

**LIGA DE ENSINO DO RIO GRANDE DO NORTE  
FACULDADE NATALENSE PARA O DESENVOLVIMENTO DO RIO GRANDE DO  
NORTE  
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO**

**Renan Mateus Wanderley Caetano**

**INTELIGÊNCIA ARTIFICIAL PARA JOGOS DE ESTRATEGIA BASEADA EM  
TURNOS**

**NATAL/RN  
2011**

**RENAN MATEUS WANDERLEY CAETANO**

**INTELIGÊNCIA ARTIFICIAL PARA JOGOS DE ESTRATEGIA BASEADA EM  
TURNOS**

Monografia apresentada como título parcial para a  
obtenção de título de Bacharel em Gestão em  
Sistemas de Informações da Faculdade Natalense  
para o desenvolvimento do Rio Grande do Norte -  
FARN.

Orientador: Prof. Alexandre Luiz Galvão  
Damasceno.

.

**NATAL/RN  
2011**

**LIGA DE ENSINO DO RIO GRANDE DO NORTE  
FACULDADE NATALENSE PARA O DESENVOLVIMENTO DO RIO GRANDE DO  
NORTE  
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO**

**Renan Mateus Wanderley Caetano**

**INTELIGÊNCIA ARTIFICIAL PARA JOGOS DE ESTRATEGIA BASEADA EM  
TURNOS**

Monografia apresentada e aprovada em 24 de novembro de 2011, pela banca examinadora  
composta pelos seguintes membros:

---

Prof. MSc. Alexandre Luiz Galvão Damasceno  
Orientador

---

Prof. MSc. Joseane Alves Pinheiro

---

Profa. MSc. Ytalo Rosendo do

Natal/RN 24 de novembro de 2011

**Dedico esse trabalho aos meus pais, colaboradores e a todos os meus professores.**

## **AGRADECIMENTOS**

Agradeço imensamente por esse trabalho ao meu orientador Prof. Alexandre pela paciência e dedicação. Agradeço também a todos que colaboraram com todos os momentos e informações da construção desse trabalho, muito obrigado.

## SUMÁRIO

|                                                      |           |
|------------------------------------------------------|-----------|
| <b>LISTA DE FIGURAS.....</b>                         | <b>7</b>  |
| <b>LISTA DE ABREVIATURAS, SIGLAS E SIMBOLOS.....</b> | <b>8</b>  |
| <b>1. INTRODUÇÃO.....</b>                            | <b>12</b> |
| <b>1.2 OBJETIVOS.....</b>                            | <b>14</b> |
| 1.2.1 Objetivo Geral .....                           | 14        |
| 1.2.2 Objetivos Específicos.....                     | 14        |
| <b>1.3. JUSTIFICATIVA.....</b>                       | <b>15</b> |
| <b>2. REFERENCIAL TEÓRICO.....</b>                   | <b>15</b> |
| <b>3. METODOLOGIA.....</b>                           | <b>27</b> |
| <b>4. RESULTADO.....</b>                             | <b>48</b> |
| <b>5. CONCLUSÃO.....</b>                             | <b>49</b> |
| <b>BIBLIOGRAFIA .....</b>                            | <b>50</b> |

## LISTA DE FIGURAS

|                                                                                            |           |
|--------------------------------------------------------------------------------------------|-----------|
| <b>FIGURA 1</b> - Esquema funcional do agente baseado em reflexos.....                     | <b>16</b> |
| <b>FIGURA 2</b> - Algoritmo de busca em profundidade com limite incremental.....           | <b>17</b> |
| <b>FIGURA 3</b> - Algoritmo de unificação.....                                             | <b>19</b> |
| <b>FIGURA 4</b> - Métodos gerais de aprendizado do agente por indução.....                 | <b>21</b> |
| <b>FIGURA 5</b> - Exemplo de funcionamento de uma rede neural.....                         | <b>22</b> |
| <b>FIGURA 6</b> - Imagens de alguns dos primeiros jogos de estratégia baseada em turnos... | <b>26</b> |
| <b>FIGURA 7</b> - Imagem do jogo <i>Final Fantasy Tactics</i> mostrando uma batalha.....   | <b>28</b> |
| <b>FIGURA 8</b> - Imagem do jogo <i>Final Fantasy Tactics</i> da tela de personagem.....   | <b>29</b> |
| <b>FIGURA 9</b> - Tela inicial do jogo FFT Remake.....                                     | <b>29</b> |
| <b>FIGURA 10</b> - Menu principal do jogo FFT Remake.....                                  | <b>30</b> |
| <b>FIGURA 11</b> - Tela do jogo FFT Remake de configuração de um personagem.....           | <b>31</b> |
| <b>FIGURA 12</b> - Tela do jogo FFT Remake de uma batalha.....                             | <b>31</b> |
| <b>FIGURA 13</b> - Diagrama do funcionamento da arquitetura proposta.....                  | <b>34</b> |
| <b>FIGURA 14</b> - Funcionamento da parte estratégica do jogo.....                         | <b>36</b> |
| <b>FIGURA 15</b> - Algoritmos utilizados nas funções do módulo inteligente do jogo.....    | <b>37</b> |
| <b>FIGURA 16</b> - Tela do programa enquanto a simulação é executada.....                  | <b>43</b> |
| <b>FIGURA 17</b> - Imagem do registro de resultados da simulação.....                      | <b>43</b> |
| <b>FIGURA 18</b> - Mostra a tabela de resultados da simulação de vitórias.....             | <b>45</b> |
| <b>FIGURA 19</b> - Resultados das simulações para as funções objetivo de aprendizado.....  | <b>46</b> |

## **LISTA DE ABREVIATURAS SIGLAS E SIMBOLOS**

**FFT = Final Fantasy Tactics**

**IA = Inteligência Artificial**

**FPS = First Person Shooting**

**STRIPS = Stanford Research Institute Problem Solver**

**NPC = Non Player Character**

**HTN = Hierarchical Task Network**

**SME = Structural Mapping Engine**

**AT = Active Turn**

**MP = Magic Points**

*Uma pessoa inteligente aprende com os seus erros,  
uma pessoa sábia aprende com os erros dos outros.*  
(Augusto Cury)

## RESUMO

O grande crescimento, evolução e importância da indústria de jogos eletrônicos são evidentes nos dias de hoje. Porém, apesar de áreas como o hardware, os gráficos e novos modos de interação se desenvolverem com rapidez, outras áreas como a inteligência artificial começam a ser mais bem exploradas apenas recentemente e somente para alguns gêneros de jogos. Jogos de estratégia baseada em turnos são aqueles que mais se aproximam dos jogos tradicionais como xadrez e damas, assunto que a pesquisa acadêmica de IA estuda desde seus primórdios. Estuda-se os problemas e frustrações que os jogadores encontram com relação à IA dos jogos e apresenta-se uma arquitetura já proposta para a implementação de um módulo inteligente para este gênero de jogos. Posteriormente estudam-se os efeitos dos valores iniciais e das funções-objetivo durante o aprendizado através de simulações.

**Palavras-chave:** Inteligência Artificial, Jogos de Computador, Estratégia baseada em turnos e redes neurais.

## ABSTRACT

The growing, development and influence of the video-game industry is evident nowadays. While many efforts are directed to development of hardware, graphics and interactivity, little is done in other areas, like artificial intelligence. Turn-based Strategy video-games are the most related genre to traditional table games, such as chess and checkers, which were subject of AI academic research since its beginning. This work studies the problems and frustrations that the video-game players face during gameplay because of poor AI and propose architecture for intelligent agents to games of this genre. Afterwards, it studies the effect of the initial values and the learning goal-functions through game simulations.

**Keywords:** Artificial Intelligence, Video-games, Turn-based Strategy and Weightless Neural Networks.

## 1. INTRODUÇÃO

Robôs e computadores que se comportam como humanos inspiram e assustam a humanidade há muito tempo (BUCHANAN, 2007). Desconsiderando o aspecto mágico dos computadores inteligentes nos resta um ramo acadêmico que praticamente desistiu do seu sonho utópico (BROOKS, et al., 1998) e se concentra em resolver principalmente problemas práticos, como o reconhecimento de fala, a criação de educadores virtuais, entre outros. (NORDLANDER, 2001).

Inteligência artificial é na verdade um grande conjunto de técnicas que permitem que programas de computadores tomem decisões, seja através da escolha entre várias opções ou pela conclusão lógica de acordo com certas circunstâncias (RUSSELL & NORVIG, 2003). Intuitivamente, a inteligência artificial é ligada à idéia de um ser virtual que pode ser colocado em competição com um humano, fato consolidado pela história do mercado de games (DEMARIA, 2003). Apesar de existirem por menos de 40 anos, os jogos de computador formam um mercado sério. Softwares de entretenimento constituem o ramo das indústrias de entretenimento que mais cresce no mundo e atualmente possui vendas que geram lucro maior do que a quase centenária indústria cinematográfica de Hollywood (LAIRD J, 2005).

Jogos de computador afetam seriamente a venda de computadores pessoais e motivaram grande parte do avanço tecnológico envolvendo software e hardware dedicados a gráficos de ponta. O impacto de todo esse avanço é claro nos jogos atuais: gráficos mais realistas, design mais complexo, arte melhor trabalhada, jogabilidade cada vez mais inovadora. Tudo isso forma um palco ideal para o desenvolvimento de novas técnicas de inteligência artificial em *games* (RABIN, 2005), Hoje em dia jogos de computador são objetos para grandes estudos acadêmicos (LAIRD J, 2005).

O incentivo para módulos de inteligência artificial sofisticados vem crescendo: com jogos gradualmente mais elaborados, a inteligência se tornou um diferencial. Quando bem desenvolvida a IA proporciona experiências para o jogador como se estivesse interagindo com outra pessoa (RABIN, 2005), e o frustra quando mal projetada. O nível das técnicas utilizado e módulo inteligentes nos jogos evoluiu muito nos últimos anos. Mesmo estando presente desde o início, a inteligência artificial nos jogos é considerada muito mais importante atualmente, seja por causa da complexidade ou do realismo exigido.

Jogos como *The Sims* (MAXIS, 2000) e *Halo* (ISLA, 2007) se destacam pelos resultados obtidos nesta área. O gênero de estratégia baseada em turnos é um dos que melhor pode aproveitar esse desenvolvimento. Fundamentado fortemente em jogos de tabuleiro

tradicionais como xadrez e damas, este gênero funciona com os jogadores se enfrentando controlando unidades em turnos alternados num tabuleiro. Duas características facilitam a aplicação de técnicas avançadas de IA: o turno é atemporal e por isso cálculos mais pesados podem ser feitos e o ambiente é a todo tempo acessível completamente, usado como base para decidir a melhor ação (HINRICHS & FORBUS, 2007).

Porém, por possuírem um forte apelo à estratégia, a demanda dos jogadores por uma boa inteligência artificial é ainda maior. Nesse gênero ainda não há jogos que se destacam pela inteligência. Mesmo com todos esses avanços, a inteligência artificial utilizada em jogos está longe daquela que é desenvolvida em trabalhos acadêmicos (LAIRD & LENT, 2005), principalmente em jogos de tabuleiro tradicionais (SALETAN, 2007) ou jogos mais complexos. (NOWAKOWSKI & ALBERT, 2007).

Isso ocorre por uma série de fatores, o primeiro é que os algoritmos usados nas técnicas avançadas são lentos e consome muita capacidade de processamento, o que é priorizado pela parte gráfica no caso dos jogos; o segundo é que o desenvolvimento do módulo inteligente do jogo é caro e lento demais para os padrões de criação de jogos atuais; o terceiro fator é o do jogo se alterar demais durante seu desenvolvimento, o que prejudica muito a implementação do módulo inteligente (KIRBY & WOODCOCK, 2004). Isso implica em diversas decepções para os jogadores que encontram um jogo pouco inteligente ou mal projetado.

Os problemas mais visados atualmente pelos desenvolvedores são: a) falta de sensação ou impressão de inteligência, onde o jogador percebe claramente que o computador é um computador jogando; b) falta de uma progressão gradual e desafiante do nível de dificuldade, pois os jogos têm sua dificuldade definida pela quantidade de trapaças (controlar variáveis no jogo as quais o jogador não tem acesso) que a IA faz; c) falta de adaptação pela IA, pois as unidades controladas automaticamente se comportam sempre da mesma forma, sendo previsíveis demais e d) falta de interação entre as unidades inteligentes, que parecem sempre agir individualmente mesmo quando trabalhando em grupo.

## **2 OBJETIVOS**

### **2.1 OBJETIVO GERAL**

Para tornar o jogo mais atraente comercialmente e em termos de dinamismo visando os consumidores mais ávidos por inteligência artificial avançada, pretende-se avaliar a aplicabilidade do uso de inteligência artificial no jogo chamado *Final Fantasy Tactics Remake*.

