Evolução de Estratégias em Unidades Controladas Pelo Computador em Jogos de Estratégia em Tempo Real Com a Utilização de Algoritmos Genéticos

Thiago D. Mendonça Christiano L. Santos

Universidade Federal de Sergipe, Departamento de Computação, Brasil

Abstract

One of the problems faced by real-time strategy (RTS) games is the predictability of the actions made by computer-controlled opponents, due to the use of artificial intelligence techniques that do not permit the machine to learn dynamically. This article aims to propose the use of genetic algorithms in RTS games to model an evolutive strategy, as well present GenCraft, an application developed as a case study.

Keywords: Artificial Intelligence, Real-Time Strategy, Genetic Algorithms

Resumo

Um dos problemas enfrentados por jogos de estratégia atualmente é a respeito da previsibilidade das ações dos oponentes controlados pelo computador, devido ao emprego de técnicas de inteligência artificial que não permitem o aprendizado de forma dinâmica. Este artigo tem como objetivo propor a modelagem de uma estratégia evolutiva em unidades controladas pelo computador em jogos de estratégia em tempo real com a utilização de algoritmos genéticos, bem como apresentar o GenCraft, uma aplicação desenvolvida como estudo de caso.

Palavras Chave: Inteligência Artificial, Estratégia em Tempo Real, Algoritmos Genéticos

Contato dos Autores:
thiagodm@dcomp.ufs.br
christianolimasantos@yahoo.com.br

1. Introdução

Os jogos digitais atuais têm se tornado cada vez mais realistas, especialmente em termos de apresentação gráfica do mundo virtual no qual o jogo é situado. Para aumentar o realismo nos jogos, o próximo passo é fazer com que os personagens desses mundos virtuais tornem-se aptos a responder efetivamente às ações do jogador [Ponsen et al. 2006].

O campo de inteligência artificial (IA) em jogos existe desde o aparecimento dos jogos digitais nos anos 70. A percepção pública da IA em jogos, mesmo nos mais atuais, remete a jogos antigos como *Pac-Man*, onde predominam ações repetitivas [Rabin 2002].

Com o advento dos computadores pessoais cada vez mais rápidos, jogos interativos ficaram cada vez mais populares. Alguns exemplos desses jogos são os de estratégia em tempo real onde os jogadores comandam exércitos que se confrontam. Para esses jogos há uma grande demanda de um comportamento inteligente que seja capaz de tomar decisões satisfatórias e em tempo real. Entretanto, mesmo em jogos RTS (*Real-Time Strategy*), como *Starcraft* e *Warcraft* da *Blizzard Entertainment* e *Age of Empires* da *Ensemble Studios,* a performance da Inteligência Artificial é pobre quando comparada aos padrões humanos, onde o computador executa estratégias repetitivas e fáceis de combater [Buro e Furtak 2004].

Desta forma, há uma preocupação crescente com a criação de agentes inteligentes com melhor capacidade de se adaptar e evoluir durante uma partida em jogos eletrônicos. Isso nos leva a um dos grandes desafios atuais das empresas desenvolvedoras de jogos eletrônicos, que é proporcionar ao jogador uma ambiente não somente competitivo, mas adaptável às escolhas do jogador, como citam os autores em [Schadd et al. 2007], que um fator importante para a escolha de uma estratégia é a própria estratégia do oponente, conseguindo assim uma situação desafiadora capaz de oferecer uma satisfação maior com o jogo.

Para isso, propõe-se um abordagem com algoritmos genéticos, permitindo a evolução da IA envolvida. Devido à sua natureza evolutiva, os algoritmos genéticos podem ser muito bem empregados em aplicações que exijam muitas interações entre diversos agentes em um mesmo ambiente, como é o caso dos jogos de estratégia, onde há muitas unidades e civilizações, evoluindo com o tempo e a importância da adaptação de estratégia e tomada de novas decisões é fundamental para garantir a experiência do usuário.

O maior desafio no uso de algoritmos genéticos em jogos RTS é o tempo necessário para o aprendizado. O algoritmo pode demorar muito tempo para convergir para uma boa estratégia, fazendo com que o jogador não sinta uma grande dificuldade no decorrer do jogo.

Este artigo propõe a investigação e utilização de algoritmos genéticos em jogos RTS para modelar uma estratégia evolutiva em unidades controladas pelo computador, utilizando como estudo de caso o jogo GenCraft, desenvolvido durante a fase de pesquisa, discutindo os resultados obtidos.

2. Trabalhos Relacionados

[Ponsen e Spronck 2004] propõem um modelo de aplicação de algoritmos evolutivos em jogos RTS bastante similar ao proposto neste trabalho. Eles perceberam que as rotinas comuns de criação de estratégias para unidades combatentes eram deficientes ao enfrentar oponentes com estratégias mais otimizadas. Um jogo de estratégia equipado com uma IA evolutiva pode potencialmente ignorar estratégias comuns e ser criativo.

