



UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE CIÊNCIAS EXATAS E DA TERRA
DEPARTAMENTO DE INFORMÁTICA E MATEMÁTICA APLICADA
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO



Cerebrate: um modelo para personagens inteligentes de jogos de estratégia em tempo real

Caio Freitas de Oliveira

Natal-RN
Junho de 2014

Caio Freitas de Oliveira

Cerebrate: um modelo para personagens inteligentes de jogos de estratégia em tempo real

Monografia de Graduação apresentada ao Departamento de Informática e Matemática Aplicada do Centro de Ciências Exatas e da Terra da Universidade Federal do Rio Grande do Norte como requisito parcial para a obtenção do grau de bacharel em Ciência da Computação.

Orientador

Dr. Charles Andryê Galvão Madeira

Co-orientador

Dr. André Maurício Cunha Campos

UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE – UFRN
DEPARTAMENTO DE INFORMÁTICA E MATEMÁTICA APLICADA – DIMAP

Natal-RN

Junho de 2014

Monografia de Graduação sob o título *Cerebrate: um modelo para personagens inteligentes de jogos de estratégia em tempo real* apresentada por Caio Freitas de Oliveira e aceita pelo Departamento de Informática e Matemática Aplicada do Centro de Ciências Exatas e da Terra da Universidade Federal do Rio Grande do Norte, sendo aprovada por todos os membros da banca examinadora abaixo especificada:

Dr. Charles Andryê Galvão Madeira
Orientador
Instituto MetrÓpole Digital
Universidade Federal do Rio Grande do Norte

Dr. André Maurício Cunha Campos
Co-orientador
Departamento de Informática e Matemática Aplicada
Universidade Federal do Rio Grande do Norte

Dra. Anne Magaly de Paula Canuto
Departamento de Informática e Matemática Aplicada
Universidade Federal do Rio Grande do Norte

Dra. Elizabeth Ferreira Gouvêa Goldberg
Departamento de Informática e Matemática Aplicada
Universidade Federal do Rio Grande do Norte

Natal-RN, 09 de junho de 2014.

Agradecimentos

Agradeço a Deus e à Nossa Senhora que foram fontes de fé e paz, principalmente em momentos difíceis.

Agradeço à minha mãe, Joana D'Arc, pelo suporte emocional e pela paciência por esses longos anos. Por ter me ensinado e por ter me ouvido. Pelo carinho e pelo amor por todo esse tempo.

Agradeço à minha namorada, Cecília Barbosa, que teve o azar de ter sido escolhida por mim. Pela paciência nos aperreios, pelo apoio constante. Por estar comigo nos piores e melhores momentos. Por amar incondicionalmente.

Aos meus amigos Thiago Abreu e Felipe Oliveira, que seguraram minha barra. Pela força e pelas risadas. A Osmar, Oliver e Carlos, que me mostraram o que realmente significa a palavra amizade. A todos aqueles que eu esqueci e ajudaram a moldar meu caráter hoje.

A Mike Lange, pela prestatividade e pelas informações sobre o pensamento de um jogador profissional de StarCraft.

Ao meu orientador, Charles Madeira, pelas ideias, pela empolgação, e principalmente ajuda imensurável que me deu. Ao meu co-orientador, André Maurício, pela oportunidade que me deu de fazer um trabalho naquilo que gosto tanto.

I fear no enemy, for the Khala is my strength!

I fear not death, for our strength is eternal!

Fenix, *StarCraft*

Cerebrate: um modelo para personagens inteligentes de jogos de estratégia em tempo real

Autor: Caio Freitas de Oliveira

Orientador: Dr. Charles Andryê Galvão Madeira

Co-orientador: Dr. André Maurício Cunha Campos

RESUMO

Jogos de estratégia em tempo real (RTS) se tornaram foco de diversos trabalhos científicos ao longo do tempo devido à grande complexidade envolvida neles e, conseqüentemente, uma grande dificuldade em criar estratégias eficazes com o uso de técnicas de inteligência artificial. Dentre estes jogos, StarCraft tem se destacado como um ótimo laboratório de pesquisa. Este trabalho apresenta um modelo cujo objetivo é o gerenciamento de recursos nestas aplicações. Como jogos RTS são uma abstração de um ambiente de guerra, o modelo proposto se baseia em um modelo organizacional para determinar as áreas de atuação dos personagens no jogo. Este modelo foi implementado usando StarCraft como estudo de caso e experimentado com alguns cenários existentes deste jogo. Os resultados obtidos foram comparados com resultados dos métodos utilizados pelos jogadores profissionais. Estes resultados demonstram que as estratégias usadas pelo modelo proposto são coerentes com as técnicas usadas pelos jogadores profissionais.

