

Um Algoritmo Evolutivo para Aprendizado On-line em Jogos Eletrônicos

Marcio Kassouf Crocomo Eduardo do Valle Simões
Laboratório de Robótica Móvel, Universidade de São Paulo, Brasil



Figura 1: Imagem do jogo produzido

Abstract

Abstract. The goal of this work is to verify if it is possible to apply Evolutionary Algorithms to on-line learning in computer games. Some authors agree that evolutionary algorithms do not work properly in that case. With the objective of contesting this affirmation, this work was performed. To accomplish the goal of this work, a computer game was developed, in which the learning algorithm must create intelligent and adaptive strategies to control the non-player characters using an evolutionary algorithm. Therefore, the aim of the evolutionary algorithm is to adapt the strategy used by the computer according to the player's actions during the game. A review on Evolutionary Computation and the techniques used to produce intelligent behaviors for the computer controlled characters in modern game is presented, exposing the advantages, the problems and some applications of each technique. The proposed game is also explained, together with the implemented algorithms, the experiments and the obtained results. Finally, it is presented a comparison between the implemented algorithm and the Dynamic Script technique. Thus, this work offers contributions to the fields of Evolutionary Computation and Artificial Intelligence applied to games.

Keywords: games, Evolutionary Algorithms, Artificial Intelligence, online adaptation.

Authors' contact:
marciokc@gmail.com
simoes@icmc.usp.br

1. Introdução

Nos últimos anos, muita ênfase foi dada ao realismo dos jogos eletrônicos, fazendo surgir uma variedade de

motores gráficos, físicos, de áudio e de Inteligência Artificial (IA) [Bittencourt e Osório 2006]. Grandes esforços foram empregados principalmente na ambientação gráfica utilizada. Porém, já há algum tempo, a previsão para o mercado de jogos era a de que o seu próximo foco seria nos personagens que se comportam de forma realista e podem aprender e se adaptar, ao invés de personagens com maiores resoluções e com mais quadros por segundo [Sweetser 2002]. Atualmente, é possível observar essa tendência em empresas que possuem jogos que têm se destacado pela qualidade de sua IA, como Black & White¹ e The Sims².

Máquinas de estado finito, nesse contexto, vêm sendo empregadas na elaboração das estratégias utilizadas pelo computador com uma frequência maior do que outros métodos nos jogos atuais [Ponsem 2004]. Isso se deve ao fato de serem fáceis de implementar e geralmente suficientes para atingir o propósito desejado. No entanto, essa técnica torna previsível a estratégia utilizada pelo computador. Uma alternativa promissora para contornar os comportamentos previsíveis é a utilização de Algoritmos Evolutivos (AEs) [Yannakakis 2005].

Os AEs possuem diversas aplicações em jogos [Yannakakis 2005], como por exemplo, adaptar as estratégias utilizadas pelo computador àquelas utilizadas pelo usuário, encontrar caminhos e evoluir comportamentos de personagens controlados pelo computador. Apesar das possibilidades de suas aplicações serem numerosas, os AEs encontram resistência em serem utilizados pelos desenvolvedores de jogos. Alguns autores afirmam que essa técnica (assim como a técnica de Redes Neurais Artificiais) permite a obtenção de comportamentos “não

¹ Lionhead Studios. **Black & White**. Disponível em: <<http://www.lionhead.com/bw2/>>. Acesso em: 16 ago. 2007.

² Electronic Arts. **The Sims**. Disponível em: <<http://thesims.ea.com/>>. Acesso em: 16 ago. 2007.

aceitáveis” durante o processo de aprendizado dos personagens [Ponsem 2004]. Entre esses autores, Spronck [2004] afirma que AEs não são uma alternativa que funcione na prática para o aprendizado on-line em jogos, por não satisfazerem os critérios de eficiência e efetividade necessários. Este artigo é uma resposta a essas afirmações, pois propõe um AE que satisfaça esses requisitos.

Dessa forma, é esperado com este trabalho contribuir para o campo de IA em jogos respondendo a pergunta: É possível construir um AE para o aprendizado on-line de jogos que possa ser utilizado na prática, por satisfazer os requisitos de rapidez, efetividade, robustez e eficiência?

2. Trabalhos Relacionados

Diversos trabalhos envolvendo IA em jogos podem ser encontrados, utilizando técnicas como, por exemplo, Lógica Fuzzy, Flocking, Árvores de Decisão, Máquinas de Estado Finito, Scripting, RNAs e AEs. Uma revisão do uso destas técnicas em jogos pode ser encontrada em [Sweetser 2002]. Com relação ao aprendizado em jogos, o mesmo pode ocorrer de duas formas: on-line e off-line [Spronck et al. 2004]. O aprendizado off-line é aquele onde não é necessária uma interação com o usuário, podendo ocorrer até mesmo antes do jogo ser lançado. Por sua vez, o aprendizado on-line permite que no decorrer do jogo, a adaptação da estratégia utilizada pelo computador em relação ao comportamento do jogador. Spronck et al. [2004] afirma que AEs não satisfazem os dois últimos dos quatro requisitos apontados por ele como necessários no aprendizado on-line e, por isso, não são candidatos a serem utilizados nesses jogos. Os requisitos levantados por esse autor são:

- **Rápido:** o algoritmo de aprendizado deve ser computacionalmente rápido, pois é realizado durante a execução. Dessa forma, um algoritmo lento atrapalha o desempenho do jogo.
- **Robusto:** o mecanismo de aprendizado deve suportar uma quantidade de aleatoriedade significativa, normalmente presente nos jogos comerciais.
- **Efetivo:** as estratégias adaptadas devem ser pelo menos tão desafiadoras quanto as programadas explicitamente. P. Spronck afirma que este requisito exclui métodos de aprendizado aleatórios, como os AEs, pois permitem a geração de estratégias muito ruins para serem utilizadas contra o jogador.
- **Eficiente:** um pequeno conjunto de testes deve ser suficiente para que o aprendizado ocorra. P. Spronck diz que este requisito exclui técnicas de aprendizado lento, como RNAs, AEs e Aprendizado por Reforço, pois tomariam muito tempo dos jogadores para poderem ajustar suas estratégias.

No trabalho de Spronck et al. [2004] é afirmado que AEs e Redes Neurais não são técnicas utilizáveis na prática para a obtenção de um bom aprendizado on-line em jogos e uma técnica chamada Dynamic Scripting (DS) é proposta como alternativa. Como um contra-exemplo para tal afirmação, o trabalho de Stanley et al.

