INF1771 - INTELIGÊNCIA ARTIFICIAL

TRABALHO 2 – LÓGICA

Descrição:

*"Após reunir os três pingentes da virtude, Link caminha em direção a **Lost Woods** para obter a legendaria **Master Sword**. Porém, ao chegar à entrada da floresta, Link percebe que encontrar a Master Sword não será uma tarefa simples...*

***Lost Woods** é uma misteriosa floresta composta por um perigoso labirinto de caminhos, cujo mapa ninguém nunca foi capaz de desenhar. Para piorar ainda mais, a floresta é coberta por um **denso nevoeiro** que impossibilita a qualquer um ver o que existe ao seu redor.*

*Existem diversos perigos ocultos em Lost Woods, como **perigosos buracos** nos quais cairá qualquer um que tentar passar sobre eles, **Master Swords falsas** que desperdiçam a energia de quem tentar pega-las, perigosos **vórtices espaciais** que teletransportam quem se aproximar deles para qualquer outro ponto da floresta, além de perigosos **inimigos** espalhados pela floresta.*

O seu objetivo ajudar o Link a explorar Lost Woods em busca da Master Sword para que ele possa derrotar o mago Agahnim, libertar a princesa Zelda e o salvar o reino de Hyrule."

Figura 1. Master Sword em Lost Woods.

O Trabalho 2 consiste em implementar um agente baseado em conhecimento capaz de raciocinar de forma inteligente nesse ambiente desconhecido. Você deve implementar uma interface para representar visualmente esse ambiente e utilizar a linguagem Prolog para representar o conhecimento do agente.

O mapa de Lost Woods é mostrado na Figura 2.

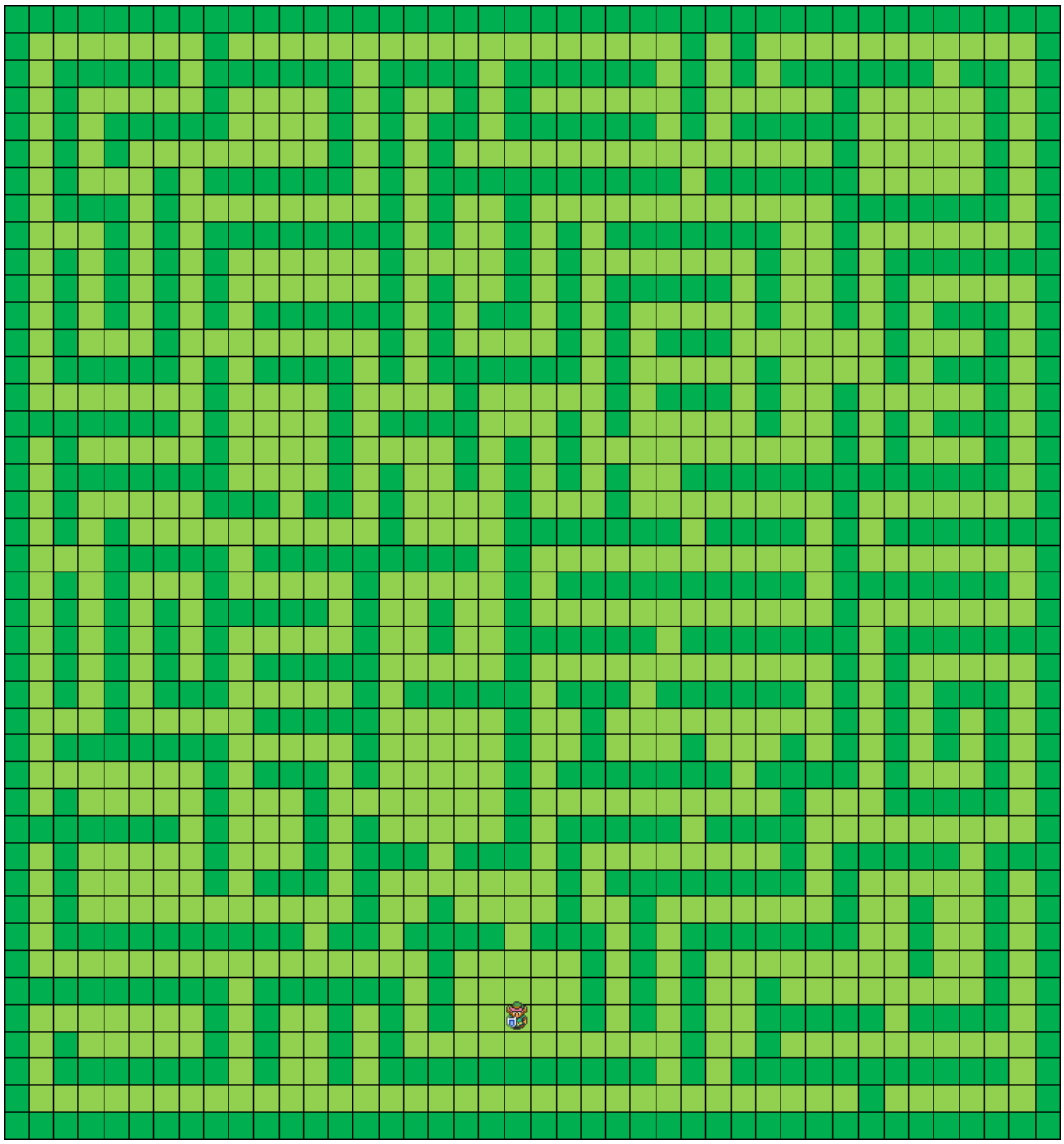

Figura 2. Mapa de Lost Woods.

