ABORDAGEM DE DECISION TREES NO CONTEXTO DE GERAÇÃO DE REGRAS PARA JOGOS COM UNITY 3D ENGINE

Addam Cauê Peres RAFACHO , Prof. MSc. Guilherme de Cleva FARTO

addamcaue@hotmail.com , guilherme.farto@gmail.com

Instituto Municipal de Ensino Superior de Assis (IMESA)
Fundação Educacional do Município de Assis (FEMA) – Assis/SP (Brasil)

RESUMO: Mecanismos de Inteligência Artificial têm sido utilizados em distintas áreas da Ciência da Computação. Recursos de classificação como, por exemplo, Árvores de Decisão contribuem com o formalismo e a definição de modelos matemáticos e lógicos na avaliação e geração de regras em sistemas computacionais. Este trabalho tem como objetivo propor e implementar uma abordagem composta por uma API e uma arquitetura que utiliza Árvores de Decisões e o algoritmo ID3 na geração de regras em jogos desenvolvidos na plataforma Unity 3D Engine. Como resultados, a API implementada em Java auxilia no processamento do algoritmo ID3, resultando em árvores de decisões baseadas em JSON que podem ser utilizadas para dinamizar o sistema de regras em jogos computacionais. Também se destaca a possibilidade de estender e evoluir a arquitetura proposta para ser adotada em distintos contextos, além de jogos e aplicações gráficas.

PALAVRAS-CHAVE: Árvores de Decisão, Algoritmo ID3, Unity 3D Engine, API, Java

ABSTRACT: Mechanisms of Artificial Intelligence have been used in different areas of Computer Science. Classification features such as Decision Trees contribute to the formalism and definition of mathematical and logical models in the evaluation and generation of rules in computational systems. This work aims to propose and implement an approach composed by an API and an architecture that uses Decision Trees and the algorithm ID3 in the generation of rules in games developed in the platform Unity 3D Engine. As results, the API implemented in Java assists in the processing of the ID3 algorithm, resulting in JSON-based decision trees that can be used to dynamize the rule system in computational games. It also highlights the possibility of extending and evolving the architecture proposed to be adopted in different contexts, in addition to games and graphic applications.

KEYWORD: Decision Trees, ID3 Algorithm, Unity 3D Engine, API, Java

1. Introdução

O mercado de computação gráfica e de jogos, cada vez mais, tem incorporado profissionais e empresas de distintas áreas como músicos, *designers*, engenheiros eletrônicos e outros (PARKER, 2012).

A criação de motores de jogos (em Inglês, *Game Engine*) tornou o desenvolvimento de jogos mais acessível, contribuindo com o aumento no número de profissionais e empresas da área. Entretanto, a diversão nos jogos pode ser facilmente afetada pela sua dificuldade e mecânica de jogo.

Desta forma, a Inteligência Artificial (IA) pode contribuir com o contexto de jogos (HUNICKE; CHAPMAN, 2004). Outro exemplo é a adoção da análise de dados por IA que pode tornar ajustável a dificuldade do jogo de acordo com as experiências e informações gerais das ações passadas de um jogador (ZOOK; RIEDL, 2012).

Este trabalho propõe o uso de Árvores de Decisões, uma subárea de IA, destacando-se o algoritmo ID3, proposto por QUINLAN (1986), como mecanismo para geração de regras em jogos desenvolvidos na plataforma Unity 3D Engine.

2. Objetivos

Esta pesquisa de iniciação científica pretende, como objetivos gerais, (i) investigar os fundamentos das áreas de Inteligência Artificial, focando-se em Árvores de Decisões (em Inglês, *Decision Trees*), bem como (ii) implementar e aplicar os conceitos do algoritmo ID3 no contexto de desenvolvimento de jogos com a plataforma Unity 3D Engine.

Para alcançar tais objetivos gerais, definiram-se os seguintes objetivos específicos:

- Explorar os conceitos de Árvores de Decisões e algoritmo ID3;
- Explorar as plataformas Java e Unity 3D Engine;
- Implementar uma *Application Program Interface* (API) em Java com base nos recursos de Árvores de Decisões e algoritmo ID3;
- Propor e implementar uma arquitetura que integre a API implementada para ID3 e a plataforma Unity 3D Engine, possibilitando a geração de regras e um mecanismo mais inteligente quanto às definições de um jogo.

