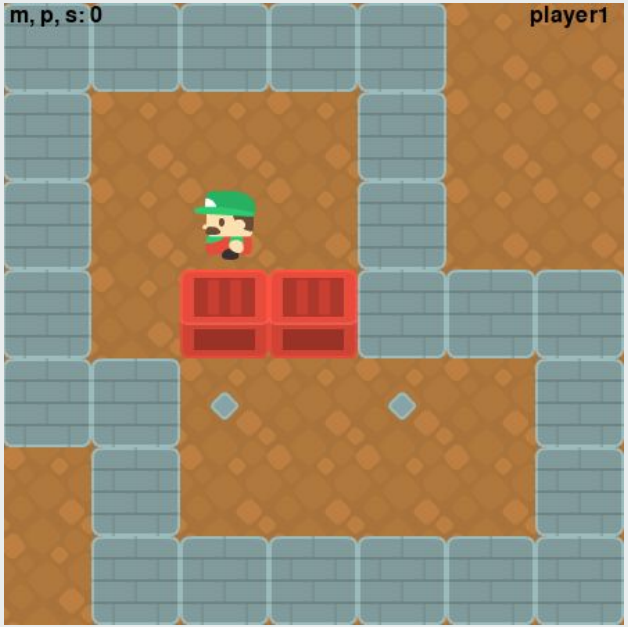
Sokoban

Luís Costa 92996
Diogo Mendonça 93002
Guilherme Pereira 93134



Overview do algoritmo

O nosso algoritmo de pesquisa divide-se em duas pesquisas em árvore:

-Uma pesquisa A*, que procura todos os movimentos possíveis das caixas que o keeper consegue efetuar, movendo a caixa e teletransportando o keeper para a antiga posição da caixa (com a devida criação de um novo nó na searchtree). Este processo repete-se até todas as caixas estarem em goals.

-Uma pesquisa em BFS, o keeper cria um caminho que une todas as suas posições para a solução da pesquisa anterior, e dá return das teclas pressionadas para resolver o nível.

Para tornar este algoritmo mais rápido implementámos também deadlocks detection e tunnel, goal room e wall macros, descritos nos próximos slides.

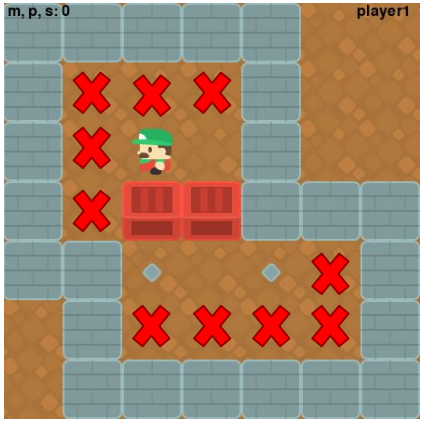
Deadlocks Detection

Para tornar a pesquisa dos movimentos das caixas mais rápido, definimos dois tipos de deadlocks:.

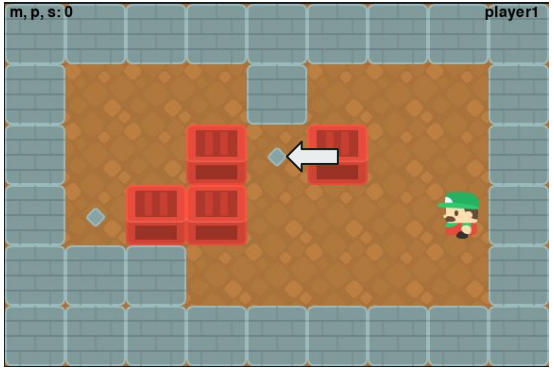
-Dead square deadlocks: Posições no mapa proibidas para as caixas, onde é impossível uma caixa chegar a um goal, independentemente do estado do mapa. Estes deadlocks são encontrados um vez no princípio do algoritmo.

-Freeze deadlocks: Causados quando o movimento de uma caixa bloqueia o movimento da mesma e de outras caixas, devido ao posicionamento destas no mapa e entre elas. Para encontrar tais deadlocks é necessário correr o algoritmo associado, todas as vezes que se tenta mover uma caixa no mapa. O algoritmo é recursivo.

Dead Square deadlocks



Freeze deadlocks



Tunnel, Room e Wall Macros

-Tunnel Macro: Quando o algoritmo detecta duas paredes ao lado da caixa que mexeu, este procura, na direção do movimento, se as posições seguintes estão livres e com duas paredes de lado, movendo enquanto isto se verificar.
-Room Macro: Quando o algoritmo detecta duas paredes ao lado da caixa que mexeu, este vê se o keeper consegue ir do lado oposto à caixa a onde está, mexendo a caixa mais uma vez na mesma direção caso isso não aconteça.
-Wall Macro: Quando o algoritmo detecta apenas uma parede de uns lados da caixa, este anda adjacente à parede até esta mudar.

Tunnel-
Room-
Wall-

Resultados e Conclusão

Com este algoritmo conseguimos resolver 137 níveis, acabando na leaderboard com um score de 6468208, 3964 pushes e 35392 steps.

Para concluir, neste trabalho sentimos que ficamos demasiado dependentes às classes já fornecidas, não permitindo, assim, construir estruturas de dados e funções mais eficientes no nosso algoritmo. Também consideramos que, tendo em conta a enorme quantidade de informação disponível para este problema (IA para Sokoban), há muitos algoritmos que poderíamos implementar no nosso, de maneira a este passar mais níveis, em alguns casos, em menos tempo; por exemplo: pi-corals.