

**UNIVERSIDADE FEDERAL DA PARÁ
CAMPUS UNIVERSITÁRIO DE TUCURUÍ
FACULDADE DE ENGENHARIA DE COMPUTAÇÃO
PROJETO DE ENGENHARIA II**

**CRYSLENE CÔELHO DE OLIVEIRA
GABRIEL CAMPELO GOMES
GLAUCIA CAROLINE DE SOUZA TELES
NATÁLIA FREITAS ARAÚJO**

O MUNDO DE WUMPUS: DESENVOLVENDO UM AGENTE PARA O JOGO

**TUCURUÍ-PA
2016**

1. INTRODUÇÃO

O presente relatório tem como principal objetivo expor todo o processo criativo e a construção do programa desenvolvido para o Mundo De Wumpus, com regras para o ambiente, o agente e o Wumpus. Para o ambiente criamos diferentes formatos e definição de tamanho, o agente sendo inteligente e o Wumpus, por sua vez, primeiro será parte do ambiente e depois poderá se mover, assim como também poderá ter uma memória e passar a ser um agente inteligente. Utilizamos os conhecimentos obtidos durante o período de aulas da disciplina de Estrutura de Dados e muitas pesquisas relacionadas à linguagem de programação C.

2. DESENVOLVIMENTO

O Mundo de Wumpus é um jogo que foi criado em 1972 por Gregory Yob, no qual contém um labirinto com um agente, um ouro e armadilhas, com fossos e um monstro, o Wumpus. Nesse jogo o agente deveria manter-se vivo e encontrar a saída para que o jogo terminasse, para isso teria que matar o Wumpus, pegar o ouro e desviar de morcegos que agarrava o agente e o transportava para qualquer lugar do ambiente. Dentro do ambiente o agente deve ficar atento as percepções que lhe serão ditas (brisa dos fossos, o mal cheiro exalado pelo Wumpus, o grito dos morcegos e brilho do ouro), pois é um lugar escuro e ele não enxergará o perigo que enfrentará para concluir sua missão. Durante todo esse período várias versões do Mundo De Wumpus foram feitas e a partir disso criamos nossa própria versão, onde não terá morcegos, já que fizemos em 2D, e o objetivo do agente será pegar o ouro, desviar dos fossos, matar o Wumpus com uma flecha e voltar para a sua casa inicial. Assim, iremos descrever durante este projeto em detalhes, e por parte, o que acontecerá e todo o código feito por trás do jogo.

2.1. AMBIENTE

O ambiente será uma matriz, alocada dinamicamente, cujo o tamanho e formato serão escolhidos pelo usuário. O tamanho mínimo que a matriz receberá é de 5 posições, para que possa ter um ambiente que suporte todos os elementos que

o compõem. Os formatos disponíveis para a escolha do ambiente serão os seguintes: 1 - quadrado, 2 – retangular, 3 – triangular, 4 – losangular e 5 – circular.

Para determinar o tamanho do quadrado o usuário informa o número de casas de um dos lados do quadrado. No retângulo, pede-se o número de casas da base e da altura. No triângulo apenas o número da altura, pois escolhemos trabalhar somente com o triângulo isósceles e um número exato de casas, para isso, a sua base terá o dobro da altura menos uma casa, a primeira linha começa com uma coluna e a cada linha baixo aumentamos duas colunas da anterior. No losango pedimos apenas a sua altura também, ele será construído de forma parecida com o triângulo, até a sua metade, abaixo dela será como um espelhamento da parte de cima, e acontecerá o inverso no formato das colunas, cada linha a baixo do meio será retirado duas colunas da linha anterior.

Para o círculo foi usado o seguinte raciocínio, o usuário informa o número de casas para o raio e essa matriz será dividida em quatro partes, representando os quadrantes do gráfico cartesiano e cada parte terá o tamanho do raio. Na origem todas as casas são válidas. Começamos a validação das casas pelo primeiro quadrante, verificando se a casa pertence ao círculo ou não. Utilizando a hipotenusa de um triângulo sendo esta o raio do nosso círculo, e a relação dela com o cateto oposto e cateto adjacente será de uma forma que ela sempre terá o mesmo valor. Assim, retiramos a raiz da subtração da hipotenusa ao quadrado pelo cateto oposto ao quadrado para saber o tamanho do cateto adjacente ($ca = \sqrt{h^2 - co^2}$), se esse tamanho for igual ou maior que a distância do centro desse quadrado para a origem, a casa irá pertencer ao círculo.