Palavras-chave: Jogos de Estratégia em Tempo Real, Personagens Inteligentes, Gerenciamento de Recursos.

Cerebrate: a model for non-playable characters of real-time strategy games

Author: Caio Freitas de Oliveira

Advisor: Dr. Charles Andryê Galvão Madeira

Co-advisor: Dr. André Maurício Cunha Campos

ABSTRACT

Real-time strategy games (RTS) have become target of many scientific works over time due to their complexity and, consequently, to the great difficulty in creating effective strategies using artificial intelligence techniques. Among these games, StarCraft has excelled as a great reasearch laboratory. This work presents a model to manage resources in these applications. Once RTS games are an abstraction of the war environments, the proposed model is based on a organizational model to determine the areas of operation of each character in the game. This model was applied in StarCraft and experimented with some existing scenarios of this game. The obtained results were compaired to results from methods used by professional players. These results show that the strategies used by the proposed model are consistent with the techniques used by professional players.

Keywords: Real-time Strategy Games, Non-Playable Characters, Resource Management.

Lista de figuras

1	Captura de tela de Dune II.	p. 15
2	Recursos de StarCraft.	p. 25
3	Indicação de base principal e natural, e indicação do significado dos principais termos no mapa Destination.	p. 29
4	Névoa de guerra em StarCraft.	p. 30
5	Estado da expansão na abertura Protoss <i>Forge Fast Expand</i>	p. 31
6	Fluxo de comunicação dos agentes.	p. 33
7	Máquina de estados dos mineradores	p. 36
8	Máquina de estados dos construtores	p. 39
9	Exemplo de muralha em Age of Empires	p. 41
10	Muralhas de cada raça de StarCraft	p. 42
11	Brechas dentro das estruturas de cada raça de StarCraft	p. 43
12	Comparação de mapa com o seu grafo correspondente	p. 47
13	Diagrama de Classes das estruturas do Ministério de Minas e do Trabalho.	p. 54
14	Diagrama de Classes das estruturas do Ministério da Economia.	p. 55
15	Diagrama de Classes das estruturas da Agência de Inteligência.	p. 56
16	Diagrama de Classes das estruturas do Ministério de Infraestrutura.	p. 58
17	Diagrama de Classes das estruturas do Ministério da Indústria.	p. 60
18	Estrutura do Cerebrate.	p. 61
19	Gráfico das funções de pertinência das distâncias	p. 62
20	Gráfico das funções de pertinência da passividade	p. 63
21	Gráficos das funções de pertinência que auxiliam a classificação das bases.	p. 66

22	Muralhas Zerg em vários mapas.	p. 69
23	Mapas usados nos experimentos.	p. 72
24	Comparação entre técnicas de mineração no mapa Destination.	p. 75
25	Ordem de expansão visto em partida profissional no mapa Destination.	p. 76
26	Comparação de ordens de expansão no mapa Fighting Spirit.	p. 77
27	Elementos desconsiderados na criação da função de avaliação de bases.	p. 77
28	Comparação de muralhas no mapa Destination.	p. 78
29	Comparação de muralhas nas posições iniciais de Fighting Spirit.	p. 79
30	Comparação dos minérios coletados em cada abertura no mapa Destination.	p. 80
31	Comparação dos minérios coletados em cada abertura no mapa Fighting Spirit.	p. 81

Lista de tabelas

1	Comparação das complexidades de jogos abordados através de IA e um RTS.	p. 19
2	Agentes do sistema e suas responsabilidades	p. 34
3	Papéis das unidades	p. 36
4	Relação entre agentes e unidades	p. 50
5	Ordens de construção usadas no confronto Zerg versus Terran.	p. 74
6	Brechas dos modelos de muralha propostos pelo Cerebrate e pelo site TeamLiquid.	p. 78

Lista de abreviaturas e siglas

RTS – Real-Time Strategy Game

API – Application Programming Interface

AIIDE – AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment

CIG – IEEE Conference on Computational Intelligence and Games

SSCAI – Student StarCraft AI Tournament

OSL – OnGameNet Starleague

Sumário

1	Introdução	p. 14
1.1	Jogos de Estratégia em Tempo Real	p. 14
1.1.1	Fases do Jogo	p. 16
1.2	RTS em Inteligência Artificial	p. 17
1.2.1	Desafios em RTS	p. 19
1.2.2	Trabalhos Existentes	p. 21
1.2.3	Competições de Bots	p. 22
1.3	Objetivos	p. 23
1.4	Organização do trabalho	p. 24
2	StarCraft	p. 25
2.1	Competição em StarCraft	p. 27
2.1.1	Mapas Competitivos	p. 28
2.2	Processo de decisão nas partidas de StarCraft	p. 30
3	Modelo organizacional de <i>game AI</i> para RTS	p. 33
3.1	Ministério de Minas e do Trabalho	p. 35
3.1.1	Atuação dos mineradores	p. 36
3.1.2	Remanejamento dos trabalhadores	p. 37
3.2	Ministério da Economia	p. 38
3.3	Ministério de Infraestrutura	p. 39
3.3.1	Atuação dos construtores	p. 39