3. Metodologia

A proposta e objetivos definidos nesta pesquisa científica forma alcançados por meio de uma metodologia inicialmente amparada pela revisão bibliográfica em fontes confiáveis como artigos técnico-científicos, monografias e livros.

Após a investigação dos trabalhos relacionados aos assuntos desta pesquisa, uma API baseada no algoritmo ID3 foi proposta e implementada na plataforma Java. Por meio desta biblioteca, os conceitos do algoritmo ID3 podem ser executados a partir das funcionalidades expostas pela API, destacando-se os cálculos de entropia e ganho, bem como a geração da própria árvore de decisão final.

4. Revisão da Literatura

4.1. Inteligência Artificial

À humanidade, deu-se o nome científico *homo sapiens*, homem sábio, devido as nossas capacidades mentais que são tão importantes para nossa vida cotidiana e senso de si mesmo. O campo da Inteligência Artificial (IA) busca compreender as entidades inteligentes. Assim, desta forma, uma razão para o estudo da IA são as descobertas sobre nós mesmos. Mas, diferente da filosofia e psicologia que também se preocupam com o estudo da inteligência, a IA se esforça para construir entidades inteligentes, bem como compreendê-las (RUSSELL, 1995).

Para MCCARTHY (2007), IA é a ciência e a engenharia de fazer máquinas inteligentes, especialmente programas de computador inteligentes. Relaciona-se com a tarefa semelhante de usar computadores para entender a inteligência humana, mas que não precisa se limitar a métodos biologicamente observáveis. Segundo COPELAND (2000), IA geralmente é definida como a ciência de fazer computadores que realizam tarefas que exigem inteligência quando realizadas por seres humanos.

Segundo RUSSEL (1995), as definições de IA, encontradas em diversos livros que abordam o tema, podem ser agrupadas em quatro categorias principais:

- (a) Sistemas que pensam como humanos;
- (b) Sistemas que agem como humanos;
- (c) Sistemas que pensam logicamente;
- (d) Sistemas que agem logicamente.

Todas definições preocupam-se com os processos de pensamento e raciocínio, porém apresentam diferentes formas para medir o desempenho da IA. As definições “a” e “b” medem o sucesso em termos de desempenho humano, enquanto as definições “c” e “d” medem de acordo com um conceito ideal de inteligência, denominado racionalidade.

4.2. Árvores de Decisões

Grande parte das aplicações de relevância prática em inteligência artificial baseia-se na concepção de modelos computacionais do conhecimento aplicado por um especialista humano. Na síntese de modelos de classificação, a associação entre as classes e o conjunto de atributos que caracterizam os objetos a serem classificados pode se dar de variadas formas, empregando processamento simbólico e/ou numérico (ZUBEN; ATTUX, 2007).

Segundo QUINLAN (1986), Árvores de Decisão visualizam o domínio da tarefa como uma classificação. A estrutura subjacente consiste em uma coleção de atributos ou propriedades que são utilizadas para descrever casos individuais, cada caso pertencente a um exato conjunto de classes. Os atributos podem ser contínuos ou discretos. O valor de um caso de um atributo contínuo é sempre um número real, enquanto seu valor de um atributo discreto é um pequeno conjunto de valores possíveis para esse atributo.

Árvores de Decisão são modelos estatísticos que utilizam um treinamento supervisionado para a classificação e previsão de dados. Em outras palavras, em sua construção é utilizado um conjunto de treinamento formado por entradas e saídas (SILVA, 2005).

Uma árvore de decisão recebe, como entrada, um objeto ou situação descrita por um conjunto de propriedades e produz uma “decisão” como, por exemplo, “sim” ou “não”. Cada nó interno na árvore corresponde a um teste do valor de uma das propriedades e os ramos do nó são rotulados com os possíveis valores do teste. Cada nó folha na árvore especifica o valor booleano, verdadeiro ou falso, a ser retornado se essa folha for atingida (RUSSEL, 1995).

4.3. Classificação em Árvore de Decisão

Uma árvore de decisão utiliza uma estratégia de dividir para conquistar, onde um problema complexo é decomposto em subproblemas mais simples e recursivamente a mesma estratégia é aplicada a cada subproblema (QUINLAN, 1986).

Segundo exemplo apresentado por BASGALUPP (2010), a fim de se ilustrar o funcionamento básico de uma árvore de decisão, pode-se considerar o problema para diagnóstico de pacientes de acordo com a Figura 1.