Os autores utilizaram *Wargus*, um jogo RTS similar a *Warcraft II*, como estudo de caso e programaram rotinas de algoritmos evolutivos para as unidades do jogo. Obtiveram questionamentos muito semelhantes aos discutidos neste trabalho, como escolha do tamanho ideal da população. Para treinar essa população, os autores utilizaram um oponente controlado pelo computador.

Este artigo visa contribuir com o estado da arte ao criar um ambiente similar ao descrito, porém com a diferença que a aplicação seja "jogável", investigando assim a utilização das técnicas em um jogo real interativo.

3. Algoritmos Genéticos

No desenvolvimento de jogos eletrônicos, técnicas de inteligência artificial normalmente são utilizadas. Dentre elas têm-se algoritmos de *pathfinding*, sistemas baseados em regras, agentes inteligentes, entre outros.

Uma técnica atualmente subutilizada em jogos digitais é o uso de algoritmos genéticos na construção do comportamento inteligente do oponente.

Os algoritmos genéticos possuem uma estrutura básica comum [Lecheta 2004]:

1. Geração aleatória de indivíduos (somente na primeira geração);
2. Seleção de indivíduos para realizar reprodução (*crossover*) aplicando-se a função *fitness* para calcular seu nível de adaptação ao meio;
3. Imposição de variações aleatórias nos cromossomos dos indivíduos resultantes (mutações);
4. Reinício do processo com os novos indivíduos.

Ao final da aplicação do algoritmo, obtêm-se um ou mais indivíduos evoluídos, ou seja, mais aptos ao ambiente.

[Dalmau 2004] cita que os algoritmos genéticos em jogos podem ser usados para a geração de uma população, criando indivíduos com diferentes "cargas genéticas", que podem influenciar seu comportamento. Essa técnica pode ser utilizada para simular gerações de unidades combatentes num ambiente de um jogo de estratégia. Dessa forma, o comportamento inteligente estaria sendo aplicado diretamente nas unidades, ou seja, cada combatente teria seu próprio comportamento autônomo e "inteligente".

Esse efeito pôde ser observado em GenCraft, onde cada unidade possui autonomia sobre suas ações, que por sua vez, são definidas de acordo com os genes de cada unidade.

4. Estudo de Caso - GenCraft

Trata-se de um jogo, ilustrado na Figura 1, onde o usuário comanda uma civilização e pode tanto recrutar unidades construtoras e combatentes, como construir depósitos para coletar recursos ou quartéis para recrutar mais unidades combatentes.

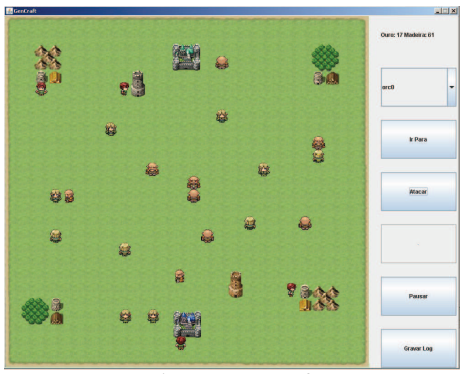

Figura 1: GenCraft

O jogo possui as seguintes unidades:

- Peão – unidade básica, não possui poder de ataque e possui poucos pontos de vida (PVs), porém constrói depósitos e quartéis;
- Orc – unidade combatente, possui muitos PVs e alto poder de ataque, porém pode atacar outras unidades apenas a curta distância;
- Elfo – unidade combatente, possui moderados PVs e baixo poder de ataque, porém pode atacar à distância.

Além disso, o jogador possui à sua disposição as seguintes construções:

- Base – construção que possibilita recrutar peões. Se destruída, o jogador proprietário da mesma perde o jogo, tornando-se este o objetivo central do mesmo;
- Depósito – construção que, se construída próxima a recursos, começa a coletá-los;
- Quartel – construção que recruta unidades combatentes.

Conforme os principais jogos estratégicos de guerra, recursos são necessários para o recrutamento de novas unidades bem como construção de novas

estruturas. Em GenCraft, dois são os recursos necessários para tal desenvolvimento:

- Madeira – recurso necessário para a criação de construções. Coletado de florestas no mapa;
- Ouro – recurso necessário para o recrutamento de unidades. Coletado de minas no mapa.

Em GenCraft, algoritmos genéticos são empregados na criação de novas unidades combatentes controladas pelo computador a fim de determinar seus comportamentos. Desta forma, a partir de indivíduos inicialmente criados aleatoriamente, espera-se que os mesmos alterem seus comportamentos a fim de melhor agirem no ambiente do jogo.