3.3.2	Posicionamento de estruturas	p. 40
3.4	Ministério da Indústria	p. 44
3.5	Ministério da Defesa	p. 44
3.5.1	Controle de unidades	p. 44
3.5.2	Composição de unidades	p. 45
3.6	Ministério de Tecnologia	p. 46
3.7	Agência de Inteligência Militar	p. 46
3.7.1	Análise do mapa	p. 46
3.7.2	Verificação da movimentação e predição de ataques	p. 47
3.7.3	Verificação do desenvolvimento estratégico do inimigo	p. 48
3.8	Comunicação dos agentes	p. 48
3.8.1	Comunicações de Dependência	p. 48
3.8.2	Comunicações de Negociação	p. 50
4	Estudo de Caso em StarCraft	p. 52
4.1	A Raça Zerg	p. 52
4.2	Simplificações do modelo	p. 53
4.3	Estrutura do modelo	p. 53
4.3.1	Ministério de Minas e do Trabalho	p. 54
4.3.2	Ministério da Economia	p. 55
4.3.3	Agência de Inteligência	p. 56
4.3.4	Ministério da Infraestrutura	p. 58
4.3.5	Ministério da Indústria	p. 60
4.3.6	Cerebrate	p. 61
4.4	Escolha de aberturas	p. 61
4.5	Expansão econômica	p. 64
4.6	Ministério de Minas e Trabalho	p. 67

4.7	Ministério de Infraestrutura	p.67
5	Experimentos	p. 71
5.1	Metodologia	p. 71
5.2	Coleta de Minérios	p. 74
5.3	Ordem de Expansão	p. 76
5.4	Posicionamento de Estruturas	p. 78
5.5	Aberturas	p. 80
6	Considerações finais	p.83
	Referências	p. 86
	Apêndice A – Papéis das Unidades em StarCraft	p. 88

1 Introdução

Este trabalho descreve um modelo de *game AI*¹ para o jogo StarCraft: Brood War. Este modelo procura produzir unidades e expandir sua economia de maneira reativa às atitudes do seu oponente. Como StarCraft se trata de um jogo de estratégia em tempo real, uma introdução ao gênero e ao jogo se faz necessária.

1.1 Jogos de Estratégia em Tempo Real

O gênero de jogos de estratégia em tempo real, ou RTS (*real-time strategy*), como o conhecemos hoje, começou em 1992 com o lançamento do jogo Dune II². O jogo feito para DOS, Amiga e Mega Drive foi o primeiro que permitia ao jogador criar unidades e estruturas e enfrentar um inimigo nas mesmas condições. Nele é possível identificar os elementos que estarão presentes em todos os jogos de estratégia a partir de então:

- Coleta de recursos,
- Construção de estruturas,
- Produção de unidades,
- Diferentes facções,
- Mini-mapa no canto da tela (Figura 1),
- Controle feito pelo cursor do *mouse*.

O jogador começa o jogo com uma pequena quantidade de recursos, uma base e alguns soldados. Para poder enfrentar seu inimigo, ele deverá adquirir mais recursos através de unidades coletoras. Com estes recursos, ele será capaz de produzir unidades e estruturas

¹Inteligência artificial voltada para jogos.

²Publicado pela Virgin Interactive, desenvolvido pela Westwood Studios.



Figura 1: Captura de tela de Dune II.

que auxiliem em seus confrontos. Após construir as estruturas obrigatórias que permitem a coleta de recursos, o jogador se depara com a quantidade de opções que ele pode executar:

- Buscar no terreno outras fontes de recurso e/ou a localização do oponente,
- Armazenar os recursos para construção de estruturas ou unidades mais caras,
- Produzir mais coletores,
- Criar unidades militares.

Além disso, existe a possibilidade de controlar as unidades enquanto elas estão combatendo, para aumentar a eficiência delas no combate. Por se tratar de um jogo em tempo real, essas decisões acabam por se tornar ainda mais complexas, pois o jogador deve constantemente: (a) refletir sobre quais aspectos ele deve focar, e (b) ser capaz de reagir ao que acontece na partida.

Ao construir mais estruturas e desbloquear novas unidades, o jogo avança de forma progressiva, expondo o jogador a desafios cada vez maiores conforme o tempo passa. Isto se deve à existência de um número maior de estruturas de construção de unidades, mais unidades para controlar, mais unidades inimigas para combater e uma área maior para defender. Esse sentimento de progressão dentro de uma partida passou a ser compartilhado por todos os RTS.