### **2.2 OBJETIVOS ESPECIFICOS**

**A)** Estudar as técnicas fundamentais da Inteligência Artificial visando formar uma base sólida para a elaboração de soluções dos problemas visados;

**B)** Estudar os métodos utilizados pelos desenvolvedores de jogos de computador nos módulos de inteligência artificial dos jogos comerciais;

**C)** Analisar uma arquitetura proposta para o módulo inteligente de jogos de estratégia baseada em turnos que atenda à demanda de inteligência exigida pelos jogadores, que resolvem os problemas de falta de impressão de inteligência, falta de controle do nível de dificuldade, falta de adaptação e falta de interação do jogo com o jogador;

**D)** Comparar os efeitos dos principais parâmetros da arquitetura proposta com a realização de simulações e testes para que seja consolidado seu funcionamento e sua eficiência na resolução dos problemas propostos, e que seja possível prever o comportamento do agente inteligente de acordo com a definição de seus parâmetros.

### **3. JUSTIFICATIVA**

Observando a críticas de jogadores e praticas diversas de empresas de games, este trabalho visa a estudar uma arquitetura proposta sobre como utilizar inteligência artificial pra adicionar dinamismo a jogabilidade de games de estratégia baseados em turnos, contra argumentando a idéia de que tal implementação tornaria o jogo impossível de ser terminado.

### **4. REFERENCIAL TEÓRICO**

Neste capítulo são apresentados estudos dos fundamentos necessários para a elaboração do trabalho. Primeiramente são discutidos os principais ramos da inteligência artificial, para que seja elaborado um panorama das técnicas que podem ser usadas num módulo de IA de um jogo.

Posteriormente será feito um estudo dos métodos e técnicas utilizados pelos desenvolvedores de jogos comerciais nos módulos inteligentes de seus jogos.

#### **4.1 INTELIGÊNCIA ARTIFICIAL**

Inteligência Artificial é definida muitas vezes como o “estudo e concepção de agentes inteligentes”, onde um agente inteligente é um sistema que percebe o ambiente externo e age de forma a alcançar seu objetivo (RUSSELL & NORVIG, 2003). Este ramo das ciências da computação foi criado na metade do século XX, baseado nas novas descobertas em neurologia, matemática e computadores digitais. Seu início foi marcado por muito otimismo, quando os cientistas achavam que em algumas décadas as máquinas substituiriam completamente os humanos. Posteriormente, na década de 70 os cientistas descobriram os verdadeiros seus desafios, o que causou uma depressão entre os profissionais do ramo (BUCHANAN, 2007). Com a descoberta de novas técnicas e a aplicação prática de muitos conceitos e programas, a inteligência artificial ganha cada vez mais importância na ciência e tecnologia atuais (NORDLANDER, 2001).

Com a grande variedade de técnicas e objetivos diferentes que compreende, a inteligência artificial é considerada fragmentada, suas principais áreas de atuação são: busca e planejamento, representação de conhecimento para sistemas especialistas, aprendizado e linguagem (MC. CARTHY, 2007). Sua maioria possui aplicações práticas, como softwares que jogam xadrez, que organizam e buscam dados, que planejam produção de indústrias, que

conseguem interagir com humanos, entre outros. Porém, não é possível ainda hoje construir máquinas que pensem ou ajam como humanos, ou que ao menos consigam raciocinar de forma a conduzir uma conversa mais complexa (BROOKS, et al., 1998).

O estudo básico dos agentes inteligentes compreende a definição de alguns conceitos fundamentais (RUSSELL & NORVIG, 2003). Um agente é algo que percebe e age num ambiente, geralmente dividido em arquitetura e programa. Agentes podem ser baseados em reflexos, são os que recebem um estímulo e reagem a ele; em objetivos, cujas ações visam atingir apenas o objetivo proposto; ou utilidades, onde o agente segue uma função específica.

Um estímulo é o que o agente percebe do ambiente, o ambiente por sua vez é tudo aquilo com o que o agente pode interagir. Essas definições estão esquematizadas na figura 1.

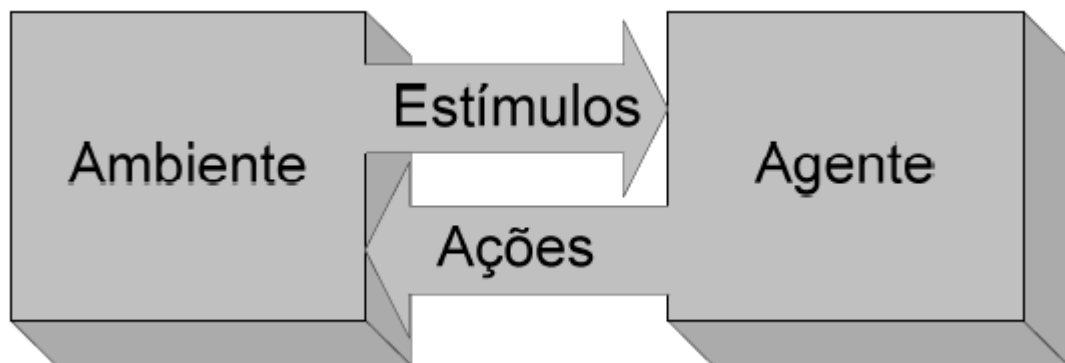


Figura 1: Esquema funcional do agente baseado em reflexos, onde o agente só reage com estímulos do ambiente.  
Fonte: Russell & Norvig. (2003)

O ambiente e o agente inteligente que percebe os estímulos e realiza ações. Ambientes podem ser acessíveis quando o agente pode saber o estado completo do ambiente, determinísticos se o próximo estado do ambiente pode ser determinado pelo estado atual e pela ação do agente, estáticos se não mudam com o tempo e discretos se possuem um número limitado de estados.

Um ambiente com todas essas características é o mais simples que um agente pode encontrar, infelizmente os ambientes reais não possuem nenhuma dessas características e por isso são os mais complexos de se trabalhar.

## 4.2. TÉCNICAS DE BUSCA

Um agente que possui um objetivo, normalmente um estado desejado para o exterior, e que procura num conjunto de vários estados uma sequência que ao fim possui o estado desejado é um agente que executa uma busca. A forma, o algoritmo, com que o agente procura essa sequência define a técnica utilizada. Para executar uma busca, o agente deve possuir uma definição do estado inicial, das ações que pode fazer e seus custos. Essas três informações formam um problema. Para realizar uma busca os estados possíveis são organizados em forma de árvore, a ordem em que o agente considera os ramos da árvore define o algoritmo utilizado. Os principais algoritmos são o de busca em largura, em profundidade, de custo uniforme, em profundidade com limite incremental e busca bi-direcional (RUSSELL & NORVIG 2003). A figura 2 mostra a descrição básica do algoritmo de busca em profundidade com limite incremental.

```

função BuscaProfundidadeIncremental(problema) retorna sequencia solução
entrada problema indicando estado inicial, ações possíveis e objetivo

for profundidade = 0 até infinito
{
    se BuscaProfundidade(problema, profundidade) resolve
    {
        retorna resultado
    }
}
retorna falha

```

```

função BuscaProfundidade(problema, profundidade) retorna sequencia
entrada problema indicando estado inicial, ações possíveis e objetivo
        profundidade indicando o comprimento máximo da fila
fila <- FazerFila(FazerNo(problema.EstadoInicial))
loop
{
    se nos é vazio retorna falha
    no <- RemoveFrente(fila)
    se problema.TesteObjetivo(no.Estado) resolve
    {
        retorna no
    }
    fila <- EnfilaFrente(fila, problema.Operadores)
}

```

Figura 2: Algoritmo de busca em profundidade com limite incremental.

Fonte: Russell & Norvig. (2003)

### 4.3. AGENTES RACIONAIS

Outra área da inteligência artificial é a de programar raciocínio lógico nos agentes. Isso é feito através de sistemas especialistas, que possuem uma forma de representação do conhecimento e utilizam uma base de sentenças lógicas para deduzir as ações do agente, que pode ser considerado, dentro de seus limites, racional. Representar o conhecimento é uma tarefa difícil.

Para realizá-la são utilizadas regras de sintaxe e de semântica. O agente pode então perceber o ambiente externo, converter o estímulo em uma cadeia semântica e combiná-la com as sentenças que já possui em sua base de conhecimento para concluir por dedução novas sentenças. Através das novas sentenças criadas e de seus objetivos, o agente pode decidir qual ação tomar. Normalmente as regras de sintaxe são definidas pela lógica de primeira ordem, que é uma linguagem de representação genérica que pode ser usada para qualquer aplicação. O agente pode usar regras de diagnóstico ou causal. Regras de diagnóstico raciocinam de estímulos para proposições sobre o mundo, enquanto que regras causais descrevem as condições do ambiente externo que poderiam ter causado os estímulos recebidos.

Para raciocinar, ou seja, para levantar conclusões utilizando as regras lógicas é preciso usar um algoritmo que receba novas regras, o inclua na base e conclua algo, como o algoritmo da unificação (RUSSELL & NORVIG, 2003), mostrado na figura 3.

```

função Unificação(x,y) retorna substituição para fazer x e y iguais
entrada x uma variavel, constante, lista ou composto
          y uma variavel, constante, lista ou composto

UnificarInterno(x,y, {})

```

```

UnificarInterno(x,y,t) retorna substituição para fazer x e y iguais dado t
entrada x uma variavel, constante, lista ou composto
          y uma variavel, constante, lista ou composto
          t uma substituição construida até agora

se t = falha retorna falha
senão se x = y retorna t
senão se Evariavel(x) retorna UnificaVariavel(x,y,t)
senão se Evariavel(y) retorna UnificaVariavel(y,x,t)
senão se Ecomposto(x) e Ecomposto(y)
    retorna UnificaInterno(x.arg, y.arg, UnificaInterno(x.op, y.op, t))
senão se Elista(x) e Elista(y)
    retorna UnificaInterno(x.resto, y.resto, UnificaInterno(x.l, y.l, t))
senão retorna falha

```

```

função UnificaVariavel(var,x,t) retorna uma substituição
entrada var uma variavel
          x uma expressão qualquer
          t uma substituição construida até agora

se substituir var por um valor pertence a t
    retorna UnificaInterno(valor,x,t)
senão se substituir x por um valor pertence a t
    retorna UnificaInterno(var,valor,t)
senão se var está em x
    retorna falha
senão retorna adicionar substituir x por var em t

```

Figura 3: Algoritmo de Unificação, programando o agente a criar novas regras baseando-se nas sentenças já conhecidas.

Fonte: Russell & Norvig. (2003)

Além de ser utilizada em agentes que possuem alguma forma de representação de conhecimento, sistemas lógicos são usados em algumas aplicações como linguagens de programação lógica, programas que provam teoremas e estudos cognitivos.

#### 4.4. PLANEJAMENTO

As técnicas de planejamento são próximas às de busca, porém possuem representações mais flexíveis dos estados, ações e objetivos, além de utilizarem previsões para resolver seus problemas.

Agentes inteligentes de planejamento usam linguagem e algoritmos específicos para cada situação com o intuito de reduzir o espaço de estados para aumentar a eficiência e assim conseguem resolver problemas mais complexos. Hoje em dia uma parte considerável dos programas de planejamento utiliza a linguagem STRIPS modificada e estendida especialmente para cada aplicação (EROL, NAU, & SUBRAHMANIAN, 1995). Essa linguagem foi desenvolvida a partir do cálculo situacional.

Outras técnicas muito utilizadas são: realizar buscas no espaço de planos, e não no espaço de estados; utilizar elos causais entre os planos e estados; e relações específicas entre a expressividade da linguagem utilizada e a eficiência do algoritmo utilizado (RUSSELL & NORVIG, 2003). Colocando agentes de planejamento para atuar em sistemas reais é necessário utilizar ainda uma série de outras astúcias, por causa de problemas como informações incompletas e incorretas sobre o ambiente exterior. Entre as principais estão o planejamento condicional, onde são planejados diversos cenários antes de ocorrer o estímulo que decidirá qual direção seguir, os módulos de re-planejamento, onde parte do planejamento é refeito para se adequar aos estímulos recebidos e o monitoramento das ações, que verificam se a ação executada realmente alterou o ambiente externo como planejado.

#### 4.5. APRENDIZADO

Outra área da inteligência artificial é a do aprendizado, onde são estudadas técnicas que permitem que o agente possua capacidades de melhorar seu desempenho com o tempo, de acordo com treinamento ou com tentativas e erros, aprendendo a melhor forma de agir. Técnicas de aprendizado são essenciais para ambientes desconhecidos, para compensar a falta de conhecimento do ambiente ou a falta de tempo durante a implementação do agente (RUSSELL & NORVIG, 2003). O aprendizado é basicamente um problema de ajuste, onde o agente recebe alguns pontos de uma função e deve calcular a posição dos outros pontos, ou seja, ele sabe como agir em algumas situações e deve aprender a agir em outras situações. Isso é feito incluindo dois blocos funcionais à concepção do agente: um elemento de desempenho que seleciona as ações a serem tomadas, e um de aprendizado que modifica o elemento de

desempenho de acordo com o método escolhido para que ele consiga melhorar sua atuação. O método escolhido depende do problema, da representação do conhecimento e das hipóteses consideradas.