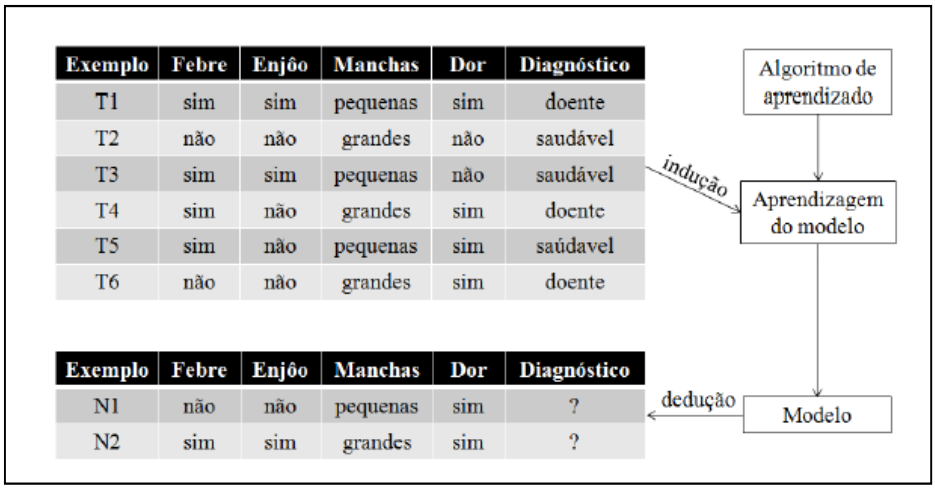

Figura 1. Exemplo de dados de treinamento (*Training Data*)

Supondo que um novo paciente chegue ao consultório médico, com a finalidade de diagnosticar o paciente, a primeira pergunta que pode ser realizada é se o paciente tem sentido dor. A seguir, diferentes perguntas podem ser realizadas. Como por exemplo, (i) se o paciente apresenta febre ou enjoos ou (ii) ainda se tem notado alguma mancha no corpo. Isto caracteriza uma forma para solucionar um problema de classificação por intermédio de perguntas referentes a uma série de características, que no caso é o paciente. A cada pergunta respondida, outra pode ser realizada até que se chegue a uma conclusão sobre a classe a que pertence o exemplo. Tal série de perguntas e suas possíveis respostas podem ser organizadas na representação de uma árvore de decisão composta de forma hierárquica por nodos, ou nós, e arestas (BASGALUPP, 2010).

Na Figura 2 é ilustrada uma possível árvore de decisão para o problema de diagnosticar pacientes, que podem ser classificados como “saudável” ou “doente”.

Dessa forma, torna-se possível a utilização de árvore de decisão para classificação de um paciente como saudável ou doente. Basta partir do nodo raiz da árvore e percorrê-la por meio das respostas aos testes dos nodos internos até que se chegue em um nodo folha, o qual indica a classe correspondente ao paciente. Além da obtenção da classe, a grande vantagem que a trajetória percorrida até o nodo folha representa uma regra, facilitando a interpretação do modelo pelo usuário, no caso um médico (BASGALUPP, 2010).

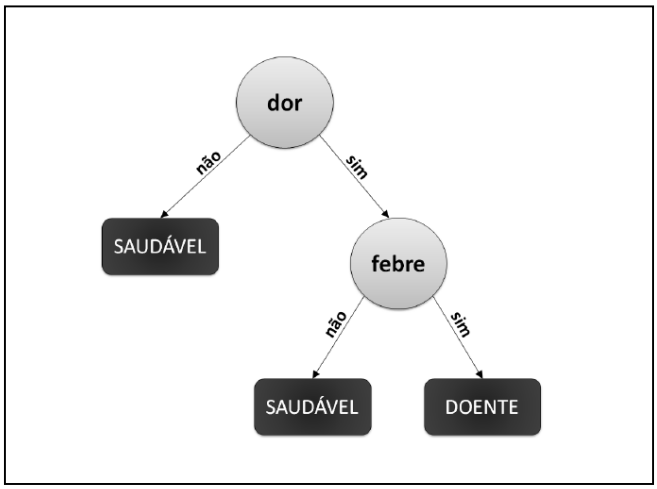

Figura 2. Exemplo de árvore de decisão (BASGALUPP, 2010)

4.4. Entropia e Ganho de Informação

De acordo com BASGALUPP (2010), grande parte dos algoritmos de indução de árvores de decisão utiliza-se de funções de divisão univariável, ou seja, cada nodo interno da árvore é dividido de acordo com um único atributo. Assim, o algoritmo tenta encontrar o melhor atributo para realizar essa divisão. Os critérios de seleção para melhor divisão são baseados em medidas, tais como (i) impureza, (ii) distância e (iii) dependência. A maior parte dos algoritmos de indução busca dividir os dados de um nodo de forma a minimizar o grau de impureza dos nodos filhos.