[2005] descreve um jogo que utiliza RNA em seu aprendizado on-line, demonstrando com sucesso a viabilidade de utilização prática dessa técnica. Tal trabalho ganhou o título de melhor artigo no CIG (IEEE Symposium on Computational Intelligence and Games) 2005. Em um trabalho desenvolvido anteriormente, um AE utilizado no aprendizado on-line apresentou o critério de eficiência, no entanto não o de efetividade [Crocomo et. al 2005].

3. O Jogo Desenvolvido

O jogo implementado segue o modelo do simulador elaborado para a realização dos experimentos de Spronck et al. [2004]. Nesse jogo, o jogador será responsável pela estratégia de quatro personagens, contra quatro personagens controlados pelo computador. O jogo é um simulador das batalhas que ocorrem em jogos de RPG do estilo Baldur's Gate³. Esse estilo utiliza os sistemas de jogos mais complexos encontrados em RPGs eletrônicos.

O jogo ocorre com o encontro de dois grupos de personagens. Cada um desses grupos (um controlado pelo computador e outro pelo jogador) é composto por dois personagens magos e dois guerreiros. O equipamento dos personagens é estático. Cada personagem possui duas poções mágicas dentre as três possíveis e cada mago possui sete magias de um total de 21 possíveis. As magias existentes são de variados tipos, como, por exemplo, maldições, invocações, danos e bênçãos.

Cada um dos personagens possui as ações que serão tomadas representadas por scripts compostos por cinco regras para um guerreiro (selecionadas de uma base contendo 20 regras) e 10 regras para um mago (selecionadas de uma base de 50 regras). A Tabela I mostra o exemplo de um script de guerreiro, no qual cada linha representa uma regra.

Tabela 1 – Exemplo de um script

Ação	Alvo	Condições
Beber Poção de Cura	Próprio personagem	Pontos de vida < 50% e personagem possuir poção
Beber Poção contra Paralisação	Próprio personagem	Personagem estar paralisado e possuir poção
Atacar	Oponente com condição paralisado	Existir oponente com condição paralisado e personagem não estar paralisado
Atacar	Mago mais próximo	Existir mago oponente vivo e personagem não estar paralisado
Atacar	Oponente com menor saúde	Personagem não estar paralisado

Quanto mais acima na tabela, maior a prioridade da regra: a cada turno, o guerreiro controlado pelo script da Tabela I testa as condições de

³ BIOWARE. **Baldur's Gate**. Disponível em: <http://www.bioware.com/games/baldurs_gate/>. Acesso em: 16 ago. 2007.

sua primeira regra. Caso elas sejam satisfeitas, a ação da regra é executada. Caso contrário, são testadas as condições da próxima regra. Esse procedimento se repete até que alguma ação seja executada. No caso de nenhuma ação poder ser executada, a ação de “passar o turno” entra em execução (isso significa que o personagem abre mão de seu movimento).

As regras que compõem os scripts são formadas a partir de cinco ações básicas: atacar, beber uma poção, se mover, executar uma magia ou passar a vez. O conjunto contendo quatro scripts (dois para magos e dois para guerreiros) constitui uma estratégia a ser usada por um time. Dois times são colocados no mesmo ambiente e entram em combate utilizando, cada um, sua respectiva estratégia. Quando um time fica sem personagens vivos, é considerado derrotado. Se o time do jogador vencer o time controlado pelo computador, um novo time é criado para dar continuidade ao jogo, resultando em uma próxima batalha.

3.1 O Algoritmo Evolutivo Proposto

O AE proposto possui como cromossomos as estratégias de comando de um time composto por quatro personagens: dois guerreiros e dois magos (um conjunto de quatro scripts). Cada regra que compõe esses scripts é considerada um parâmetro do cromossomo sendo avaliado, resultando em um total de 30 regras por cromossomo: 5 para cada guerreiro e 10 para cada mago. O cromossomo descrito pode ser visto na Figura 2.

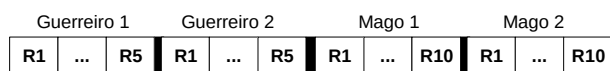


Figura 2: O cromossomo proposto

A população evoluída é composta por quatro cromossomos. O funcionamento do AE proposto é composto por três etapas básicas: geração da população inicial, teste dos indivíduos no ambiente e geração de uma nova população. A Figura 3 ilustra o funcionamento do algoritmo.

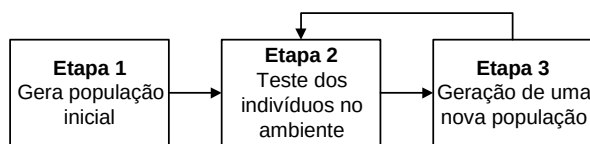


Figura 3: Funcionamento do Algoritmo Evolutivo

Na Etapa 1, é gerada a população inicial contendo quatro cromossomos. Para cada um dos quatro membros da população, são geradas oito estratégias de jogo selecionando-se de forma aleatória as regras para os scripts. É realizado um torneio entre essas estratégias, como ilustra a Figura 4, e a estratégia vencedora é o cromossomo inicial do correspondente membro da população. Os confrontos realizados entre essas estratégias ocorrem colocando as duas para

combater no ambiente de jogo em background, ou seja, esse confronto não é visto pelo usuário do jogo. Com a finalidade de gerar os quatro indivíduos da população inicial, são realizados quatro torneios de estratégias, como o mostrado pela Figura 4. Essa Etapa não compromete o desempenho do jogo, pois ocorre antes das partidas com o usuário começarem.

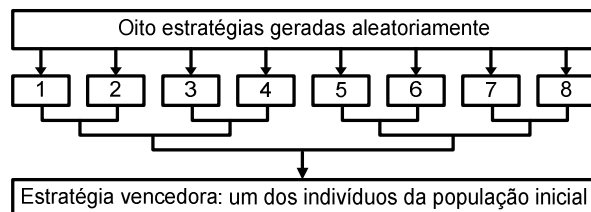


Figura 4: Torneio entre oito estratégias (geradas aleatoriamente). A vencedora irá compor um dos quatro indivíduos da população inicial.

