UNIVERSIDADE ESTADUAL DO SUDOESTE DA BAHIA – UESB
DEPARTAMENTO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
CURSO DE CIÊNCIA DA COMPUTAÇÃO

CHARLES ABREU SANTANA

Lógica Paraconsistente Anotada de Anotação com Dois Valores - LPA2v aplicada a Inteligência Artificial em Jogos Eletrônicos

Vitória da Conquista - BA
2015

CHARLES ABREU SANTANA

Lógica Paraconsistente Anotada de Anotação com Dois Valores - LPA2v aplicada a Inteligência Artificial em Jogos Eletrônicos

Monografia apresentada ao Curso de Graduação em Ciência da Computação da Universidade Estadual do Sudoeste da Bahia, como requisito para obtenção do Grau de Bacharel em Ciência da Computação.

Orientador (a): Prof.ª Dra. Alzira Ferreira da Silva

Vitória da Conquista

2015

CHARLES ABREU SANTANA

"LÓGICA PARACONSISTENTE ANOTADA DE ANOTAÇÃO COM DOIS VALORES – LPA2V APLICADA A INTELIGÊNCIA ARTIFICIAL EM JOGOS ELETRÔNICOS".

Monografia apresentada no Curso de Bacharelado em Ciência da Computação da Universidade Estadual do Sudoeste da Bahia, campus de Vitória da Conquista, como exigência parcial para obtenção do grau de Bacharel em Ciência da Computação, na área de concentração em Inteligência Artificial.

Trabalho aprovado pela banca examinadora em 05/02/2015.

Alzira Ferreira da Silva.

Prof.ª D.ra. Alzira Ferreira da Silva - UESB

Giovanilton Ferreira da Silva.

Prof.ª D.r Giovanilton Ferreira da Silva

Prof.º D.r Roque Mendes Prado Trindade

AGRADECIMENTOS

À minha família, em especial à minha mãe Eliene pelo suporte fundamental para realização desse trabalho. Ao meu pai Messias, *"In Memorian"*, que apesar de não poder presenciar este momento em minha vida, foi fundamental para que ele acontecesse. Aos meus amigos, os velhos e os que fiz durante este período de graduação. Aos professores e funcionários da UESB que contribuíram para minha formação e em especial à minha orientadora Alzira Ferreira da Silva, obrigado pela paciência em me guiar neste projeto.

"Podes dizer-me, por favor, que caminho eu devo seguir para sair daqui?"

"Isso depende muito de para onde queres chegar." - Disse o Gato.

Lewis Carroll – Alice no País das Maravilhas

Lógica Paraconcistente Anotada com Anotação de Dois Valores - LPA2v aplicada a Inteligência Artificial em Jogos Eletrônicos

RESUMO

O mercado de jogos está em constante crescimento e técnicas inovadoras garantem vantagem competitiva. O uso de técnicas de inteligência artificial para jogos de computador amplia sua interatividade e oferece um jogo cada vez mais atrativo garantindo a satisfação do usuário.

A Lógica Paraconsistente tem demonstrado bons resultados em diferentes áreas do conhecimento, porém com aplicação pouco explorada no desenvolvimento de jogos. Devido a esse fato surgiu a ideia de interligar esses dois campos de conhecimento, a Lógica Paraconsistente e o desenvolvimento de jogos. Para isso foi feito uma investigação sobre as técnicas de inteligência artificial utilizadas atualmente em jogos de computador e um estudo sobre os fundamentos da Lógica Paraconsistente. O presente trabalho, baseando-se em diferentes ideias colhidas na literatura, consiste na implementação de um algoritmo de inteligência artificial fundamentado na Lógica Paraconsistente Anotada para aperfeiçoar a eficácia das ações dos personagens controlados pelo computador em um jogo.

Palavras chaves: Lógica Paraconsistente, jogos de computador, Inteligência Artificial, desenvolvimento de jogos.

Paraconcistent Annotated Logic of Annotation with Two Values LPA2v applied Artificial Intelligence in Electronic Games

ABSTRACT

The gaming market is constantly growing, then innovative techniques ensure competitive advantage. The use of artificial intelligence techniques for computer games has increased its interactivity and provides an increasingly attractive game ensuring user satisfaction.

The Paraconsistent Logic has shown good results in different areas of knowledge, but with little explored application in game development. Due this fact the idea to connect these two fields of knowledge, Paraconsistent Logic and game development. For this was done a research on artificial intelligence techniques currently used in computer games and a study on the foundations of Paraconsistent Logic. This work, based on different ideas collected in the literature, is the implementation of an artificial intelligence algorithm based on Annotated Paraconsistent Logic to improve the effectives actions of non-player characters in a game.

Keywords: Paraconsistent Logic, computer games, Artificial Intelligence, game development.

LISTA DE FIGURAS

LISTA DE GRÁFICOS

LISTA DE SIGLAS E ABREVIATURAS

¬	– Conectivo Lógico de Negação
μ	– Grau de crença
λ	– Grau de descrença
G_c	_ Grau de certeza
Gct	– Grau de Contradição
HP	– Hit Points
IHC	– Interação Humano Computador
IA	– Inteligência Artificial
IP	– Inimigo Paraconsistente
LPA	– Lógica Paraconsistente Anotada
LPA2v	–Lógica Paraconsistente Anotada de anotação com dois valores
MP	– Magic Points
NPC	– Non- Player Character
p	– Proposição
QUPC	– Quadro Unitário no Plano Cartesiano
SBR	– Sistema Baseado em Regras
SDL	– Simple Direct Media Layer
RNAP	– Redes Neurais Artificiais Paraconsistentes
V	–Verdadeiro
F	–Falso
⊤	–Inconsistente
⊥	–Paracompleto
Qf→ ⊤	– Quase falso tendendo a inconsistente
Qf→⊥	–Quase falso tendendo a paracompleto

Qv→⊥	–Quase verdadeiro tendendo a paracompleto
Qv→ ⊤	–Quase verdadeiro tendendo a inconsistente
⊤→f	–Inconsistente tendendo a falso
⊤→v	– Inconsistente tendendo a verdadeiro
⊥→f	– Paracompleto tendendo a falso
⊥→v	– Paracompleto tendendo a verdadeiro
Vicc	– Valor inferior de controle de certeza
Vsci	– Valor superior de controle de incerteza
Vscc	– Valor superior de controle de certeza
Vici	– Valor inferior de controle de incerteza

SUMÁRIO

1. INTRODUÇÃO

O mercado de jogos se encontra em crescimento e muito se tem feito para ganhar competitividade nessa área. Características como jogabilidade, interação e design gráfico são constantemente lapidadas no intuito de tentar oferecer ao usuário a melhor experiência possível.

Segundo o SEBRAE, em 2012 o Brasil ocupava a posição de quarto maior mercado do mundo no segmento de jogos digitais, com cerca de 35 milhões de usuários. O mercado nacional de games movimentou R$ 5,3 bilhões em 2012, com crescimento de 32% em relação ao ano anterior. Segundo o instituto de pesquisa New Zoo (UOL, 2014), estimava-se que o Brasil iria consumir quase US$ 1,4 bilhões em jogos digitais no ano de 2014, o colocando como maior mercado na América Latina.

O foco do desenvolvimento de jogos se concentrou, através dos anos, em sua composição gráfica. Porém a nova tendência a ser desvendada no mercado de jogos será a interatividade. Em depoimento ao site GamesIndustry.biz, Yves Jacquier, diretor executivo de produção da Ubisoft[1], disse acreditar que os consoles da próxima geração terão os sistemas de inteligência artificial como grande foco, em vez da parte gráfica.

Ao contrário de gráficos rebuscados e incrivelmente realistas, a inteligência artificial é o quesito que será o foco dos jogos futuramente. Personagens que se comportam de forma realista e podem aprender e se adaptar, será mais atrativo que personagens com maiores resoluções e com mais *frames* por segundo. Isso se torna cada dia mais evidente, pois à medida que as plataformas de jogos eletrônicos aumentam sua capacidade computacional um maior volume de processamento torna-se disponível para a inteligência dos jogos.

Além de sua ascensão no mercado, do ponto de vista acadêmico, os jogos se apresentam como uma excelente plataforma para construção de novas metodologias e algoritmos. Jogos de computador envolvem diversas áreas como computação gráfica, redes, Interação Homem e Computador (IHC) e Inteligência Artificial (IA). “Em função da diversidade de campos envolvidos, os jogos se mostram um bom ambiente para a pesquisa de novas tecnologias para estas áreas da ciência da computação.” (CROCOMO, 2006, p.8). “Tais

[1]Ubisoft é uma empresa publicadora de jogos eletrônicos. É uma das maiores publicadoras do mundo e também é uma grande desenvolvedora, tendo entre suas franquias mais famosas as séries Assassin'sCreed, Far Cry, Driver, Just Dance, Rayman, Myst, Splinter Cell, Imagine, Prince of Persia e Watch Dogs.

Jogos apresentam novos desafios e oportunidades sobre o ponto de vista da inteligência artificial, devido ao alto grau de interação com o usuário e por possuírem ambientes ricos e dinâmicos, aproximando-se assim do mundo real, mas mantendo-os ainda sob ambiente controlado.” (TATAI, 2003, p. 2).

No campo da inteligência artificial, jogos de computador possuem caráter multidisciplinar, prestando-se a demonstrar a validade das mais diversas técnicas, desde reconhecimento de linguagem natural, modelos de cognição e interação, até mecanismos complexos de planejamento, busca e aprendizagem (TATAI, 2003).

Geralmente, no desenvolvimento de jogos tem-se utilizado da lógica clássica. Estas não levam em consideração a mutação de eventos. Para solucionar este problema tem-se as extensões, as lógicas modais que permitem raciocinar sobre possibilidades e certezas. No entanto a lógica clássica é ineficiente para modelar sistemas especialistas, em que aparecem incertezas, ambiguidades e contradições. Desta forma como avaliar situações inconsistentes?

Uma proposta é a utilização das lógicas paraconsistentes. Estas tratam das leis das contradições e isto permite a sua utilização em qualquer situação em que exista a necessidade de se analisar o conhecimento incerto. Por essa flexibilidade no tratamento de informações as lógicas paraconsitentes tem sido aplicadas na matemática, física, computação, inteligência artificial e robótica, dentre outras áreas. (LEMES NETO E VENSON, ?)

A Lógica Paraconsistente tem obtido bons resultados em diferentes áreas do conhecimento como: reconhecimento de caracteres, diagnóstico médico, robótica autônoma e aplicações industriais (ABE, 2013). Uma aplicação ainda pouco explorada é o desenvolvimento de jogos, uma área em que o tratamento de situações que levam a contradições podem ser exploradas. Devido a esse fato, o intuito nesse trabalho é mostrar como a abordagem paraconsistente pode ser interessante para inteligência artificial em jogos de computador, podendo levar a resultados tão bons quanto aos obtidos com as abordagens clássicas tradicionais.

Observando este caráter multidisciplinar, este trabalho visa apresentar uma abordagem diferenciada no campo da inteligência artificial, nesse caso a Lógica Paraconsistente, que sirva como alternativa para incrementar o estado da arte e complementar as técnicas tradicionais, auxiliando-as nas áreas em que as mesmas encontram dificuldades.

