UD3 - Unity Decision Tree Integrator: Uma extensão de Árvores de Decisão para Unity

Andrei A. Müller[1], João R. Bittencourt[2]

Universidade do Vale do Rio dos Sinos (UNISINOS)
Caixa Postal 93.022-000 – São Leopoldo – RS – Brasil

Abstract

This paper describes the proposed plug-in for the Unity game engine: UD3 – Unity Decision Tree Integrator, that aims to help any Unity developer to apply Artificial Intelligence (AI) in the projects developed with the engine. The paper presents the main goals and the justification of the project, followed by a contextualization about Decision Trees. Then, the paper presents info about some AI solutions already released for Unity. A proposed implementation for UD3 and comments about the project developed to date are shown next. Finally, the paper describes the final considerations about the first stage of the project and the next steps that will be made in its development.

Resumo

Este artigo descreve o *plug-in* proposto para o motor de jogo Unity: UD3 – Unity Decision Tree Integrator, que visa facilitar o trabalho do desenvolvedor Unity quanto ao uso de técnicas de Inteligência Artificial (IA) nos projetos criados no motor. São apresentados os objetivos principais e a justificativa do projeto, seguido de uma contextualização sobre Árvores de Decisão. Em seguida, são apresentadas informações sobre algumas soluções de IA já lançadas para Unity. A seguir, é mostrada uma proposta de implementação para o UD3 e é comentado o projeto desenvolvido até o presente momento. Por fim, o artigo descreve as considerações finais sore a primeira fase do projeto e os próximos passos que serão feitos em seu desenvolvimento.

Palavras-chave: Inteligência Artificial, Árvores de Decisão, Unity

Contato dos autores:
[1] andreiagmu@gmail.com
[2] joaorb@unisinos.br

1. Introdução

Inteligência Artificial (IA) é um módulo de grande importância no desenvolvimento de Jogos Digitais e possui grande influência na imersão do jogador [Ram, Ontañón et al. 2007]. Uma IA bem planejada é essencial para que as entidades de jogo chamadas Agentes Autônomos [Russell e Norvig 2009] interajam com o jogador escolhendo as ações mais apropriadas para o contexto do jogo (por exemplo: pular, correr, atirar), de forma que não impacte negativamente na imersão [Ram, Ontañón et al. 2007].

O motor de jogo Unity é um motor muito utilizado por desenvolvedores independentes e possui diversas ferramentas e funcionalidades prontas, por exemplo, gráficos, som, física que auxiliam o desenvolvedor de jogos para criar mais rapidamente seu projeto, sem precisar programar desde o início tais módulos [Gregory 2009]. Também é possível criar e adquirir novas extensões para o motor, chamadas de *plug-ins*, que adicionam mais funcionalidades ao motor original. Estes *plug-ins* estão disponíveis para o usuário no *website Unity Asset Store*, sendo o download das extensões gratuito ou pago, dependendo do *plug-in*.

A criação, testes e aplicação de uma IA robusta para um projeto de jogo digital pode consumir boa parte do tempo de desenvolvimento do jogo. Logo, há a necessidade de ferramentas que auxiliem e tornem mais rápido esse processo. Este artigo propõe a criação de uma ferramenta desse tipo, o *plug-in* "UD3 – Unity Decision Tree Integrator", que integrará ao motor Unity ferramentas que visam auxiliar na criação e aplicação de IA que utilizem árvores de decisão nos projetos criados no motor.

Um importante foco do UD3 é a implementação de um algoritmo de Aprendizagem de Máquina (*Machine Learning*). Mitchell [1997] define que um programa de computador "aprende" quando seu desempenho melhora à medida que adquire experiência. A implementação proposta do UD3 usará algoritmos baseados em árvores de decisão, como o ID3 [Quinlan 1986].

Desenvolver um sistema de Inteligência Artificial robusto para um projeto de jogo digital é uma tarefa árdua e há grande potencial de mercado para ferramentas que facilitem a criação e a integração de sistemas de IA no motor de jogo. Na *Unity Asset Store* existem diversos *plug-ins* relacionados com IA, como o *coAdjoint Orbit*, o gratuito *RAIN{indie}* e diversos outros *plug-ins* comerciais ou gratuitos, que implementam diferentes técnicas, por exemplo, *path finding* (*Simply A**), Redes Neurais Artificiais (*Neural Nets Source Code – Tong Framework*), o que comprova o potencial de mercado deste tipo de

extensão. Entretanto nenhum desses implementa a técnica de árvores de decisão.