Na Etapa 2, ocorre o confronto entre as estratégias da população sendo evoluída com a estratégia do jogador. Os seguintes procedimentos são realizados:

1. O valor aptidão de cada cromossomo é iniciado como -1 , significando que os cromossomos ainda não possuem um valor de aptidão. É selecionado o primeiro cromossomo da população.
2. Os scripts que compõem o cromossomo são utilizados para controlar o grupo de personagens do computador.
3. Ocorre uma batalha entre os personagens do computador e os personagens do jogador. Após o término da batalha, calcula-se uma pontuação para a estratégia utilizada pelo computador, dada pela função de aptidão do grupo: $Fit(p)$, representada pela Equação 1 a seguir.

$$Fit(p) = \begin{cases} 0,5 - 0,125 \sum_{m \in O} \frac{h(m)}{H(m)} & \{ \forall n \in p \mid h(n) \leq 0 \} \\ 0,5 + 0,125 \sum_{n \in p} \frac{h(n)}{H(n)} & \{ \exists n \in p \mid h(n) > 0 \} \end{cases}$$

Equação 1

sendo:

- $F(p)$: Função de aptidão do grupo p
- $H(n)$: Saúde total do personagem n
- $h(n)$: saúde restante do personagem n após a batalha (número natural entre 0 e $H(n)$)

A aptidão do grupo possui um valor real no intervalo $[0,1]$. É 0 se o grupo perdeu a luta e 0,5 mais a metade da média da porcentagem de saúde restante dos personagens do time.

4. Caso a estratégia do computador tenha sido vitoriosa, a mesma estratégia é mantida para a próxima batalha (retorna para o procedimento 3).
5. Caso o computador tenha sido derrotado pelo jogador, o indivíduo com o cromossomo responsável pela estratégia utilizada recebe um valor de aptidão (*fitness*) igual à média aritmética das pontuações obtidas por sua estratégia no procedimento dois, dada pela Equação 2 abaixo:

$$\text{aptidão} [j] = \frac{\sum_{i=1}^n x_i}{n} \quad \text{Equação 2}$$

sendo:

- j: cromossomo sendo avaliado
- aptidão[j] : valor de aptidão do cromossomo j
- n: número de vitórias + 1 (número de batalhas utilizando a estratégia de j)
- xi: pontuação obtida na i-ésima batalha dada pela Equação 1

6. Se ainda houver um cromossomo na população sem um valor de aptidão (aptidão = -1), então retorna para o procedimento 2, testando o cromossomo em questão. Caso contrário, a etapa dois chega ao fim.

Após todos os indivíduos serem avaliados, é executada a etapa 3, com o objetivo de aproveitar as estratégias que obtiveram maior sucesso para gerar uma nova população, da qual é esperado um maior potencial para superar a estratégia utilizada pelo jogador. Para se gerar a nova população, é inicialmente utilizado o elitismo [Bramlette 1991] como técnica de seleção, escolhendo o melhor indivíduo da população atual, isto é, o indivíduo com maior valor de aptidão (chamado de “pai” da população). Após o “pai” da população ser selecionado, o operador evolutivo de crossover [Bramlette 1991] é responsável por determinar três cromossomos da nova população, tendo 50% de chance de receber o valor do cromossomo do “pai” e 50 % de chance de receber o valor do cromossomo antigo; o quarto cromossomo é exatamente igual ao cromossomo do “pai”.

Após realizado o crossover, cada parâmetro do cromossomo resultante possui 5% de chance de ser substituído por uma regra aleatória que não pertença ao script em questão. Após a nova população ter sido gerada, a mesma é avaliada chamando novamente a etapa dois. O laço entre as etapas dois e três (Figura 3) representa a capacidade de adaptação do algoritmo, presente durante todo o jogo. O laço só termina no momento em que o jogo for encerrado pelo jogador.

3.2 Os experimentos

Com a finalidade de responder a questão apresentada ao final do capítulo 1 (Introdução), é necessário verificar se o AE implementado durante o trabalho apresenta os quatro requisitos para um aprendizado on-line utilizável na prática: Rápido, Efetivo, Robusto e Eficiente. O AE proposto foi projetado de maneira a cumprir pelo menos três dos requisitos:

- Rápido: o AE proposto é rápido, pois funciona entre as batalhas exigindo apenas uma pequena quantidade de operações lógicas e aritméticas. Desta forma a próxima batalha é rapidamente carregada, sem que o processamento feito pelo computador seja notado pelo usuário, ou seja, o jogador não precisa ficar

esperando uma grande quantidade de processamento para continuar jogando.

- Robusto: o AE proposto é robusto, pois nenhuma regra é excluída em momento algum, ou seja, todas as regras podem voltar a ser utilizadas em algum momento do jogo devido ao operador evolutivo de mutação. Desta forma, mesmo que uma regra seja prejudicada devido ao estilo de um jogador, a mesma não é excluída permanentemente. Ela poderá reaparecer devido à mutação e caso o jogador mude de estratégia, ela poderá ter um efeito positivo na pontuação, sendo incorporada na população de scripts.
- Efetivo: P. Spronck afirma que AEs são incapazes de garantir que as soluções geradas sejam pelo menos tão desafiadoras quanto soluções projetadas manualmente [Spronck et al. 2004]. No entanto, também afirma que a técnica DS desenvolvida sempre produz soluções pelo menos tão desafiadoras quanto aquelas projetadas manualmente, por utilizar regras baseadas em domínio de conhecimento (as regras pertencentes à base de regras são projetadas manualmente) [Spronck et al. 2004]. O AE proposto neste trabalho possui como função realizar a seleção de regras também baseadas em domínio de conhecimento. Sendo assim, utilizando a mesma argumentação de P. Spronck, o AE proposto também cumpre o requisito de ser efetivo.

Foi necessário realizar experimentos para verificar se o quarto requisito (eficiência) é cumprido pelo algoritmo. Para esta finalidade, foram repetidos os testes realizados durante o trabalho de P. Spronck para verificar a eficiência da técnica DS. Tais experimentos se baseiam em projetar scripts manualmente que simulem um jogador humano. Após o script ser projetado, o jogo é realizado, sendo que um conjunto de personagens (personagens do jogador) é controlado pelo script pré-concebido, e os outros (personagens do computador) controlados pelos scripts gerados pelo AE proposto. As estratégias projetadas para simular um jogador humano são:

- Ofensivo: os personagens do jogador possuem como prioridade diminuir a vida de seus oponentes na maior velocidade possível: guerreiros sempre atacam o inimigo mais próximo, enquanto os magos usam suas magias de dano mais fortes.
- Deteriorante: os guerreiros iniciam a batalha utilizando poções contra paralisia; após isso atacam o inimigo mais próximo. Os magos utilizam todas as suas magias para debilitar os oponentes (como, por exemplo, magia de paralisação) durante os primeiros turnos.
- Amaldiçoante: guerreiros sempre atacam o inimigo mais próximo; os magos utilizam magias de invocação, redução de atributos, e magias de controle.
- Defensivo: guerreiros começam bebendo poções que reduzam dano do elemento fogo (diminuindo dano de algumas magias); após isso, atacam o inimigo mais

próximo. Os magos usam todas suas magias defensivas e magias de invocação.