O objetivo geral é analisar a viabilidade da aplicação da lógica paraconsistente anotada para modelagem de jogos eletrônicos

Para alcançar este objetivo, foi realizado uma investigação sobre as técnicas de inteligência artificial, utilizadas atualmente em jogos de computador e sobre alguns fundamentos da Lógica Paraconsistente Anotada. Assim, buscou-se as contribuições para o campo da inteligência artificial para jogos, utilizando uma abordagem da Lógica Paraconsistente Anotada e a possibilidades do uso do algoritmo paraconsistente para aperfeiçoar a eficácia das ações dos *non-player characters* (NPC's)[2]. Desta maneira, esta pesquisa se configura como uma pesquisa exploratória, uma vez que não se encontrou até o presente momento a utilização desta lógica em tais situações. Esta tem um caráter qualitativo, pois buscou-se avaliar a quão eficiente é um jogo paraconsistente com base em testes, por isso, tem-se uma pesquisa também experimental.

Para realização dos testes foi desenvolvido o Sistema de Batalhas Paraconsistente, sistema este que teve como base o Sistema de Batalhas baseada em regras desenvolvido por Barbosa (BARBOSA, 2012), descrito no Capítulo 4.

Ressalto que, a pretensão desse estudo não é apresentar a Lógica Paraconsistente como uma alternativa a outros métodos já consolidados, e sim como uma complementação a estes. Este mecanismo foi baseado em diferentes ideias colhidas na literatura no que diz respeito à Lógica Paraconsistente, e tem como objetivo ampliar características como a interatividade e jogabilidade do software, aumentando assim sua atratividade.

Para viabilizar o desenvolvimento do trabalho proposto, alguns objetivos específicos foram realizados:

1. Definir os fatores (preposições) aos quais serão atribuídos graus de crença e descrença.
2. Identificar a estrutura do algoritmo para-analisador adequada ao sistema de batalhas utilizado como estudo de caso, verificando as respostas do sistema através do plano cartesiano (QUPC).
3. Identificar os requisitos funcionais e não funcionais do Sistema de Batalhas paraconsistente tendo como base o Sistema de Batalhas(Barbosa,

[2]NPCé um personagem que não pode ser controlado por um jogador, mas se envolve de alguma forma no enredo de um jogo exercendo papéis específicos.

2012), utilizado como estudo de caso, para adaptá-lo ao algoritmo implementado.

4. Comparar o desempenho do algoritmo paraconsistente implementado com o modelo clássico para verificar a eficiência do algoritmo.

1.1 Estrutura da Monografia

Este trabalho encontra-se estruturado da seguinte forma:

No Capítulo 2, durante a revisão bibliográfica, são apresentados tópicos pertinentes à Inteligência Artificial em jogos eletrônicos. É comentado sobre a situação atual dos jogos assim como alguns dos principais algoritmos que constituem o estado da arte do tema em questão.

No Capítulo 3 são apresentados fundamentos da Lógica Paraconsistente. São apresentados alguns conceitos fundamentais e explicado o funcionamento do algoritmo para-analisador que será implementado no jogo proposto.

No Capítulo 4, os conceitos principais do projeto são apresentados. É explicada a estrutura e funcionalidades do jogo utilizado como ambiente de testes e é apresentado o algoritmo para-analisador implementado.

No Capítulo 5 são apresentados pontos relevantes em relação aos testes realizados e o funcionamento do algoritmo.

No Capítulo 6 encontram-se as conclusões sobre o projeto e propostas de como continuá-lo ou modificá-lo.

2. IA E JOGOS ELETRÔNICOS

O desenvolvimento de jogos possui diferenças em relação a outras esferas de desenvolvimento. Ele possui seu próprio mundo técnico, de linguagem, de habilidades e desafios (MILLINGTON, 2006). Para a Inteligência Artificial em jogos não é diferente, pois este ramo também possui suas próprias técnicas e modelos. No desenvolvimento de jogos o termo IA diverge em relação a IA acadêmica. Para distinguir a inteligência artificial utilizada em jogos e no meio acadêmico os desenvolvedores adotaram o termo *Game IA*. "Na indústria de jogos, o termo IA é utilizado para denotar os módulos de software responsáveis pelo comportamento "inteligente" realizado por componentes do jogo durante seu funcionamento." (TATAI, 2003, p.1).

A principal diferença entre as abordagens está no objetivo que cada uma propõe. Enquanto a IA acadêmica é fundamentada na resolução de problemas para entender e construir agentes, a IA para jogos eletrônicos visa a diversão. Sua importância é quanto aos resultados que o sistema irá gerar, e não como o sistema chega até os resultados; ou seja, o problema não é como o sistema pensa, mas sim como ele age (KISHIMOTO, 2004).

Para jogos, uma IA que não possui um resultado visível para o usuário, é um desperdício computacional (KIRBY, 2011). O jogador não tem que necessariamente ver o comportamento inteligente no momento em que a decisão está sendo feita, mas a evidência da inteligência deve ser disponibilizada para o jogador para que o mesmo possa desfrutá-la.

Para Kirby (2011), IA não é física. Muitos confundem conceitos como colisão ou movimentos físicos dos personagens com IA. Pelo contrário, esses são apenas mecanismos utilizados para simular o mundo real, mas não para oferecer um comportamento inteligente. Por exemplo, uma pedra não faz uma escolha ótima em curto prazo para se beneficiar em longo prazo, sobre como ela irá cair de um penhasco. Ela simplesmente cai, e isso não é IA.

O cerne da IA para jogos está em oferecer ao usuário, através de resultados notáveis, um jogo divertido, desafiador e com alto grau de interatividade. Como um produto de entretenimento, deve-se sempre visar o usuário. A IA deve tornar o jogo mais divertido e desafiador, não mais difícil. Um jogo difícil cria jogadores frustrados. Em um jogo, a diversão é o objetivo principal.

Em entrevista ao site jovemnerd.com.br, Tarquinio Teles, presidente da Hoplon, uma desenvolvedora nacional de jogos, disse que uma inteligência artificial deve ser como um ser humano jogando tênis com seu chefe: ganhar, mas não de muito, e perder sem parecer que está entregando o jogo. Para Teles, programar IA é fácil, mas isso faz com que a máquina seja perfeita e infalível. O desafio é balancear a programação de forma que o jogador possa participar da ação.

A IA em jogos aumenta a experiência e imersão do jogo, melhorando sua jogabilidade. Personagens não controlados pelo jogador podem ser utilizados para melhorar a experiência em jogos. Os NPC's inteligentes são necessários em qualquer gênero de jogo para criar a ilusão que o jogador está num mundo com outros jogadores inteligentes (KISHIMOTO, 2004). Logo respostas coerentes às decisões dos jogadores são o que constitui uma IA de qualidade. Algoritmos simples, robustos e flexíveis são de fundamental importância para garantir respostas plausíveis e diversificadas diante a mudança dos estados do jogo, devido às decisões dos usuários.

Empresas, como a Eletronic Arts, apostam no aprimoramento da inteligência artificial utilizada em seus jogos para comandar os personagens não controlados pelo jogador. Em função disso, alguns jogos têm se destacado pela qualidade de sua Inteligência Artificial, como The Sims4 (Eletronic Arts, 2014), veja a Figura 1. The Sims4 é um jogo de simulação de vida, onde os jogadores controlam personagens em várias atividades e devem estabelecer relações de uma forma semelhante à vida real. Nesse jogo os personagens possuem personalidades distintas definidas pelas suas aspirações e habilidades, além de simular emoções, tornando-os mais interativos.

Figura 1 - The Sims4

Fonte: https://www.thesims.com/pt_BR/

Jogos *open world* (mundo aberto), onde o personagem tem a possibilidade de percorrer livremente um mundo virtual, também trabalham no aprimoramento de sua inteligência artificial. Devido à liberdade de navegação dentro do jogo, o jogador pode se deparar com diversas situações e agir de maneiras distintas exigindo uma IA mais robusta e eficaz. O jogo Watch Dogs (Ubsoft, 2014), utiliza esse tipo de abordagem, onde o personagem deve interagir com vários NPC's enquanto percorre o mapa do jogo para realizar seus objetivos (Figura 2).

Figura 2-Watch Dogs

Fonte: http://watchdogs.ubi.com/watchdogs/pt-br/

Nessa mesma categoria de jogos, tem-se também a série *Assassin'sCreed* (Ubsoft, 2014). É uma série de grande sucesso, que gerou grande popularidade, sendo lançado em várias plataformas e estando em sua oitava série principal, além de várias outras publicações paralelas, veja na Figura 3.

Esse jogo é do gênero *stealth* (discrição), que é um estilo de jogo onde o jogador deve evitar ser percebido, fazendo uso de estratégias para evadir ou criar uma emboscada para antagonistas. Jogos desse tipo empregam mecânicas como se esconder na sombra, espionagem, aproximação sigilosa, disfarces, e barulhos que podem alertar os inimigos. Esse tipo de comportamento gera uma sobrecarga muito grande para a IA, forçando-a a ser robusta e adaptável a novas condições. Segundo Kirby (2011), uma das características mais importantes de uma IA é adaptabilidade, se a IA não agir de forma diferente em diferentes situações, será como assistir várias vezes a uma queda de vareta ao chão quando você deixá-la cair.

Figura 3-Assassin'sCreedUnity

Fonte: http://assassinscreed.ubi.com/pt-BR

Algumas abordagens utilizam IA também como uma ferramenta para trazer vantagens ao desenvolvimento dos jogos. Um exemplo dessa abordagem é a utilização de redes neurais e aprendizagem para definir comportamentos dentro do jogo, sem a necessidade de programá-los.

2.1 Técnicas e Algoritmos de IA Implementados em Jogos

Existem uma grande variedade de técnicas de Inteligência Artificial em jogos eletrônicos que vão desde a inteligência artificial clássica, baseada em lógica de primeira ordem e sistemas especialistas, como a que incorpora sistemas nebulosos, redes neurais artificiais e sistemas evolutivos (TATAI, 2003).

Segundo Kishimoto (2004), os primeiros jogos criados se baseavam em implementações de padrões pré-programados de movimento para os NPC’s. Isto se deve ao fato de os computadores inicialmente possuírem limitações na memória e velocidade de processamento.

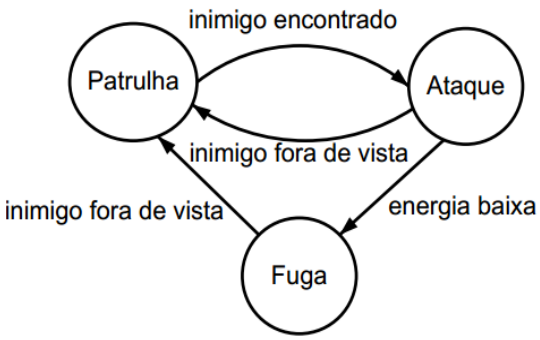
Atualmente várias técnicas têm sido utilizadas, cada qual em seu contexto, para melhorar a interatividade dos jogos. As mais utilizadas em jogos tradicionais são as máquinas de estados finitos, sistema baseado em regras e busca em árvores, devido sua característica

determinística. Recursos mais avançados como: redes neurais, algoritmos genéticos e sistemas nebulosos, que incorporam as técnicas de inteligência computacional, ainda estão em desenvolvimento. Muita pesquisa se tem feito na tentativa de incorporar essas técnicas aos jogos.

