3º ano do Mestrado Integrado de Engenharia Informática e Computação, 2014/2015

Inteligência Artificial

Box World 2

Relatório Intercalar

Francisco Lopes, 201106912
Miguel Mendes, 201105535

18/04/2015

Objetivo:

O objetivo deste trabalho consiste em desenvolver um programa que demonstre inteligência suficiente para resolver o problema em questão. Focando-nos no jogo "Box World 2", pretendemos que, em vez de ser o jogador a resolver os puzzles propostos, que o computador seja capaz de o fazer autonomamente, encontrando caminhos através dos mesmos, e empurrando quaisquer blocos necessários para que tal seja possível.

Descrição:

1) Especificação

O nosso trabalho é baseado no jogo "Box World 2". Neste jogo, o objetivo do jogador é simplesmente chegar à saída, desta forma conseguindo avançar para o nível seguinte. No entanto, os caminhos para a saída poderão estar bloqueados por buracos que impeçam a travessia até ao objetivo. Para resolver isto, existirão vários blocos que podem ser deslocados pelo jogador, e colocados nestes buracos para que chegar ao objetivo seja ultimamente possível. O desafio neste jogo reside no quão complexos poderão ser os movimentos do jogador, em congruência com a colocação destes blocos, para que este consiga resolver os vários puzzles. Além dos blocos mais comuns, que apenas movem uma unidade quando o jogador os empurra, ainda existe uma segunda categoria de blocos, blocos de gelo, que se deslocam ininterruptamente na direção em que são empurrados até que encontrem uma parede ou bloco.

O nosso objetivo consiste em desenvolver inteligência para o programa, de forma a que não seja necessário um jogador, e o computador seja capaz de resolver os vários puzzles apresentados, presumindo que existe solução, respeitando todas as regras implícitas ao correr do jogo.

Tendo em conta as prioridades deste trabalho, pretendemos começar pelo desenvolvimento e implementação do algoritmo A* que se adapta à resolução do problema, inicialmente apenas tendo em conta a existência do tipo de blocos mais comum. Assim que possuirmos um algoritmo estável que funcione de forma previsível, com resultados corretos, serão introduzidos os blocos de gelo e, se possível, o algoritmo será reajustado para que a resolução dos vários puzzles se mantenha correta. Finalmente, iremos desenvolver o ambiente gráfico para que toda a aplicação seja minimamente apresentável. Partindo do princípio que nesta etapa o algoritmo já se comporta de forma correta, sem falhas e com todas as condições de aresta testadas, concentrar-nos-emos apenas no melhoramento da componente gráfica do programa.

Considerando especificamente a aplicação do algoritmo, consideramos que será vantajosa a preferência de inserção de blocos em buracos, pelo que iremos dar o custo a este tipo de movimento um valor negativo. Considerando que a única heurística até agora considerada se baseia na distância entre o nó atual e a saída, um

valor negativo na inserção de blocos em buracos irá garantir uma preferência do algoritmo por este tipo de escolha. No entanto, como a colocação de blocos em buracos irá alterar de forma um pouco significante a estrutura do puzzle a ser resolvido, consideramos melhor a reanálise do puzzle e reaplicação do algoritmo, para ter as alterações realizadas em consideração. Inicialmente, pretende-se encontrar o menor caminho possível até ao objetivo com o mínimo número de buracos. Após a seleção de blocos a colocar em cada buraco pode-se aplicar o algoritmo, não ao jogador, mas ao bloco. Fazendo parte da heurística a condição de que um movimento de um bloco possua um espaço vazio no sentido oposto ao seu movimento, assim garantido que serão preferenciais ao algoritmo movimentos em que seja possível o jogador se colocar detrás do bloco a empurrar para que a ação seja possível.

2) Trabalho efetuado

Até este momento já desenvolvemos a representação básica de puzzles, e embora ainda não exista o jogador, já foi desenvolvido um algoritmo inicial de A* que levaria o jogador à saída, caso não existissem buracos que bloqueassem o caminho até à mesma.

3) Resultados esperados e forma de avaliação

Para testar a eficácia do trabalho desenvolvido iremos introduzir puzzles de complexidade e dificuldade cada vez maiores, e iremos verificar se o programa consegue chegar à saída do nível que lhe é apresentado, presumindo que existe solução para o mesmo. Caso contrário, o programa deverá tornar evidente que não consegue encontrar solução para o problema, o que poderá ser válido num puzzle sem solução. Caso o mesmo ocorra num puzzle que tem de facto solução, seremos obrigados a analisar e consertar quaisquer falhas e lapsos no algoritmo utilizado.

Conclusões:

Presentemente, estamos dependentes dos resultados de implementação do algoritmo que prevemos que seja o correto para a resolução do problema. Iremos manter o processo de testar casos mais simples, mantendo a ideologia que poderemos iterar até possuirmos um algoritmo estável que se apresente como solução definitiva ao problema. Evidentemente, irão surgir vários problemas e lapsos ao longo do desenvolvimento do trabalho, assim como na implementação gráfica da representação do problema e da sua resolução. No entanto, visamos resolver quaisquer obstáculos que se apresentem.

Cremos que nos encontramos adiantados em termos de trabalho, tendo já tratado das estruturas básicas que concernem o funcionamento do jogo, e tendo desenvolvido uma versão inicial do algoritmo, procederemos a todas as iterações necessárias para alcançar a solução final.

Recursos:

Os seguintes elementos foram e serão utilizados como recursos para o desenvolvimento deste projecto:

- Visual Studio: para desenvolvimento do código, em C++ da aplicação final e implementação do algoritmo principal, destinado à resolução do problema em questão.

- SFML: biblioteca/framework utilizada para criação de ambiente GUI e elementos gráficos necessários para representação do problema e do processo da sua resolução.