Uma das medidas baseadas na impureza é o Ganho de Informação, o qual usa a Entropia como medida de impureza. O algoritmo ID3 (QUINLAN, 1986), pioneiro em indução de árvores de decisão utiliza essa medida. Para determinar o quão boa é uma condição de teste realizada, é necessário comparar o grau de entropia do nodo pai, antes da divisão, com o grau de entropia dos nodos filhos, após a divisão. O atributo que resultar na maior diferença é escolhido como condição de teste na árvore (BASGALUPP, 2010).

4.5. Algoritmo ID3

O algoritmo ID3, desenvolvido por Quinlan em 1986, foi um dos primeiros algoritmos de árvore de decisão, tendo como base sistemas de inferência e em conceitos de sistemas de aprendizagem. O algoritmo constrói árvores de decisão a partir de um dado conjunto de exemplos, também chamado de *Training Data*, sendo a árvore resultante usada para classificar amostras futuras (GARCIA; ALVARES, 2001).

No algoritmo ID3, cada regra é um conjunto de instâncias que, por sua vez, possui certos atributos discretos. Um exemplo de instância pode ser “luz da sala” cujos atributos são “acesa” ou “apagada”. Outro exemplo pode ser a instância “tempo”, cujos atributos são “manhã”, “tarde” e “noite” (TAKIUCHI et al., 2004).

O ID3 é um algoritmo recursivo que realiza uma busca Heurística sobre um conjunto de atributos, procurando aqueles que melhor dividem os exemplos gerando sub árvores. A principal limitação apresentada pelo ID3 é que sua capacidade se restringe a lidar com atributos categóricos, não sendo possível aplica-lo a conjuntos de dados com atributos contínuos. Neste caso, os atributos contínuos devem ser previamente discretizados. Além desta limitação, o ID3 também não apresenta nenhuma forma para o tratamento de valores desconhecidos, ou seja, os exemplos do conjunto de treinamento devem ter valores conhecidos para seus atributos. Dessa forma, necessita-se de uma boa quantidade de tempo para o pré-processamento dos dados com ID3 (BASGALUPP, 2010).

O ID3 utiliza o ganho de informação para selecionar a melhor divisão. Entretanto, esse critério não leva em consideração o número de divisões (número de arestas), e isso pode acarretar em árvores mais complexas (BASGALUPP, 2010).

4.6. Unity 3D Engine

O Unity 3D Engine apresenta-se como um motor de desenvolvimento de jogos (em Inglês, *Game Engine*). Para que uma ferramenta seja considerada um motor de jogos completo, módulos e funcionalidades auxiliares se tornam necessárias. Em especial, um sistema de renderização 3D e um sistema de simulação física são fundamentais. Além disto, necessita-se de uma boa arquitetura para a programação de *scripts*, um editor de cenas integrado, bem como a capacidade de se importar modelos 3D, imagens e efeitos de áudio produzidos em ferramentas externas. É desejável que os jogos desenvolvidos possam ser distribuídos em múltiplas plataformas (PASSOS et al., 2009).

O motor de jogos Unity 3D Engine abstrai do desenvolvedor de jogos a necessidade de utilizar DirectX ou OpenGL diretamente, suportando a criação de elementos da computação gráfica complexos com a linguagem Cg da NVidia. Internamente, o sistema de simulação física é o popular PhysX, também da NVidia. Para a execução de scripts, o Unity 3D Engine utiliza uma versão de alto desempenho da biblioteca Mono, implementação do *framework* .NET da Microsoft (PASSOS et al., 2009).

4.7. JSON

JSON (em Inglês, *JavaScript Object Notation*) é um formato leve para intercâmbio de dados. Apresenta fácil leitura e escrita por humanos e máquinas. É baseado em um subconjunto da linguagem de programação *JavaScript*. JSON é um formato de texto que é completamente independente da linguagem, mas usa convenções que são familiares aos programadores da família C, incluindo C, C++, Java, JavaScript, Perl, Python e outros. Essas propriedades tornam o JSON uma linguagem ideal para intercâmbio de dados (JSON, 2016). Os Apêndices A e B apresentam fragmentos de arquivos JSON.