As técnicas propostas até o momento não apresentam variações durante a batalha. Isto geralmente não ocorre com um oponente humano, portanto outras três técnicas compostas foram criadas:

- Técnica aleatória: a cada batalha, uma das quatro táticas apresentadas anteriormente é selecionada de maneira aleatória.
- Técnica aleatória para cada personagem: para cada batalha, cada um dos personagens utiliza aleatoriamente um script relativo a uma das primeiras quatro técnicas apresentadas anteriormente. A escolha de um script para um personagem não depende da escolha dos scripts para os outros personagens do grupo.
- Mudança de estratégias: O grupo começa utilizando uma das quatro primeiras técnicas aleatoriamente. Enquanto a técnica utilizada pelo grupo é vitoriosa, ela é mantida; quando a técnica é derrotada, outra é selecionada.

Para avaliar a eficiência das técnicas desenvolvidas pelo AE em relação às técnicas apresentadas para simular o jogador, a cada batalha foi calculada a aptidão média dos grupos durante as últimas 10 batalhas. Quando este valor for maior para o grupo controlado pelo computador, é dito que a estratégia do computador dominou a estratégia do usuário. A partir da estatística apresentada, serão verificados dois valores:

1. Ponto de equilíbrio médio: número da primeira batalha após a qual o grupo controlado pelo computador domina o grupo do usuário por pelo menos 10 batalhas consecutivas.
2. Ponto de equilíbrio absoluto: número da primeira batalha após a qual um número consecutivo de batalhas em que o grupo do computador vence o grupo do usuário nunca é seguido por um número maior de batalhas consecutivas em que o grupo do jogador vence o grupo do computador.

Os valores destes pontos de equilíbrio são avaliados para revelar a eficiência do algoritmo, sendo que quanto menores os valores encontrados mais eficientes são os algoritmos. As simulações realizadas são mostradas abaixo.

1. DS X Jogador Humano
2. AE X Jogador Humano
3. AE X DS

Os resultados coletados durante estas simulações foram avaliados com o objetivo de verificar a eficiência do AE implementado. Estes resultados também possibilitam a comparação entre os dois algoritmos de aprendizado on-line: o DS proposto por P. Spronck e o AE proposto neste trabalho.

3.3 Resultados Obtidos

Depois de implementados os algoritmos de aprendizado, foram realizados experimentos para verificar se os mesmos apresentam capacidade de adaptação. Para isso, os personagens adaptados pela técnica DS disputaram duas mil partidas contra personagens com a estratégia ofensiva, explicada no capítulo anterior. Em cada partida, as pontuações das duas equipes são comparadas. A equipe com maior pontuação é considerada a vencedora na partida em questão.

Para melhor compreensão dos dados coletados neste experimento, os resultados foram representados em um gráfico, utilizando média móvel com largura 10. A equação da média móvel utilizada está representada pela Equação 2 abaixo:

$$p_i = \frac{\sum_{k=i-9}^i f(k)}{10} \quad 10$$

Equação 2

sendo:

p_i : “pontuação média” explicada no texto acima.

$f(k)$: A pontuação da equipe sendo avaliada, explicada pela função de avaliação da técnica DS.

A função $f(k)$ acima, é dada pela Equação 3 abaixo. Tal equação é utilizada na técnica DS de maneira análoga a Equação 1, utilizada no AE deste trabalho.

$$F(p) = \begin{cases} 0 & \{\forall n \in p \mid h(n) \leq 0\} \\ 0,5 + 0,125 \sum_{n \in p} \frac{h(n)}{H(n)} & \{\exists n \in p \mid h(n) > 0\} \end{cases}$$

Equação 3

sendo:

- $F(p)$: Função de aptidão do grupo p
- $H(n)$: Saúde total do personagem n
- $h(n)$: saúde restante do personagem n após a batalha (número natural entre 0 e $H(n)$)

Os pontos de equilíbrio médio e absoluto encontrados no experimento representado pela Figura 5 foram 54 e 92 respectivamente. O mesmo experimento foi realizado para se verificar a capacidade de adaptação do AE. Desta vez, para a exibição dos dados, foi utilizada a média móvel feita com a pontuação obtida pela função de avaliação da equipe utilizada no AE, representada pela Equação 1. Os dados coletados desta forma estão representados no gráfico da Figura 6, e apresentaram pontos de equilíbrio médio e absoluto de 41 e 43 respectivamente.

A Figura 5 e a Figura 6 permitem verificar que os algoritmos implementados fornecem ao computador a capacidade de gerar estratégias que vençam a estratégia sendo enfrentada. Ambos os algoritmos necessitaram de um pequeno número de partidas para superar a estratégia pré-elaborada, sendo isso uma evidência de que os algoritmos satisfazem o critério de eficiência.

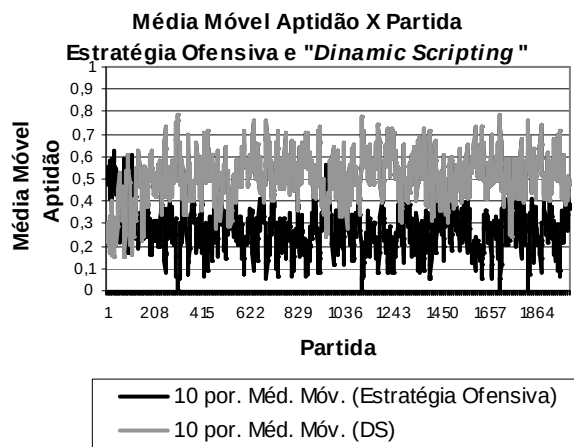


Figura 5 - DS Disputando Contra a Estratégia Ofensiva, Gráfico de Média Móvel Aptidão por Partida

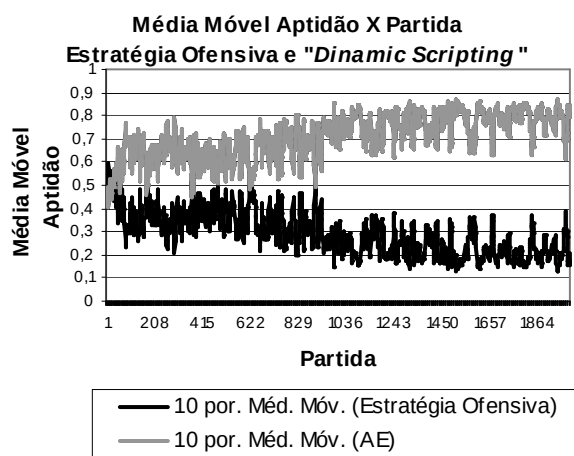


Figura 6 - AE Disputando Contra a Estratégia Ofensiva, Gráfico de Média Móvel Aptidão por Partida