Aprender através de exemplos é chamado de aprendizado por indução, neste contexto o agente deve ser exposto a um conjunto de dados chamado de dados de treinamento, que formarão os exemplos e posteriormente a um conjunto de teste, onde a desempenho do treinamento será verificada. Quanto maior o conjunto de dados de treinamento, melhor é a desempenho do agente nos dados de teste. A figura 4 mostra a diferença entre os métodos de aprendizado com exemplos e com recompensas.

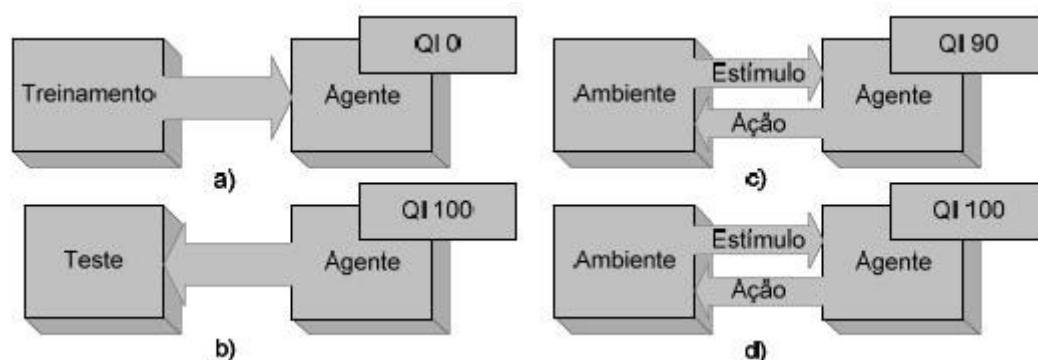


Figura 4: Métodos gerais de aprendizado do agente por indução mostrando seus resultados.

Fonte: Russell & Norvig. (2003)

O agente também pode aprender através de recompensas, ou seja, através de tentativa e erro. Neste caso é decidido qual o tipo de informação que deve ser aprendido, um modelo do ambiente ou um mapeamento entre ações e recompensas. Sempre é usada uma função de utilidade, que é calculada para determinar a influência de cada ação para alcançar o objetivo, que é aprender totalmente e nunca mais errar dadas as situações consideradas.

O método de calcular a função de utilidade varia para cada caso e pode alterar muito o desempenho do agente, podem ser usados o método dos mínimos quadrados, programação dinâmica adaptativa e de diferença temporal. Redes neurais são formas de representar funções, e que são usadas nos módulos de aprendizado dos agentes inteligentes.

Elas são classificadas em simples e realimentadas e de acordo com o número de camadas internas. O número de entradas e saídas é determinado com a natureza do problema, mas o número de camadas, e o número de neurônios em cada camada ainda não são determinados por nenhuma teoria. Cada neurônio é associado a uma função matemática que

recebe as entradas e devolve às saídas, o aprendizado da rede consiste em alterar os pesos dos elos entre os neurônios, para que a saída da rede seja mais precisa em relação ao objetivo, como mostrado na figura 5.

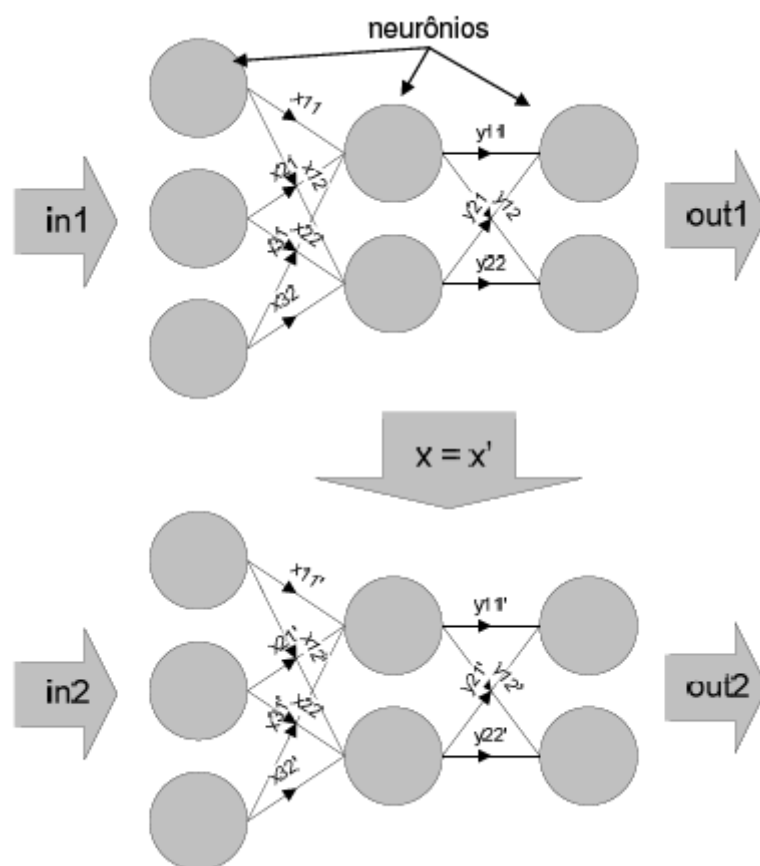


Figura 5: Exemplo de funcionamento de uma rede neural.

Fonte: Russell & Norvig. (2003)

#### 4.6. INTELIGENCIA ARTIFICIAL DE JOGOS TRADICIONAIS

Jogos foram um prato cheio para testar técnicas de inteligência artificial desde os primeiros programas inteligentes, por isso jogos como o xadrez (SHANNON, 1950) e damas são explorados a fundo (NOWAKOWSKI & ALBERT, 2007) há muito tempo. Talvez o acontecimento mais conhecido envolvendo programas inteligentes tenha sido quando o programa Deep Blue derrotou o campeão mundial de xadrez Garry Kasparov (SALETAN, 2007). Jogos são interessantes pelo fato de ser abstratos, possuírem regras bem definidas, serem baseados em turnos, permitirem uma informação completa do que está acontecendo e também por permitir apenas uma pequena quantidade de diferentes ações.

O programa feito especialmente para jogar esses jogos tradicionais utiliza-se de várias técnicas, diversas áreas dentro da inteligência artificial, mas abusam principalmente dos algoritmos de busca. Programas mais recentes, aplicados para jogos mais complexos como Xadrez e o Go utilizam técnicas mais exóticas como aprendizado, algoritmos genéticos ou até técnicas específicas criadas pelos programadores. Para os jogos mais simples como Othello e damas existem programas que jogam perfeitamente e nunca perdem (IWATA & KASAI, 1994) (SCHAEFFER, 2007). Mas há ainda os que persistem.

Um jogo interessante que possui uma alta complexidade é o Gamão. Além de um grande espaço de estados, ele não é determinístico, depende de números aleatórios para definir as possíveis jogadas do jogador. O software que consegue atingir o nível dos grandes mestres do jogo utiliza técnicas especiais de busca e aprendizado (TESAURO, 1995).

Xadrez é o jogo mais tradicional da cultura ocidental, ele sempre foi o que recebeu mais atenção dos programas inteligentes. Atualmente não é consenso, mas os programas se mostram melhores jogadores do que os grandes mestres (CHESSBASE, 2007). As técnicas utilizadas pelos melhores programas variam, mas praticamente todos abrimam mão da representação do conhecimento para terem maior força bruta e conseguirem aplicar algoritmos de busca em movimentos mais distantes.

Definitivamente o mais complexo de todos, Go é um jogo oriental de mais de dois mil anos, sendo o jogo de tabuleiro mais popular do Japão. É enorme a diferença entre o nível do melhor programa jogador de go e o dos grandes mestres (BOUZY, 2001). Isso é causado pelo tamanho do tabuleiro, pela grande influência que uma pedra pode fazer, pelos conceitos empregados durante a partida como territórios e pedras mortas, mas o principal é que uma partida não possui um fim bem definido, uma vez que os estados do jogo podem avançar e recuar em direção ao fim. Da mesma forma que com xadrez, os melhores programas jogadores de Go utilizam diferentes técnicas, alguns utilizam conhecimento, alguns utilizam muitas simulações com o algoritmo Monte Carlo (GELLY & WANG, 2006), alguns utilizam aprendizado ou até uma combinação de técnicas. O vencedor dos campeonatos mundiais de go muda todos os anos, por isso nada pode ser dito quanto a uma estratégia predominante.

#### 4.7. INTELIGÊNCIA ARTIFICIAL EM JOGOS COMERCIAIS

Formando hoje um importante aspecto da vida contemporânea, jogos de computador são interessantes por vários aspectos: alta tecnologia, beleza e arte, imergir o jogador em outros mundos, desenvolver capacidades mentais e físicas, entre outras (RABIN, 2005). Desde que começaram a invadir as casas das pessoas os jogos evoluíram bastante, junto com os computadores eles se tornaram mais complexos, intrigantes e envolventes. Para alimentar essa demanda por entretenimento e os aspectos mencionados acima, a indústria de desenvolvimento de jogos trabalha bastante e recebe bem por isso (DEMARIA, 2003).

O projeto de um jogo utiliza muitas pessoas de diversas especializações, principalmente arte, concepção de jogos, computação gráfica, inteligência artificial e redes de computador. O módulo inteligente de jogos de computador atuais está num período de valorização (LAIRD J, 2005). Hoje em dia, jogos que não possuem uma inteligência que atende as expectativas dos jogadores são mal vistos. Todos os jogos de computador precisam de um módulo inteligente, até os primeiros jogos como Pong<sup>1</sup> o possuíam (BUCHANAN, 2007). Apesar de ganhar cada vez mais importância durante o desenvolvimento dos jogos, a parte lógica nunca foi uma das prioritárias devido a várias limitações (LAIRD & LENT, 2005), ou talvez ao costume que os produtores possuam.

Essa é uma tendência que tende a mudar. As limitações para o uso de técnicas mais avançadas de inteligência artificial em jogos são várias: falta de tempo, dinheiro, mão-de-obra especializada durante o desenvolvimento, capacidade de processamento, de memória e o fato de que os jogos mudam muito durante a criação, e como o módulo inteligente do jogo é muito sensível a tudo isso, os programadores adotam a grande maioria das vezes técnicas de busca básicas sobre as ações possíveis que cada agente pode realizar (LAIRD & van LENT, 2005). Infelizmente, para determinar a dificuldade dos aspectos controlados pelo computador, geralmente são usados aspectos que o jogador não possui acesso, ou seja, o computador usa *cheating*<sup>2</sup> para aumentar o nível do desafio.

Nas principais convenções de desenvolvedores de jogos fica evidente que o maior esforço para melhorar a inteligência dos jogos é dispendido em jogos de tiro em primeira

---

<sup>1</sup> **Pong**, jogo em que o objetivo era acertar a esfera (bola) com sua raquete (barra vertical) e lançar para o campo adversário, marcando ponto quando a bola passasse ao campo adversário.

<sup>2</sup> **Cheating**, prática usada pelo jogador ou agente para trapacear no jogo modificando suas características comuns, dando-lhe vantagem sobre o jogador rival ou o agente rival.

pessoa, FPS ou *First Person Shooting*<sup>3</sup> (KIRBY & WOODCOCK, 2004). Neles as decisões tomadas pelos *NPCs*<sup>4</sup> devem ser rápidas mesmo envolvendo muitas possibilidades e mais importante do que tudo, eles devem trabalhar em grupo e seus movimentos devem parecer inteligentes (ISLA, 2007). A sensação de que o computador realmente pensa como humano para agir é um grande desafio para os programadores, e o cenário mais gratificante para os jogadores.

As tentativas de implementar técnicas mais avançadas de inteligência artificial nos jogos em sua grande maioria falharam (KIRBY & WOODCOCK, 2004), porém algumas pequenas inovações conseguiram destacar consideravelmente alguns jogos em especial, como *Black and White* (LIONHEADSTUDIOS, 2001) e *The Sims* (MAXIS, 2000). Outros jogos que possuem uma boa inteligência artificial não alcançaram tanto destaque por esse aspecto em especial por terem apenas tentado melhorar módulos inteligentes de jogos precedentes.

#### 4.8. INTELIGENCIA ARTIFICIAL PARA ESTRATEGIA BASEADA EM TURNOS

Estratégia baseada em turnos é um gênero de jogos de computador que simula jogos de tabuleiro, onde cada pessoa possui a sua vez de jogar e realiza ações apenas em seu turno. Os jogos mais tradicionais como xadrez e damas inspiraram esse gênero que adaptou essa estrutura para introduzir temas como guerra espacial e batalhas de fantasia medieval. Normalmente o jogador controla diferentes tipos de unidades (infantaria, espaçonaves, arqueiros, entre outros...) de acordo com a sua estratégia para derrotar seus inimigos (RABIN, 2005). Jogos de estratégia baseada em turnos possuem uma grande vantagem sobre os outros em relação ao tempo e à capacidade de processamento dedicada à inteligência (HINRICHS & FORBUS, 2007).