Em Lost Woods existem somente 2 **tipos de terrenos**: grama (região verde claro) e floresta (região verde escuro). O agente pode passar livremente pela grama, mas não pode passar de nenhuma maneira por regiões de floresta.

Inicialmente o agente não conhece nada sobre o Lost Woods, ele deve explorar o ambiente e utilizar seus **sensores** para obter informações.

Informações Adicionais:

- O mapa deve ser representado por uma matriz 42 x 42 (igual à mostrada na Figura 2).

- Em Lost Woods existem os seguintes **elementos**:
    - **Buracos** – se o agente cair em um buraco ele morre imediatamente;
    - **Monstros** – o agente morre imediatamente se ele entrar em um local onde existe um monstro;
    - **Master Sword's Falsas** – são idênticas a Master Sword real e o agente somente fica sabendo que é falsa depois de pega-la;
    - **Vórtices Espaciais** – ao entrar em um vórtice espacial o agente é teletransportado para algum outro local em Lost Woods;
    - **Corações** – corações que são usados para recuperar a energia do agente;
    - **Rupias** – rupias que dão pontos extras para o agente;
    - **Master Sword** – a verdadeira Master Sword (principal objetivo do agente);

- O agente sempre **inicia** a jornada na entrada de Lost Woods (ponto onde está o personagem no mapa). A aventura **termina** quando o agente conseguir encontrar a verdadeira Master Sword.

- O agente pode executar as seguintes **ações**:
    - Mover para Frente;
    - Virar a Direita (rotação de 90°);
    - Virar a Esquerda (rotação de 90°);
    - Atacar – Para dar um golpe com a espada na direção em que o personagem estiver olhando;
    - Pegar Coração – Para pegar o coração utilizado para recuperar a energia do agente. A ação somente pode ser executada uma vez em cada local que exista um coração;
    - Pegar Rupia – Para pegar uma rupia existente do local onde o agente estiver. A ação somente pode ser executada uma vez em cada local que exista uma rupia;
    - Pegar Master Sword – Para pegar a Master Sword (verdadeira ou falsa). A ação somente pode ser executada em locais que exista uma Master Sword (verdadeira ou falsa);

- Cada ação executada pelo agente possui um **custo**:
    - Mover para Frente = -1;
    - Virar a Direita = -1;
    - Virar a Esquerda = -1;
    - Atacar = -5;
    - Pegar Coração = -10;
    - Pegar Rupia = +10;
    - Pegar Master Sword = -100;
    - Cair em um Buraco = -10000;
    - Ser atacado por um Monstro = -10000;

- Além do custo de cada ação, Link também possui uma determinada quantidade de **energia**. O valor inicial e máximo da energia é 100.

- Ao **atacar um mostro** com a sua espada, Link perde -10 de sua energia e o monstro é destruído.

- Ao **Pegar um coração**, Link recupera +50 de sua energia.

- O agente não tem acesso a nenhuma informação do mapa, mas ele possui alguns **sensores** para perceber o ambiente. O agente possui os seguintes sensores:
    - Em locais adjacentes a buracos, exceto diagonal, o agente sente uma leve brisa;
    - Em locais adjacentes a monstros, exceto diagonal, o agente ouve os barulhos emitidos pelo inimigo;
    - Em locais adjacentes a vórtices espaciais, exceto diagonal, o agente percebe distorções espaciais;
    - Em locais onde existe uma Master Sword (verdadeira ou falta), o agente percebe um aumento no brilho emitido pelos pingentes da virtude;
    - Em locais onde existe um coração, o agente sente a presença de fadas;
    - Em locais onde existe uma rupia, o percebe o brilho da rupia;

- O mapa tem a estrutura ilustrada na Figura 2. Mas **é desconhecida a localização** dos buracos, monstros, master swords, vórtices espaciais, corações e rupias. Sabe-se apenas que existem:
    - 10 Buracos;
    - 100 Monstros;
    - 30 Master Sword’s Falsas;
    - 10 Vórtices Espaciais;
    - 30 Corações;
    - 50 Rupias;
    - 1 Master Sword Verdadeira

- As posições dos buracos, monstros, master swords, vórtices espaciais, corações e rupias devem ser **sorteadas aleatoriamente** no inicio do programa. Mas o agente NÃO PODE ter acesso direto a essas informações.