Para cada uma das estratégias pré-elaboradas explicadas no capítulo anterior, foram realizados 21 testes como os descritos acima contra cada algoritmo de aprendizado implementado, e para cada uma das estratégias compostas (técnica aleatória, aleatória para cada personagem, e mudança de estratégia) foram realizados 11 testes. O número de testes realizados foi assim escolhido para reproduzir os testes realizados pelo criador da técnica DS no artigo [Spronck et al. 2004]. Os resultados baseados nos pontos de equilíbrio médio e absoluto podem ser visualizados na Tabela II e na Tabela III. É importante observar que, de acordo com as definições dos pontos de equilíbrio, quanto menores os valores encontrados, mais eficiente é o algoritmo.

A Tabela I representa os resultados obtidos pela técnica DS neste trabalho. A tabela referente aos mesmos experimentos realizados pelo criador da técnica DS pode ser encontrada em [Spronck et al. 2004]. Nos dois trabalhos, a estratégia “Mudança de Estratégia” foi a indicada pela mediana como a mais difícil de ser vencida. É possível verificar também que as estratégias pré-elaboradas possuem algumas diferenças:

Tabela II – Resultados obtidos confrontando os scripts desenvolvidos pela técnica DS com as estratégias pré-elaboradas da primeira coluna.

Dynamic Scripting – Ponto de Equilíbrio Médio				
Estratégias	Menor	Maior	Média	Mediana
Ofensiva	12	1129	88	38
Deteriorante	11	36	20	18
Amaldiçoante	11	51	26	22
Defensiva	11	48	18	12
Aleatória	11	214	70	53
Personagens Aleatórios	11	35	17	16
Mudança de Estratégia	11	102	60	56
Dynamic Scripting – Pontos de Equilíbrio Absoluto				
Estratégias	Menor	Maior	Média	Mediana
Ofensiva	15	1919	158	47
Deteriorante	1	832	112	28
Amaldiçoante	6	115	39	31
Defensiva	1	80	21	12
Aleatória	70	---	406 ⁴	174
Personagens Aleatórios	1	70	21	18
Mudança de Estratégia	62	1938	507	254

1. A estratégia “Deteriorante” apresentou maior dificuldade a ser vencida neste trabalho do que em [Spronck et al. 2004].
2. A estratégia Defensiva apresentou maior facilidade a ser vencida neste trabalho do que em [Spronck et al. 2004].
3. A estratégia “Aleatória” apresentou neste trabalho um caso em que a técnica DS não conseguiu obter um ponto de equilíbrio absoluto (tal verificação encontra-se explicada mais adiante). Tais diferenças são naturais dado que o ambiente construído neste trabalho não é exatamente igual, embora tenha sido montado de maneira a se assemelhar ao ambiente criado por Spronck [2004]. Além disso, as estratégias foram montadas por especialistas diferentes, embora sigam a mesma idéia nos dois trabalhos.

O importante é verificar a eficiência da técnica DS, independente das diferenças do ambiente e das estratégias enfrentadas. Com relação a isso, observações feitas em [Spronck et al. 2004] são também válidas para a Tabela II e encontram-se realizadas mais adiante.

Tabela III – Resultados obtidos confrontando os scripts desenvolvidos pelo AE com as estratégias pré-elaboradas da primeira coluna.

Algoritmo Evolutivo – Ponto de Equilíbrio Médio				
Estratégias	Menor	Maior	Média	Mediana
Ofensiva	11	167	61	40
Deteriorante	11	170	30	17
Amaldiçoante	11	354	54	22
Defensiva	11	124	18	11
Aleatória	11	214	70	53
Personagens Aleatórios	11	35	17	16
Mudança de Estratégia	11	102	60	56

⁴ Neste caso, o valor da média foi calculado utilizando apenas os 10 testes em que um ponto de equilíbrio absoluto foi encontrado.

Algoritmo Evolutivo – Ponto de Equilíbrio Absoluto				
Estratégias	Menor	Maior	Média	Mediana
Ofensiva	1	1331	192	107
Deteriorante	1	328	81	28
Amaldiçoante	1	1326	174	38
Defensiva	1	137	13	4
Aleatória	6	1798	505	102
Personagens Aleatórios	1	340	67	25
Mudança de Estratégia	4	1996	776	506

Analisando as duas tabelas (Tabela II e Tabela III), é possível observar que:

1. Em todos os experimentos, e para os dois pontos de equilíbrio, a média sempre possui valor maior que a mediana. Isso se explica pelo fato de serem raras as ocorrências de pontos de equilíbrio com valores altos.
2. Ambos os algoritmos mostraram médias e medianas factíveis para os pontos de equilíbrio médio, isto é, para que a adaptação ocorra, é necessário que o jogador jogue uma quantidade de partidas aceitável (levando em consideração que cada partida leva geralmente menos de um minuto no jogo implementado). Tal constatação indica que os algoritmos são eficientes.
3. Comparando as duas tabelas, é difícil afirmar qual é o melhor algoritmo, sendo que em alguns testes, a técnica DS obteve melhores resultados, e em outros a utilização do AE teve melhor desempenho.
4. Ao enfrentar a estratégia “Aleatória”, A técnica DS não conseguiu obter um turning point absoluto em um dos experimentos. A ocorrência de tal acontecimento é dada ao fato de que em algumas partidas, devido ao fator sorte, a técnica DS pode aumentar o peso de regras não adequadas, e até mesmo excluir regras necessárias para obter uma estratégia vencedora. Tal limitação do algoritmo já havia sido constatada pelos desenvolvedores da técnica. O fato de permitir que o peso mínimo para uma regra seja zero, resulta em uma grave desvantagem, pois com o passar do tempo, a técnica perde a capacidade de adaptar suas estratégias.

Para chegar a uma melhor comparação entre os algoritmos, a próxima etapa dos testes foi realizada, confrontando no jogo as duas técnicas. Ou seja, enquanto um grupo de personagens é evoluído pelo AE, o grupo oponente foi adaptado pela técnica DS. Os pontos de equilíbrio obtidos pelo AE podem ser verificados na Tabela IV. Enquanto os obtidos pela técnica DS se encontram na Tabela V. Para obtenção destes pontos de equilíbrio foram realizados 21 testes, cada um composto por duas mil partidas.