Algumas das principais técnicas de inteligência artificial utilizadas em jogos de computador são:

- **Máquinas de Estado Finito:** é uma máquina abstrata que define os estados em que o personagem pode se encontrar, e quando o mesmo muda de estado em determinada circunstância. O estado corrente da máquina determina como o personagem deve atuar no jogo e as transições de estado são feitas quando a condição de mudanças de estado é alcançada (KARLSSON, 2005). Máquinas de estados são frequentemente usadas para implementação de IA em jogos, devido ao seu fácil entendimento, implementação e depuração de erros (KISHIMOTO, 2004). No entanto por ser determinística, esta técnica torna previsível a estratégia utilizada pelo computador (CROCOMO, 2006). Um exemplo de uma máquina de estados para o comportamento de um personagem é mostrado na Figura 4;

Figura 4 - Exemplo de uma máquina de estados para um jogo fictício.



Fonte: TATAI, 2003, p.10

- **Path-Finding:** são também conhecidos como algoritmos de busca e geralmente são aplicados em estrutura de dados chamada de árvore. Têm como objetivo encontrar um caminho (ótimo ou não) do estado inicial até um estado final, explorando sucessivamente os nós conectados aos nós já explorados até a obtenção de uma solução para o problema (TATAI, 2003). A* é a principal técnica de busca utilizada, geralmente aplicada para realizar busca de caminhos

entre dois pontos do cenário ou para desviar de possíveis obstáculos, inimigos. Nesse tipo de implementação o personagem caminha através de "grades" (*grids*), onde cada célula pode representar um nó de um grafo. Um custo é associado para cada célula do *grid*, utilizado pela heurística do A*. (KISHIMOTO apud LAMOTHE, 1999).

- **Flocking:** Simulação de comportamentos naturais de grupos de indivíduos, como um grupo de pássaros ou um rebanho de ovelhas. Utilizada para criar ambientes realistas a serem exploradas pelo jogador (CROCOMO, 2006). Esse tipo de técnica é normalmente explorada em jogos nos quais é mais importante o comportamento do time de personagens ao invés do comportamento individual.
- **Árvores de Decisão:** Utilizada para classificação ou predição. Permite encontrar a sequência de ações que provavelmente apresenta os melhores resultados para uma determinada situação. Apesar de não ser uma técnica muito explorada em jogos, árvores de decisão são de fácil implementação (CROCOMO, 2006). O jogo Black Withe2 (Eletronic Arts), faz uso dessa técnica permitindo que uma criatura seja capaz de aprender com um jogador e com outras criaturas no desenrolar do jogo.
- **Redes Neurais:** Tem como objetivo de construir mecanismos similares às redes neurais biológicas, a tomada de decisões complexas, processamento de informações, otimização e aprendizagem. Utilizada principalmente como ferramenta para imitar o comportamento do jogador e aprender com o mesmo (CROCOMO, 2006). Vários desenvolvedores tem uma resistência em utilizar este método. Por ser não-determinístico, a rede neural pode ocasionar comportamentos não desejados para os personagens.
- **Lógica Fuzzy:** A lógica fuzzy ou difusa é uma lógica que admite valores intermediários entre o falso e o verdadeiro, representando desta forma informações vagas, como o "talvez". Nos jogos atuais ela é uma abordagem limitada a complexos comandos condicionais. Esse tipo de lógica analisa as sentenças e procura representar as tomadas de decisão humanas (KARLSSON, 2005). É essencial para tratar problemas onde as preposições assumem valores diversos, não só binários como na lógica clássica, como, por exemplo, o nível de afetividade de um personagem com relação ao jogador: muito baixo, baixo, médio, alto, etc. (CROCOMO, 2006).

- **Algoritmos Evolutivos:** A computação evolutiva tem como objetivo a resolução de problemas (ou a modelagem de processos evolutivos) de forma inspirada no processo biológico de propagação genética e seleção natural proposto por Charles Darwin. Segundo Crocomo (2006), é utilizado para estratégias de adaptação em relação às estratégias utilizadas pelo jogador e encontrar caminhos evoluindo o comportamento de personagens controlados pelo computador. Apesar de existirem inúmeras aplicações ainda há uma resistência para com o uso desse tipo de algoritmo, devido ao fato do mesmo apresentar comportamentos não aceitáveis durante o jogo.

Outro mecanismo de IA para jogos é o Sistema Baseado em Regras (SBR), que veremos mais detalhadamente na próxima seção, pois foi a abordagem utilizada para os testes no presente trabalho.

2.2 Sistemas Baseado em Regras

Em um Sistema Baseado em Regras (SBR), o conhecimento é definido através de um conjunto de parâmetros (variáveis) e um conjunto de regras que trabalham sobre esses parâmetros, de modo que durante a "tomada de decisão" essas regras são então processadas (KARLSSON, 2005). A saída exibida por essas regras pode ser uma ação do personagem, por exemplo.

Um SBR é constituído basicamente de duas partes: um banco de dados contendo conhecimento acessível, e um conjunto de regras "se-então". As regras examinam o banco de dados para determinar se uma condição é válida. Regras que tem suas condições atendidas são acionadas e sua respectiva ação é executada. A ação é responsável pelo comportamento do personagem, ou atualização de alguma informação da base de dados (KIRBY, 2011).

O acionamento das regras é feito a partir de uma máquina de inferência, que a partir do estado atual da base de dados, verifica se cada regra pode ser acionada através de iterações. Essa verificação pode ser feita de duas maneiras: encadeamento direto (forward-Chaining), ou encadeamento reverso (backward-chaining).

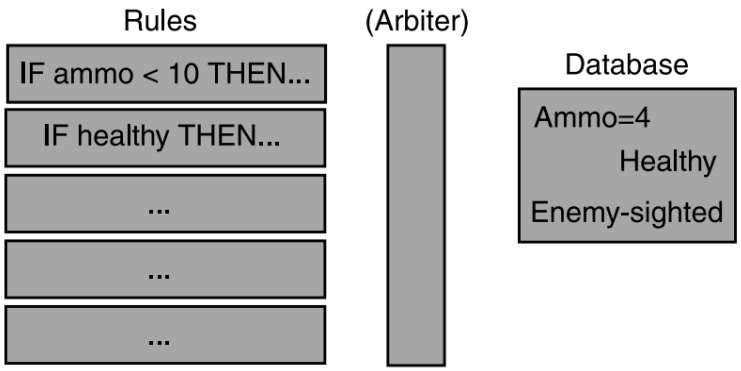
O sistema de encadeamento direto é um raciocínio guiado por fatos (NEGNEVITSKY, 2002). Considerado o mais comum dos sistemas de inferência. Parte-se

dos fatos e executam-se as regras adicionando novos fatos na base de dados. Nele, o sistema inicia com um conjunto de fatos, e a partir destes, aplica as regras repetidamente até que o resultado desejado seja alcançado (FUJITA, 2005).

O encadeamento reverso é o raciocínio guiado por objetivos. No encadeamento reverso, o sistema especialista recebe um fato a ser demonstrado como verdadeiro (esse é o objetivo) e a máquina de inferência busca encontrar evidências que provem a veracidade daquele fato (NEGNEVITSKY, 2002). Devido à sua natureza recursiva, o encadeamento reverso não é amplamente utilizado, por questões de desempenho computacional (FUJITA, 2005).

Em um dado ciclo de inferência várias regras podem ser acionadas ao mesmo tempo. Quando mais de uma regra é acionada, um método de arbitragem é necessário para a resolução do conflito. O árbitro em um SBR é um pedaço de código responsável pelas normas de arbitragem, que decide qual regra deve ser executada, no caso de múltiplas regras serem aplicáveis. A Figura 5 mostra o esquema de um SBR.

Figura 5 - Esquema de um sistema baseado em regras



Fonte: MILLINGTON,2006, p.404

Ao avaliar todas as regras, garantir que a melhor resposta seja dada, implica diretamente na eficiência. Portanto existem diferentes métodos de arbitragem para atender a necessidade do programador. Millington (2006) cita alguns desses métodos:

- **Primeiro Aplicável:** É o algoritmo mais simples. As regras são dispostas em uma ordem fixa e a primeira regra da lista que for ativada é executada. A ordenação impõe prioridade estrita: as regras anteriores na lista têm prioridade sobre as regras posteriores. Porém há alguns problemas: se uma regra

executada não modifica o estado atual da base de dados, a mesma regra continuará a ser executada cada vez que o sistema executa a IA.

- **Ultimo Recentemente Usado:** As regras do sistema são alocadas em uma lista encadeada e segue o mesmo mecanismo de ativação do primeiro aplicável. Porém quando alguma regra é executada, ela é removida de sua posição na lista e é colocada no final da lista. Depois de algum tempo, a lista irá conter as regras em ordem inversa de uso, então escolher a primeira regra a ser acionada, é o mesmo que escolher a regra menos usada recentemente. Esta abordagem evita *looping* e garante que todas as regras, algum momento terão a oportunidade de ser executada.
- **Regra Randômica:** Se várias regras são acionadas, escolhe-se uma aleatoriamente para ser executada. Ao contrário das duas ultimas abordagens, este tipo de árbitro precisa checar cada regra acionada na lista de regras, e não somente a primeira que for acionada. Nesse quesito este esquema é ineficiente em relação aos anteriores.
- **Atribuição de prioridade dinâmica:**Um valor de prioridade é atribuído às regras de acordo com sua importância no estado atual do jogo. Então o árbitro checa todas as regras, selecionando as que foram acionadas, e verifica as que possuem maiores valores para serem executadas. Assim como a regra randômica, essa abordagem realiza uma busca na lista de regras antes de decidir quem será executado. Esse método oferece maior flexibilidade, porém exige um custo de tempo maior.

3. LÓGICA PARACONSISTENTE

"A lógica estabelece as leis do raciocínio. A maneira certa de como a razão deve operar, pouco importando se o raciocínio tem ou não fundamento na realidade" (DA SILVA FILHO; ABE 2000). "O estudo da lógica está fundamentado na Lógica Clássica que se baseia no cálculo de predicados clássicos de primeira ordem, podendo ser estendida pela teoria dos conjuntos, teoria dos tipos e teoria das categorias" (NETO; VENSON, 2002).

Na lógica clássica uma proposição é classificada como verdadeira ou falsa. Não há qualquer outra possível alternativa, ou algo é Verdadeiro ou exclusivamente Falso. Não é possível que uma proposição seja simultaneamente Verdadeira e Falsa, como também não existe proposição que não possa ser classificada em uma das duas categorias. Também não é possível relativizar o que venha a ser verdadeiro ou falso, ou uma proposição é totalmente verdadeira ou totalmente falsa (DEMARIO, 2013).

Apesar de ser fundamentada em axiomas bem definidos, a Lógica Clássica não consegue abstrair todos os acontecimentos do mundo real. Segundo Platão (348 a.C.) existe dois mundos diferentes: o real caótico e mutável, que captamos pelos nossos sentidos; e mundo das ideias que é perfeito e representável pela matemática (DA SILVA FILHO e ABE, 2000). Ainda segundo Da Silva Filho e Abe (2000), para conseguir atender às características do mundo real, a Lógica Clássica, fundamentada nas rígidas leis binárias, pode não ser suficiente, sendo necessária uma abordagem de representação mais flexível. As situações de inconsistências, indefinições e de conhecimentos parciais são descritas no mundo real com muita frequência, de forma que se necessita de uma lógica que englobe todos esses comportamentos.

A partir do aparecimento de situações que não se enquadram nas rígidas regras da Lógica Clássica, surgiram estudos paralelos que culminaram com a instituição das lógicas alternativas da clássica. Com isso foram surgindo novas lógicas chamadas Lógicas Não Clássicas, cujo objetivo era de se estudar o tratamento de situações; como as indefinições, as inconsistências e os paradoxos que apareciam no mundo real, mas que não podem ser tratadas pela lógica clássica, pelo menos diretamente (DA SILVA FILHO, 1999).