JSON é constituído de duas estruturas (JSON, 2016):

- **Uma coleção de pares de nome/valor**: Em várias linguagens, isto é realizado como um objeto, registro, estrutura, dicionário, lista ou matriz;
- **Uma lista ordenada de valores**: Na maioria das linguagens, isto é realizado como uma matriz, vetor, lista ou sequência.

Um arquivo JSON apresenta uma estrutura de dados universal. Assim, o formato de dados tem sido adotado por ser intercambiável entre plataformas e linguagens de desenvolvimentos (JSON, 2016).

4.8. Java

A história do Java iniciou-se em 1991, quando um grupo de engenheiros da *Sun*, denominados "*Green Team*", liderados por James Gosling, criaram uma linguagem de programação portável. Em 1995, com o anúncio da incorporação ao navegador de *Internet Netscape*, a tecnologia e então plataforma Java expandiu (LINDHOLM; YELLIN, 2011).

Em uma linguagem de programação, ao compilar um programa, o código fonte é transformado em código de máquina específico para um sistema operacional. O código binário resultante será executado e, por esta razão, o código deve ser capaz de se comunicar com o sistema operacional em questão (DEITEL; DEITEL, 2010). A linguagem Java adota o conceito denominado "máquina virtual", onde, entre o sistema operacional e a aplicação, existe uma camada extra que interpreta as ações da aplicação para realizar as respectivas chamadas ao sistema operacional (CAELUM, 2016).

A *Java Virtual Machine* (JVM) é uma máquina abstrata. Assim como uma máquina real de computação, a JVM apresenta um conjunto de instruções e manipula várias áreas de memória em tempo de execução (LINDHOLM; YELLIN, 2011).

5. Desenvolvimento

A API projetada e implementada na plataforma Java podem ser adotada em distintos contextos de maneira interoperável. Dessa forma, não somente aplicações Java podem utilizar os recursos da API resultante desta pesquisa científica, bem como diferentes linguagens e plataformas de programação como, por exemplo, Python e Microsoft .Net.

Na Figura 3, é apresentada a representação arquitetural proposta para a abordagem de ID3 no contexto de desenvolvimento de jogos com Unity 3D Engine. A estratégia adotada pode ser dividida em três importantes etapas:

- (1) Entrada de dados de treinamento (*Training Data*): Como mencionado anteriormente, as Árvores de Decisões e, particularmente a esta pesquisa, o algoritmo ID3, utilizam um conjunto de dados existente, ou *Training Data*, para auxiliar na montagem da árvore de decisão final com base na melhor divisão dos atributos e classes de atributos. O Apêndice A apresenta um fragmento de exemplo de JSON para um possível *Training Data*;
- (2) Processamento da API: Os conceitos do algoritmo ID3 foram implementados com o auxílio dos recursos da plataforma Java, como *collections* e *maps*, bem como da própria Orientação a Objetos. Dessa forma, classes, interfaces e métodos incorporam as funcionalidades necessárias para importar, manipular, extrair e processar arquivo de *Training Data*. Por meio da API, as operações de cálculos de Entropia e Ganho de Informação podem ser executadas.
- (3) Geração da árvore de decisão final: Após o processamento e cálculos de Entropia e Ganho de Informação, a API proposta realiza a montagem da árvore de decisão resultante. A saída final é um arquivo JSON que representa a árvore de decisão otimizada para o arquivo utilizado como *Training Data*. O Apêndice B apresenta um JSON de saída para uma árvore de decisão gerada;
- (4) Uso da árvore de decisão gerada, em JSON, pela Unity 3D Engine: Por fim, o arquivo JSON, que representa a árvore de decisão extraída pela API, pode ser utilizado para distintas finalidades. Nesta pesquisa científica, tal árvore é empregue como uma estratégia para geração de regras em jogos. Por exemplo, regras para (i) atacar um determinado monstro ou inimigo, (ii) gerar ou criar monstros de diferentes tipos a partir das características do personagem (por exemplo, modo de jogar), (iii) gerar ou combinar cenários e objetos do ambiente do jogo e até mesmo (iv) sugerir o uso de itens e/ou recompensas durante fases/mapas de um jogo.