4.1 Arquitetura do Cromossomo

Em GenCraft, somente as unidades combatentes, ou seja, os orcs e os elfos, são criados a partir de técnicas de algoritmo genético.

Para que possa, de fato, haver um melhoramento no comportamento das unidades no decorrer de uma partida, os genes que compõem o cromossomo de cada indivíduo devem conter atributos que possam ser utilizados para calcular a estratégia final da unidade. No estudo de caso, utilizamos sete atributos, denominados a_0, a_1, a_2, a_3, b_0, b_1 e b_2.

Os atributos de a_0 a a_3 são parâmetros para o cálculo do nível de ameaça que cada unidade inimiga oferece à unidade portadora deste gene. Já os atributos de b_0 a b_2 determinam qual estratégia será escolhida pela unidade.

Existem duas estratégias básicas implementadas em cada unidade:

- Atacar – A unidade que decide pela estratégia de atacar determina qual unidade inimiga causa maior ameaça e ataca-a;
- Recuar – A unidade que decide pela estratégia de recuar movimenta-se em direção às proximidades de sua base para refugiar-se.

Os atributos possuem valores inteiros. Segue uma descrição mais detalhada dos genes das unidades, assim como sua faixa de valores:

- a_0 – influência que os PVs de uma unidade inimiga sobre a ameaça que esta causa à esta unidade (0-100);
- a_1 – influência que a quantidade de unidades inimigas ao redor de uma certa unidade inimiga tem na ameaça que esta causa à esta unidade (0-100);
- a_2 – influência do fato de uma certa unidade inimiga ser uma criatura (elfo, orc ou peão) ou uma construção (base, depósito ou quartel) na ameaça que esta causa à esta unidade (0-100);
- a_3 – influência da proximidade de uma certa unidade inimiga tem na ameaça que esta causa à esta unidade (0-100);
- b_0 – quantidade mínima de PVs que a unidade deve ter para decidir pela estratégia de atacar (0-70);
- b_1 – quantidade mínima de unidades aliadas em sua proximidade (reforços) que a unidade deve ter para decidir pela estratégia de atacar (0-5);
- b_2 – quantidade mínima de unidades aliadas a mais que unidades inimigas em todo o campo que a unidade deve ter para decidir pela estratégia de atacar (0-3).

Em cima desses atributos será feito um cálculo que irá determinar finalmente a estratégia tomada pela unidade neste momento até que seu objetivo (destruir o inimigo alvo ou recuar com sucesso à base) seja cumprido ou o ambiente mude drasticamente (seus pontos de vida diminuam, o inimigo recrute muitas unidades, seus aliados morram, etc.), quando, então, a unidade deverá perceber o ambiente novamente para decidir sua nova estratégia.

Quando, uma unidade morre, calcula-se para ela um valor de *fitness*, que indica a probabilidade dessa unidade ser escolhida para passar seus genes para a próxima geração. Para o cálculo do *fitness*, levamos em conta vários fatores – tempo de vida da unidade, quantidade de ataques que ela desferiu e de construções e criaturas que ela derrotou.

Para a implementação do algoritmo genético, foi utilizada seleção por roleta, *crossover* por cruzamento uniforme, mutação por inversão de bit e não foi implementado elitismo.

4.2 Resultados

A partir de discussões sobre o tempo de jogo necessário para o aprendizado do oponente e a quantidade de gerações necessária para tal, resolveu-se utilizar o valor de cinco indivíduos por geração. Isso significa que a cada cinco unidades que morrem e têm seu *fitness* calculado é iniciada uma nova iteração, e essas mesmas cinco unidades farão parte da seleção para propagar seus genes, e assim por diante.

Foram realizadas algumas partidas entre um jogador humano e o computador, utilizando os parâmetros já definidos para os algoritmos genéticos, gravando-se os logs das partidas para interpretação dos resultados. Escolheu-se paralisar os jogos com trinta e cinco minutos de partida, como um valor limite para a observação do aprendizado do oponente.

Durante os primeiros minutos de jogo, em geral percebia-se que as unidades adotavam estratégias aleatórias e pouco inteligentes, recuando ou atacando quando desnecessário. Após, em média, quinze minutos, havia alguma mudança no comportamento

geral. Algumas unidades ainda apresentavam comportamentos não muito eficientes, entretanto defendiam prontamente sua base quando unidades inimigas aproximavam-se.