Tabela IV– Pontos de Equilíbrio obtidos pelo AE ao confrontar a técnica DS

AE – Ponto deEquilíbrio Médio			
Menor	Maior	Média	Mediana
11	393	49	12
AE – Ponto deEquilíbrio Absoluto			
Menor	Maior	Média	Mediana
2	--	598	391

Tabela V– Pontos de Equilíbrio obtidos pela técnica DS ao confrontar o AE

DS – Ponto deEquilíbrio Médio			
Menor	Maior	Média	Mediana
11	240	55	38
DS – Ponto deEquilíbrio Absoluto			
Menor	Maior	Média	Mediana
76	--	1172	1997

Olhando os pontos de equilíbrio obtidos a partir deste experimento, é possível observar que:

1. Tanto o AE quanto a técnica DS encontraram casos em que não foi possível obter um ponto de equilíbrio absoluto. Dos 21 testes realizados, o AE não conseguiu obter um ponto de equilíbrio absoluto em 2 testes. Já a técnica DS não foi capaz de obter este ponto de equilíbrio em 5 testes.
2. A mediana do ponto de equilíbrio absoluto obtida pela técnica DS foi de 1997. Este valor é muito alto considerando o número de partidas realizadas em cada teste (2000), o que é um indicador de que o AE dominou a técnica DS na maioria das partidas.
3. As médias e as medianas apresentaram melhores resultados para as estratégias desenvolvidas pelo AE.

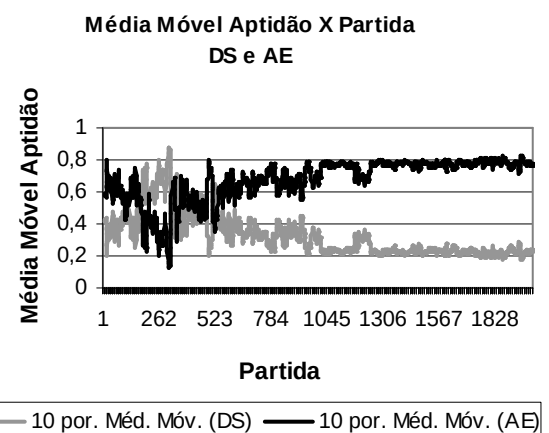


Figura 7 - AE Disputando Contra a Técnica DS, Gráfico de Média Móvel Aptidão por Partida

A fim de verificar as constatações 2 e 3 realizadas, foram verificados os comportamentos dos testes realizados. Foi constatado que o grupo adaptado pela técnica DS perde a capacidade de adaptação após alguns jogos. Tal constatação é coerente com a observação realizada no experimento anterior, no qual a técnica DS não encontra um ponto de equilíbrio absoluto ao confrontar a estratégia “Aleatória”. A Figura 7 mostra um dos testes realizados onde é possível visualizar a perda da capacidade de adaptação da técnica DS.

Para colher dados capazes de confirmar a superioridade do AE sobre a técnica DS ao longo de um jogo composto por 2000 partidas, foi utilizado um método que utiliza como critério o erro máximo de estimação com um nível de confiança de 95% [Bussab e Morettin 2006]. O método utilizado é explicado pela Figura 8.

1 Repita
2 Realize um jogo composto por 2000 partidas;
3 Calcule a frequência das partidas vencidas pelo grupo adaptado pelo Algoritmo Evolutivo;
Até que (Número de Jogos > 200) ou (erro máximo < 0.05 * (Média dos valores calculados no passo 3)).

Figura 8 - Método Adotado para a Coleta dos Dados

No passo 2, o jogo é realizado como no experimento relatado anteriormente, realizando 2000 partidas. No passo 3 o valor calculado da amostra (frequência das partidas vencidas pelo grupo adaptado pelo Algoritmo Evolutivo durante o jogo i) é chamado de x_i e dado pela razão abaixo:

$$x_i = \frac{VAE}{NP}$$

sendo:

- **VAE** Número de vitórias obtidas pelo grupo cuja estratégia é evoluída pelo AE
- **NP** Número total de partidas realizadas, (2000 neste caso).

No critério de parada, o erro máximo (e) é calculado a partir dos valores amostrados até o momento [Bussab e Morettin 2006]. A fórmula para cálculo deste erro é dada pela Equação 7:

$$e = \sqrt{\frac{s^2}{n}}$$

Equação 7

sendo:

- e erro máximo
- s^2 variância amostral
- n número de jogos realizados (amostras)

A variância amostral (s^2) é estimada através da Equação 8 abaixo.

$$s^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2$$

Equação 8

sendo:

- x_i Valor da amostra
- $\bar{x}$ Média das amostras
- n Nmero de jogos realizados (amostras)

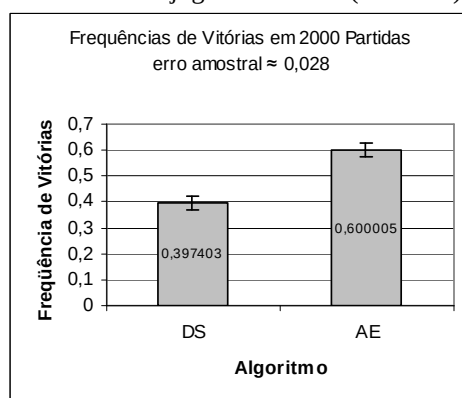


Figura 9 – Frequência de Vitórias das Técnicas AE e DS em 2000 Partidas

Os resultados obtidos encontram-se no gráfico da Figura 9, onde as colunas indicam as frequências obtidas de vitórias do AE e da técnica DS, e o intervalo indicado sobre as colunas representa o intervalo de possíveis valores para as frequências esperadas, considerando o erro máximo encontrado.

Tais resultados permitem concluir, com um nível de confiança de 95%, que em um jogo composto por 2000 partidas, as estratégias geradas pelo AE vencem as estratégias geradas pela técnica DS na maioria das vezes. A afirmação é possível de ser realizada, pois é visível no gráfico que o pior caso do AE (frequência encontrada de vitórias do AE **menos** o erro máximo) é melhor que o melhor caso da técnica DS (frequência encontrada de vitórias da técnica DS **mais** erro máximo). Tal resultado é esperado, dada a constatação de que a técnica DS perde a capacidade de adaptar sua estratégia após um determinado número de partidas, sendo que 2000 partidas são suficientes para se verificar tal acontecimento, como visto no experimento anterior. Os casos de empate não foram exibidos no gráfico, pois se mostraram desprezíveis (casos de empates e erro máximo de aproximadamente 0%).