1.1.1 Fases do Jogo

As etapas conceituais de uma partida de qualquer RTS são nomeadas a partir das etapas das partidas de xadrez:

1. Abertura
2. Meio-jogo
3. Final

Assim como no xadrez, a abertura é uma série de movimentos (no caso dos RTS, uma ordem de construção ou *build order*) com a qual chega-se a uma posição desejada dentro do jogo. Há vários tipos de aberturas, comumente classificados como:

- *Cheese*: Abertura não-convencional desenvolvida para pegar o oponente de surpresa. Geralmente, *cheese* é difícil de vencer se não for observado com antecedência, mas fácil de defender caso contrário.
- *Rush*: Abertura que corta alguns custos a fim de produzir algum tipo de agressão no início do jogo. Normalmente a economia do jogador fica atrasada com relação ao esperado de outra abertura considerada mais tradicional.
- *Greedy*: Aberturas gananciosas são similares aos *rushes*, mas com um foco econômico. Essas aberturas podem ser punidas com um ataque na fase inicial do jogo, pois o jogador terá cortado custos para desenvolver a economia, normalmente com a construção precoce de uma base. Caso não seja punida, essa abertura dará ao jogador uma economia maior, o que possibilitará um meio-jogo mais forte.
- *Standard*: Este tipo de abertura foca inicialmente no desenvolvimento econômico, sem abrir mão da defesa. Essas aberturas tendem a ser mais passivas, para defender-se de possíveis *cheeses* e *rushes*.

A partir da abertura escolhida, e se o jogador não venceu ou perdeu o jogo devido a um *cheese* ou *rush*, pode ser possível fazer uma transição para o meio-jogo. Para isso, avança-se na árvore tecnológica de forma que o jogador consiga produzir uma composição sólida de unidades, e então parte-se para o ataque. Embora normalmente haja um foco único, isso não implica a utilização de mais de um tipo de ataque durante o decorrer do jogo. Os principais tipos de ataques são:

- Ataques à economia, através do transporte de unidades às linhas de minério ou através de unidades de assédio³;
- Ataques em múltiplas localidades, para tentar reduzir o potencial de defesa do inimigo ao mesmo tempo em que foca-se na destruição de bases ou de estruturas de produção;
- Ataques frontais, forçando o oponente a se preocupar mais em defender-se do que em aumentar seu lucro ou sua capacidade de produção.

O final do jogo é marcado pela escassez de recursos no mapa e pelo alto nível tecnológico dos jogadores, de forma que, geralmente, as batalhas envolvem uma disputa por recursos. Muito raramente as partidas chegam a esse estágio, terminando em uma fase tardia do meio-jogo, quando pelo menos um dos jogadores chega ao último nível tecnológico.

A qualquer momento do jogo, como em um jogo de apostas, o jogador pode se colocar numa posição de “tudo ou nada”, por ter ficado em uma posição de extrema desvantagem com relação ao outro jogador, seja na coleta de recursos, no desenvolvimento tecnológico ou na capacidade de produção. Assim, deve-se fazer algum ataque à economia ou produção do seu inimigo para conseguir ao menos deixá-lo em uma situação similar à sua, caso contrário não conseguirá vencer a partida.

1.2 RTS em Inteligência Artificial

A partir do trabalho de Buro (2003), cujo foco é estimular pesquisas de inteligência artificial em cima de RTS, muitos trabalhos foram feitos usando StarCraft como objeto de estudo. Como mostrado por Ontañón et al. (2013), jogos deste gênero são ambientes complexos, onde as técnicas de IA ainda têm dificuldades, pois possui um espaço de estados muito grande.

A complexidade de um ambiente pode ser classificada de acordo com várias propriedades (RUSSEL; NORVIG, 2010), e RTS se enquadra no pior caso na maior parte delas:

- **Parcialmente observável:** a névoa de guerra impede o jogador de ver o que o oponente está fazendo, e saber dados como sua taxa de coleta de recursos e a composição de seu exército, por exemplo, de antemão.

³Unidades rápidas e que podem causar grande dano econômico ao destruir muitos trabalhadores em pouco tempo.