Em 1910, o russo Nikolai A. Vasil'ev (1880-1940) e o polonês Jan Lukasiewicz (1878-1956), publicaram de forma independente, trabalhos aos quais tratavam da possibilidade de uma lógica que não eliminasse as contradições (DA SILVA FILHO, 1999). Ainda, segundo Da Silva Filho (1999), em 1948, o lógico polonês Stanislaw Jáskowski formalizou baseando-se na lógica discursiva um cálculo proposicional paraconsistente chamado Cálculo Proposicional Discursivo. Independentemente, no mesmo período, o lógico brasileiro Newton C. A. da Costa desenvolveu vários Sistemas Paraconsistentes que vem sendo um campo de pesquisa muito progressivo e promissor.

A Lógica Paraconsistente pertence ao grupo das Lógicas Não Clássicas e foi edificada para se encontrar meios de dar tratamento não trivial às situações contraditórias. Ela apresenta alternativas às proposições cujas conclusões sejam valores além do verdadeiro e do falso, como o indeterminado e o inconsistente (NETO; VENSON, 2002). As argumentações sobre uma determinada proposição se limitam a certificar que as premissas constituem evidencias apenas parciais para suas conclusões. Neste caso é considerado o grau de credibilidade ou crença que as premissas conferem à conclusão. Diremos que uma proposição é Inconsistente quando uma evidência sugere que ela seja Verdadeira e outra evidência sugere que ela seja Falsa, diremos que uma preposição é Indeterminado ou Paracompleta se não tivermos evidência de que ela seja Verdadeira nem tampouco que ela seja Falsa (DEMARIO, 2013).

Segundo NETO e VENSON (2002), uma lógica é dita paraconsistente se admite contradições não triviais. Para isso é necessário que a mesma seja estruturada numa teoria Inconsistente e não-trivial. Uma teoria é dita inconsistente se existir uma sentença **p** tal que **p**e ¬**p** sejam teoremas dessa teoria, em outras palavras, é inconsistente se existirem dois teoremas onde um é a negação do outro. Um teorema é trivial se qualquer sentença expressável de sua linguagem for teorema, caso contrário é dito não-trivial.

3.1 Lógica Paraconsistente Anotada de anotação com dois valores LPA2v

A Lógica Paraconsistente Anotada de anotação de dois valores- LPA2v é uma classe da Lógica Paraconsistente que trabalha considerando as proposições acompanhadas de anotações. Nesse tipo de lógica as conclusões são obtidas a partir de informações que não são categóricas, mas que apenas trazem evidências do fato a ser analisado (DA SILVA FILHO; ABE, 2000).

Essas evidências são quantificadas através de valores algébricos denominados grau de crença, ou evidência favorável, e grau de descrença, ou evidência desfavorável. Dada uma preposição p, o grau de evidência favorável à p é simbolizado por μ e o grau de evidência desfavorável à p é simbolizado por λ. Desse modo considera-se (μ, λ) como uma anotação de p. Esse método facilita o tratamento da informação, principalmente quando manipulada matematicamente. Os valores μ e λ são normalizados para que sofra uma variação apenas entre o intervalo real de 0 a 1.

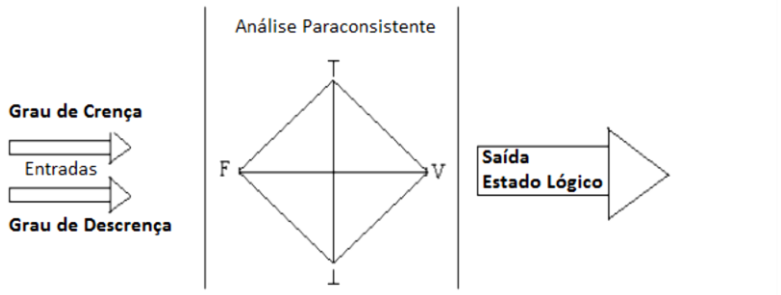
Para melhor representação a LPA2v é associada a um reticulado, em cujos vértices são alocados os símbolos que indicam os estados lógicos. O estado lógico é encontrado através dos valores de anotação que são os graus de evidencia favorável e desfavorável atribuídos à proposição.

μ = Grau de evidência favorável

λ = Grau de evidência desfavorável

Estes valores são considerados como informações de entrada do sistema e têm como objetivo solucionar o problema de sinais contraditórios ao coletar evidências, e por meio de análises modificar o comportamento do sistema para que a intensidade das contradições diminua (DA SILVA FILHO, 2006). Veja a figura 6.

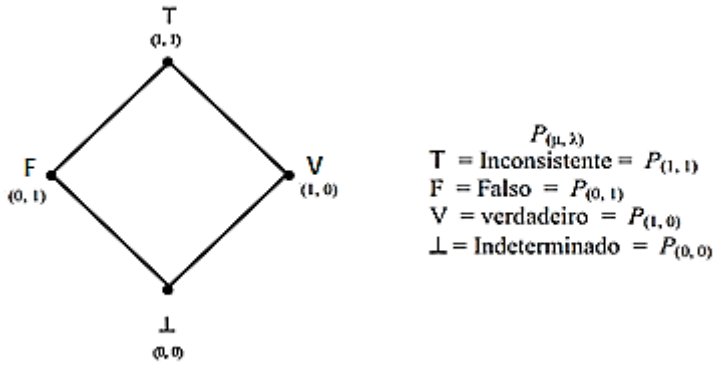
Figura 6 - Demonstração de um sistema básico de análise paraconsistente.



Fonte: NETO & VENSON, 2002, p.3

Os estados lógicos, com os valores do grau de crença e descrença podem ser relacionados como na Figura 7:

Figura 7: Reticulado associado à Lógica Paraconsistente Anotada LPA2v



Fonte: DA SILVA FILHO, 2000.

Os estados lógicos serão dados de maneira discretizada a partir de um algoritmo para-analisador. Esse algoritmo, implementado em linguagem computacional convencional, traduz a análise paraconsistente através dos valores do grau de evidência favorável μ, e o grau de evidência desfavorável λ, resultando em valores do Grau de Certeza G_ce do Grau de Contradição G_{ct} pelas equações:

$$G_c = \mu - \lambda \qquad \text{Eq. I}$$

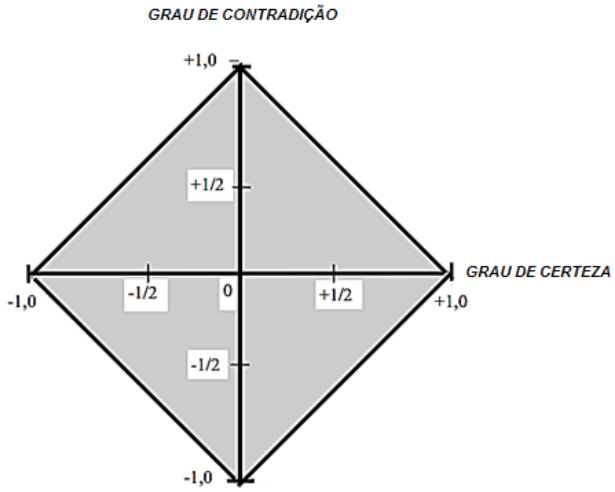
$$G_{ct} = \mu + \lambda - 1 \qquad \text{Eq. II}$$

Portanto o G_c (Eq. I) e o G_{ct} (Eq. II) variam em um intervalo fechado, [-1, 1].

Os valores de G_c e G_{ct} quando considerados no reticulado permitem que possamos verificar o quanto o ponto de interpolação entre esses dois valores está, ou não, próximo dos estados lógicos extremos representados nos vértices do reticulado.

Dispondo os dois eixos, tanto o do Gc quanto do Gct e cruzando-os no ponto de indefinição zero, temos o reticulado da LPA, construídos com os valores de G_c e G_{ct} (Figura8).

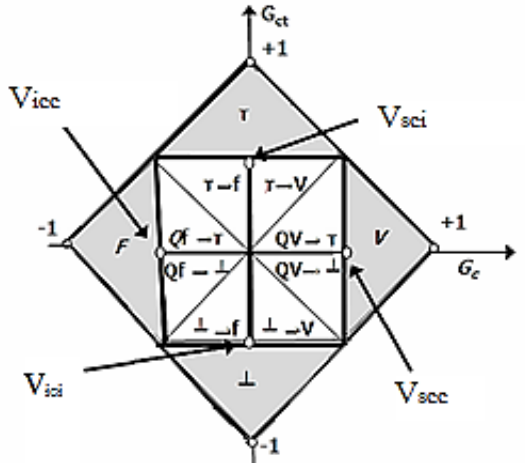
Figura 8-Representação dos graus de *certeza* e de *incerteza* inter-relacionados.



Fonte: DA SILVA FILHO, 1999

Na discretização do reticulado apenas um estado lógico estará ativo no final de cada análise. Assim o Sistema Paraconsistente pode formular uma conclusão e realizar uma ação baseada na região onde está localizado o ponto resultante. Assim o reticulado pode ser dividido em regiões, conforme a Figura 9.

Figura 9 - Reticulado da LPA2v dividido em 12 regiões



LEGENDA:

V = VERDADEIRO
F = FALSO
T = INCONSISTENTE
⊥ = PARACOMPLETO
Qf→ T =QUASE FALSO TENDENDO A INCONSISTENTE
Qf→⊥= QUASE FALSO TENDENDO A PARACOMPLETO
Qv→⊥= QUASE VERDADEIRO TENDENDO A PARACOMPLETO
Qv→ T = QUASE VERDADEIRO TENDENDO A INCONSISTENTE
T→f= INCONSISTENTE TENDENDO A FALSO
T→v= INCONSISTENTE TENDENDO AVERDADEIRO
⊥→f= PARACOMPLETO TENDENDO A FALSO
⊥→v = PARACOMPLETO TENDENDO A VERDADEIRO
Vicc = VALOR INFERIOR DE CONTROLE DE CERTEZA
Vsci = VALOR SUPERIOR DE CONTROLE DE INCERTEZA
Vscc = VALOR SUPERIOR DE CONTROLE DE CERTEZA
Vici = VALOR INFERIOR DE CONTROLE DE INCERTEZA

Fonte: Modificado, DA SILVA FILHO, 2006.

A saída do controlador paraconsistente apresenta dois tipos: situações de *estados extremos*, que são, Falso, Verdadeiro, Inconsistente e Paracompleto, a as situações de estados denominados *não-extremos*, todos representados no reticulado. Os estados extemos e não

extremos são separados pelos *Valores de controle limite* que podem ser ajustados conforme for necessário para obter resultados mais satisfatórios.

O Algoritmo abaixo descreve como são obtidas as regiões que definem os estados lógicos da saída do Controlador Paraconsistente.