Porém, nenhum deles se destacou até hoje por uma boa inteligência artificial. Normalmente ela possui diversos defeitos e situações que os jogadores descobrem após o lançamento do jogo e utilizam muito o *cheating* para aumentar o desafio e conseguir oferecer alguma resistência a jogadores avançados.

---

<sup>3</sup> **First Person Shooting** é um estilo de jogo, no qual se enxerga apenas o ponto de vista do protagonista, como se o jogador e personagem do jogo fossem o mesmo observador.

<sup>4</sup> **NPCs**, sigla vem de “Non Player Characters” em ingles, refere-se a um personagem que interpreta respostas, simula comportamentos e interage com outros jogadores de uma forma logica imitando um jogador comum, porem não controlado por um dos jogadores participantes.

Foi realizado um estudo sobre a melhoria de técnicas de aprendizado e planejamento através de analogias, e um modelo qualitativo das cidades no jogo *FreeCiv* (HINRICHS & FORBUS, 2007).

Neste estudo proposto, é utilizada a técnica de planejamento HTN ou *Hierarchical Task Network*<sup>5</sup> em paralelo com técnicas de aprendizado baseado em analogias como a SME ou *Structural Mapping Engine*<sup>6</sup>. O estudo conclui que o modelo de representação é básico e que os esforços sobre o aprendizado estão ainda em fase inicial, mas que esse é um bom caminho.

Um segundo estudo propõe um método de desenvolvimento para o módulo inteligente do jogo que é adaptativo, o que em suma é uma combinação de técnicas de adaptações de planos e informações ontológicas que relacionam objetos no jogo (RUIZY, URBAN, AVILA, AGUDOY, & CALERO, 2004).

. As informações ontológicas são facilmente lidas em jogos do tipo estratégia baseadas em turnos. A proposta é de usar uma base de estratégias e um modo de escolher qual é a estratégia que se adapta melhor à situação atual do jogo através de uma base de conhecimento. A técnica foi comprovada na prática, mas os testes feitos não possuíam um número razoável de estratégias na base, o que impede de testar seu desempenho.



Figura 6: Imagem comparativa de alguns dos primeiros jogos de estratégia baseada em turnos, inteligência artificial avançada superando a necessidade de gráficos elaborados.

Fonte: Meyer e Intelligent Systems. (1991)

<sup>5</sup> **Hierarchical Task Network**, inteligência artificial que trabalha de forma a aprender e “discutir” seu aprendizado com um mais de um agente ou inteligência artificial disponíveis em uma rede.

<sup>6</sup> **Structural Mapping Engine**, inteligência artificial com conceitos de ciência cognitiva, onde o agente conhece estruturalmente seu ambiente ao ponto de fazer comparações e saber características físicas dos objetos os quais ele interagiu.

Comparando jogos de computador de estratégia baseada em turnos e jogos tradicionais, o módulo inteligente dos primeiros não pode ser exclusivamente baseado em força bruta (algoritmos de busca sem uma forma de representação do conhecimento), pois as jogadas possíveis são muito mais numerosas. Por isso, uma proposta para uma boa inteligência nestes jogos deve se basear prioritariamente em conhecimento e não em busca.

## 5. METODOLOGIA

Neste capítulo são apresentados os materiais e métodos utilizados durante o trabalho: o jogo para o qual foi implementado o módulo inteligente, a estrutura deste módulo e suas funcionalidades, e finalmente como foram organizadas as simulações para o estudo dos parâmetros deste módulo.

### 5.1 O JOGO FINAL FANTASY TACTICS REMAKE (FFT REMAKE)

O jogo FFT Remake é um jogo de estratégia baseada em turnos, criado Dalmon (2006), ex-aluno da Universidade de Campinas.

Surgiu como um projeto de um simulador chamado *Final Fantasy Tactics Remake* que foi baseado no jogo *Final Fantasy Tactics* (SQUARESOFT, 1997). Este jogo permite que o jogador controle um grupo de personagens alternando turnos num tabuleiro, batalhando contra o grupo de personagens controlado pela inteligência artificial, até que um deles seja derrotado. A figura 7 mostra uma batalha típica do jogo *Final Fantasy Tactics*.



Figura 7: Imagem do jogo *Final Fantasy Tactics* mostrando uma batalha.

Fonte: Squaresoft (1997)

Além da batalha, o jogo possui um menu de opções onde o jogador pode escolher a forma de configuração do personagem, ou seja, definir quais serão suas habilidades, seus atributos físicos, equipamentos (armas, escudos, armaduras) e sua classe.

Nesta figura percebem-se claramente os personagens e o tabuleiro. O personagem com as letras AT sobre ele é aquele que possui o turno ativo neste momento, apenas personagens com o turno ativo podem realizar ações. Durante um turno, o personagem pode se mover (mudar de posição no tabuleiro) e realizar uma ação (atacar um inimigo, curar um aliado, entre outras). As ações disponíveis para um personagem dependem de sua classe (seja cavaleiro, mago ou ladrão) e das habilidades que adquiriu ao longo do jogo.

A figura 8 mostra uma tela do jogo durante a montagem de um personagem. A forma com que o jogador age durante a batalha depende diretamente da configuração dos personagens de seu grupo. A eficiência das ações do personagem depende de seus atributos.

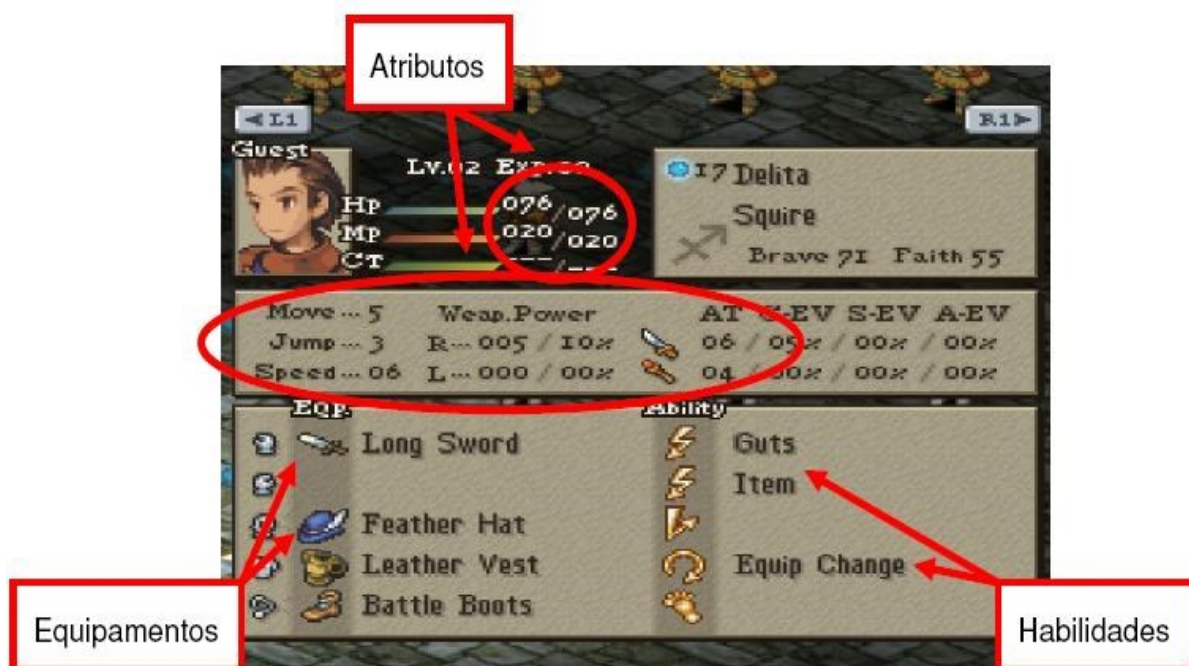


Figura 8: Imagem do jogo *Final Fantasy Tactics* da tela de configuração de personagem.

Fonte: Squaresoft. (1997)

O jogo FFT Remake é um jogo equivalente ao *Final Fantasy Tactics*, mas em modo texto. Ele possui as principais funções: salvar e carregar um jogo, menu de configuração dos personagens e a batalha. As figuras de 9 a 12 mostram respectivamente a tela inicial, o menu principal, o menu de configuração de personagens e uma batalha.



Figura 9: Tela inicial do jogo FFT Remake, com as opções de iniciar um novo jogo ou carregar um jogo já existente salvo na memória, permitindo continuar a partir do ponto onde a simulação parou.

Fonte: Dalmon (2006).

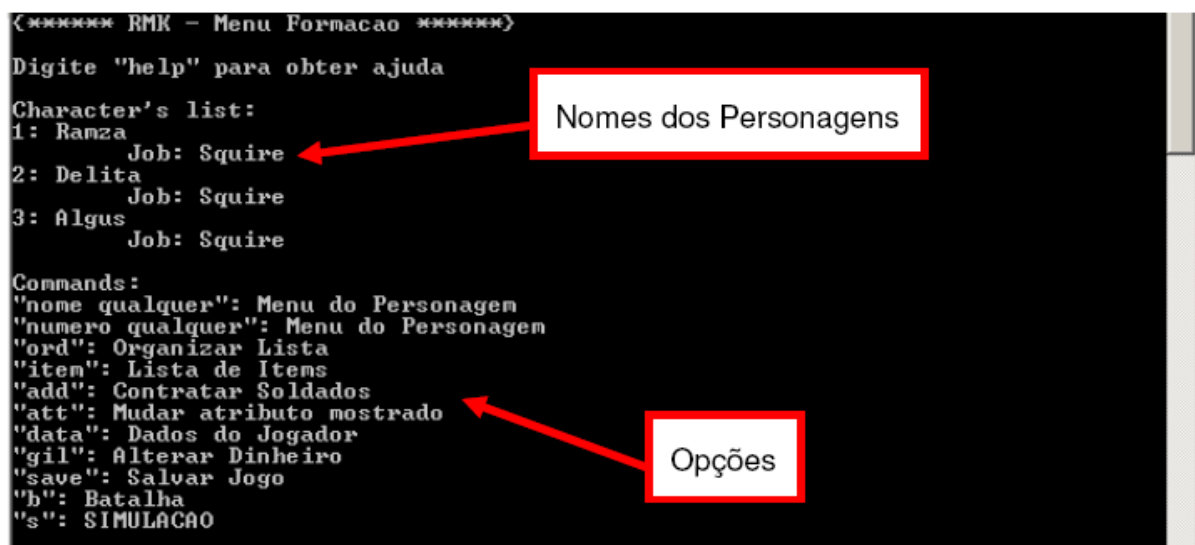


Figura 10: Menu principal do jogo FFT Remake, Tela onde mostra as opções na simulação e quais dados serão adicionados ao jogo.

Fonte: Dalmon. (2006).

Apesar de sua simplicidade gráfica, o jogo FFT Remake possui a mesma mecânica de jogo daquele em que foi baseado, isso somado ao fato de possuir todas as funções e opções para o jogador, faz com que ele seja um jogo adequado para a aplicação de um módulo de inteligência artificial equivalente àqueles usados em jogos comerciais.

A figura 12 representa os personagens pelos nomes “P 1”, “E 1”, “P 2”, assim por diante, as alturas do tabuleiro pelos números próximos aos vértices dos retângulos, o tipo de terreno (grama, pedra, etc...) nos números no centro dos retângulos, e as posições para as quais o personagem pode se movimentar com a marca “####”.

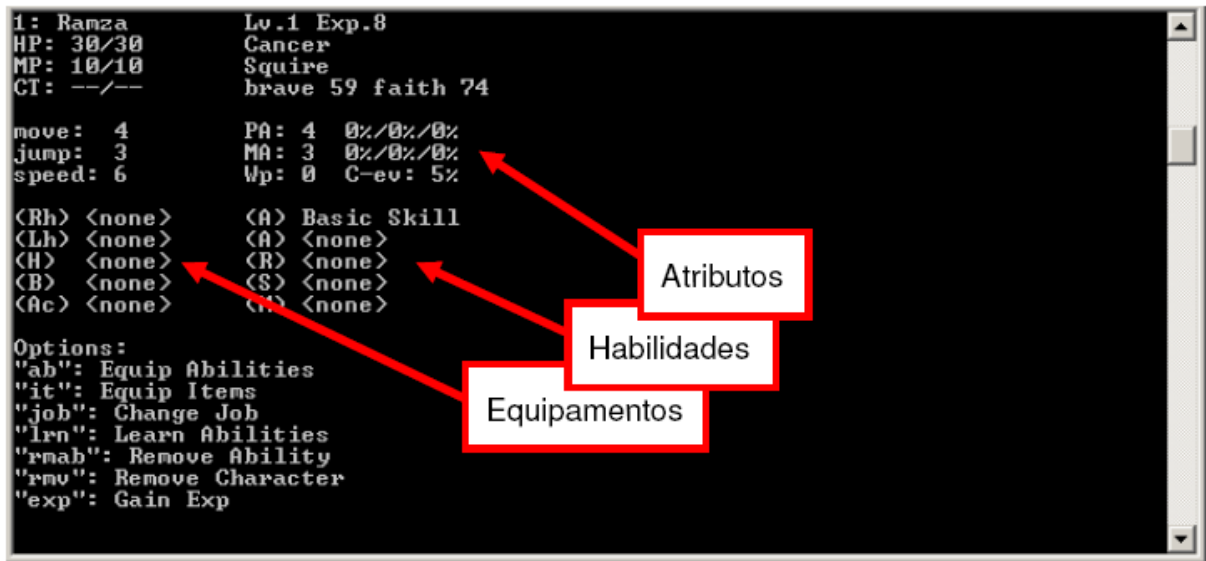


Figura 11: Tela do jogo FFT Remake de configuração de um personagem.