A partir disso aumentamos de um em um o intervalo do meio de cada quadrado, já que precisaremos sempre do meio das linhas, fazendo com que sempre conheça o valor do cateto oposto e da hipotenusa que vão gerar um valor válido para esse quadrado. Com o passar do tempo a hipotenusa vai ser rotacionada e seu tamanho vai diminuindo ao mesmo tempo que o tamanho do cateto adjacente vai aumentando, gerando um formato de semicírculo, no primeiro quadrante que depois somando com a coordenada de i e j que será invertido para completar o círculo com os outros quadrantes que foram implementados para a adaptação do círculo.

Para prosseguir com o trabalho, representamos cada elemento com uma numeração diferente, o número -1 representa o agente, o 1 o ouro, o 2 o Wumpus, o 3 os fossos, e o 8 uma casa vazia.

Como já foi mencionado acima, a cada fase o ambiente passa por pequenas mudanças, a fim de aumentar a dificuldade para o agente. Na primeira o Wumpus é fixo em uma casa, faz parte do ambiente e as bordas do ambiente tem uma especie de proteção nas duas primeiras fases. A partir da segunda fase o Wumpus se torna um outro agente, tratamos ele como o vilão do jogo; flechas poderão ser distribuidas pelo ambiente através da proporção: flechas = (fórmula da área) / 100. E na terceira fase o ambiente não possui nenhum tipo de proteção nas suas extremidades.

2.2. JOGO

Na primeira fase o agente ganha três vidas para usá-las nas três fases, inicia o jogo com uma pontuação de 1.000 pontos, e com uma flecha apenas.

A cada passo que o agente der ele perde um ponto, caso o agente queira atirar uma flecha para tentar matar o Wumpus ele perde 100 pontos, mas se ele conseguir matar o Wumpus ganha 500 pontos, e essas distribuições de pontos serão as mesmas em todas as fases. O agente apenas passa de fase se pegar o ouro e voltar para a casa inicial. A sua pontuação é acumulativa, sendo somada com um valor específico a cada fase que o agente passar. A segunda fase soma-se o acumulado mais 2.000 pontos e na terceira fase soma-se com 3.000 pontos.

A partir da segunda fase o ambiente poderá ter flechas extras, se o agente encontrar algumas dessas flechas ele poderá usa-la quando quiser e até acumular flechas, mas se o Wumpus encontrar ele quebra essa flecha extra. O Wumpus, agora o vilão do jogo possui movimentação e percepções ao seu redor, dar um passo a cada dois do agente, e caso ele caia um fosso ele não volta ao ambiente. E se o agente der um passo para fora do ambiente nas duas primeiras fases, ele apenas recebe uma mensagem, e o ambiente não o deixa cair.

Na terceira fase o agente e o Wumpus dão um passo por vez, intercaladamente, se o agente der um passo para fora das extremidades do ambiente ele perde uma vida e volta para a casa anterior à queda.

E em qualquer fase se o agente cair em um fosso ele perde uma vida e volta ao jogo na casa anterior a que ele estava do fosso, enquanto ele tiver vidas. Se ele encontrar o Wumpus o jogo acaba, independentemente do número de vidas que ele possui, e o agente perde. O agente ganha o jogo, se passar com sucesso pelas três fases.

Saindo dessa seleção seguimos para a função jogo, onde tem todas as jogadas até o agente perder ou ganhar. O jogo começa com o usuário escolhendo, se deseja vê o agente desenvolvido em ação, ou se deseja jogar por si só. Após ele decide se quer Torneio ou Treino, se o usuário escolher um torneio será vários jogados, cada um terá a sua vez, ou se ele simplesmente encerra, pois ele escolheu um treino, então é só um jogador.

A função termina registrando a pontuação, vidas, flechas e fase do jogador.

2.3. AGENTE

Para a construção da inteligência do agente foi utilizado conceitos de programação probabilística, utilizando a probabilidade para desenvolver a lógica do agente. Utilizamos uma struct e uma matriz, as quais não serão mostradas na execução do código, e que estão alocadas dinamicamente, para registrar percepções, decisões e consequências a cada passo dado pelo Agente.