Algoritmo Para-Analisador

/Definição dos Valores/

$V_{scc}= C_1$ */Definição do valor superior de controle de certeza*/
$V_{icc}= C_2$ */Definição do valor inferior de controle de certeza*/
$V_{sci}= C_3$ */Definição do valor superior de controle de incerteza*/
$V_{ici}= C_4$ */Definição do valor inferior do controle de incerteza*/

/Variáveis de Entrada/

μ

λ

/Variáveis de Saída/

Saída discreta = S_1

Saída analógica = S_{2a}

Saída analógica = S_{2a}

*/Expressões matemáticas */ sendo :$0 \leq \mu \leq 1$ e $0 \leq \lambda \leq 1$

$G_{ct}= \mu + \lambda - 1$

$G_c= \mu - \lambda$

/determinação dos estados extremos/

Se $G_c \geq C_1$ então $S_1= V$

Se$G_c \leq C_2$ então $S_1= F$

Se $G_{ct} \geq C_3$ então $S_1=T$

Se$G_{ct} \leq C_4$ então $S_1= \perp$

/determinação dos estados não-extremos/

Para: $0 \leq G_c < C_1$ e $0 \leq G_{ct} < C_3$

se $G_c \geq G_{ct}$ então $S_1 = Qv \rightarrow T$

Senão $S_1 = T \rightarrow v$

Para: $0 \leq G_c < C_1$ e $C_4 < G_{ct} \leq 0$

se $Gc \geq |Gct|$ então $S_1 = Qv \rightarrow \perp$

Senão $S_1 = \perp \rightarrow v$

Para: $C_2 < G_c \leq 0$ e $C_4 < G_{ct} \leq 0$

se $|G_c| \geq |G_{ct}|$ então $S_1 = Qf \rightarrow \perp$

Senão $S_1 = \perp \rightarrow f$

Para: $C_2 < G_c \leq 0$ e $0 \leq G_{ct} < C_3$

se $|G_c| \geq G_{ct}$ então $S_1 = Qf \rightarrow T$

Senão $S_1 = T \rightarrow f$

$G_{ct} = S_{2a}$

$G_c = S_{2b}$

/FIM/

3.2 Aplicações da Lógica Paraconsistente

A Lógica Paraconsistente tem obtido bons resultados em diferentes áreas do conhecimento. Muitos pesquisadores utilizam-se das características singulares da Lógica Paraconsistente Anotado (LPA), como tratamento de contradições, para buscar através de seus trabalhos soluções em que a lógica é devidamente efetiva.

Um dos trabalhos que podemos citar é o desenvolvimento do robô móvel autônomo Emmy (DA SILVA FILHO, 2006). Emmy usa um Sistema de Controle construído com os fundamentos teóricos da LPA, para controlar seus movimentos. Seus circuitos são devidamente interligados destinando-se a sensoriar o ambiente e tratar os sinais de informação conforme os conceitos da LPA. Este trabalho foi de importância fundamental para mostrar a possibilidade de construir Sistemas de Controle de natureza paraconsistente com bom desempenho.

Demário (2013), em seu trabalho: Lógica paraconsistente no processamento de imagens, utiliza os recursos da LPA para resolver problemas de segmentar imagens mamográficas. Utilizando abordagens clássicas e combinando-as convenientemente no reticulado, conseguiu obter resultados promissores.

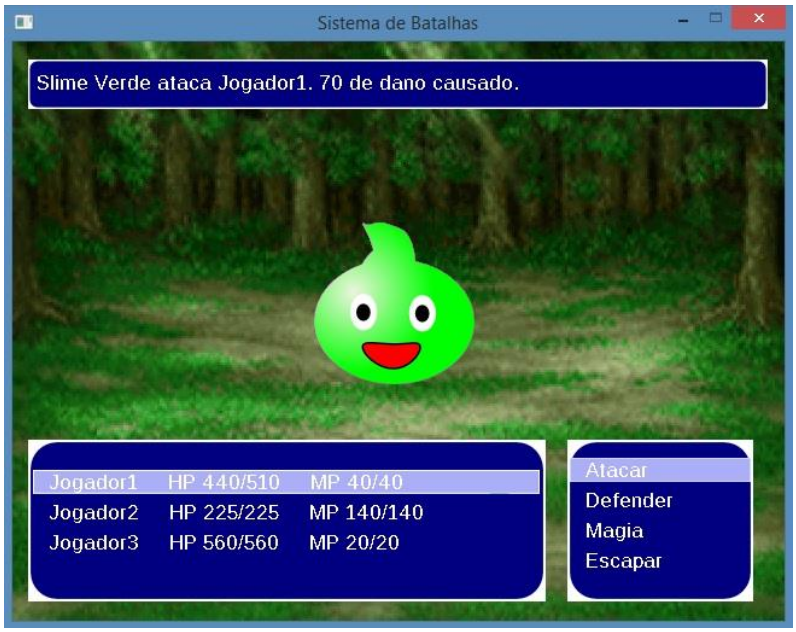
No campo de diagnóstico médico Mário (2006), em seu trabalho, faz uso de Redes Neurais Artificiais Paraconsistentes - RNAP, para sugerir diagnóstico e propor tratamento para maloclusão a partir de variáveis craniométricas. As RNAPs são construídas com algoritmos baseados na LPA, que trata informações de variáveis craniométricas e classificam o tipo de maloclusão existente.

Podemos observar que a LPA está presente nos diversos campos da ciência, auxiliando na resolução de problemas onde a contradição e a incompletude se encontram presentes. Porém, apesar da pesquisa aprofundada, não houve nenhum resultado encontrado quando se trata da utilização da LPA para o campo de desenvolvimento de jogos. Esse fenômeno encorajou o autor a realizar a pesquisa e implementar o Sistema de Batalhas Paraconsistente, que será descrito mais detalhadamente na próxima seção.

4. APLICAÇÃO DO ALGORITMO PARACONSISTENTE NO SISTEMA DE BATALHAS

Para a aplicação do algoritmo paraconsistente proposto no trabalho desenvolvemos o Sistema de Batalhas Paraconsistente, baseado no Sistema de Batalhas desenvolvido por Barbosa (2012), que foi implementado para estudo sobre a inteligência artificial em jogos, utilizando um Sistema Baseado em Regras para definir o comportamento inteligente do personagem, veja a Figura 10. Foi feita algumas modificações na estrutura do código para que o mesmo pudesse atender ao algoritmo para-analisador implementado.

Figura 10 - Interface do Sistema de Batalhas desenvolvido por Barbosa



Fonte: BARBOSA, 2012.

O Sistema de Batalhas (BARBOSA, 2012), é composto por três personagens controlados pelo usuário e um inimigo que usa o algoritmo de IA para tomar suas decisões. Cada personagem, inclusive o inimigo, possui os atributos HP (Hit Points), MP (Magic Points), Força, Destreza e Agilidade (BARBOSA, 2012). Uma das modificações realizadas foi alterar a quantidade de personagens, para facilitar os experimentos, o Sistema de Batalhas Paraconsistente terá apenas um personagem e não três como em Barbosa (2012).

Os atributos força, destreza e agilidade são utilizados para o cálculo de dano durante a batalha, os HPs são os pontos de vida do personagem, a batalha termina quando o HP do

inimigo ou dos personagens controlados pelo jogador acaba. O MP são os pontos de magia usados para a cura, ou seja, recuperação dos pontos de HP.

4.1 O Sistema de Batalhas

Para o desenvolvimento do Sistema de Batalhas (BARBORSA, 2012), foi utilizado o ambiente de desenvolvimento Code::Blocks na versão 13.12. O sistema foi desenvolvido em linguagem C/C++ e para criação da interface multimídia foi utilizada *Simple Directmedia Layer* (SDL).

A SDL é uma biblioteca de desenvolvimento em multiplataforma destinado a prover acesso a baixo nível a funções de áudio, teclado, mouse, joystick, e *hardwaregráficos* via OpenGL[3]. Oficialmente suporta Windows, Mac OS X, Linux, iOS, e Android. SDL é escrito em linguagem C, funciona nativamente com C++, e pode ser utilizada com linguagens como C#, Lua e Python (SDL, 2015).

O inimigo foi desenhado em uma das atividades realizadas na disciplina de Desenvolvimento de Jogos no primeiro semestre de 2013 e foi reutilizado no Sistema de Batalhas Paraconsistente. Para o plano de fundo e os sons foram utilizados recursos livres da internet.

4.1.1 A Interface

Foi escolhido o desenvolvimento do sistema com interface gráfica simplificada, para tornar a demonstração do algoritmo mais didática, veja a figura 11.

No painel superior é exibida a ultima ação ocorrida no jogo. No centro da tela se localiza o inimigo e acima dele foi inserida uma barra que representa os pontos de vida do mesmo.

No painel inferior esquerdo possui algumas informações sobre o personagem controlado pelo jogador. Mais a esquerda está uma imagem que representa o jogador. Ao lado da imagem, na primeira linha está o nome do jogador, na segunda seu *Hit Points* e na terceira os *Magic Points*. O dano causado ao inimigo será exibido no painel superior. Cada vez que um ataque for realizado o comprimento da barra de vida irá diminuir, o tamanho da barra é diretamente proporcional à quantidade de vida do inimigo.

[3]OpenGL é uma biblioteca de desenvolvimeto livre utilizada para construção de aplicativos gráficos, ambientes 3D e jogos

Figura 11 - Interface do Sistema de Batalhas Paraconsistente

No painel inferior direito está o *menu* com as ações que podem ser realizados pelo jogador. O personagem, seja do jogador ou da máquina, tem direito a uma ação por turno. O personagem poderá atacar, defender, usar magia ou fugir.

4.1.2 Ações dos Personagens

Cada personagem por turno pode realizar uma de quatro ações possíveis: atacar, defender, usar magia ou fugir.

Caso o personagem escolha *Atacar* será calculado o dano no oponente escolhido baseado nos seus atributos e nos do oponente. O ataque é calculado através da fórmula utilizada por Barbosa (2012):

$$Dn = F \cdot \left(\frac{De}{Ag}\right) \qquad \text{Eq. III}$$

Onde *Dn* é o dano, *F* é a força do personagem, *De* é a destreza do personagem e *Ag* é a agilidade do oponente.

Se a escolha do personagem for *Defender*, ele ficará em posição de defesa em todo turno corrente, voltando à posição normal apenas quando chegar sua vez de executar uma

ação novamente. Quando um personagem está na posição de defesa, danos causados a ele são reduzidos, pois o foco está na defesa, o que no caso deste sistema de batalhas, aumenta sua agilidade em um determinado percentual.

Ao escolher a opção *Magia* o personagem poderá usar magia para recuperar seus pontos de vida. Cada vez que a magia é utilizada gasta 20 MP e recupera 20 de HP. Primeiro é verificado se o personagem possui a quantidade de MP necessária para executar a magia. Caso personagem não possua os pontos de magia necessários, a magia irá falhar. Se a magia funcionar será recuperado o de HP do personagem selecionado.

Por último, a opção *Escapar* que permite que o personagem tente escapar da batalha. As chances de um personagem escapar são baixas, por isso essa opção só deve ser usada em últimos casos. Caso a fuga seja bem sucedida, a batalha termina sem haver um vencedor.

Figura 12 - Algoritmo do Sistema Baseado em Regras do Inimigo

```
// Normaliza seu status no caso de uso de defesa no turno anterior
normaliza_status_inimigo();

int acao = 0;

        if (inimigo1.hp_atual > 1000){

            inimigo_ataca();
        }

        if ( inimigo1.hp_atual < 1000 && (inimigo1.mp_atual >= 20))
        {
            //Escolhe um ação randômica
                acao = rand() % 3;

                if( acao == 0){
                    inimigo_ataca();
                }
                if( acao == 1){
                    inimigo_defende();
                }
                if( acao == 2){
                    inimigo_magia();
                }
                if( acao == 3){
                    inimigo_fuga();
                }
        }

        if ((inimigo1.hp_atual < 1000) && (inimigo1.mp_atual < 20))
        {
                //Escolhe um ação randômica
                acao = rand() % 2;

                if( acao == 0){
                    inimigo_ataca();
                }
                if( acao == 1){
                    inimigo_defende();
                }
                if( acao == 2){
                    inimigo_fuga();
                }

        }
```

Fonte: BARBOSA, 2012, p.17.