- **Multiagente:** cada unidade do jogador possui um propósito e deve cooperar a fim de atingir objetivos globais. Da mesma forma, elas devem levar em conta as unidades inimigas que estão no mapa e tentar combatê-las de alguma forma.
- **Estocástico:** embora o resultado de qualquer ação de um agente seja determinístico, o ambiente é parcialmente observável, a quantidade de ações que o oponente pode tomar é enorme e essas ações podem ser feitas simultaneamente. Ele é complexo o bastante para ser considerado estocástico, ao invés de determinístico.
- **Sequencial:** todas as ações tomadas em um determinado momento da partida tem consequências, grandes ou pequenas, que se repercutem ao longo do jogo.
- **Dinâmico:** o ambiente do jogo está mudando constantemente com as ações do jogador. Enquanto o jogador pensa no que fazer, jazidas de minérios e gêiseres de gás vespeno estão se exaurindo, unidades são criadas e eliminadas, e unidades inimigas estão se movimentando.

Jogos clássicos, como Xadrez e Go, possuem uma grande complexidade com regras inteligentes e objetivos simples. Eles foram alvo de diversos trabalhos (BOUZY; CAZENAVE, 2001) devido à essa complexidade e simplicidade de regras. Conforme ONTAÑÓN et al. explicam, jogos de estratégia em tempo real possuem uma complexidade muito superior às complexidades destes dois jogos, que até hoje são explorados por diversas técnicas de inteligência artificial.

Outra maneira de medir a complexidade do jogo é olhando o fator de ramificação, b , e a profundidade do jogo, d , [...] , com uma complexidade total de b^d . Em Xadrez, $b \approx 35$ e $d \approx 80$. Em jogos mais complexos, como Go, $b \approx 30$ a 300, e $d \approx 150$ a 200. A fim de determinar o fator de ramificação em StarCraft quando uma IA o joga, nós devemos ter em mente que a IA pode dar ações simultaneamente a quantas unidades no jogo for desejado. Então, considerando que, em um jogo típico, um jogador controla entre 50 e 200 unidades, o fator de ramificação ficaria entre u^{50} e u^{200} , onde u é o número médio de ações que cada unidade pode executar. Estimar o valor de u não é fácil, pois o número de ações que uma unidade pode tomar é altamente dependente do contexto. [...] Agora, se tivermos em mente que ações possuem tempos de *cool-down*⁴, e assim nem todas as unidades podem executar todas as ações a todo quadro, podemos tomar uma estimativa conservadora de cerca de 10 possíveis ações por unidade por quadro do jogo. Isto resulta em uma estimativa conservadora para o fator de ramificação entre $b \in [10^{50}, 10^{200}]$, somente

⁴Tempo de *cool-down* é o tempo em que uma habilidade ou ataque fica indisponível entre um uso e outro. Esse conceito não se restringe aos RTS e pode ser observado em jogos como *World of Warcraft*.

considerando unidades (ignorando as ações que construções podem executar). Agora, para computar d , nós simplesmente consideramos o fato de que os jogos típicos duram em torno de 25 minutos, o que resulta em $d \approx 36000$ (25 minutos \times 60 segundos \times 24 quadros por segundo) (ONTAÑÓN et al., 2013, p. 294).

Esta comparação de complexidade (mais evidente na Tabela 1), mostra uma diferença grande entre a quantidade de estados dos jogos. Devido a esse fator, pesquisadores como BURO e ONTAÑÓN et al. acreditam que o investimento de pesquisas com foco em jogos de estratégia em tempo real possibilitam um amplo melhoramento ou criação de técnicas de inteligência artificial.

Jogo	Fator de ramificação	Profundidade do jogo
Xadrez	≈ 35	≈ 80
Go	≈ 30 a 300	≈ 150 a 200
StarCraft (25 minutos)	$[10^{50}, 10^{200}]$	≈ 36000

Tabela 1: Comparação das complexidades de jogos abordados através de IA e um RTS.

1.2.1 Desafios em RTS

No intuito de estimular pesquisas de inteligência artificial em RTS, Buro (2003) observou seis desafios encarados ao tentar criar uma *game AI* para um jogo desse gênero:

1. **Gerenciamento de recursos:** desafio que envolve a coleta e o gasto dos recursos, seja em unidades, tecnologias ou estruturas, em um ambiente mutável e em tempo real.
2. **Tomada de decisão sob incerteza:** devido à névoa de guerra (*fog of war*), não é possível saber a localização e intenções do(s) oponente(s), e o jogador deve tomar decisões sem esse conhecimento.
3. **Raciocínio temporal e espacial:** análises de terreno estática e dinâmica bem como entender a relação temporal das ações é imprescindível no decorrer do jogo.
4. **Colaboração:** em partidas cooperativas, duas *game AIs* aliadas devem saber interagir de modo a vencer a partida.
5. **Modelagem e aprendizado de oponentes:** ao enfrentar o mesmo oponente várias vezes, *game AIs* devem conseguir se adaptar ao seu estilo de jogo.

6. **Planejamento adversário em tempo real:** ao invés de planejar micro-ações, a *game AI* deve utilizar abstrações para manter o espaço de busca em tamanho gerenciável.