Após os trinta e cinco minutos de jogo, paralisava-se para analisar os resultados. Em termos de jogabilidade, nessa etapa da partida, as unidades já apresentavam um melhor comportamento, atacando em momentos propícios, apesar de haver ainda certa quantidade de unidades optando somente pela tática de recuar. A Figura 2 demonstra um gráfico com a evolução da média dos valores dos genes que obtiveram uma mudança mais significativa durante os testes.

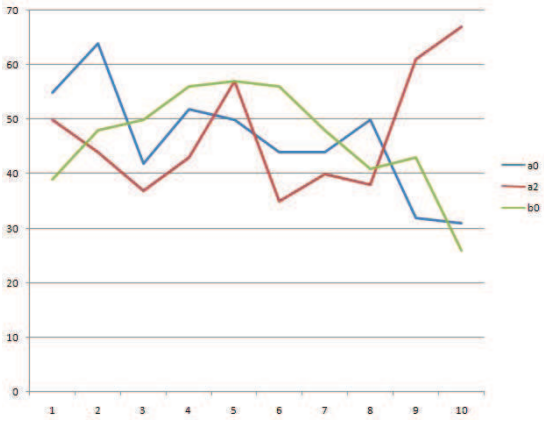

Figura 2: Gráfico de Atributos

No eixo horizontal estão as gerações (da 1ª à 10ª) e no eixo vertical estão os valores absolutos dos atributos em questão. A linha azul corresponde ao atributo a_0, a vermelha ao atributo a_2 e a verde a b_0.

Percebe-se que as unidades optaram por dar pouca importância aos PVs do inimigo ao atacar (a_0), como também decidiram que construções causam muito mais ameaça do que criaturas (a_2) e, por fim, deram pouca importância à quantidade de PVs necessária para escolher por atacar (b_0). Isso demonstra que as unidades decidiam atacar com muito mais frequência.

Demonstra-se assim a adaptação das unidades às circunstâncias do jogo, a ocorrência do aprendizado e a eficácia do método em tempo hábil, requisitos essenciais para algoritmos de inteligência artificial aplicados a jogos.

5. Considerações Finais

Este trabalho apresentou-se como o primeiro passo para a investigação de soluções mais robustas no campo de desenvolvimento de jogos eletrônicos e, mais especificamente, jogos de estratégia em tempo real. Por meio deste, confirmou-se a aplicabilidade dos algoritmos genéticos.

Por explorar uma área ainda pouco visada no mercado atual de jogos, é válida a tentativa de fomentar a preocupação com a criação de uma IA mais flexível e com um maior poder de evolução durante o jogo.

6. Futuras Extensões

Como futura extensão, em GenCraft, pretende-se investigar outras técnicas de IA como *dynamic scripting* para diversificar e melhorar o comportamento do oponente controlado pelo computador, além de criar estratégias mais complexas com a finalidade de conseguir resultados mais contundentes, realizando testes também com jogadores humanos com estratégias variadas, com o intuito também de verificar a validade de suportar oponentes aleatórios no início da partida.

Pretende-se ainda aperfeiçoar o experimento, expandindo-o para SPRING, uma *engine open source* para jogos RTS, com o objetivo de melhorar a jogabilidade, aumentar a complexidade das unidades e principalmente para poder realizar testes mais eficientes e comparar os resultados com outras IAs desenvolvidas por outros usuários.

Referências

BURO, M. E FURTAK, T., 2004. RTS Games and Real-Time AI Research. *Em: Advances in Computer Games X, pp. 159-174. Kluwer Academic Press*.

DALMAU, D., 2004. Core Techniques and Algorithms in Game Programming. *Indianapolis, New Riders.*

LECHETA, E., 2004. Algoritmos Genéticos para Planejamento em Inteligência Artificial. *Tese de Mestrado, Setor de Ciências Exatas, Universidade Federal do Paraná, Curitiba, Brasil.*

PONSEN, M., MUÑOZ,-AVILA, H., SPRONCK, P. E AHA D., 2006. Automatically Generating Game Tactics through Evolutionary Learning, *Em: AI Magazine Vol. 27 Nº 3, pp. 75-84.*

PONSEN, M., SPRONCK P., 2004. Improving Adaptive Game AI with Evolutionary Learning, *Em: Computer Games: Artificial Intelligence, Design and Education (CGAIDE 2004). pp. 389-396, University of Wolverhampton.*

RABIN, S., 2002. AI Programming Wisdom. *Hingham, MA, USA: Charles River Media, Inc.*

SCHADD, F., BAKKER, S. E SPRONCK, P., 2007. Opponent Modeling in Real-Time Strategy Games. *Em: 8th International Conference on Intelligent Games and Simulation (GAME-ON 2007), M. Rocetti, Ed.2007, pp. 61-68.*

SPRING. [online] Disponìvel em: http://springrts.com/ [Acessado em 4 de setembro de 2009].