Fonte: Dalmon. (2006).

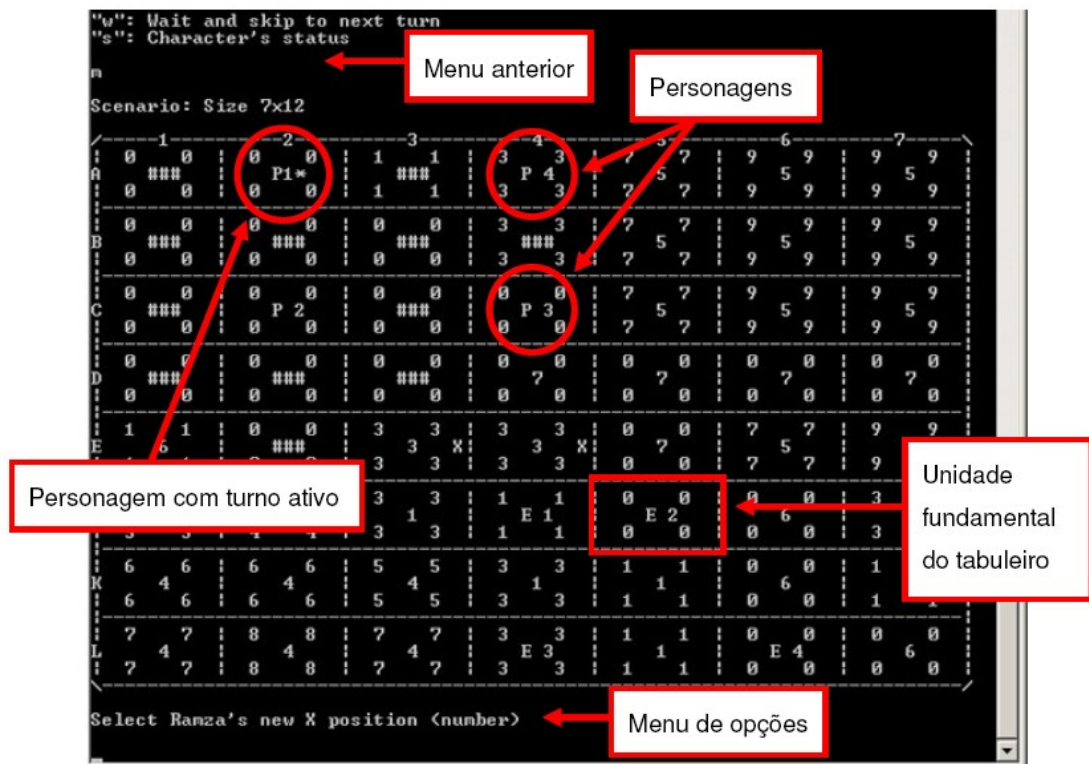


Figura 12: Tela do jogo FFT Remake de uma batalha.

Fonte: Dalmon. (2006)

Durante uma sessão de jogo, o jogador normalmente começa um novo jogo ou inicia um jogo em estado anterior. Em seguida prepara seus personagens da maneira que ele quiser e então parte para batalhar contra o “computador” (agente que controla os personagens inimigos). As decisões do jogador são feitas através de menus, como pode ser observado na parte de baixo da figura 10, e nas partes superior (menu precedente, onde foi feita a escolha da opção “m”) e na parte inferior da figura 11.

## **5.2. PROPOSTA DA ARQUITETURA DO MÓDULO DE IA**

A arquitetura proposta utiliza os conceitos estudados na revisão bibliográfica e as características que se destacam dos jogos de estratégia baseada em turnos. Além disso, a arquitetura pretende ser modelável de acordo com a necessidade do jogo e resolver todos os problemas enfrentados pelos jogadores por causa dos módulos de IA mal projetados dos jogos comerciais anteriores.

### **5.2.1. OBJETIVOS DO MÓDULO INTELIGENTE**

Para a elaboração da arquitetura proposta foram utilizados vários critérios para que depois de implementada, ela cumpra todos os objetivos propostos. Estes critérios listam todas as características que o módulo de inteligência artificial deve possuir, sua arquitetura deve atender a todos. Abaixo segue a lista de todos os critérios utilizados para a Inteligência Artificial do jogo:

- A) Deve induzir o divertimento e não ser o mais forte possível;
- B) Ser adaptável com relação ao objetivo da batalha;
- C) Ser adaptável com relação à estratégia que será seguida;
- D) A estratégia seguida deve ser passível de mudança durante a batalha, para ser ainda mais imprevisível e poder resolver situações bem distintas em uma única batalha.
- E) Ser adaptável também em relação ao nível de dificuldade;

**F)** Deve se adaptar à estratégia utilizada pelo oponente (neste caso o jogador) para que não seja previsível demais;

**G)** Deve organizar as ações das unidades em conjunto, para que elas não pareçam agir sozinhas;

**H)** Parecer ser inteligente, dando a impressão de inteligência ao jogador. Para tanto deve considerar todas as características da batalha para tomar suas decisões e realizar ações que surpreendam o jogador;

**I)** Os cálculos necessários para a determinação da melhor ação não precisam ser tão rápidos, pois a estrutura do jogo baseado em turnos torna o limite de tempo grande;

### **5.2.2. DESCRIÇÃO GERAL DA ARQUITETURA**

A arquitetura proposta possui três grandes partes: a estratégica, a tática e a adaptativa. As partes estratégicas e táticas são sistemas especialistas com representação de conhecimento que realizam as ações. A parte estratégica compreende a análise do ambiente externo, ou o estado da batalha e determina as tendências para cada tipo de ação a ser tomada, se as ações terão tendência a serem mais ofensivas, a utilizar um tipo específico de ação, entre outras. A parte tática recebe essas tendências e determina qual é a melhor sequência de ações que as unidades controladas devem realizar.

Essa divisão foi escolhida para atender aos critérios de ser adaptável ao objetivo da batalha e à estratégia desejada, e também ao critério de permitir que a estratégia seja alterada durante a batalha.

A parte adaptativa modifica a parte estratégica de forma a mudar o comportamento do módulo inteligente durante uma batalha. Após cada ação realizada, seja do módulo inteligente do computador ou do jogador, a parte adaptativa é chamada para modificar se necessária, a estratégia atual, determinando se os resultados das ações condizem com o objetivo da estratégia. Essa parte é necessária para atender aos critérios de ser adaptável à estratégia do oponente e para modificar a estratégia utilizada durante uma batalha, além de aumentar a impressão de inteligência, o que são pontos cruciais para o êxito da arquitetura.

Para que a arquitetura possa ser adaptável quanto ao nível de dificuldade, devem ser utilizadas várias estratégias diferentes, umas mais elaboradas e outras menos, considerando mais ou menos características do ambiente externo para tomar as decisões.

Para que a arquitetura possa organizar as ações das unidades em conjunto, a parte tática deve determinar a sequência de ações para todas as unidades de uma só vez, elaborando um trabalho em grupo, até que o ambiente externo seja modificado e que seja necessário determinar todas as ações novamente.

A parte estratégica deve ser facilmente modificável (pela parte adaptativa) e parametrável (para podermos definir vários níveis de dificuldade e diferentes estratégias). A figura 13 mostra um diagrama esquemático com o funcionamento descrito.

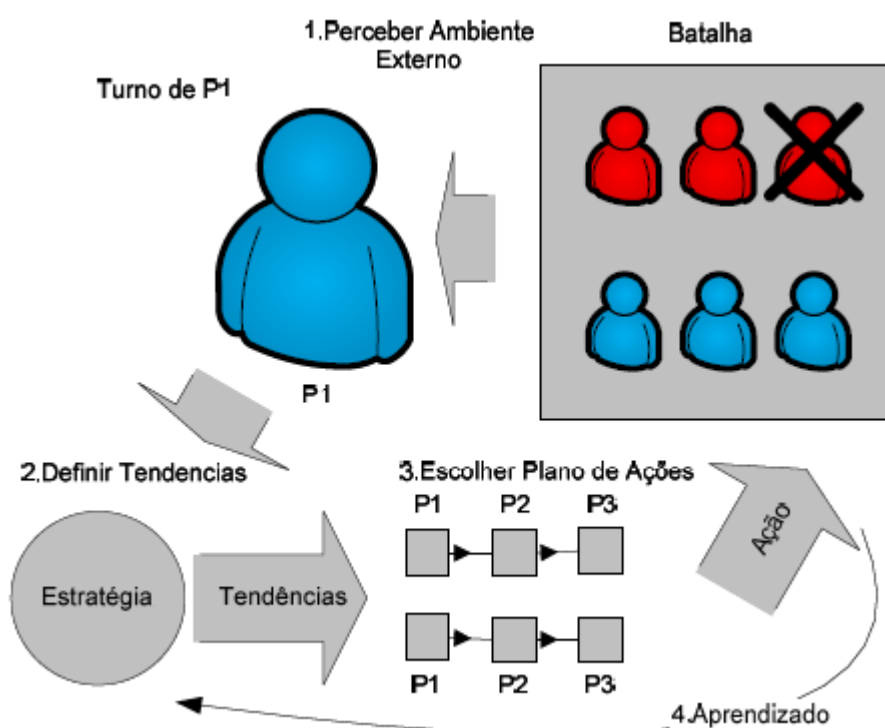


Figura 13: Diagrama do funcionamento da arquitetura proposta.

Fonte: Dalmon. (2006)

Vale salientar que o aprendizado de uma batalha não é transmitido para outra batalha, pois dessa forma o agente inteligente ficaria forte demais com o avanço no jogo, e ficaria especialista demais ao mesmo tempo. O conceito que o aprendizado proposto traz é que o agente controlado pelo computador possui uma estratégia inicial, elaborada de acordo com o

estágio do jogo, e ela se adapta em função do desenrolar da batalha, mas os valores iniciais que utiliza não dependem das batalhas travas anteriormente.

### 5.2.3 IMPLEMENTAÇÃO

Durante a implementação, a estratégia é um conjunto de pesos que associam uma característica da batalha com um tipo de habilidade. Os tipos de habilidade são classificações que definem o objetivo com que a habilidade deve ser usada durante o jogo. A tendência é um conjunto de números, um para cada tipo de habilidade, e é o resultado da função estratégica, conforme mostrado na figura 14.

A lista de tendências é passada para a função de escolha da melhor ação, que a executa, o funcionamento dessa função é da seguinte maneira: cada ação disponível para cada personagem é classificada de acordo com o seu tipo, e logo em seguida classificada de acordo com a alteração que provocará no ambiente externo (quanto maior, melhor); assim a ação disponível do tipo indicado pela lista de tendências cujo efeito no ambiente externo é maior, é escolhida como a ação a ser realizada.