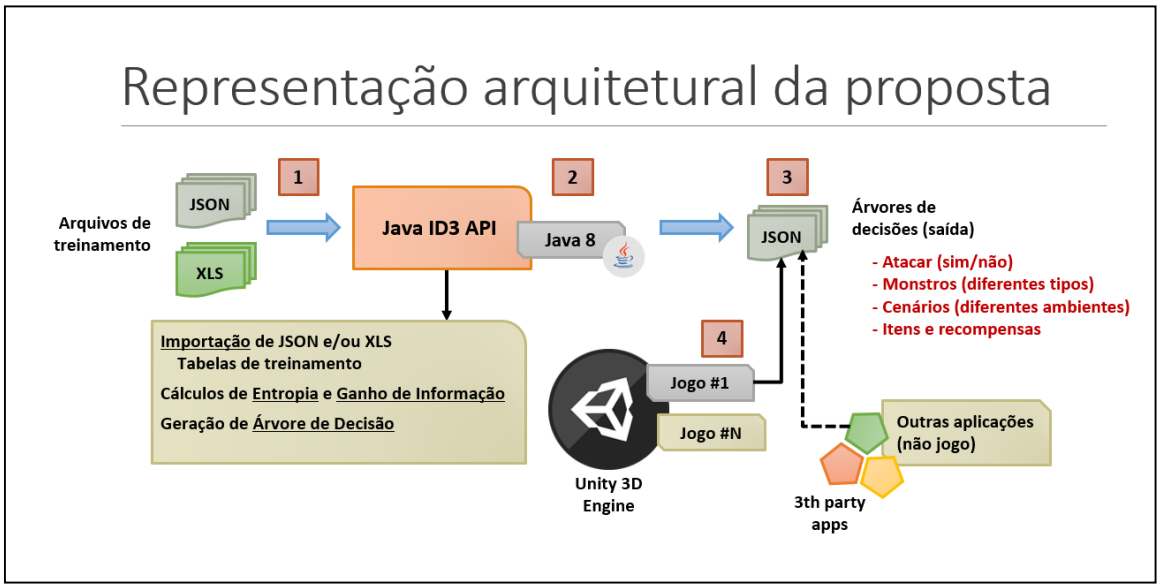

Figura 3. Representação arquitetural da abordagem proposta

5.1 Vantagens da Abordagem Proposta

Pode-se afirmar que a abordagem proposta, composta pela arquitetura e API modeladas e implementadas, contribuem como sendo um mecanismo dinâmico para regras e decisões em jogos. Isto, pois diferentes regras podem estar contidas nas árvores de decisões, ao invés de estarem incorporadas diretamente no código fonte do jogo.

A API e a abordagem exploradas nesta pesquisa são independentes do contexto e/ou tipo de jogo, tornando-os mais adaptativos e customizados de acordo com o nível de adoção das regras e das árvores de decisões geradas. Além disto, a abordagem proposta pode ser aplicada a quaisquer cenários, inclusive em contextos que não se baseiam em jogos.

A arquitetura auxilia equipes de desenvolvimento de jogos, otimizando e melhorando a maneira como regras e controles são incorporadas em jogos. O uso pode ser estendido como parte de simulações de diversos cenários e situações em jogos, permitindo que novas regras sejam testadas e avaliadas. Por fim, futuramente, os próprios jogadores podem utilizar a abordagem para gerar regras e desafios a partir de árvores de decisões.

6. Resultados

Ressalta-se como resultados desta pesquisa de iniciação científica, o estudo da subárea de Árvores de Decisão (área de Inteligência Artificial), mais precisamente dos fundamentos do algoritmo ID3 proposto por Quinlan (1986), para a concepção e implementação de jogos desenvolvidos na plataforma Unity 3D Engine.

Para alcançar os objetivos, uma *Application Program Interface* (API) foi modelada e implementada na plataforma Java para expor as funcionalidades e regras do algoritmo ID3 na forma de uma biblioteca escalável. A partir da API de ID3 desenvolvida nesta pesquisa, os cálculos de Entropia e Ganho, bem como a importação de informações de *Training Data* e a geração da árvore de decisão podem ser realizadas não somente com Java, porém em qualquer linguagem de programação ou plataforma de desenvolvimento.

Para este trabalho, buscou-se utilizar a API junto à plataforma Unity 3D Engine, possibilitando o uso dos recursos de árvores de decisão e ID3 na geração de regras para jogos e aplicativos baseados em computação gráfica.

Após a aplicação e experimentação dos conceitos de Árvores de Decisões e ID3 com base na API desenvolvida, verificou-se que a abordagem e arquitetura proposta para integração em jogos com Unity 3D Engine evidenciam a sua validade. Assim, por meio da arquitetura e integração modelada, os conceitos de ID3 puderem ser aplicados para a geração de regras em jogos desenvolvidos com Unity 3D Engine.