Após 10 anos de publicação do trabalho de Buro e vários artigos sobre StarCraft e seu gênero, Ontañón et al. (2013) expuseram os seis atuais desafios na criação de *game AI* para RTS:

1. **Planejamento:** o planejamento em RTS depende de múltiplos níveis de abstração. No nível mais elevado, o jogador precisa ter capacidades de planejamento a longo prazo, para desenvolver uma economia forte no jogo; no nível mais baixo, as unidades devem ser controladas levando em consideração o terreno e o oponente.
2. **Aprendizado:** os autores distinguiram três tipos de problemas de aprendizado em jogos de estratégia em tempo real:
 - *Aprendizado a priori:* o aprendizado que se faz antes da partida começar, analisando *replays* ou mapas para aprender as estratégias apropriadas.
 - *Aprendizado na partida:* implantação de técnicas de aprendizado *online*, como aprendizado por reforço e modelagem de oponentes, para adaptar-se em tempo real.
 - *Aprendizado entre partidas:* estudar o que aconteceu nas últimas partidas e usar esse conhecimento para aumentar as chances de vitória nas próximas partidas.
3. **Incerteza:** em RTS existem dois principais tipos de incerteza. O primeiro é pelo jogo ser parcialmente observável, forçando o jogador a patrulhar para saber o que o oponente está fazendo. O segundo é pelo jogo ser competitivo e o jogador não conseguir prever as ações do(s) oponente(s).
4. **Raciocínio temporal e espacial:** raciocínio espacial engloba cada aspecto da exploração do terreno. Está relacionado a posicionamento de estruturas, expansão de bases e controle de unidades. O posicionamento de estruturas pode ajudar o jogador a criar as construções de forma a proteger sua base ou evitar configurações que impeçam unidades de mover. Antes de criar novas bases, o jogador precisa avaliar as possibilidades e escolher o melhor posicionamento, levando em consideração as suas bases e as do oponente. O controle das unidades no campo de batalha deve evitar ficar em gargalos e posicionando as unidades de maneira favorável, como um terreno elevado.

5. **Exploração do conhecimento de domínio:** como a quantidade de conhecimento de domínio é significativamente grande, não se sabe como esse conhecimento pode ser explorado por *game AIs*. Os trabalhos se focaram em: (a) implementar as estratégias existentes diretamente nas *game AIs*, e (b) analisar grandes conjuntos de *replays* para aprender estratégias, tendências ou planos.
6. **Decomposição de tarefas:** por todas as razões acima, a maior parte dos trabalhos decompõe o problema em uma coleção de pequenos problemas, a serem resolvidos independentemente. A divisão mais comum é:
 - *Estratégia*: está relacionada com a área mais elevada do processo de tomada de decisão. Deve identificar estratégias eficientes contra um dado oponente.
 - *Tática*: implementa a estratégia escolhida de forma eficiente. Cuida de movimentação do exército, posicionamento de construções, timings de ataques e produções.
 - *Controle reativo*: implementação das táticas. Determina os alvos das unidades, sua movimentação em batalha e fuga.
 - *Análise de terreno*: consiste em identificar pontos importantes no mapa e saber as regiões navegáveis dentro dele. Pontos importantes incluem posições de minérios e gás, gargalos, terrenos elevados ou rebaixados.
 - *Coleta de informação*: devido à névoa de guerra, jogadores devem regularmente mandar patrulhas para localizar e espionar bases inimigas.

Porém como integrar todas essas áreas em uma única arquitetura ainda é um problema aberto.

1.2.2 Trabalhos Existentes

Durante os últimos anos, vários trabalhos científicos focaram nas diferentes áreas desse jogo.

- Na área que cuida da produção de unidades e da pesquisa de tecnologias, trabalhos focaram em abordagens baseadas em aprendizado de máquina (WEBER; MATEAS, 2009) (DERESZYNSKI et al., 2011) (SYNNAEVE; BESSIÈRE, 2011) (YOUNG; HAWES, 2012) (HSIEH; SUN, 2008).

- Na área de análise de terreno Perkins (2010) encontrou uma maneira de identificar as regiões navegáveis e separou-as através de gargalos, também identificados pela decomposição de Voronoi.
- Em posicionamento de estruturas, Čertický (2013) apresentou uma abordagem para isso usando *answer-set programming*.
- Outra área que foi foco de vários trabalhos é a área de controle de unidades militares. Dessa área, pode-se dividir os trabalhos focados na área de decisão e na área de movimentação. Trabalhos na área de decisão usaram *case-based reasoning* (CADENA; GARRIDO, 2011) (SYNNAEVE; BESSIÈRE, 2012) para o planejamento tático.
- Trabalhos sobre movimentação utilizaram *potential fields* (SAJJAD; ISLAM, 2011) (HAGELBÄCK, 2012) e aprendizado por reforço (WENDER; WATSON, 2012).