Este artigo está organizado em 5 seções. Na seção 1 são apresentados o tema principal do artigo, as delimitações e os objetivos do projeto e uma breve justificativa do potencial mercadológico do UD3; na seção 2 são feitas explicações sobre Árvores de Decisão, os algoritmos ID3 e a possibilidade de uso do ID3 em um projeto criado no motor de jogo Unity; na seção 3 é apresentada uma proposta de modelo de implementação para o UD3, descrevendo a arquitetura e as classes do projeto; a seção 4 mostra como que o modelo proposto está sendo implementado e descreve o que já funciona no UD3; por fim, na seção 5 são feitas considerações sobre o atual progresso do projeto e os próximos passos do desenvolvimento.

2. Árvores de Decisão

De acordo com Mitchell [1997, p. 52, tradução do autor]: *"A aprendizagem com árvore de decisão é um dos métodos mais utilizados e práticos para inferência indutiva. A aprendizagem com árvore de decisão é um método para aproximar funções alvo com valores discretos, em que a função aprendida é representada por uma árvore de decisão"*.

A árvore de decisão, como mostra a Figura 1 [Mitchell 1997, p. 53], é uma estrutura semelhante a um fluxograma, com nodos e galhos, onde cada nodo interno representa um teste feito num determinado atributo, cada galho representa uma consequência do teste e cada nodo-folha representa um rótulo de classe, isto é, a decisão final predita após o teste de todos os atributos. O caminho percorrido entre o nodo-raiz da árvore e um nodo-folha qualquer representa uma regra de classificação, que é o conjunto formado pelos atributos e por um rótulo de classe.

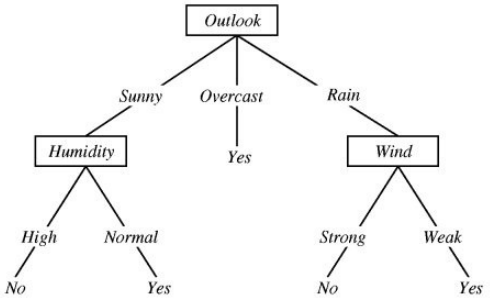

Figura 1: Exemplo simples de Árvore de Decisão

O algoritmo ID3, desenvolvido por Quinlan [1986], é um algoritmo tipicamente utilizado para Aprendizagem de Máquina (*Machine Learning*). O ID3 possui a funcionalidade de gerar uma árvore de decisão a partir de uma base de dados. Nessa base constam diversos exemplos (também chamados de instâncias), representados por pares atributo-valor, sendo o último atributo a classe (classificação prevista) do exemplo. O algoritmo itera entre cada atributo e calcula sua **entropia** (ou seu ganho de informação). Em seguida, o algoritmo seleciona o atributo com a menor entropia ou o maior ganho de informação, e subdivide a base de dados em relação ao valor deste atributo que cada exemplo possui (por exemplo: clima = ensolarado, clima = chuvoso, clima = nublado). O algoritmo então recursa em cada subdivisão da base de dados, iterando entre cada atributo não selecionado anteriormente; se todos os elementos de uma determinada subdivisão pertencem à mesma classe, é gerado um nodo-folha que é rotulado com a classe dos elementos da subdivisão. O algoritmo é executado de forma recursiva até não sobrar mais atributos para serem selecionados. Cada subdivisão na base de dados gera um nodo interno referente ao atributo divisor e novos galhos correspondentes aos possíveis valores deste atributo.

2.1 Utilização do ID3 em um projeto Unity

Em um projeto de jogo, um Agente Autônomo (por exemplo: um inimigo, adversário do jogador) [Russell e Norvig 2009] possui diversas variáveis internas e métodos (funções) que descrevem seu estado atual e fazem o seu controle (por exemplo: variáveis "vida", "munição"; métodos "mirar no inimigo", "atirar", "fugir do inimigo"). Além disso, o agente pode observar variáveis externas, que descrevem o mundo do jogo e observar o estado de outros objetos que existem nesse mundo (por exemplo: outros agentes, o personagem controlado pelo jogador, *waypoints* definidos no mapa). Os atributos da base de dados podem ser definidos como as variáveis citadas (do agente, do mundo do jogo e de outros objetos) e a classe pode representar a ação/decisão que o agente deverá tomar, levando em consideração os valores atuais dos atributos. Logo, os rótulos de classe representam as possíveis ações (métodos) que o agente pode executar em determinado momento do jogo.