Os resultados explicados até o momento são suficientes para apontar os Algoritmos Evolutivos como uma melhor opção a ser utilizada em jogos nos quais a adaptação deve ocorrer continuamente. Em jogos em que ocorrem poucas partidas, as duas técnicas seriam aceitáveis, pois a técnica DS só perde sua capacidade de adaptação após um número de partidas suficiente para reduzir os pesos de suas regras a zero (a quantidade de partidas varia de acordo com fatores como o espaço de busca e a quantidade de aleatoriedade do jogo). No entanto, isso não permite comparar as duas técnicas com relação ao critério de eficiência, pois o mesmo diz respeito a um número pequeno de partidas para que o aprendizado ocorra. Para comparar este critério, o último experimento foi repetido, mas desta vez o número de partidas por jogos foi diminuído, realizando testes para 300, 200, 100 e 50 partidas por jogo. Os resultados do teste podem ser visualizados na Tabela VI abaixo.

Tabela VI – Comparação das frequências de vitórias entre o AE implementado e técnica DS. Resultados coletados utilizando 300, 200, 100 e 50 partidas por jogo.

Partidas por jogo	Frequência de Vitórias		
	DS	AE	Erro amostral ≈
300	0,514	0,475	0,026
200	0,505	0,487	0,025
100	0,509	0,479	0,025
50	0,463	0,521	0,023

É válido avisar que a soma das frequências de uma linha da tabela não é exatamente um, pois existem os casos de empate (que possuem valores a partir da quarta casa decimal) e também devido a arredondamentos realizados pelo computador. Estes casos de empate, mais uma vez não foram mostrados na tabela acima, pois se mostraram desprezíveis. Observando os dados resultantes dos experimentos, é possível verificar que as frequências de vitórias obtidas

para o AE e para a técnica DS foram similares. Só é possível concluir que a eficiência de uma técnica de aprendizado 'A' é superior à eficiência da outra técnica de aprendizado 'B', se o pior caso de 'A' for superior ao melhor caso de 'B'. Ao dizer pior ou melhor caso, é considerado o erro máximo, assim como feito no experimento com 2000 partidas por jogo. De outra maneira, podemos afirmar a superioridade da técnica 'A' com relação à técnica 'B' caso a inequação abaixo seja satisfeita [Bussab e Morettin 2006]:

$$FVA - ea > FVB + eb$$

Sendo:

- FVA Frequência de Vitórias do Grupo cujos scripts foram adaptados pela técnica 'A'
- FVB Frequência de Vitórias do Grupo cujos scripts foram adaptados pela técnica 'B'
- ea erro máximo da Percentagem de Vitórias do Grupo cujos scripts foram adaptados pela técnica 'A'
- eb erro máximo da Percentagem de Vitórias do Grupo cujos scripts foram adaptado pela técnica 'B'

Como consequência dos casos de empate terem se mostrado desprezíveis, o erro amostral aproximado (representado na Tabela VI) é igual para **ea** e **eb** ($ea \approx eb$). Em nenhum dos resultados apresentados pela Tabela VI pode ser constatada superioridade na eficiência da técnica DS sobre o AE, já que, embora alguns experimentos apontem a técnica DS como vencedora (experimentos com 300, 200 e 100 partidas por jogo), ao considerar o erro amostral encontrado tal superioridade não pode ser admitida, pois nenhum dos experimentos torna verdadeira a inequação apresentada. De maneira análoga, ao se comparar a superioridade do AE sobre a técnica DS, é possível verificar a superioridade do AE no último experimento realizado (50 partidas por jogo), pois os dados coletados tornam válida a inequação verificada.

É possível concluir, com base nos dados obtidos, que a eficiência apresentada pelo AE se mostrou melhor ou próxima a da técnica DS, que já teve sua eficiência verificada no trabalho de Spronck. Com tal verificação tornou-se evidente que Algoritmos Evolutivos podem satisfazer o critério de eficiência para um aprendizado *on-line* que funcione na prática. Atingido este objetivo, os experimentos foram finalizados.

3. Conclusão

A proposta deste trabalho foi a de verificar a possibilidade de se construir um Algoritmo Evolutivo (AE) capaz de ser aplicado de fato no aprendizado on-line em jogos, capacidade esta apontada como impossível por alguns autores [Spronck et al. 2004] [Ponsem 2004]. Para o cumprimento de tal objetivo, inicialmente foi realizada uma revisão bibliográfica que apresentou fatos encorajadores, pois possibilitou encontrar trabalhos que diziam ter obtido aprendizado

on-line eficiente utilizando Algoritmos Evolutivos e Redes Neurais Artificiais.

Para constatar que o AE funciona de fato no aprendizado on-line, foi verificado se o mesmo satisfaz os requisitos: rapidez, robustez, efetividade e eficiência. O AE proposto foi projetado de maneira a cumprir os primeiros três dos requisitos. Para mostrar a presença destes critérios, foi realizada uma comparação da estrutura do algoritmo a técnica de aprendizado DS que satisfaz os quatro requisitos. Para verificar o critério de eficiência, foi necessário:

1. Produzir um jogo como ambiente de testes assim como uma linguagem de scripting capaz de controlar o comportamento dos personagens. Para a produção deste jogo, foi procurado recriar o ambiente de testes produzido pelos criadores da técnica DS, cujos detalhes explicados no artigo [Spronck et al. 2004] exigiram grande atenção para estarem presentes também no jogo produzido neste trabalho.
2. Propor e codificar um AE, realizando testes sobre a função de avaliação utilizada, e otimizando suas funções evolutivas. Inicialmente a função de avaliação utilizada seria igual a função de avaliação do time utilizada pela técnica DS. Os testes realizados durante esta etapa mostraram a necessidade de se modificar esta avaliação para o funcionamento junto ao AE implementado. A taxa de mutação e o tamanho da população do AE foram definidos empiricamente, realizando alguns testes nesta etapa do projeto.
3. Implementar a técnica DS e realizar testes para verificar a capacidade de adaptação da mesma. Para isso foi necessário um estudo cuidadoso da técnica a fim de não cometer enganos em sua implementação. Os testes realizados permitiram verificar a correta implementação desta, assim como mostrar sua capacidade de adaptação.
4. Elaborar estratégias de jogo, com o objetivo de simular um jogador humano para permitir a realização de grandes quantidades de testes com as técnicas de aprendizado. Estas estratégias foram extraídas do trabalho de Spronck [2004], e procuraram ser reproduzidas neste trabalho.
5. Reproduzir, para os dois algoritmos de aprendizado, os testes realizados para verificar a eficiência da técnica DS no trabalho onde a mesma foi proposta, assim como realizar testes confrontando diretamente os dois algoritmos de aprendizado. Nesta etapa foi necessário um estudo detalhado sobre os testes realizados no artigo que apresenta a técnica DS [Spronck et al. 2004].
6. Avaliar os dados coletados segundo os critérios estatísticos adotados durante a etapa seis. Dificuldades foram encontradas com o rigor estatístico a ser utilizado. Foi então buscado o auxílio de um especialista da área de estatística, que se mostrou de grande ajuda para o cumprimento desta etapa.