A matriz do agente será como uma réplica do ambiente original, porém sem nenhum elemento inicialmente, durante a exploração do agente ele irá marcar nessa sua matriz o que encontrou, como se ele estivesse conhecendo um local e desenhando o seu mapa. Na struct temos um vetor p para guarda todas as sensações que o agente tem na atual posição, a variável m que guardará a decisão que o agente tomou, ou seja, para qual lado ele irá ou se irá atirar uma flecha, e as variáveis i e j que são as coordenadas cartesianas da casa que ele está.

A cada percepção as orientações que o agente tem para se movimentar (cima / baixo / esquerda / direita) ganha um peso o qual será fundamental para escolher a direção que ele andará. A tomada de decisão e movimentação do agente foi dividida em duas partes: antes de pegar o ouro e depois do ouro ser encontrado.

Antes de pegar o ouro, o agente prioriza andar por todo o ambiente, mas caso ele sinta uma brisa (sensação de um fosso ao seu redor) ele “pondera” a sua decisão para voltar a uma casa já conhecida e mais segura, ou seja, a casa anterior que ele estava ganhará um peso a mais. Caso ele perceba que alguma das orientações possui um fosso, ou tenha certeza disso, por já ter caído neste fosso essa coordenada ganha o peso zero, anulando a possibilidade de o agente ir para aquela casa novamente. Mas, se o agente não tem nenhuma sensação o retorno a casa anterior ganha o peso zero, uma vez que o objetivo dele neste momento é explorar o máximo possível do ambiente, até achar o ouro. Caso o agente sinta fedor (sensação de um Wumpus ao seu redor) e tenha uma flecha ou mais, ele usará o mesmo sistema de probabilidade para atirar a flecha.

Depois do agente pegar o ouro, caso ele consiga, ele irá analisar em qual coordenada ele está e fará um caminho para chegar a casa inicial. Priorizando caminhos já conhecidos, e evitando armadilhas que ele já enfrentou. Porém ele poderá fazer um novo caminho, e não apenas voltar pelo exato caminho que vez primeiramente, buscando dessa forma regressar pela menor distância. Para tal, a coordenada de cima recebe o peso zero, pois todas as casas iniciais, independentemente do formato do ambiente, estão na última linha da matriz, a orientação para cima apenas ganhará algum peso se o agente se sentir ameaçado. Os pesos para a esquerda ou direita irão depender de onde ele estiver para a coordenada da casa inicial, ele tentará ajustar a coordenada i e j até chegar a primeira casa.

Depois dos pesos distribuídos para cada orientação dividimos o espaço amostral de valor 100 pelas orientações validas. Temos variáveis que irão contar caso uma orientação seja invalidada ou se uma orientação recebe um peso a mais. O resultado dessa divisão será multiplicado pelo peso de cada orientação. Assim, teremos porcentagens para cada posição que o agente possui para se movimentar. É feito um sorteio aleatório de valores de 1 a 100, e o resultado fará parte de alguma parcela das porcentagens já determinadas, e desta forma sairá qual orientação o agente irá seguir.

2.4. WUMPUS

O Wumpus segue a mesma ideia, com diferentes objetivos. Se ele sentir o cheiro do agente ele tem uma maior probabilidade de buscar o agente, tomando um cuidado maior quando sente uma brisa, sendo mais simples a criação do código do Wumpus.

3. CONCLUSÃO

O principal objetivo deste trabalho foi a criação de um jogo de muitos desafios. Foram feitas várias reuniões, tanto via web quanto pessoalmente, para um melhor aprimoramento de um trabalho em equipe, e o domínio dos assuntos da disciplina de Estrutura de Dados foi essencial para a execução desse código. Com isso percebemos que com organização conseguimos fazer um projeto eficiente através das ferramentas utilizadas. A conclusão deste trabalho foi bastante satisfatória, criando o Mundo de Wumpus de acordo com todos os desejos iniciais que tivemos desde o início da proposta do jogo, ultrapassando limites e com dedicação.

4. REFERÊNCIAS BIBLIOGRÁFICAS

Portal do Instituto de Matemática e Estatística da Universidade de São Paulo. Disponível <<http://www.ime.usp.br/~leliane/LabVIA/historia.htm>>. Acesso em 20 de Julho de 2016.

Pontifícia Universidade Católica de São Paulo. Disponível <http://www.dbd.puc-rio.br/pergamum/tesesabertas/0321287_08_cap_03.pdf>. Acesso 18 de Julho de 2016.