Mais referências de artigos que utilizam esse jogo como objeto de estudo podem ser vistos no trabalho de Ontañón et al. (2013).

1.2.3 Competições de Bots

Com a criação de uma API que permitiu fazer interface entre o jogo StarCraft e sistemas desenvolvidos para controlar automaticamente os personagens deste jogo, pesquisadores em *game AI* criaram um *framework* de competições entre diferentes sistemas que substituem o jogador humano. As competições começaram em 2010 nas conferências científicas AIIDE e CIG, organizadas respectivamente pela AAAI (*Association for Advancement of Artificial Intelligence*) e pela IEEE (*Institute of Electrical and Electronics Engineers*). Outras duas competições surgiram nos anos seguintes. A SSCAI⁵ surgiu em 2011 com a ideia de fazer uma competição entre *game AIs* feitas por estudantes, e entre as feitas pelos estudantes contra o estado da arte. Em 2012, foi criada a liga de robôs de StarCraft, *StarCraft BroodWar Bots Ladder*⁶ que visa ser um ambiente de competição ininterrupta e contém 17 *bots* classificados de acordo com o sistema de *ranking* Elo. Detalhes sobre as competições da AIIDE, CIG e sobre a liga podem ser vistos em (ONTAÑÓN et al., 2013).

⁵<http://sscaitournament.com>

⁶<http://bots-stats.krasi0.com/>

1.3 Objetivos

Apesar do crescente número de *game AIs* que apareceram nas competições acadêmicas e a possibilidade de vê-las funcionando em tempo real, pela SSCAI, ainda percebe-se um negligenciamento do estudo do funcionamento das mentalidades dos jogadores profissionais. Com o intuito de estudar este problema, esse trabalho propõe um modelo que trata de diferentes áreas de interesse a fim de criar uma *game AI* que consiga agir de acordo com os comportamentos vistos em campeonatos profissionais de StarCraft.

Ao analisar a demografia das *game AIs* que entraram nas competições AIIDE 2011, AIIDE 2012 e CIG 2011, foi constatado que havia poucas soluções voltadas ao tratamento da raça *Zerg*. Além disso, as estratégias adotadas por essas *game AIs* eram muito pouco variadas, e até mesmo mal geridas de acordo com o confronto. Juntando com o fato do autor deste trabalho ter uma grande experiência prática jogando com essa raça, a decisão de me aventurar com os *Zerg* foi natural. Devido à complexidade dos confrontos, que requer *mindsets*⁷ diferentes quando se joga contra cada raça, o projeto se focará nas partidas contra a raça Terran.

Este trabalho tem como objetivo principal comparar as aberturas mais usadas pelos jogadores profissionais no confronto *Zerg* versus Terran através do ponto de vista econômico e militar, observando a coleta de recursos e o tempo de criação das primeiras unidades militares.

O conjunto de aberturas possíveis da *game AI* foi escolhido a partir de uma entrevista com Mike Lange, um jogador *Zerg* de alto nível, que já ajudou na avaliação de *game AI* de StarCraft em campeonatos da AIIDE (ONTAÑÓN et al., 2013). Ele expôs as aberturas usadas por todas as estratégias que foram consolidadas em torneios profissionais. Dentre todas essas aberturas, apenas algumas foram selecionadas:

- *5 pool*: um *cheese* que consiste em criar várias unidades militares no início do jogo;
- *9 pool*: um *rush* menos agressivo que a *5 pool*, mas mais seguro e com o potencial de negar expansões territoriais do inimigo por um bom período de tempo;
- *Overpool*: uma abertura mais econômica que pode fazer o inimigo achar que se trata de uma *9 pool*;
- *12 pool*: uma abertura mais defensiva, boa para prevenir contra *rushes* da raça Terran;

⁷Conjuntos de suposições e atitudes.

- *12 hatch*: a abertura mais tradicional, que possibilita transições sólidas para o meio-jogo.

Pela entrevista com o jogador, algumas hipóteses foram criadas:

1. As aberturas possuem taxas de coleta diferentes. Elas podem ser ordenadas em forma crescente, de acordo com esta taxa, da seguinte maneira: *5 pool*, *9 pool*, *overpool*, *12 pool* e *12 hatch*.
2. As aberturas também possuem tempos de criação de *zerglings* diferentes. A ordem crescente das aberturas pelo tempo de criação destas unidades é descrita por: *5 pool*, *9 pool*, *overpool*, *12 pool* e *12 hatch*.