- Os elementos (buracos, monstros, master swords, vórtices espaciais, corações e rupias) somente podem estar localizados em um terreno do tipo grama. Além disso, não pode existir mais de um elemento do mesmo tipo na mesma posição. Também não podem existir combinações dos seguintes elementos em um mesmo local: buraco, monstro, vórtice espacial e master sword verdadeira (somente pode existir um desses elementos em cada local). Durante a geração

aleatória da posição dos elementos, o seu programa deve garantir que essas regras sejam respeitadas.

- Ao **entrar em um vórtice espacial** o agente é teletransportado para uma nova posição no mapa. Essa posição deve ser um local com terreno do tipo grama sorteado aleatoriamente. Podendo ser um local onde existem buracos, monstros ou qualquer outro elemento, inclusive um novo vórtice espacial.

- O **jogo acaba** quando o agente conseguir encontrar a verdadeira Master Sword ou quando o agente morrer ao cair em um buraco, ao ser atacado por um monstro ou ao ficar sem energia.

Requisitos:

- O programa deve ser implementado em C/C++ ou Java utilizando a biblioteca do SWI-Prolog que permite acessar diretamente o Prolog. Também é permitido utilizar outras linguagens, mas antes você deve verificar se ela é compatível com o SWI-Prolog. Exemplos:
  - C# (http://www.swi-prolog.org/contrib/CSharp.html)
  - Python (http://code.google.com/p/pyswip/)
  - PHP (http://www.j-paine.org/dobbs/prolog_from_php.html)

- O Prolog deve ser utilizado somente para **representar o conhecimento do agente,** a interface visual e demais controles devem ser implementados em C/C++ ou Java.

- Não é permitido realizar nenhum processo de tomada de decisão em C/C++ ou Java, a decisão de quais ações o agente vai realizar deve ser feita exclusivamente pelo Prolog.

- Deve existir uma maneira de **visualizar os movimentos** do agente, mesmo que a interface seja bem simples. Podendo até mesmo ser uma matriz desenhada e atualizada no console.

- **O mapa de Lost Woods deve ser configurável**, ou seja, deve ser possível modificar o tipo de terreno em cada local. O mapa pode ser lido de um arquivo de texto ou deve ser facilmente editável no código.

- O programa deve exibir um **log das consultas e inserções** realizadas na base de conhecimento Prolog.

- O programa também deve **exibir a pontuação** do agente enquanto ele executa as ações. Assim como a pontuação final.

- O trabalho pode ser feito **individualmente** ou em **grupos** de no máximo 3 pessoas.

- O programa deve ser apresentado durante a aula por **todos os membros do grupo**:

    - O membro do grupo que **não comparecer** receberá nota **zero**;

    - O membro do grupo que **não souber explicar** algo relacionado ao trabalho perderá 5.0 pontos.

Dicas:

- Planeje e defina exatamente quais vão ser os predicados necessários no Prolog para codificar o conhecimento que o agente tem do mundo. Exemplos:

    - em(3, 3). - define a posição atual do agente;

    - buraco(10, 6). - identifica que existe um buraco na posição (10, 6);

    - rupia(10, 13). - identifica que existe uma rupia na posição (10, 13);

- Lembre-se de codificar predicados para identificar locais seguros e também locais visitados.

- A maneira mais simples de codificar a comunicação entre o Prolog e o C/C++ ou Java é definindo um predicado "*melhorAção*" no Prolog. Esse predicado deve retornar a melhor ação para ser executada naquele momento. Comece codificando os comportamentos mais simples, como por exemplo:

    - melhorAcao(pegar_rupia(X,Y)) :- em(X,Y), rupia(X, Y).

- A ação "*andar*" não necessariamente precisa ser para um local adjacente a posição do agente. Pode ser um "*andar*" para outro local (X, Y) ainda não visitado. Nesse caso, você pode executar o A* para calcular o melhor caminho para chegar até a posição (X, Y) passando por locais seguros, mas lembre-se de tomar cuidado com os elementos existentes no mapa e também de aplicar os custos de movimentação.

Forma de Avaliação:

Será avaliado se:

(1) O trabalho atendeu a todos os requisitos especificados anteriormente;
(2) Os algoritmos foram implementados e aplicados de forma correta;
(3) O código foi devidamente organizado;
(4) O trabalho foi apresentado corretamente em sala de aula;

Bônus:

(1) O agente que conseguir encontrar a Master Sword com o menor custo, dado uma determinada configuração de elementos (monstros, buracos, master swords, etc.), receberá 2 pontos extras na nota. Para participar dessa competição é necessário que o programa inclua uma forma simples de definir manualmente a posição dos buracos, monstros, master swords, vórtices espaciais, corações e rupias. Em caso de empate, ambos os trabalhos receberão a nota extra.

Data de Entrega:

05/05

Forma de Entrega:

O programa deve ser apresentado na aula do dia 05/05 (segunda) e enviando até o mesmo dia para o email edirlei.slima@gmail.com.

Trabalhos entregues atrasados perderam 0.5 pontos para cada dia de atraso.