Ao final da última etapa descrita acima, foi possível verificar que o algoritmo de aprendizado proposto neste trabalho satisfaz o requisito de eficiência. Também foi observado que o algoritmo em questão

apresenta eficiência melhor ou similar a da técnica DS, cuja eficiência já havia sido comprovada no trabalho [Spronck et al. 2004]. Tal constatação permite responder à pergunta chave deste trabalho: sim, é possível desenvolver um AE que satisfaça os quatro requisitos: rapidez, robustez, efetividade e eficiência.

Os testes realizados permitiram verificar uma deficiência da técnica DS: esta perde a capacidade de adaptação após um determinado tempo de funcionamento. Uma forma para corrigir tal deficiência seria definindo um peso mínimo de valor maior que zero a ser utilizado. Assim, nenhuma regra seria permanentemente descartada e, portanto, a técnica preservaria sua capacidade de adaptação. É provável que com esta correção a eficiência da técnica diminua, pois regras que já tenham sido testadas e mostradas inadequadas contra a estratégia utilizada continuariam apresentando a possibilidade de serem testadas.

A obtenção de um AE como o implementado neste trabalho traz contribuições para o campo de Inteligência Artificial aplicada a jogos, no qual é reconhecido que as possíveis aplicações para AE são numerosas, embora a técnica encontre certa resistência para ser aceita pelos desenvolvedores de jogos. Tal resistência é reforçada por afirmações como as contestadas e negadas neste trabalho. Portanto, espera-se com este trabalho, que os desenvolvedores de jogos tornem-se mais receptivos às técnicas evolutivas de IA.

Uma aplicação em potencial de AEs seria em jogos como MMORPGs, que estão em destaque atualmente. Este tipo de jogo permite que uma grande quantidade de usuários jogue simultaneamente. A avaliação de seus indivíduos poderia ocorrer de forma paralela, sendo que cada jogador funcionaria como uma função de avaliação para os oponentes que os enfrentassem.

O nível de complexidade do jogo produzido foi escolhido de maneira a ser alto quando comparado a outros jogos mais simples (espaço de busca de tamanho 25*1027 aproximadamente) [Fairclough et al., 2001]. Desta forma, ao se constatar que o AE apresentou bom desempenho neste jogo, é esperado que em jogos similares, mas com soluções contidas em um espaço de busca menor do que o deste projeto o AE apresente um desempenho ainda melhor. Em suma, foram obtidos neste trabalho:

1. Uma revisão sobre técnicas utilizadas na Inteligência Artificial dos jogos atuais. Um jogo5 com alta complexidade que serve como ambiente de testes para técnicas de aprendizado on-line.
2. Um AE capaz de produzir estratégias inteligentes adaptadas às estratégias utilizadas pelo usuário em tempo de jogo. Além disto, foi verificado que este algoritmo funciona na prática, pois satisfaz quatro requisitos: rapidez, efetividade, robustez e eficiência.

3. Resultados experimentais que: (1) mostram que o AE implementado é mais eficiente ou apresenta eficiência similar a da técnica DS desenvolvida por Spronck; (2) apontam deficiências da técnica DS e sugestões de melhorias para a mesma; e (3) negam afirmações que dão suporte à resistência encontrada para a aceitação de AEs por desenvolvedores de jogos.

Ao se concluir este trabalho, várias alternativas para sua continuação foram encontradas, dentre as quais podem ser citadas:

1. Implementar a correção proposta para a técnica DS e realizar outros experimentos para verificar se o mesmo continua satisfazendo o critério de eficiência.
2. Comparar outros algoritmos de aprendizado com os utilizados neste trabalho.
3. Tentar encontrar AEs que se mostrem ainda mais eficientes que o deste trabalho

Referencias

- BITTENCOURT, J. AND OSÓRIO, F., 2006. Motores de jogos para criação de jogos digitais – gráficos, áudio, interface, rede, inteligência artificial e física. In: *Anais da V ERI-MG SBC*, v. 1, 1–36.
- BRAMLETTE, M., 1991. Initialization, mutation and selection methods in genetic algorithms for function optimization. In: *Proceedings of the 4th International Conference on Genetic Algorithms*, San Mateo, CA, USA, 100–107.
- BUSSAB, W.O., Morettin, P. A. 2006. **Estatística Básica**. 5. ed. SARAIVA. P. 540. ISBN 8502034979.
- CROCOMO, M. K.; MIAZAKI, M.; SIMÕES, E.V. 2005. Algoritmos evolucionários no desenvolvimento de jogos adaptativos, In: *Simpósio Brasileiro de Jogos Para Computador e Entretenimento Digital – WJogos*, São Paulo, Brail, 342-347.
- FAIRCLOUGH, C. ET AL., 2001. *Research Directions for AI in Computer Games*. In: *Irish conference on AI and cognitive science*, 12. Kildare, AICS, 1 – 12.
- PONSEM, M., 2004. *Improving adaptive game AI with evolutionary learning*. Dissertação de Mestrado, Faculty of Media & Knowledge Engineering, Delft University of Technology.
- SPRONCK, P., YEE, N., SHPRINKHUIZEN-KUYPER, I. AND POSTMA, E., 2004. Online adaptation of game opponent AI in theory and practice. In: *Proceedings of the 4th International Conference on Intelligent Games and Simulation*, London, UK, 45–53.
- STANLEY, K., BRYANT, B. AND MIKKULANINEN, R., 2005. Evolving neural network agents in the NERO video game. In: *Symposium on Computational Intelligence and Games*, Piscataway, NJ, USA, IEEE Press, 182–189.
- SWEETSER, P., 2002. *Current AI in games: a review*. Relatório Técnico, School of ITEE, University of Queensland.
- YANNAKAKIS, G., 2005. *AI in computer games: generating interesting interactive opponents by the use of evolutionary computation*. Tese de doutorado, Faculty of Media & Knowledge Engineering, University of Edinburgh.

⁵ **Site do jogo ZantArena**. Disponível em: <<http://quadros.no-ip.org/zantarena>>. Acesso em: 02 fev. 2008.