Sendo assim, é possível utilizar árvores de decisão na IA do agente: primeiro é criada uma base de dados, por exemplo, usando informações obtidas de partidas de jogadores humanos. O agente consulta a base de dados e executa o algoritmo do ID3 para criar uma árvore de decisão e extrair regras a partir dessa árvore. Então, o agente utiliza as informações de entradas obtidas pelos seus sensores, como *raycasts* e a área ao seu redor para perceber o estado atual do ambiente (mundo do jogo) e o estado atual de suas variáveis internas. Com essas informações, o agente faz uma inferência sobre as regras extraídas da árvore para decidir qual ação ele deverá tomar.

3. Proposta de Implementação do UD3

O módulo UD3 é o principal módulo do protótipo, onde será feita a ligação entre os métodos de Árvores de Decisão do UD3 com os controladores de IA dos agentes autônomos (objetos do jogo em si). Como mostrado na Figura 2, o módulo UD3 depende dos módulos ID3 e C4_5, nestes dois serão implementados os algoritmos de IA baseados em Árvores de Decisão.

No módulo *UD3Editor* será feita a integração do *plug-in* com o editor da Unity, para facilitar a aplicação das Árvores de Decisão nos objetos do jogo.

Quando o jogo é compilado, são usadas as DLLs (bibliotecas de vínculo dinâmico) UD3.dll, ID3.dll e C4_5.dll, que farão a ligação das funções do UD3 para o arquivo executável do jogo, com exceção das funções necessárias apenas pelo editor da Unity, contidas na biblioteca UD3Editor.dll.

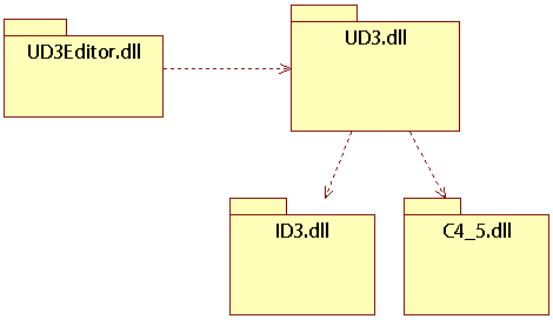

Figura 2: Diagrama base dos módulos do UD3

4. Implementações iniciais

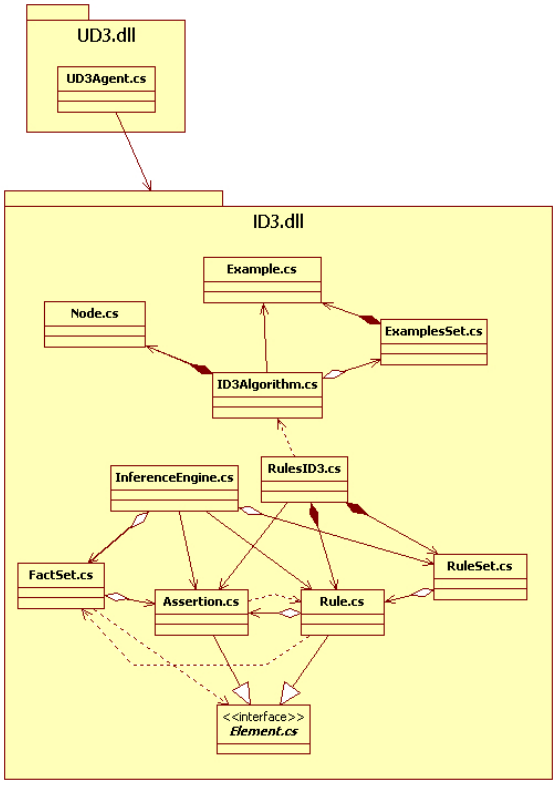

Figura 3: Diagrama UML ilustrando o relacionamento entre as classes já implementadas do UD3

A implementação do UD3 está sendo desenvolvida em em C#, no ambiente de desenvolvimento MonoDevelop integrado com a Unity. A versão atual possui os projetos ID3 e UD3 com classes já implementadas (Figura 3).