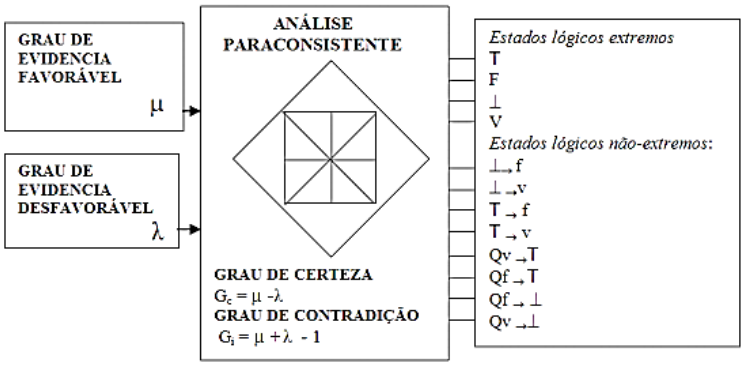
O inimigo poderá executar as mesmas ações que o jogador (atacar, defender, magia e escapar). Em Barbosa (2012) as ações do inimigo são decididas de acordo com o algoritmo implementado no formato de um Sistema Baseado em Regras, como é mostrado na figura 12. O inimigo possui três regras definidas. Para a escolha da ação referente a uma determinada regra uma função *rand( )* foi utilizada. A função *rand( )* gera um número inteiro pseudorrandômico dentro do escopo passado a ela e a partir desse número escolhe-se uma ação. Esse modelo é utilizado para garantir um pouco de imprevisibilidade por parte do personagem controlado pela IA.

O jogo acaba quando os pontos de vida do personagem do jogador ou da máquina acabarem. O grupo que tiver todos os seus personagens com os pontos de vida zerados perde o jogo.

4.2 O sistema de Batalhas Paraconsistente

O personagem controlado pelo Sistema de Batalhas Paraconsistente deve tomar suas decisões com base em evidências que serão tratadas através do Algoritmo Para-Analisador e que substituirá o Sistema Baseado em Regras randômico de Barbosa (2012), conforme o esquema apresentado na Figura 13. (descreva)

Figura 13 – Análise Paraconsistente da LPA2v

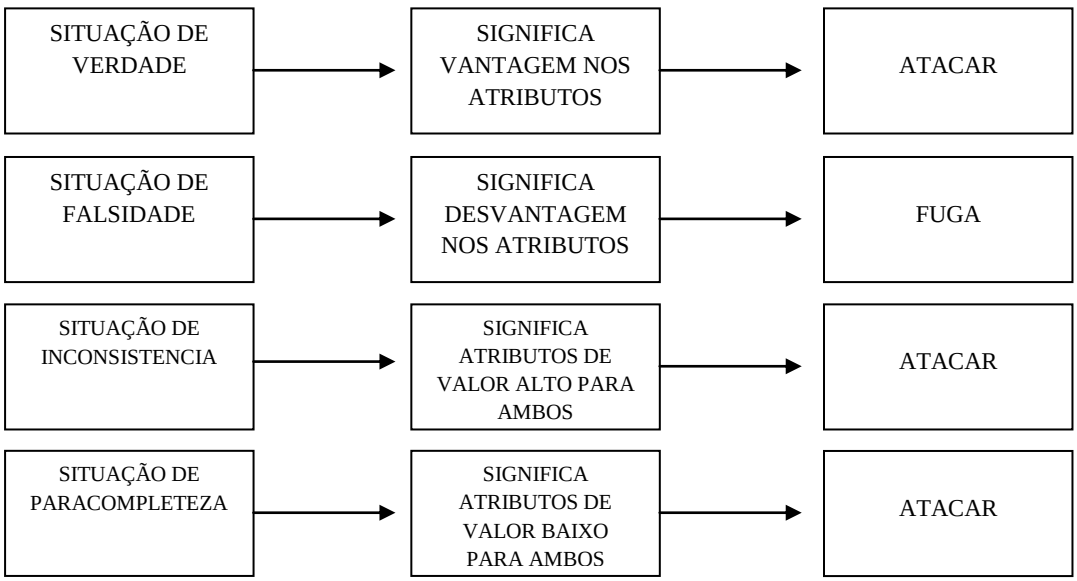


Fonte: DA SILVA FILHO, 2006, p.23.

As informações utilizadas para obter as evidências favorável e desfavorável virão do atributo *Hit Points* do personagem e do adversário. A evidência favorável será diretamente proporcional ao *HP* do personagem controlado pelo jogador e a evidência desfavorável será diretamente proporcional ao *HP* do adversário. O algoritmo recebe esses valores calcula o grau de certeza Gc e o grau de contradição Gct e faz a determinação dos estados lógicos representados pelas regiões do reticulado. Com a informação do estado lógico resultante uma sub-rotina vai direcionar as ações que o personagem irá tomar frente a cada uma das condições encontradas. Essa estrutura é semelhante à utilizada por Da Silva Filho (2006) no robô autônomo Emmy, onde cada comportamento do robô, para contornar obstáculos, está ligado a algum estado lógico resultante no reticulado.

A proposição analisada no Sistema será "enfrentar inimigo". Logo a análise paraconsistente irá analisar as condições de cada personagem para saber se diante de uma determinada situação é viável ter uma atitude ofensiva ou defensiva. Para cada estado lógico resultante referente à proposição "enfrentar inimigo" temos uma rotina programada que possibilitará ao personagem controlado pelo Sistema escolher uma ação. O algoritmo será implementado de forma que toda ação tomada pelo personagem inimigo seja similar a uma atitude coerente tomada por um ser humano.

Algumas situações encontradas e o comportamento programado são esquematizados a seguir:



Para as situações referentes aos estados lógicos *não-extremos* as sequencias de rotina são relacionadas da seguinte forma:

ESTADO RESULTANTE Qf→ T	MAGIA	ESTADO RESULTANTE T→f	DEFESA
ESTADO RESULTANTE Qf→⊥	MAGIA	ESTADO RESULTANTE T→v	DEFESA
ESTADO RESULTANTE Qv→⊥	ATAQUE	ESTADO RESULTANTE ⊥→ f	DEFESA
ESTADO RESULTANTE Qv→ T	ATAQUE	ESTADO RESULTANTE ⊥→v	DEFESA

Os Valores de controle limite foram ajustados para atender o raciocínio do personagem controlado pelo computador. O reticulado da LPA2v com os valores dos graus de certeza e contradição G_c e G_{ct} e na configuração do quadro unitário com 12 regiões, fica ajustado da seguinte forma: $V_{scc} = 1/4$, $V_{icc} = -1/2$, $V_{sci} = 1/2$, $V_{ici} = -4/5$.

A Figura 14 mostra todos os estados lógicos do reticulado com os Valores de controle limite considerados nessa configuração. A Figura 15 mostra o reticulado modelado conforme as ações tomadas pelo personagem controlado pelo computador.

Figura 14 - Representação do reticulado com novos valores limite

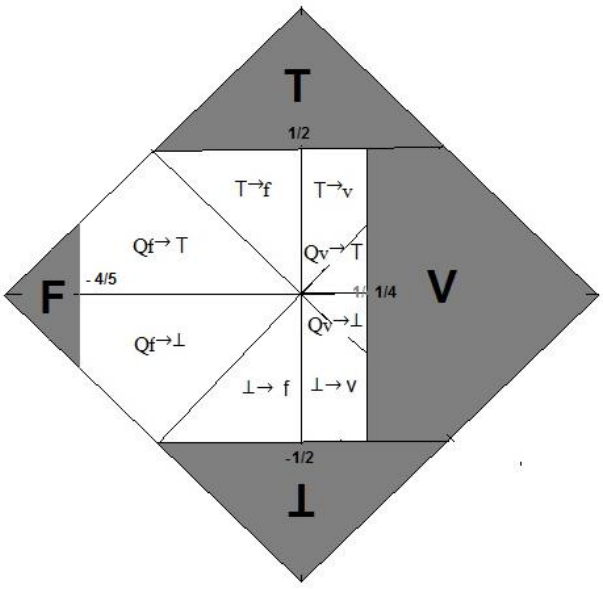
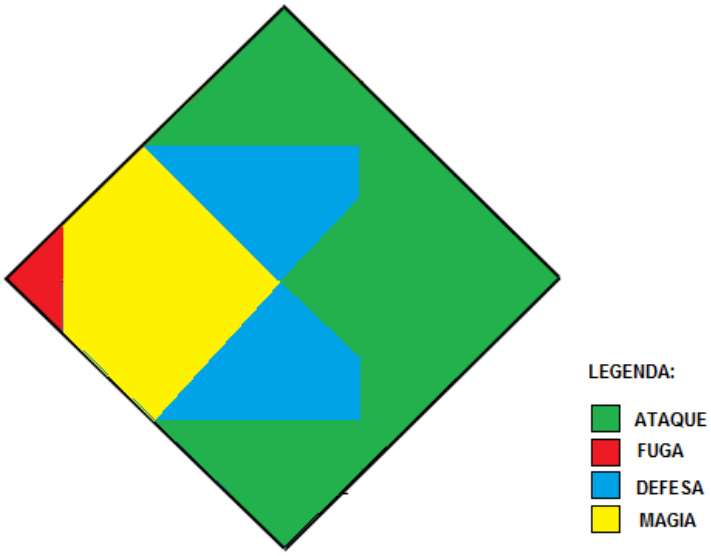


Figura 15 - Reticulado baseado nas ações do inimigo



4.3 Descrição dos testes

Com a finalidade de verificar a viabilidade do algoritmo para-analisador implementado no Sistema de Batalhas Paraconsistente foi feito um conjunto de experimentos.

Tais experimentos se baseiam em projetar *scripts*[4] manualmente para simular jogadores humanos. Após projetar o *script* o jogo é realizado, e o personagem que representa o jogador humano será controlado pelo *script* programado.

Para enriquecer o experimento, foram projetados *scripts* com estratégias diversificadas. As estratégias escolhidas para representar o jogador humano foram:

- **Ofensiva:** Nessa estratégia a prioridade é diminuir a vida do oponente o mais rápido e o máximo possível. Logo há um excesso de ataques, deixando as jogadas defensivas para último caso.
- **Defensiva:** O personagem tem preferencia por começar a partida com ações defensivas para aumentar seus atributos defensivos e manter sua vida o maior possível.
- **Técnica Aleatória:** Cada tática apresentada anteriormente é selecionada de maneira aleatória durante as partidas.

[4]Scripts são módulos com ações pré-programadas.

Para fazer uma comparação mais direta serão feitas partidas também contra o Sistema Baseado em Regras implementado por Barbosa (2012).

Os experimentos foram organizados em dois grupos. O primeiro grupo de experimentos foi realizado com personagens instanciados com atributos estáticos. Tanto o inimigo implementado com o algoritmo para-analisador como o *script* terá um personagem. Esses personagens terão atributos idênticos, e permanecerão com esses atributos durante todas as partidas. O segundo grupo de testes será feito com personagens instanciados de maneira aleatória. Cada implementação terá um personagem com atributos instanciados aleatoriamente e a cada partida esses atributos serão modificados, também de maneira aleatória.