Também se deve destacar que a API e arquitetura propostas são extensíveis e escaláveis, ou seja, podem ser facilmente estendidas para incorporar novas funcionalidades e diferentes algoritmos baseados em Árvores de Decisões. Dessa forma, recursos adicionais podem contribuir com a aplicabilidade de IA no contexto de jogos e de outras aplicações.

Por fim, destaca-se que os resultados foram expostos durante o IX Fórum Científico na FEMA e durante a Feira de Profissões e Mostra de Software e Games, em Setembro/2016.

7. Considerações Finais

Este projeto de pesquisa, ainda que desenvolvido e aplicado em um contexto experimental, apresentou uma abordagem composta por uma API e uma arquitetura para auxiliar na adoção de Árvores de Decisões, subárea da Inteligência Artificial, no contexto de jogos. Particularmente, a abordagem sugerida neste trabalho foca no uso do algoritmo ID3 para a geração de regras em conjunto com a plataforma Unity 3D Engine.

A arquitetura possibilita a manipulação de arquivos XLS ou JSON que representam dados de treinamento para o algoritmo ID3, chamados de *Training Data*. A API desenvolvida em Java expõe classes, interfaces e métodos para os cálculos de Entropia e de Ganho de Informação, bem como os recursos necessários para a geração ou montagem da árvore de decisão final. O arquivo de saída é estruturado em JSON e pode ser utilizado na plataforma Unity 3D Engine para dinamizar o sistema de regras em jogos.

A abordagem investigada e proposta resulta em vantagens, não somente no contexto de jogos, como também em diferentes cenários. Destaca-se que os estudos conduzidos nesta pesquisa contribuem com um mecanismo dinâmico para regras em jogos, pois não se torna necessário que as decisões estejam presentes no código fonte da aplicação. Assim, os jogos podem ser mais adaptativos e customizados.

Além disto, a API e arquitetura otimizam e melhoram a maneira como regras e estruturas condicionais são aplicadas nos jogos e aplicações gráficas. Inclusive, os desenvolvedores de jogos poderem utilizar a abordagem deste trabalho para realizar testes e simulações em jogos sem que alterações sejam realizadas no projeto original, incorporando as mudanças de regras nas árvores de decisões.

7.1. Trabalhos futuros

Como trabalhos futuros, distintos tópicos podem ser explorados. Os autores desta pesquisa sugerem, entre outras evoluções:

- Melhorias e refatorações na API para incorporar novas funcionalidades como, por exemplo, implementações para os algoritmos C4.5 e CART (em Inglês, *Classification And Regression Trees*), algoritmos resultantes das evoluções do ID3;
- Avaliação experimental da API proposta em distintos contextos como, por exemplo, em jogos educacionais e de treinamento ou aprendizado, bem como em diferentes tipos de aplicações, não somente no desenvolvimento de jogos;
- Aplicação e avaliação da abordagem quando integrada ao contexto de *Machine Learning* e demais áreas de IA, como Algoritmos Genéticos, em jogos.

Referências

BASGALUPP, M. P. **LEGAL-Tree: Um algoritmo genético multi-objetivo lexicográfico para indução de árvores de decisão**. Tese de Doutorado, 2010, ICMC-USP, São Carlos.

CAELUM. **Java e Orientação a Objetos**. 2016. Disponível em <https://www.caelum.com.br/apostila-java-orientacao-objetos/>. Acesso em 08/02/2016.

COPELAND, J. **What is Artificial Intelligence?**. 2000. Disponível em <http://www.alanturing.net/turing_archive/pages/reference%20articles/what%20is%20ai.html>. Acesso em 27/11/2016.

GARCIA, S. C.; ALVARES, L. O. **Cadernos de Informática**. 2001. Disponível em <http://seer.ufrgs.br/index.php/cadernosdeinformatica/article/view/v1n1p52-55/8809>. Acesso em 30/11/2016.

HUNICKE, R., CHAPMAN, V. **AI for Dynamic Difficulty Adjustment in Games**. 2004. Disponível em: <https://drive.google.com/file/d/0BzpsESbe1T0aZzQtd0R1a1lYdmc/view?ts=565886fd>. Acesso em 01/10/2016.

JSON. **Introducing JSON**. 2016. Disponível em <http://www.json.org/>. Acesso em 30/11/2016.