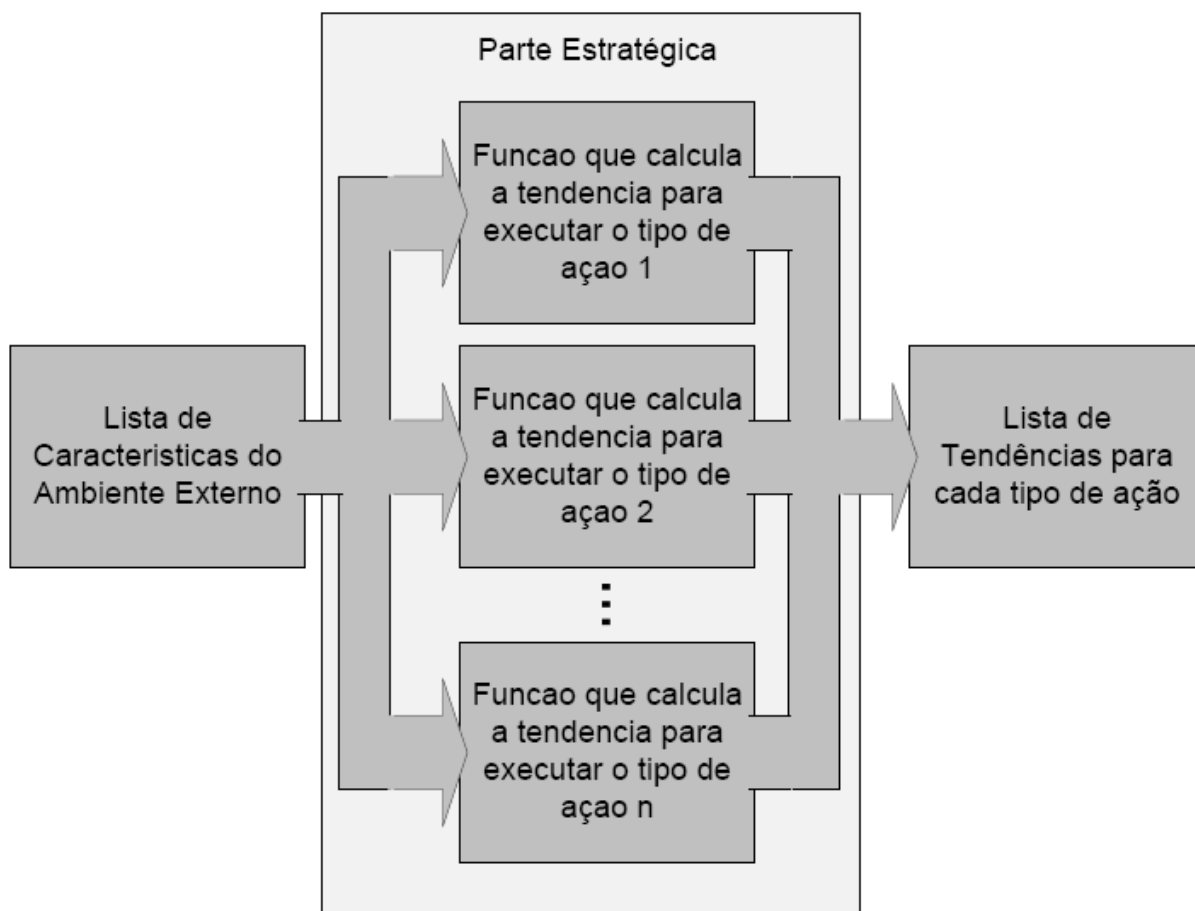


Figura 14: Funcionamento da parte estratégica do jogo.

Fonte: Dalmon. (2006)

Após cada ação, seja do jogador ou do módulo inteligente, a função de aprendizado é chamada, o que altera a estratégia de acordo com o êxito da habilidade utilizada. O funcionamento da função aprendizado é verificar a alteração pra avaliar se a ação realizada no ambiente externo promoveu um bom resultado ou um resultado ruim (se atendeu os objetivos previstos pelo tipo da ação executada ou não) e calcular modificações em cada uma das funções internas da parte estratégica de acordo com o êxito ou não da ação executada. Por exemplo, se um ataque direto não foi efetivo o suficiente, essa função modificará a função interna da parte estratégica que determina a tendência de realizar ataques diretos, reduzindo em geral o valor que ela retorna.

A figura 15 mostra os algoritmos utilizados para a implementação das três funções;

- 1) Utilização da estratégia para determinação das tendências;
- 2) Utilização das tendências para determinar a melhor sequência de ações;
- 3) A função de aprendizado.

|                                                                         |
|-------------------------------------------------------------------------|
| função tendencia                                                        |
| <b>parametros :</b> lista de personagens, cenário                       |
| <b>retorna :</b> lista de tendencias                                    |
| <b>como :</b> para cada tipo de ação                                    |
| para cada informação no estado da batalha                               |
| multiplica a informação pelo peso correspondente para este tipo de ação |
| soma tudo resultando na prioridade que sera dada ao tipo de ação        |

|                                                            |
|------------------------------------------------------------|
| função agir                                                |
| <b>parametros :</b> personagem, lista de tendencias        |
| <b>retorna :</b> ação                                      |
| <b>como :</b> para cada ação que o personagem pode fazer   |
| compara o tipo da ação com a lista de tendencias           |
| calcula a provavel mudança do estado da batalha            |
| escolhe a ação onde a mudança do estado da batalha é maior |

|                                                                                      |
|--------------------------------------------------------------------------------------|
| função aprender                                                                      |
| <b>parametros :</b> estado anterior, estado atual, ação realizada                    |
| <b>retorna :</b> modificações nos pesos internos da função tendencia                 |
| <b>como :</b> determina o resultado da ação, se boa ou ruim                          |
| para cada tipo de ação                                                               |
| para cada peso                                                                       |
| verifica se o tipo da ação realizada é parecido ou não                               |
| verifica se o peso influenciou                                                       |
| usando essa comparação e o resultado da ação associa um valor de modificação do peso |

Figura 15: Algoritmos utilizados nas funções do módulo inteligente do jogo FFT Remake.

Fonte: Dalmon. (2006)

A arquitetura implementada não possui todas as funcionalidades da arquitetura proposta. Impedida por fatores de tempo, as funções de planejar as ações de todas as unidades do grupo de uma vez não foram implementadas. Além disso, apenas um objetivo de batalha (derrotar todos as unidades do adversário) foi estudado.

### 5.3. SIMULAÇÕES PARA O ESTUDO DOS PARÂMETROS

O estudo dos parâmetros mais influentes no resultado obtido pelo módulo inteligente do jogo é essencial para a formalização da sua eficiência na solução proposta para os problemas citados e para determinar o procedimento que deve ser utilizado para conseguir diferentes efeitos quando aplicado em um jogo, como diferentes estratégias, níveis de dificuldade, entre outros.

### 5.3.1. ORGANIZAÇÃO DAS SIMULAÇÕES

Os principais parâmetros para o funcionamento da arquitetura proposta são: a estratégia (conjunto de pesos que associam um valor numérico para cada par formado por uma característica da batalha e um tipo de habilidade); a função objetivo utilizada para determinar a melhor ação; e a função objetivo para determinar a eficiência de uma ação na função de aprendizado. Neste trabalho são estudados o primeiro e o terceiro.

As simulações foram organizadas da seguinte maneira: uma batalha é realizada num cenário (o tabuleiro onde se encontram os personagens), com os agentes utilizando estratégias e funções de objetivo de aprendizado, sendo que cada um utiliza unidades cuja configuração é definida por um arquivo de jogo salvo. Foram criados 2 cenários diferentes, 3 estratégias (valores iniciais) e 3 funções objetivo de aprendizado. Estes parâmetros serão analisados em todas as suas combinações.

As configurações para as unidades dos jogadores possuem um número muito grande de possibilidades, pois contam com as habilidades disponíveis para cada personagem, sua classe e seus equipamentos, sendo que cada jogador controla quatro personagens. Por isso o número de habilidades que os personagens possuem foi definido em três níveis (nenhuma habilidade, uma habilidade de 3 tipos, e várias habilidades de todos os tipos), e são utilizados quatro classes, com equipamentos idênticos. Para explorar uma quantidade razoável de combinações entre classes e nível de habilidades entre os 8 personagens da batalha, foram preparadas 20 configurações de personagens para as simulações.

Além disso, em cada uma das condições acima, 10 batalhas são realizadas com os personagens em posições iniciais na batalha ligeiramente diferentes, este número foi considerado bom o suficiente para podermos estudar o desvio padrão e a variância dos resultados em função do ruído (provocado pelas posições iniciais diferentes dos personagens). As simulações são rodadas a partir de uma função especial colocada no código do jogo que realiza para um dado arquivo de configuração dos personagens todas as batalhas em cada condição possível e salva num arquivo os resultados obtidos (sequencia de ações e jogador vencedor).

Foram realizadas cerca de 1600 simulações (9 estratégias, 9 funções de aprendizado, 2 cenários e 10 batalhas) com cada arquivo de configuração de personagens. Os resultados do vencedor de cada uma destas batalhas foram analisados e comparados aos parâmetros que definem o funcionamento do agente inteligente. Primeiramente analisamos o efeito dos

valores iniciais da estratégia do agente, e posteriormente os efeitos das funções de aprendizado utilizadas no seu número de vitórias ao longo das simulações.

### 5.3.2. PARÂMETROS DAS SIMULAÇÕES

Para melhor compreender os resultados das simulações é necessário compreender a natureza dos parâmetros e seus valores utilizados durante cada uma das batalhas.

**Estratégia:** foram consideradas 3 estratégias diferentes: a estratégia 1 é a chamada neutra, que possui a mesma tendência de realizar ações ofensivas e defensivas; a estratégia 2 é a chamada ofensiva, que possui tendência maior em realizar ações de tipo ofensivo; finalmente a estratégia 3 é o oposto, possuindo tendência maior em realizar ações de tipo defensivo.

**Aprendizado:** foram considerados 3 tipos de aprendizado: o aprendizado 1 é o aprendizado lento, onde os valores dos pesos da estratégia são variados pouco a pouco; o aprendizado 2 é o aprendizado rápido, onde a cada ação os pesos são modificados consideravelmente; o aprendizado 3 é uma função sem aprendizado, ou seja, os pesos nunca são modificados.

**Cenários e posições iniciais:** foram criados 2 cenários para as simulações, um onde não há acidentes de terreno e as unidades podem se mover livremente e um onde há vários obstáculos. As posições iniciais dos personagens são limitadas pra cada time (grupo de unidades controlado por um agente) em 12 posições diferentes, sendo que cada time começa a batalha numa extremidade do cenário.

**Habilidades de um personagem:** unidades ou personagem utilizados nas simulações possuíam 3 tipos de conjunto de habilidades: o conjunto 1 não possui nenhuma habilidade, assim o personagem possui apenas uma ação disponível, o ataque simples; o conjunto 2 possui 3 habilidades de tipos (funções durante a batalha) distintos, por exemplo causar dano, curar e reduzir atributo do inimigo; o conjunto 3 abrange todas as habilidades que a classe permite, o que normalmente, depende da classe do personagem, possui de 4 a 8 tipos de ação.

**Classes de um personagem:** as classes consideradas foram 4: a primeira é o cavaleiro que possui fortes ataques à curta distância e pode reduzir os atributos e quebrar equipamentos dos inimigos; a segunda é o mago de ataque que possui ataques poderosos à distância, podendo

também bloquear seus adversários, mas é fraco em combate corpo-a-corpo; o mago de defesa que possui habilidades de cura à distância e o arqueiro que possui ataques físicos à distância mas é fraco em ataques próximos.

**Times ou grupos de personagens:** um arquivo pronto para a simulação contém 8 personagens configurados com um conjunto de habilidade e com uma classe, por exemplo o arquivo 8cv1 possui 8 cavaleiros com o conjunto 1 de habilidades. Dessa forma foram estudados, nos 20 arquivos citados, times de cavaleiros, times de magos de ataque e times mistos, com várias combinações de conjuntos de habilidades.

## 5.4 RESULTADOS E DISCUSSÕES

Neste capítulo são mostrados e discutidos os resultados obtidos com o módulo de inteligência artificial implementado pelo autor no jogo FFT Remake. A partir do funcionamento do módulo aplicado no jogo normal, ou seja, no caso de um jogador humano interagir com o jogo, pode-se notar os resultados quanto aos problemas apresentados pelos jogadores deste tipo de jogo, o que é apresentado na seção anterior. O papel de cada parte interna do módulo implementado e de seus parâmetros foi estudado através de simulações, onde um programa foi implementado à parte para conduzir e rodar as simulações explorando os diferentes valores dos parâmetros estudados.

### 5.4.1 RESULTADOS DE IMPRESSÃO DE INTELIGENCIA

Os resultados obtidos com o módulo de inteligência artificial implementado com relação à impressão de inteligência, ou seja, com a sensação percebida pelo jogador durante sua interação com o agente inteligente. Para analisar estes efeitos, foram realizadas diversas batalhas (sessões de jogo) com a sequência de todas as ações tomadas registrada. A partir da análise da natureza das sequências de ações, é possível verificar o comportamento do módulo inteligente frente a uma situação onde normalmente os jogadores percebem falhas e defeitos na inteligência artificial do jogo. Abaixo são mostrados exemplos de sequências de ações tomadas durante as batalhas analisadas e uma análise sobre a impressão de inteligência que o jogador pode ter durante a sessão de jogo.

**Exemplo 1:** Situação 3 personagens vivos para cada time, no caso de times de configurações iguais, o time ofensivo continua atacando e o time defensivo deixa de atacar para curar e reviver seus personagens.

**Exemplo 2:** Um time tenta atacar seus adversários, porém este último o atrasa com habilidades que o deixam sem agir durante um determinado número de turnos. Dessa forma fica evidente que uma habilidade de remover esses bloqueios fará uma grande diferença para o estado da batalha, e quando dois ou mais personagens são afetados por esses bloqueios habilidades de remoção são usadas.

**Exemplo 3:** Um time ataca os adversários primeiramente com habilidades que quebram seus equipamentos, e só então ataca-os. Porém, quando a defesa proveniente dos equipamentos é pequena em comparação à defesa total do personagem, esse time ataca normalmente desde o início.