As principais classes já definidas do protótipo são: *ID3Algorithm*, que possui a implementação do algoritmo do ID3 para construir a árvore de decisão; *Node*, que define um nodo usado pelo ID3; *ExamplesSet*, que define a base de dados como uma lista de exemplos, construídos na classe *Example*; *RulesID3*, que carrega uma árvore de decisão e extrai regras a partir dela; *InferenceEngine*, que executa a inferência a partir de uma lista de regras (implementada pelas classes *RuleSet* e *Rule*) e de uma lista de fatos (implementada pelas classes *FactSet* e *Assertion*); a classe abstrata *Element*, que contém métodos relacionados com o valor de confiança de um fato, que define se um fato é verdadeiro, falso ou duvidoso; e a classe *UD3Agent* do módulo UD3, que controla a execução dos métodos de árvores de decisão e de inferência pelo agente, durante o jogo.

Primeiramente, foram implementadas as classes *ID3Algorithm*, *Node*, *Example*, *ExamplesSet* e *UD3Agent*. Com estas classes prontas foi possível testar uma compilação inicial das DLLs do projeto, que são salvas no diretório *Assets\Plugins* do projeto Unity usado para testar o UD3. A classe *UD3Agent* da biblioteca UD3.dll é a classe que faz os algoritmos de árvores de decisão funcionarem no jogo e faz a ligação entre as funções implementadas do UD3 e o objeto de jogo (*Game Object*) da Unity. Testando essa classe, foi possível nesse ponto gerar uma árvore de decisão, a partir de uma base de dados.

Após esse teste inicial, foram implementadas as classes *InferenceEngine*, *RulesID3*, *Rule*, *RuleSet*, *FactSet*, *Element* e *Assertion*. Então, a classe *UD3Agent*, após gerar a árvore a partir da base de dados, inicializa uma lista de regras (criando as regras a partir da árvore de decisão gerada), uma lista de fatos e o motor de inferência. Como teste, foram adicionados alguns fatos na lista de fatos, então o motor de inferência comparou os novos fatos com as regras já definidas e fez a inferência, mostrando seu resultado na tela.

Na versão atual do projeto, a árvore de decisão e o motor de inferência já funcionam com variáveis e métodos criados pelo usuário, em um jogo desenvolvido na Unity. O usuário deve primeiro adicionar a classe *UD3Agent* como um componente dos *Game Objects* referentes aos agentes (por exemplo: inimigos no mapa); neste componente o usuário pode definir para cada agente seus Sensores (variáveis que serão monitoradas pelo agente, podendo ser referentes à *scripts* do agente ou de outros *Game Objects* – por exemplo: a vida atual do agente ou do personagem controlado pelo jogador) e Ações (possíveis métodos

que o agente deverá chamar após a execução do motor de inferência – por exemplo: Atacar, Fugir; o método a ser chamado depende do resultado do motor de inferência). No mesmo componente, o usuário pode carregar um arquivo de texto com a base de dados a ser usada pelo algoritmo e definir os rótulos (*labels*) referentes aos atributos e às classes contidos na base de dados. Tendo esses dados corretamente definidos, o usuário deve executar o jogo.

Após o jogo ser carregado são inicializadas a árvore de decisão e a lista de regras. Então, a cada *frame* do jogo o algoritmo executa a inferência considerando o valor atual dos sensores e a lista de regras, e a partir do resultado da inferência, o algoritmo decide qual ação o agente deverá tomar naquele *frame*.