LINDHOLM, T.; YELLIN, F. **Java Virtual Machine Specification**. 2011. Disponível em <http://docs.oracle.com/javase/specs/jvms/se7/html/jvms-1.html>. Acesso em 08/02/2016.

MCCARTHY, J. **What is Artificial Intelligence?**. 2007. Disponível em <http://www-formal.stanford.edu/jmc/whatisai/whatisai.html>. Acesso em 27/11/2016.

PARKER, L. **The Digital Revolution: How Consumers Are Driving the Future of Games Retail**. 2012. Disponível em: <http://www.gamespot.com/articles/thedigital-revolution-how-consumers-are-driving-the-future-of-games-retail/1100-6396713/>. Acesso em 13/09/2016.

PASSOS, E. B.; JÚNIOR, J. R. S.; RIBEIRO, F. E. C.; MOURÃO, P. T. **Tutorial: Desenvolvimento de Jogos com Unity 3D**. VIII Brazilian Symposium on Games and Digital Entertainment, 2009.

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

RUSSELL, S. J.; NORVIG, P. **Artificial Intelligence: A modern approach**. Prentice-Hall, 1995.

SILVA, L. M. O. **Uma Aplicação de Árvores de Decisão, Redes Neurais e KNN para a Identificação de Modelos ARMA Não-Sazonais e Sazonais**. Tese de Doutorado, 2005, PUC-RIO.

TAKIUCHI, M.; MELO, E.; TONIDANDEL, F. **Domótica Inteligente: Automação Baseada em Comportamento**. 2004. Centro Universitário da FEI - UniFEI. Disponível em <http://fei.edu.br/~flaviot/pub_arquivos/cba2004_final.pdf>. Acesso em 30/11/2016.

ZOOK, A. E., RIEDL, M. O. **A Temporal Data-Driven Player Model for Dynamic Difficulty Adjustment**. In: Proc. of the 8th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, 2012.

ZUBEN, F. J., ATTUX, R. R. F. **Árvores de Decisão**. 2007. Disponível em <ftp://ftp.dca.fee.unicamp.br/pub/docs/vonzuben/ia004_1s10/notas_de_aula/topico7_IA004_1s10.pdf>. Acesso em 29/11/2016.

Apêndices

Apêndice A: Fragmento de JSON para *Training Data* do cenário “Jogar Tênis”

```
{
    "idAttribute": {
        "name": "ID"
    },
    "attributes": [{
        "name": "ID"
    },
    {
        "name": "Perspectiva"
    },
    {
        "name": "Temperatura"
    },
    {
        "name": "Umidade"
    },
    {
        "name": "Vento"
    },
    {
        "name": "Jogar tênis"
    }],
    "decisionAttribute": {
        "name": "Jogar tênis"
    },
    "rows": [{
        "values": [[{
            "name": "ID"
        },
        "D1"],
        [{
            "name": "Perspectiva"
        },
        "Ensolarado"],
        [{
            "name": "Temperatura"
        },
        "Quente"],
        [{
            "name": "Umidade"
        },
        "Alta"],
        [{
            "name": "Vento"
        },
        "Fraco"],
        [{
            "name": "Jogar tênis"
        },
        "Não"]]
    } ... ] // demais valores para o Training Data completo
}
```

Apêndice B: Árvore de decisão resultante da API, em JSON, para o cenário “Jogar Tênis”


```
{
    "rootNode": {
        "description": "Perspectiva",
        "nodes": [{
            "description": "Ensolarado",
            "isEdge": true,
            "nodes": [{
                "description": "Umidade",
                "nodes": [{
                    "description": "Alta",
                    "isEdge": true,
                    "nodes": [{
                        "description": "Não"
                    }]
                },
                {
                    "description": "Normal",
                    "isEdge": true,
                    "nodes": [{
                        "description": "Sim"
                    }]
                }]
            }]
        },
        {
            "description": "Nublado",
            "isEdge": true,
            "nodes": [{
                "description": "Sim"
            }]
        },
        {
            "description": "Chuvoso",
            "isEdge": true,
            "nodes": [{
                "description": "Vento",
                "nodes": [{
                    "description": "Fraco",
                    "isEdge": true,
                    "nodes": [{
                        "description": "Sim"
                    }]
                },
                {
                    "description": "Forte",
                    "isEdge": true,
                    "nodes": [{
                        "description": "Não"
                    }]
                }]
            }]
        }]
    }
}
```