**Exemplo 4:** Um time enfrenta um outro que possui personagens magos, personagens magos precisam de MP para usar suas habilidades. Caso a quantidade de dano no MP do alvo que um personagem deste time possa causar seja considerável, ele elimina o MP do alvo antes ao invés de atacar normalmente.

**Exemplo 5:** Um time que ataca os personagens adversários, mas existe um personagem adversário que curam os outros, fazendo com que ataques normais não sejam tão eficientes. Dessa forma, além de mudar o tipo de habilidade preferida pelo time, o time preferirá atacar o personagem que possui habilidades de cura, para poder vencer a batalha.

Estes são resultados que não são vistos com frequência em jogos comerciais de estratégia baseado em turnos, o que revela algumas melhorias que a aplicação desta arquitetura pode provocar num jogo. O Exemplo 1 nos mostra que é possível implementar times com configurações idênticas mas com comportamentos durante a batalha completamente diferentes, isso faz com que o jogador perceba que o computador possui um objetivo diferente, que reage de maneira diferente e se sinta desafiado de outra forma. Os Exemplos 2 e 5 mostram que a estratégia utilizada muda com o passar da batalha, enfrentando situações ruins, o agente inteligente muda aos poucos suas preferências e passa a agir de uma outra forma, procurando se adaptar e resolver o problema, vencendo a batalha. Em jogos

comuns, os inimigos sempre agem da mesma maneira, não importa se estão perdendo ou não, se o jogador utiliza a mesma estratégia desde o início ou não. Isso surpreende o jogador acostumado com a inteligência artificial de jogos comuns, pois ele sempre agiu da mesma maneira, e se esta maneira foi eficiente num dado momento, ela será em todos.

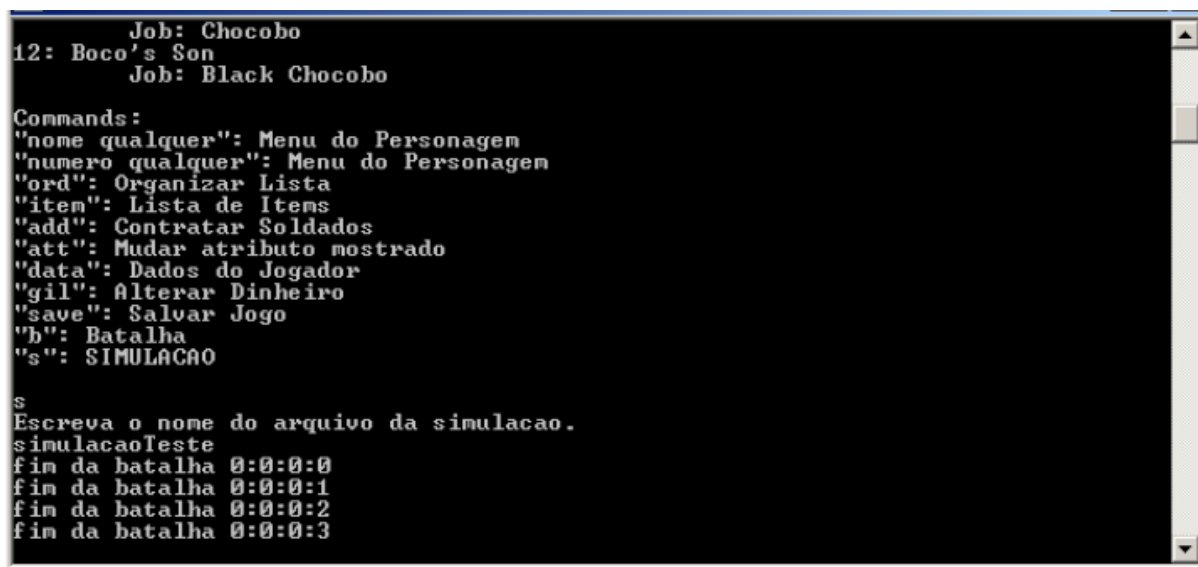
No caso dos Exemplos 3 e 4 o agente muda de estratégia conforme a estratégia do adversário, que depende de características de seus personagens, e ele considera essas características para agir de forma diferente com cada personagem. Esta estratégia portanto, considera a estratégia utilizada pelo oponente de forma diferente da consideração dos Exemplos 2 e 5, que é uma consideração ao longo do passar da batalha.

Todos estes fatores funcionando em conjunto trazem ao jogador experiências muito mais atrativas do que as proporcionadas pelos módulos de inteligência artificial de jogos comerciais comuns.

É importante ressaltar que a arquitetura proposta permite a inclusão de conceitos que estenderiam a atratividade para os jogadores que não foram implementados e estudados como o nível de dificuldade, movimentos de personagens com diferentes objetivos, ações de personagens sincronizadas (trabalho em grupo) e batalhas com objetivos diferentes do que apenas derrotar todas as unidades do adversário. Para a implementação de diferentes níveis de dificuldade, por exemplo, é necessário preparar diferentes valores iniciais para a estratégia do agente inteligente, considerando um número diferente de características do estado da batalha, quanto maior esse número, mais difícil e mais inteligente é o agente.

#### **5.4.2 RESULTADOS DAS SIMULAÇÕES**

A estrutura dos resultados das simulações é constituída de dois arquivos: um com todos os movimentos e ações tomadas pelos personagens nas batalhas simuladas, e outro arquivo com apenas os parâmetros utilizados em cada batalha e o time vencedor. A figura 16 mostra a tela do programa de simulações, a figura 17 mostra a estrutura dos arquivos que são resultado da simulação.



```

Job: Chocobo
12: Boco's Son
Job: Black Chocobo

Commands:
"nome qualquer": Menu do Personagem
"numero qualquer": Menu do Personagem
"ord": Organizar Lista
"item": Lista de Items
"add": Contratar Soldados
"att": Mudar atributo mostrado
"data": Dados do Jogador
"gil": Alterar Dinheiro
"save": Salvar Jogo
"h": Batalha
"s": SIMULACAO

s
Escreva o nome do arquivo da simulacao.
simulacaoTeste
fim da batalha 0:0:0:0
fim da batalha 0:0:0:1
fim da batalha 0:0:0:2
fim da batalha 0:0:0:3

```

Figura 16: Tela do programa enquanto a simulação é executada.

Dalmon. (2006).

A partir do menu principal do jogo FFT Remake, mostrado também na figura 10, é possível lançar as simulações com a configuração de personagens carregada do arquivo escolhido. Em seguida é necessário definir o nome do arquivo para identificar o resultado da simulação, e as simulações das batalhas começam. O fim de cada batalha é indicado com uma linha que é impressa na tela indicando os parâmetros utilizados em sua simulação: Estratégia, Aprendizado, Cenário, Batalha. Ou seja, a batalha 0:0:0:3 é a terceira batalha com as estratégias em 0, o aprendizado em 0 e o cenário 0.

| simulacaoTeste                                   | simulacaoTeste_resultados              | simulacaoTeste | simulacaoTeste_resultados           |
|--------------------------------------------------|----------------------------------------|----------------|-------------------------------------|
| 1 0: P 1 Delita esta na posicao Ax3              | 1 Batalha 0:0:0:0                      | 1              | Batalha 0:0:0:0                     |
| 2 1: P 2 Agrias esta na posicao Bx1              | 2 The Battle is over. Team One WINS!!  | 2              | The Battle is over. Team One WINS!! |
| 3 2: P 3 Albus esta na posicao Bx4               | 3                                      | 3              |                                     |
| 4 3: P 4 Ovelia esta na posicao Cx3              | 4 Batalha 0:0:0:1                      | 4              | Batalha 0:0:0:1                     |
| 5 4: E 1 Gafgarion esta na posicao Lx6           | 5 The Battle is over. Team Two WINS!!  | 5              | The Battle is over. Team Two WINS!! |
| 6 5: E 2 Meliadul esta na posicao Jx4            | 6                                      | 6              |                                     |
| 7 6: E 3 Ramza esta na posicao Kx6               | 7 Batalha 0:0:0:2                      | 7              | Batalha 0:0:0:2                     |
| 8 7: E 4 Alma esta na posicao Kx7                | 8 The Battle is over. Team Two WINS!!  | 8              | The Battle is over. Team Two WINS!! |
| 9 P1* Delita andou de Cx1 para Dx4               | 9                                      | 9              |                                     |
| 10 P2* Agrias andou de Ax2 para Cx4              | 10 Batalha 0:0:0:3                     | 10             | Batalha 0:0:0:3                     |
| 11 P3* Albus andou de Dx2 para Fx4               | 11 The Battle is over. Team Two WINS!! | 11             | The Battle is over. Team Two WINS!! |
| 12 P4* Ovelia andou de Cx3 para Fx4              | 12                                     | 12             |                                     |
| 13 E1* Gafgarion andou de Fx12 para Hx6          | 13 Batalha 0:0:0:4                     | 13             | Batalha 0:0:0:4                     |
| 14 E2* Meliadul andou de Dx10 para Fx4           | 14 The Battle is over. Team Two WINS!! | 14             | The Battle is over. Team Two WINS!! |
| 15 E2* Meliadul causou 6 de dano em P 4 Ovelia!! |                                        |                |                                     |

Figura 17: Imagem do registro de resultados da simulação.

Fonte: Dalmon. (2006)

A partir destes arquivos é possível analisar o comportamento dos agentes inteligentes em cada batalha, e é possível estudar o êxito que cada agente, definido pela combinação de parâmetros, obteve ao longo de todas as batalhas simuladas. As próximas seções apresentam e discutem sobre os resultados obtidos desta forma.

#### **5.4.3 EFEITOS DOS VALORES INICIAIS**

Para a computação das vitórias de cada time durante as várias simulações foi confeccionado um software específico para ler os arquivos de texto e contar as vitórias de cada time para cada combinação de parâmetros da simulação. Esses resultados foram agrupados e analisados em uma planilha, de forma a estudar o efeito dos valores iniciais da função estratégia do módulo inteligente do jogo com a variação dos outros parâmetros: configuração dos personagens e função de aprendizado utilizada.

Os resultados foram resumidos da seguinte maneira: os arquivos com configurações de personagens com os conjuntos de habilidades 2 e 3 não apresentaram grande variação nos resultados, e por isso são apresentados juntos; além disso, os resultados das simulações com arquivos de configuração de personagens de times mistos (que utilizam todas as classes) não apresentaram taxa de vitória ou derrota superior a 4%, e portanto não permitem nenhuma conclusão concreta. A tabela 1 mostra um resumo dos resultados obtidos com as simulações com relação à variação dos valores iniciais para os pesos da função estratégia do módulo de inteligência artificial do jogo FFT Remake.

O principal resultado da tabela 1 é que nenhuma porcentagem de vitória nas batalhas em questão foi muito maior do que 60%, a grande maioria das porcentagens de vitória fica entre 45 e 55%, o que permite a conclusão de que muitas vezes os parâmetros escolhidos, ou a forma com que foram utilizados, não foi crítico para o resultado das batalhas. Isto pode indicar dois fatos: a as mudanças nos parâmetros da arquitetura proposta não alteram suficientemente o resultado das batalhas; além disso, os valores parâmetros escolhidos não foram estudados a fundo para determinar a faixa de maior influência, e, portanto os parâmetros escolhidos para serem estudados não são os mais influentes sobre o vencedor das batalhas.

| Configuração |                | Porcentagem de vitórias do time1 |                     |                     |
|--------------|----------------|----------------------------------|---------------------|---------------------|
|              | time1 vs time2 | Estratégia 1 contra              | Estratégia 1 contra | Estratégia 2 contra |
|              |                | Estratégia 2                     | Estratégia 3        | Estratégia 3        |
| 1            | 4cv1 vs 4cv1   | 48                               | 60                  | 58                  |
| 2            | 4cv1 vs 4cv2   | 41                               | 48                  | 47                  |
| 3            | 4cv2 vs 4cv2   | 42                               | 53                  | 62                  |
| 4            | 4ma1 vs 4ma1   | 49                               | 57                  | 56                  |
| 5            | 4ma1 vs 4ma2   | 52                               | 47                  | 48                  |
| 6            | 4ma2 vs 4ma2   | 40                               | 42                  | 42                  |
| 7            | 4cv1 vs 4ma1   | 52                               | 55                  | 53                  |
| 8            | 4cv1 vs 4ma2   | 42                               | 54                  | 63                  |
| 9            | 4cv2 vs 4ma1   | 41                               | 51                  | 47                  |
| 10           | 4cv2 vs 4ma2   | 52                               | 53                  | 49                  |
| 11           | misto1         | 51                               | 61                  | 52                  |
| 12           | misto2         | 47                               | 53                  | 49                  |

Figura 18: Mostra a tabela de resultados da simulação de vitórias.