Para testar o funcionamento do UD3, foi desenvolvido um jogo bastante simples, no estilo *First Person Shooter* (Tiro em Primeira Pessoa), onde o jogador se encontra em um cenário com alguns inimigos (agentes que usam o UD3) e diversos itens de cura. O jogador pode atirar nos inimigos, fazendo-os perder "vida". A base de dados é formada por alguns exemplos definidos pelos atributos "vida do agente" e "vida do jogador" (durante o jogo, ambos os atributos podem ter os valores "fraco", "médio" ou "forte"); e cada exemplo é classificado com uma ação: Atacar ou Fugir. Em um *script* dos inimigos, foram definidos alguns métodos que fazem o inimigo tomar ações no mapa (ir atrás do jogador, fugir do jogador, buscar item de cura, etc.), mas estes métodos não são chamados diretamente por esse *script* – cabe ao UD3 decidir qual método será chamado. E após ser executada a inferência com as regras extraídas da base de dados, é possível acompanhar o seu resultado: com a base criada inicialmente, quando os inimigos estão com sua vida elevada, eles perseguem o jogador para atacá-lo (o UD3 chama o método definido pelo usuário para a ação Atacar); quando algum inimigo fica com a vida baixa, ele foge do jogador e vai atrás de um item de cura (este foi o método definido para a ação Fugir). Por fim, se o usuário modificar a base de dados (por exemplo: alterando o valor dos atributos ou a classe de cada exemplo; adicionando ou removendo exemplos), isso altera a tomada de decisões do agente durante o jogo, pois as regras extraídas pelo algoritmo do UD3 podem se tornar completamente diferentes, dependendo das modificações feitas na base.

5. Considerações finais

Foi possível programar um protótipo do UD3 com os métodos de *machine learning* (criação das árvores de decisão usando o ID3, extração das regras da árvore, inferência) funcionando na Unity. Também foi implementado o uso de sensores do agente para a criação de fatos a serem inferidos pelo motor de inferência, com isso já é possível testar e ver os resultados do UD3 dentro do jogo. O próximo passo é implementar o algoritmo C4.5 [Quinlan, 1993] no UD3, para disponibilizar ao usuário mais uma alternativa de métodos para *machine learning*, através do *plug-in*.

Será continuada a integração do UD3 com o editor da Unity, com o objetivo de tornar o *plug-in* mais acessível e intuitivo para o usuário final. Para validar o resultado do projeto, em um determinado período do desenvolvimento serão feitos testes com usuários, que serão solicitados a criar um ou mais agentes autônomos em um projeto de Unity proposto pelo autor. Será disponibilizado um formulário onde os testadores poderão dar *feedback* de sua experiência ao usar o *plug-in*, relatar *bugs* encontrados ou dificuldades para usar determinada ferramenta ou função. Como grupo de controle alguns testadores não vão receber o UD3 e deverão implementar desde o inicio o módulo de IA. Dois grupos de teste serão criados: um receberá o UD3 sem nenhuma explanação e o outro irá usar somente as demonstrações e a documentação técnica como base. Espera-se que o grupo que receberá explicações e o UD3 tenha um desempenho melhor e mais eficaz do que o grupo que não receba o *plug-in*. Após os testes com o grupo de usuários, no caso do UD3 obtiver um resultado satisfatório, serão feitos os procedimentos necessários para lançar o *plug-in* na *Unity Asset Store*.

Referências

coAdjoint Orbit by coAdjoint Limited [online]. Disponível em: http://goo.gl/oTIjZU [Acessado 18 Jun. 2014].

GREGORY, J., 2009. *Game Engine Architecture.* A K Peters / CRC Press.

MITCHELL, T. M., 1997. *Machine Learning.* Boston: McGraw-Hill.

Neural Nets – Tong Framework [online]. Disponível em: http://goo.gl/eFYplP [Acessado 18 Jun. 2014].

Neural Nets Source Code – Tong Framework [online]. Disponível em: http://goo.gl/sk8ZPb [Acessado 18 Jun. 2014].

Neural Network AI [online]. Disponível em: http://goo.gl/QUUFnD [Acessado 19 Jun. 2014].

QUINLAN, J. R., 1986. Induction of Decision Trees. *Machine Learning*, 1, 81-106.

QUINLAN, J. R., 1993. *C4.5 : programs for machine learning*. Morgan Kaufmann, San Mateo.

RAIN{indie} by Rival{Theory} [online]. Disponível em: http://goo.gl/aJq6mF [Acessado 18 Jun. 2014].

RAM, A.; ONTAÑÓN, S.; MEHTA, M., 2007. Artifical Intelligence for Adaptive Computer Games. *In: FLAIRS Conference 2007*. 22-29.

RUSSELL, S. J.; NORVIG, P., 2009. *Artificial Intelligence: A Modern Approach*. 3. ed. Upper Saddle River, New Jersey: Prentice Hall.

*Simply A** [online]. Disponível em: http://goo.gl/hUvp2z [Acessado 18 Jun. 2014].

Unity – Game Engine [online]. Disponível em: https://unity3d.com/ [Acessado 18 Jun. 2014].