grupo



Grupo: 93264_93086_92948_p5
Rodrigo Martins, Lúcia Sousa, Raquel Pinto

universidade de aveiro

# Descrição do Algoritmo

- O algoritmo de procura foi baseado no tree_search das aulas práticas, sendo que um estado representa a posição das caixas e do keeper.
- Os novos nós vão sendo gerados com base nas ações possíveis num determinado estado. Estes serão organizados por ordem crescente da sua heurística mais o seu custo (estratégia A*).
- Para reduzir o número de nós gerados são usadas funções que detetam deadlocks. A função de deadPositions verifica se as colunas e as linhas encostadas às bordas do mapa têm posições viáveis (nunca terão a não ser que tenham pelo menos um goal). A função de deadlocks só verifica deadlocks em cantos.
- O algoritmo começa por gerar as ações possíveis num determinado estado (andar numa das 4 direções), executa uma dessas ações no estado atual e verifica se alguma caixa é empurrada, se for, é calculada a posição onde a caixa ficaria para se poder verificar se está em deadlock ou se ficaria numa posição que correspondesse a outra caixa ou a uma parede.

# Descrição do Algoritmo

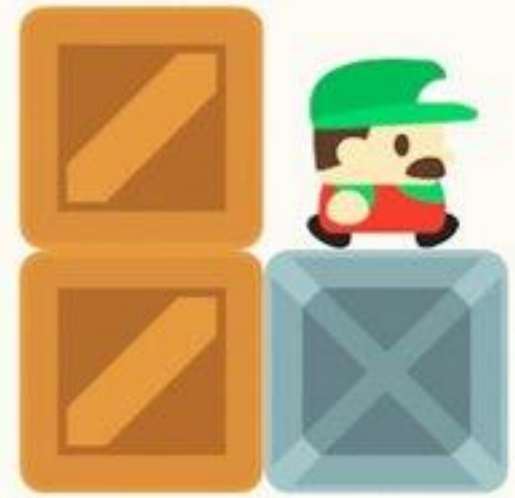
- Caso seja uma posição válida, um novo estado é gerado com a posição das caixas atualizadas e é adicionada essa posição a um set de posições, que não são deadlocks para que no futuro não tenhamos que verificar outra vez essa posição (isto é válido porque só são detetados deadlocks em cantos com paredes). Os estados gerados serão organizados pela sua heurística, que consiste na soma das distâncias das caixas ao seu goal correspondente.
- A associação das caixas aos goals é feita quando o problema é criado (uma caixa fica associado ao goal mais próximo no estado inicial do problema). A criação de novos estados é repetida até que o número de caixas em goals seja o mesmo que o número de goals existentes.



# Resultados obtidos

- Com o nosso algoritmo o agente consegue chegar ao nível 131 com a estratégia A*.
- O algoritmo demora relativamente pouco tempo para resolver cada nível. Nos níveis 68 e 117 é onde o algoritmo encontra mais problemas, demorando assim muito mais tempo a resolver estes 2 níveis.





# Conclusões

- Neste projeto decidimos representar os estados com as caixas e o keeper, o que levou à criação de muitos estados, por isso, tentamos reduzir ao máximo o número de estados repetidos e de estados que impossibilitam a solução. Esta escolha levou-nos a usar sets em vez de listas pois estes são mais eficientes (tanto para procurar se um elemento existe, como para adicionar um elemento) e a criar funções que eliminam posições no mapa (isto reduz o número de possibilidades) o que tornou o algoritmo mais rápido. Estas escolhas provaram ser bastante úteis e eficientes.
- Ficou por melhorar o algoritmo que deteta ainda mais deadlocks, isto para que nos níveis com muitas caixas não fossem gerados tantos nós desnecessários para chegar à solução.