Para realizar o grupo de testes foi feito no código um laço com mil iterações. É um número relativamente alto em termos de partidas de videogame, e rápido em tempo de processamento, portanto suficiente para colher os dados em tempo hábil. Essas partidas foram disputadas contra cada uma das estratégias escolhidas para representar o jogador humano: ofensiva, defensiva e técnica aleatória. Após cada confronto é coletado dados referente ao número de vitórias de cada algoritmo, assim como o número de fugas que houve durante todas as mil iterações. Finalmente mais mil partidas foram disputadas entre o algoritmo paraconsistente e o SBR. A eficiência do sistema será obtida a partir das frequências de vitórias durante as partidas.

5. RESULTADOS OBTIDOS

Para referenciar o algoritmo paraconsistente durante os resultados dos experimentos iremos chamá-lo de Inimigo Paraconsistente – IP. Para garantir que nenhum dos algoritmos envolvidos nos experimentos levasse vantagem por começar a partida e ficar um passo a frente de seu adversário, cada grupo de experimentos foi dividido em duas etapas. A primeira etapa consiste em mil partidas realizadas contra cada *script* projetado com o IP começando a partida. Na segunda etapa foram realizadas mil partidas contra cada *script*, porém os *scripts* iniciam a partida.

Para a primeira etapa do primeiro grupo de testes, onde os atributos dos personagens permanecem estáticos e o IP inicia a partida, tivemos uma dominância do IP sobre os demais *scripts* implementados para simular o comportamento humano. Como podemos observar no Gráfico1 há uma diferença considerável no número de vitórias do IP em cada experimento.

Gráfico 1: Comparação das frequências de vitórias entre o IP e os scripts para personagens com atributos estáticos. IP iniciando as partidas.

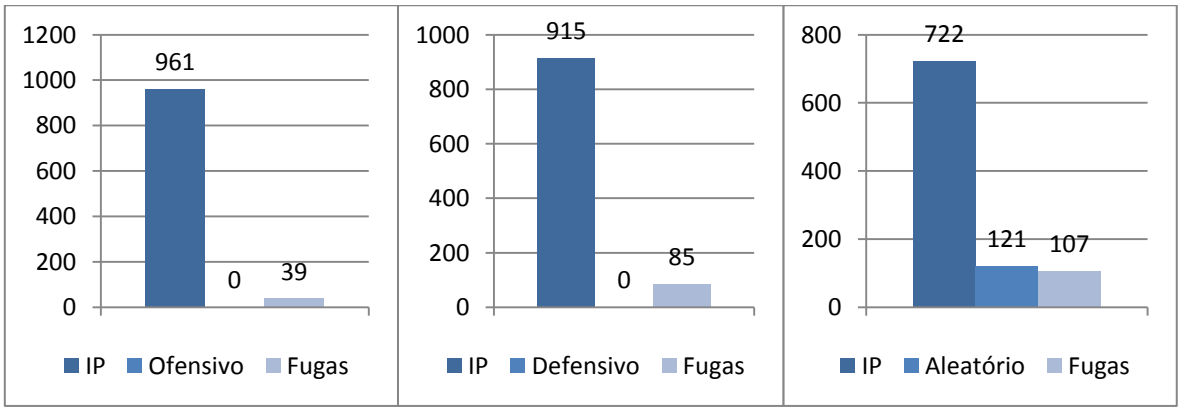
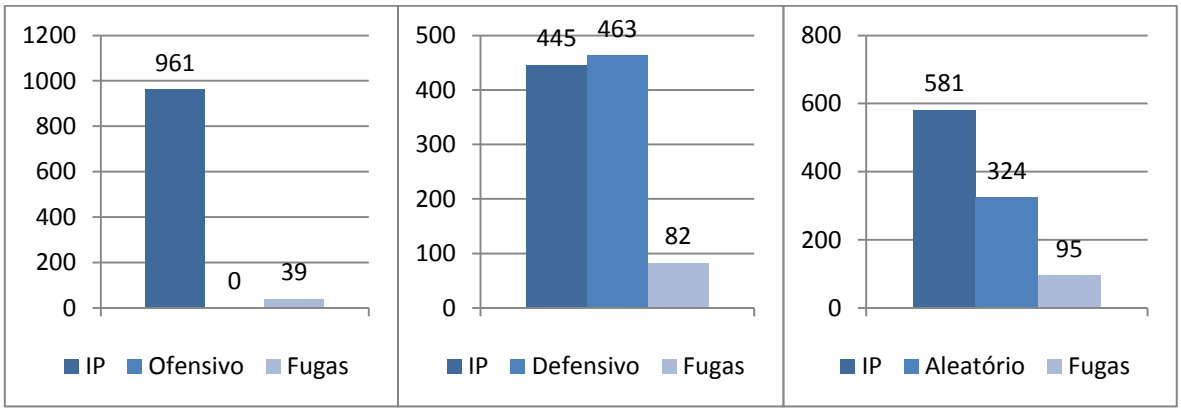


Gráfico 2: Comparação das frequências de vitórias entre o IP e os scripts para personagens com atributos estáticos. Scripts iniciando as partidas.



Na segunda etapa, para as partidas que os *scripts* iniciam, não houve total dominância por parte do IP. Como podemos observar no Gráfico 2, contra a estratégia defensiva encontramos um equilíbrio entre as frequências de vitória, com mínima vantagem do *script* defensivo. Contra a estratégia aleatória o IP também conseguiu um número superior de vitórias, porém com uma diferença menor em relação ao Gráfico 1.

Gráfico 3: Comparação das frequências de vitórias entre o IP e os scripts para personagens com atributos aleatórios. IP iniciando as partidas.

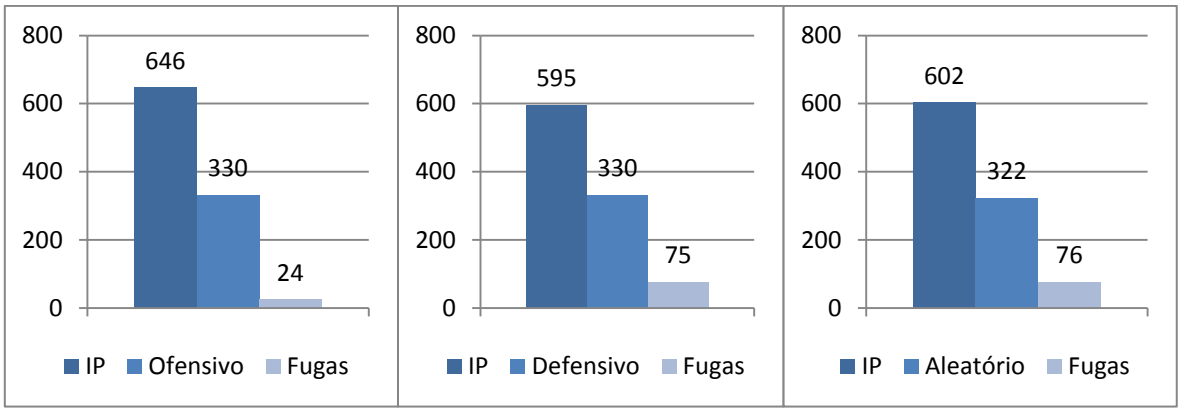
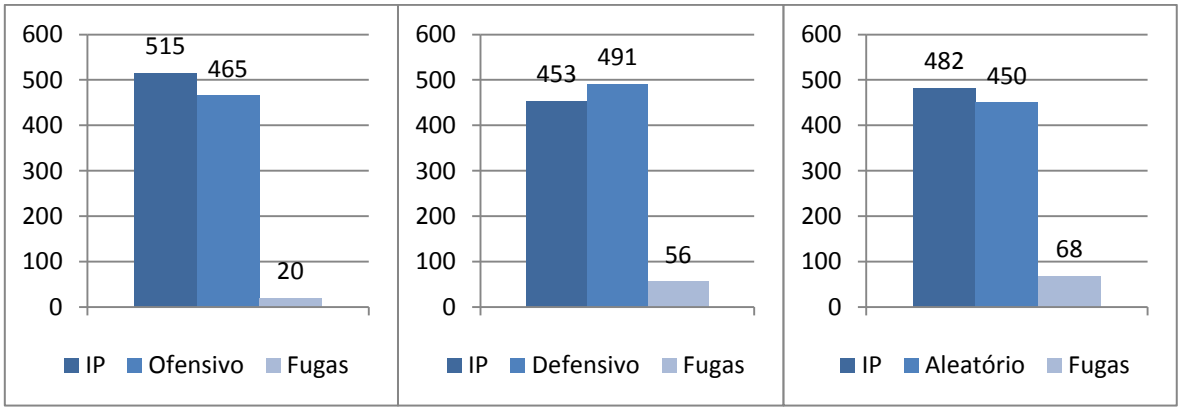


Gráfico 4: Comparação das frequências de vitórias entre o IP e os scripts para personagens com atributos aleatórios. Scripts iniciando as partidas.

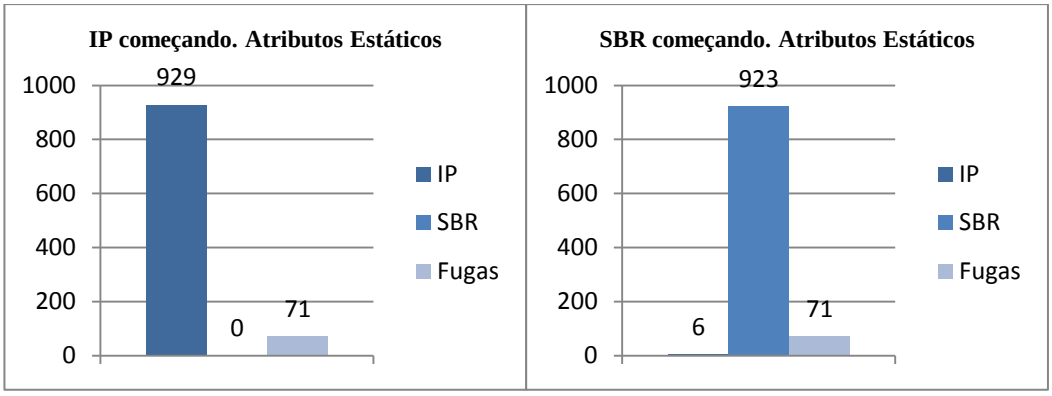


No segundo grupo de testes, onde os personagens possuem atributos aleatórios, encontramos mais equilíbrio dentre as partidas. Na primeira etapa de testes, representado pelo Gráfico 3, o IP ainda apresenta uma frequência de vitórias dominante. Porém a diferença entre os valores são menores, se comparados ao primeiro grupo de testes, e o IP obtém vitória em cerca de 60% das partidas disputadas. Na segunda etapa, Gráfico 4, com os *scripts* iniciando as partidas, encontramos uma diferença mínima entre as frequências de vitórias. Mais uma vez a estratégia defensiva se mostrou mais vitoriosa com uma diferença pequena de partias.

Finalmente, como ultimo teste, é feito um confronto direto entre o algoritmo paraconsistente, representado pelo Inimigo Paraconsistente – IP, e o Sistema Baseado em Regras – SBR implementado por Barbosa (2012).