Fonte: Dalmon. (2006)

Considerando que uma diferença de mais ou menos 5% é significativa, podemos concluir diversos fatos a partir desta tabela:

**A)** A estratégia 3 é fraca com personagens sem habilidades (linhas 1 e 4) e também com personagens magos com habilidades (linha 8);

**B)** Em compensação, a estratégia 3 é a mais forte no caso de confronto entre times de personagens magos;

**C)** No caso de personagens cavaleiros com habilidades (linhas 2 e 3), a estratégia 2 é a melhor;

**D)** Para personagens cavaleiros com habilidades enfrentando magos (linha 9), a estratégia 2 é predominante;

E) Finalmente, no caso de times mistos, a única conclusão que pode ser tomada é de que para personagens sem habilidade (linha 11), a estratégia 1 é mais forte contra a 3.

Em suma, o efeito dos diferentes valores iniciais das estratégias é um pouco mais forte em alguns casos específicos. Um estudo mais aprofundado sobre os valores iniciais utilizados, e a proporção com que eles são colocados nos diferentes parâmetros, deve ser feito. Esperava-se que a diferença nas estratégias provocasse uma diferença maior nas porcentagens de vitória de times com personagens que possuísem habilidades de vários tipos, porém, apesar dos resultados apresentados na seção 5.1., esses comportamentos não são importantes quando é analisada a influência no resultado da batalha.

#### 5.4.4 EFEITOS DA FUNÇÃO OBJETIVO DE APRENDIZADO

De maneira análoga à forma apresentada na seção 5.3., os resultados obtidos nas simulações em relação ao modo de aprendizado (função objetivo de aprendizado) estão resumidos na figura 18:

| Configuração |                | Porcentagem de vitórias do time1  |                                   |                                   |
|--------------|----------------|-----------------------------------|-----------------------------------|-----------------------------------|
|              | time1 vs time2 | aprendizado 1 vs<br>aprendizado 2 | aprendizado 1 vs<br>aprendizado 3 | aprendizado 2 vs<br>aprendizado 3 |
| 1            | 4cv1 vs 4cv1   | 39                                | 48                                | 52                                |
| 2            | 4cv1 vs 4cv2   | 51                                | 59                                | 52                                |
| 3            | 4cv2 vs 4cv2   | 42                                | 45                                | 44                                |
| 4            | 4ma1 vs 4ma1   | 55                                | 53                                | 54                                |
| 5            | 4ma1 vs 4ma2   | 52                                | 51                                | 48                                |
| 6            | 4ma2 vs 4ma2   | 43                                | 45                                | 52                                |
| 7            | 4cv1 vs 4ma1   | 51                                | 55                                | 52                                |
| 8            | 4cv1 vs 4ma2   | 47                                | 54                                | 53                                |
| 9            | 4cv2 vs 4ma1   | 55                                | 56                                | 55                                |
| 10           | 4cv2 vs 4ma2   | 54                                | 52                                | 55                                |
| 11           | misto1         | 50                                | 51                                | 58                                |
| 12           | misto2         | 52                                | 47                                | 56                                |

Figura 19: Resultados das simulações para as funções objetivo de aprendizado utilizado.

Fonte: Dalmon. (2006)

As conclusões que podem ser tiradas desta tabela são similares às da tabela 1, nenhum valor ultrapassa a diferença de 10 pontos percentuais da média, ou seja, nenhuma combinação entre os modos de aprendizado foi extremamente significativa para o resultado do vencedor das batalhas. Porém, da mesma forma, podemos ainda concluir algumas tendências em casos isolados de combinações de modos de aprendizado:

- 1) Na linha 2, o aprendizado 1 é melhor contra o aprendizado 3, ou seja, no caso de um time sem habilidades enfrentar um outro com habilidades;
- 2) No caso dos personagens de ambos os times possuírem habilidades (linha 2), o modo de aprendizado se sobressai sobre o modo 2, que por sua vez supera o modo 1;
- 3) Porém, para personagens magos com habilidades (linha 6), o modo de aprendizado 2 predomina frente aos outros;
- 4) para times de várias classes, o aprendizado 2 supera na maioria das vezes o aprendizado 3.

Mais uma vez deve ser observado que essas tendências apresentadas possuem valores próximos à média, e podem não possuir exatamente o significado apresentado. Foi assumido que a influência dos valores iniciais da estratégia sobre o vencedor da batalha independe da influência da função objetivo de aprendizado utilizado.

## 5.5. SUGESTÕES PARA TRABALHOS FUTUROS

Para próximos trabalhos com o objetivo de aplicar a arquitetura sugerida em jogos de estratégia baseada em turnos, é necessário antes de estudar o efeito de cada parâmetro sobre o resultado da batalha, ou sobre as impressões que o jogador terá durante o jogo, determinar os valores ótimos para os parâmetros de cada uma das funções utilizadas para aumentar a influência da estratégia e do aprendizado nas ações tomadas durante a batalha e consequentemente sobre o resultado da mesma.

Uma das maiores dificuldades durante este trabalho foi à administração de todos os elementos do jogo em relação à inteligência artificial. E com isso vários aspectos do jogo tiveram de ser deixados de lado, assim como funcionalidades da própria arquitetura proposta para o módulo inteligente. Por isso sugere-se que jogos mais simples sejam analisados considerando todos os aspectos possíveis do jogo do que jogos mais complexos analisados mais superficialmente, como é o caso deste trabalho.

Sugere-se, portanto, para novos trabalhos neste mesmo tema, que os valores dos parâmetros sejam estudados prioritariamente para aperfeiçoar sua influência no resultado de cada seção de jogo, e só então seja estudada a influência de cada sub-módulo e seus parâmetros dentro desta arquitetura.

## 6. RESULTADOS

Os resultados das simulações permitiram analisar o efeito de parâmetros como as ações disponíveis para as unidades no jogo, os valores iniciais da estratégia do módulo inteligente, assim como a função de aprendizado utilizada. Apesar do grande número de simulações realizadas, dados pouco claros foram obtidos e apenas conclusões para casos específicos de relações entre as unidades, a estratégia e o modo de aprendizado utilizado puderam ser tomadas.

## 7. CONCLUSÃO

Este trabalho apresentou um panorama geral sobre a Inteligência Artificial, seus principais conceitos, seus desafios atuais e sua aplicação em jogos comerciais. Este panorama, junto com a lista de problemas de IA enfrentados pelos jogadores, forneceu material necessário para a elaboração de uma arquitetura para o módulo inteligente de jogos de estratégia baseada em turnos.

A arquitetura proposta foi implementada no jogo FFT Remake, um jogo criado para simular *Final Fantasy Tactics* que foi usado como base para provar conceitos apresentados nas simulações. O efeito do módulo inteligente sobre a jogabilidade e a impressão de inteligência sentida pelos jogadores é evidente, e foi mostrado através de exemplos de situações que não são vistas em jogos comerciais comuns.

## 7. BIBLIOGRAFIA

ALLIS, V. (1994). **Searching for Solutions in Games and Artificial Intelligence.**

BOUZY, B. (2001). **Computer Go: an AI Oriented Survey.**

BROOKS, R., BREZEAL, C., BREAZEAL, C., IRIE, R., KEMP, C., MAJANOVIC, M., et al. (1998). **Alternative Essences of Intelligence.** 15th National Conference on AI, (pp. 961-967).

BUCHANAN, B. G. (2007). **A (Very) Brief History of Artificial Intelligence.** AI Magazine.

CHESSBASE. (2007). **Once Again the Computer Defeats a Chess Master.** Extraído de : [http://www.nytimes.com/2006/12/05/crosswords/chess/05cnd-chess.html?\\_r=1: http://www.chessbase.com/newsdetail.asp?newsid=2476](http://www.nytimes.com/2006/12/05/crosswords/chess/05cnd-chess.html?_r=1: http://www.chessbase.com/newsdetail.asp?newsid=2476).> em 16/11/2010

DEMARIA, R. (2003). **High Score: The Illustrated History of Electronic Games.** McGraw-Hill Osborne Media.

DALMON, Danilo L.(2006) **Final Fantasy Tactics Remake.** Extraído de <http://www.orkut.com.br/Main#Community?cmm=10865623>> em 07/04/2011

EROL, K., NAU, D. S., & SUBRAHMANIAN, V. S. (1995). **Complexity, decidability and undecidability results for domain-independent planning\*.** Artificial Intelligence , pp. 75-88.

GELLY, S., & WANG, Y. (2006). **Exploration and Exploitation in Go: UCT for Monte-Carlo Go.**

HINRICHS, T. R., & FORBUS, K. D. (2007). **Analogical Learning in a Turn-Based Strategy Game.** Game Developers Conference.

INTELLIGENTSYSTEMS. (1990). **Fire Emblem.** Extraído de: [http://en.wikipedia.org/wiki/Fire\\_Emblem:\\_Ankoku\\_Ry%C5%AB\\_to\\_Hikari\\_no\\_Ken40](http://en.wikipedia.org/wiki/Fire_Emblem:_Ankoku_Ry%C5%AB_to_Hikari_no_Ken40)> em 16/11/2010

ISLA, D. (2007). **Handling Complexity in the Halo 2 AI.** Game Developers Conference. San Francisco.

IWATA, S., & KASAI, T. (1994). **The Othello game on an  $n \times n$  board is PSPACE-complete.** Theory on Computer Science, pp. 329-340.

KIRBY, N., & WOODCOCK, S. (2004). **GDC AI Roundtables Report.** Game Developers Conference. San Francisco.

LAIRD, J. E., & van LENT, M. (2005). **Human-level AI's Killer Application: Interactive Computer Games.** Game Developers Conference. San Francisco.

LAIRD, J. (2005). **Using Computer Game to Develop Advanced AI**. Game Developers Conference.

LIONHEADSTUDIOS (2001). **Black and White**. Extraído de:  
 <[http://en.wikipedia.org/wiki/Black\\_and\\_White\\_\(game\)](http://en.wikipedia.org/wiki/Black_and_White_(game)): e  
<http://www.gameai.com/games.html#BLACKANDWHITE>.> em 16/11/2010

MAXIS. (2000). **The Sims**. <[http://en.wikipedia.org/wiki/The\\_sims](http://en.wikipedia.org/wiki/The_sims): Extraído de:  
<http://www.gameai.com/games.html#THESIMS>.> em 24/11/2010

MCCARTHY, J. (2007). **What is Artificial Intelligence**. Extraído de:  
 <<http://www.formal.stanford.edu/jmc/whatisai/whatisai.html>.> 24/11/2010

MEYER, S. (2001). **Civilization: Call to Power**. Extraído de:  
 <[http://en.wikipedia.org/wiki/Civilization:\\_Call\\_to\\_Power](http://en.wikipedia.org/wiki/Civilization:_Call_to_Power): e  
<http://www.gameai.com/games.html#CIV>.> 5/12/2010

NORDLANDER, T. E. (2001). **AI Surveying: Artificial Intelligence In Business**. MS Thesis, Montford University.

NOWAKOWSKI, R., & Albert, M. (2007). **Lessons in Play: In Introduction to Combinatorial Game Theory**.

ORKIM, J. (2006). **Three States and a Plan: The A.I. of F.E.A.R.** Game Developers Conference. San Francisco.

RABIN, S. (2005). **Introduction to Game Development**. Hingham, Massachusetts: Charles River Media.

RUIZY, A. S., URBAN, S. L., AVILA, H. M., AGUDOY, B. D., & CALERO, P. G. (2004). **Game AI for a Turn-based Strategy Game with Plan Adaptation and Ontology-based retrieval**. Game Developers Conference.

RUSSELL, S. J., & Norvig, P. (2003). **Artificial Intelligence: A Modern Approach**. Upper Saddle River: NJ: Prentice Hall.

RUSSEL, Stuart; NORVIG, Peter. **Inteligência Artificial**. 2. ed. Rio de Janeiro: Campus, 2003.

SALETAN, W. (2007). **Chess Bump: The triumphant teamwork of humans and computers**. Extraído de: <<http://www.slate.com/id/2166000/> e  
<http://researchweb.watson.ibm.com/deepblue/>.> em 8/12/2010

SCHAEFFER. (2007, Julho 19). **Checkers Is Solved**. Science, pp. 1518-1522.

SHANNON, C. E. (1950). **Programming a Computer for Playing Chess**.

SQUARESOFT. (1997). **Final Fantasy Tactics**. Extraído de:  
 <[http://en.wikipedia.org/wiki/Final\\_fantasy\\_tactics](http://en.wikipedia.org/wiki/Final_fantasy_tactics).> em 9/12/2010

STEEL JR., G. L., & Gabriel, R. P. (1990). **The Evolution of Lisp**.  
<<http://www.dreamsongs.com/NewFiles/Hopl2.pdf>.> em 15/12/2010

TESAURO, G. (1995). **Temporal Difference Learning and TD-Gammon**. *Communications of the ACM*, pp. Vol. 38, No. 3.

van LENT, M., LAIRD, J., BUCKMAN, J., HARTFORD, J., STEINKRAUS, K., TEDRAKE, R., et al. (2000) **Intelligent Agents in Computer Games**. Game Developers Conference.