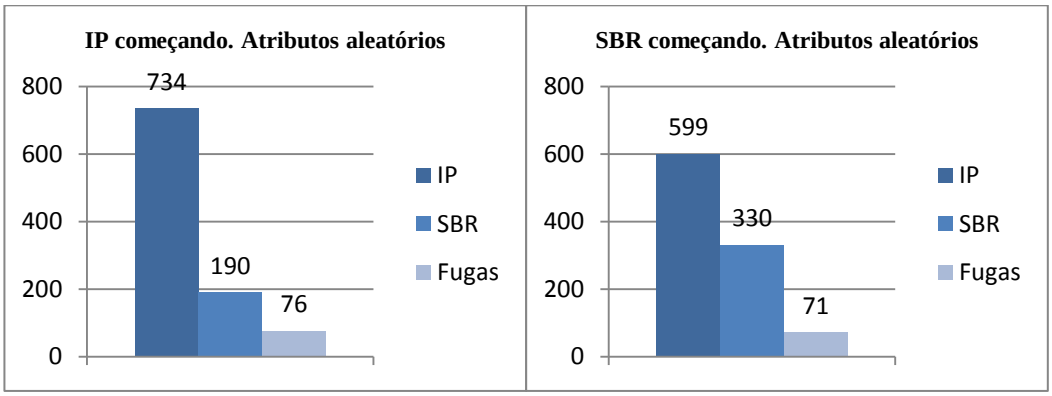
No Gráfico 5 temos os experimentos feitos com personagens de atributos estáticos. O gráfico está dividido em duas seções, a primeira que representa mil partidas iniciadas pelo IP, e a segunda que representa mil partidas iniciadas pelo SBR. Podemos observar que nessa configuração há um domínio por parte do algoritmo que inicia a partida. O IP ainda conseguiu algumas vitórias na segunda seção, porem não suficientes para diminuir a diferença entre o numero de partidas.

Gráfico 5: Comparação das frequências de vitórias entre o IP e SBR personagens com atributos estáticos.



No Gráfico 6 encontram-se os experimentos para atributos aleatórios. Diferente do gráfico anterior, o IP conseguiu dominar as duas seções. Na segunda seção, onde o SBR inicia as partidas, o mesmo conseguiu um maior número de vitórias em relação às primeiras seções, porém não o suficiente para diminuir a diferença contra o IP.

Gráfico 6: Comparação das frequências de vitórias entre o IP e SBR personagens com atributos aleatórios.



Durante os experimentos foram verificados algumas características relevantes sobre o algoritmo:

- Durante os testes o algoritmo demostrou ser rápido, pois funciona entre os turnos de cada personagem apenas com uma pequena quantidade de operações lógicas e aritméticas.
- Ele demonstrou ser robusto, pois garante que nenhuma ação será excluída em momento algum e que sempre, em qualquer ocasião, será escolhida alguma ação. Além disso, devido ao poder de extensão da quantidade de estados lógicos na LPA2v (DA SILVA FILHO, 1999), pode-se elaborar qualquer número de ações necessárias para o personagem e mapeá-las devidamente no reticulado.
- Mostrou também ser efetivo, pois dependendo do modo que é projetado, pode produzir soluções suficientemente desafiadoras e diferenciadas.
- E finalmente demonstrou ser eficiente ao dominar a maioria dos experimentos com um número grande de vitórias. Utilizando apenas duas evidências e pouco processamento mostrou-se como um algoritmo desafiador para o adversário.

Com essas características destacadas podemos afirmar que a Lógica Paraconsistente é uma abordagem alternativa para que um personagem controlado pelo computador possa tomar suas decisões dentro de um jogo. De acordo com o experimento realizado o algoritmo paraconsistente demonstrou ser melhor na maioria das partidas realizadas contra *scripts*, e no confronto direto contra o SBR teve resultados promissores, ficando, no geral, com uma média superior de vitórias.

No entanto é necessário levar em consideração o fato de o Sistema proposto ser simplificado. Para sistemas com um maior espaço de busca a complexidade aumenta no momento de tratar as evidências. Porém como demonstrado em Da Silva Filho (1999), a LPA2v possui operadores para tratar uma quantidade grande de dados e dar os resultados de maneira similar. Para o que foi proposto o algoritmo demonstrou resultados satisfatórios e mais desafiador do que o Sistema Baseado em Regras implementado por Barbosa (2012).

6. CONSIDERAÇÕES FINAIS E TRABALHOS FUTUROS

A proposta deste trabalho foi a de verificar a possibilidade de se construir um Algoritmo alternativo capaz de ser aplicado em inteligência artificial para jogos eletrônicos. Para isso, inicialmente, foi realizada uma pesquisa bibliográfica sobre Lógicas Não-Clássicas, o que apresentou fatos encorajadores, onde foi encontrado trabalhos que diziam ter bons resultados em relação a LPA2v em diversas áreas do conhecimento. Por tanto se iniciou este trabalho na busca de intercalar o campo da Game IA com o da Lógica Paraconsistente Anotada.

Para constatar a viabilidade do uso do algoritmo da LPA2v em jogos foi necessário implementá-lo em linguagem de programação convencional em um ambiente de testes, no caso, um Sistema de Batalhas (BARBOSA, 2012). Para isso algumas modificações, tanto na parte lógica quanto na interface do jogo foram feitas. Foi necessário elaborar estratégias de jogo, como o objetivo de simular o comportamento humano, e modelá-las dentro do algoritmo para-analisador. Por fim foi substituído o SBR existente no jogo pelo algoritmo proposto e testes foram feitos, onde o algoritmo demonstrou resultados satisfatórios.

Uma dificuldade encontrada para o desenvolvimento do trabalho foi durante a pesquisa bibliográfica, sendo que, apesar de se encontrar muito material sobre Inteligência Artificial em Jogos e aplicações da Lógica Paraconsistente Anotada, não se encontrou nada relevante sobre a aplicação simultânea de ambos. No entanto, esta dificuldade mostrou-se motivante, pois se baseando no material estudado verificou-se que o presente trabalho tem característica de inovação.

O fato de trabalhar com Lógicas Não Clássicas também foi um fator motivante. Sabe-se que esta área de conhecimento ainda é pouco trabalhada na graduação, porém muita pesquisa se tem feito com esta área. Portanto o desafio de se pesquisar conteúdos que vão além do escopo do currículo de uma graduação estimulou ainda mais o espirito de pesquisador do autor.

O Sistema de Batalhas utilizado é bem simplificado devido à pequena quantidade de evidências e de ações. Para trabalhos futuros é proposto que a LPA2v seja utilizada em um sistema mais complexo e com um espaço de busca maior. É demonstrado que o algoritmo pode tratar tais sistemas, já que para um número de evidências grande pode-se utilizar-se de

operadores, como AND e OR, para reduzir as entradas (DA SILVAFILHO, 1999). Quanto às ações pode-se ampliar o número de estados lógicos do reticulado para atender a demanda.

Espera-se que este trabalho traga contribuições positivas para o campo da Inteligência Artificial aplicada a jogos, e que trabalhos similares possam fazer uso da abordagem proposta para incrementar o estado da arte e fornecer ferramentas diversificadas para a resolução de problemas pertinentes ao universo da Game IA.

REFERÊNCIAS

ASSASSIN'S CREED UNITY. <*http://assassinscreed.ubi.com/pt-BR*>. Acesso em: Dezembro de 2014.

BARBOSA, S.T.; VEIGA, J.; CARVALHO, C.V.A. *Estudo do uso de técnicas de Inteligência Artificial em Jogos 2D.* Revista eletrônica TECCEN, Vassouras, v-5, n. 1, p. 5-20, 2012.

CROCOMO, M. K. *Desenvolvimento de um Jogo Adaptativo Utilizando um Algoritmo Evolutivo.* Monografia de Conclusão de Curso. IMC-USP: Universidade de São Paulo, 2006.

CROCOMO, M. K. *Um Algoritmo Evolutivo para Aprendizado On-line em Jogos Eletrônicos.* Dissertação de mestrado. IMC-USP: Universidade de São Paulo, 2008.

DA SILVA FILHO, J. I. *Métodos de Aplicações da Lógica Paraconsistente Anotada de anotação com dois valores-LPA2v com construção de algoritmo e Implemantação de Circuitos Eletrônicos.* Tese de doutorado. São Paulo, 1999. 226p.

DA SILVA FILHO, J. I.; ABE J. M. *Introdução à Lógica Paraconsistente Anotada com Ilustrações.* 1.ed. Santos, SP, 2000.

DA SILVA FILHO, J. I.; TORRES, C.R.; ABE J. M. *Robô Móvel Autônomo Emmy: Uma aplicação eficiente da Lógica Paraconsistente Anotada.* Santos, São Paulo. 2006.

DEMARIO, J.L.C. *Lógica paraconsistente no processamento de imagens.* Pontifícia Universidade Católica de São Paulo, São Paulo, Brasil, 2013.

ELETRONIC ARTS (2014). <*http://www.ea.com/*>. Acessado em Julho de 2014.

FUJITA, E. *Algoritmos de IA para jogos*. 2005. Monografia (Graduação em Ciência da Computação) – Universidade Estadual de Londrina.

JOVEM NERD. <http://jovemnerd.com.br/nerdcast/nerdcast-238-profissao-desenvolvedor-de-games/>. Acesso em Janeiro de 2015.

KARLSSON, B. F. F. *Um Middleware de Inteligência Artificial para Jogos Digitais*. Dissertação (Mestrado pelo Programa de Pós Graduação em Informática do Departamento de Informática da Puc-Rio) – Puc-Rio. Setembro de 2005.

KIRBY, N. *Introduction to game AI.*Course Technology, a part of Cengage Learning, Boston, 2011.

KISHIMOTO, A. *Inteligência Artificial em Jogos Eletrônicos.* São Paulo, SP, 2004.

MARIO, M.C.; ABE, J.M.; ORTEGA, N.R.S.; DEL SANTO JR, M. *Análise cefalométrica para auxílio ao diagnóstico ortodôntico utilizando Redes Neurais Artificiais Paraconsistentes.* Workshop on Intelligent Computing Systems, WICS. São Paulo. 2013.

MARTINS, H. G.; TORRES, G. L.; POTIN, L. F. *A lógica paraconsistente anotada.* ComunicarEditora. Santos, SP. 2007.

MILLINGTON, I. *Artificial Intelligence for Games.*Elsevier, San Francisco, 2006.

NEGNEVITSKY, M. (2002).Artificial Intelligence.A Guide to Intelligent Systems.Pearson Education.

NETO, M. C. L.; VENSON N. *Lógica Paraconsistente.* Universidade Federal de Santa Catarina, Santa Catarina, 2002.

SEBRAE.*Brasil tem um dos maiores mercados de games do mundo em 2012.* Disponível em:<http://www.sebrae2014.com.br/Sebrae2014/Alertas/Brasil-tem-o-maior-mercado-de-games-no-mundo-em-2012#.VM7uQJ3F964>. Acesso em: Janeiro de 2015.

SDL. *SimpleDirectMediaLayer*.Disponível em: <*http://www.libsdl.org/*>. Acesso em: Janeiro de 2015.

TATAI, V. K. *Técnicas de Sistemas Inteligentes Aplicadas ao Desenvolvimento de jogos de Computador.* UNICAMP, Campinas, São Paulo, SP, 2003.

THE SIMS 4. Disponível em: <*www.thesims.com/pt_BR/*>. Acesso em: Dezembro de 2014.

UOL. *Brasil será maior mercado de games da América Latina, diz pesquisa.*<http://jogos.uol.com.br/ultimas-noticias/2014/06/25/brasil-sera-maior-mercado-de-games-na-america-latina-em-2014-diz-pesquisa.htm>. Acesso em: Janeiro de 2015.

WATCHDOGS. Disponível em:<*http://watchdogs.ubi.com/watchdogs/pt-br/*>. Acesso em: Dezembro de 2014.