Para atingir este objetivo, a *game AI* deve ser capaz de realizar as atividades: (a) manter uma ordem de construção e executá-la, (b) coletar minérios com os trabalhadores, (c) posicionar e construir estruturas e (d) identificar locais de expansão. Com base nisso, os objetivos secundários desse trabalho são:

- Criar um sistema que maximize a quantidade de recursos explorados em um determinado período de tempo,
- Criar um sistema de posicionamento e construção de estruturas que dificulte ao acesso dos inimigos à base,
- Criar um sistema que defina uma sequência de expansão territorial adequada à estratégia adotada no jogo.

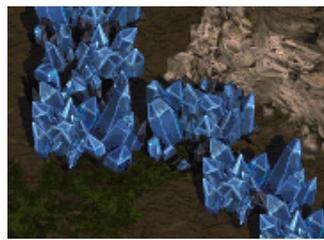
1.4 Organização do trabalho

O restante deste trabalho está dividido em:

- Capítulo 2: apresentação dos problemas específicos do jogo StarCraft.
- Capítulo 3: o modelo conceitual do sistema que irá controlar os personagens do jogo.
- Capítulo 4: a aplicação do modelo no jogo StarCraft, voltada à raça Zerg.
- Capítulo 5: os resultados dos experimentos feitos com a implementação.
- Capítulo 6: apresentação de conclusões e trabalhos futuros.

2 StarCraft

StarCraft é um RTS em que o jogador deve coletar dois tipos de recursos, *minérios* e *gás vespeno*, para poder enfrentar um grupo de inimigos. Esses recursos são normalmente distribuídos pelo mapa de forma tal que se possa criar uma base de extração, também chamada de expansão, próxima a todas as fontes de recursos¹, jazidas de minérios e gêiseres de gás vespeno (Figura 2). O conjunto de jazidas de minérios próximo a uma base é chamado de linha de minério.



(a) Jazidas de minérios



(b) Gêiser de gás vespeno

Figura 2: Recursos de StarCraft.

Desses recursos naturais, os minérios são os principais, usados para produção de todas as unidades, estruturas e pesquisas dentro do jogo. O gás vespeno, por sua vez, é usado para unidades especiais e pesquisas tecnológicas. Para aquisição desses recursos são necessários trabalhadores, que se encarregarão de coletar os recursos a partir de alguma fonte e entregá-los à base mais próxima.

Além de recursos naturais, o jogador precisa de recursos secundários. Dentre estes recursos secundários, os mais notórios são os *suprimentos*, sem os quais não é possível produzir unidades. Cada unidade consome uma quantidade pré-determinada de suprimentos, e para poder produzir mais unidades é necessária a criação de mais fontes de suprimento, até que se chegue na capacidade máxima determinada pelo jogo. Outro recurso secundário é o *tempo*, uma vez que toda produção e movimentação está restrita ao

¹Estruturas neutras que são o alvo da coleta de recursos.

tempo que vai ser levado para terminá-los.

A criação de unidades, incluindo trabalhadores, é feita em estruturas de produção. Essas estruturas podem ter pré-requisitos estruturais, como por exemplo a *factory*² que necessita de uma *barracks*³ para ser construída. Unidades também podem ter esses pré-requisitos, como o *dragoon*⁴ que precisa da *cybernetics core*⁵. As estruturas de produção, bem como as estruturas de pesquisa, possuem uma fila de produção com a qual pode-se antecipar a ordem de construção de alguma unidade ou pesquisa, isto é, sem precisar futuramente mandar a estrutura produzir no tempo exato. Isso ajuda o jogador a otimizar o tempo em algumas situações, como quando ele quer produzir muitas unidades.

Antes de cada partida, o jogador deve escolher uma entre três raças, cada uma com suas próprias árvores tecnológicas e peculiaridades. As raças disponíveis são:

- *Protoss*: é uma raça que possui unidades muito resistentes e com elevado poder de fogo, porém consomem muito de todos os recursos - minérios, gás, suprimento e tempo - para serem produzidas. Suas estruturas de produção são especializadas: *gateway* é encarregado de produzir as unidades terrestres básicas de ataque, a *robotics facility* produz unidades utilitárias, isto é, unidades que precisam de unidades básicas para que seu uso no exército tenha sentido, e o *stargate* produz unidades aéreas de ataque. Sua estrutura-base é o *nexus*.
- *Zerg*: voltada a produzir muitas unidades de uma só vez, esta raça possui as unidades mais frágeis e mais baratas do jogo. Ao contrário das outras raças, ela possui apenas uma estrutura de produção, que também é sua estrutura-base, a *hatchery*. A *hatchery* gera larvas, unidades especiais que servem apenas para produzir outras unidades, e não possui fila de produção, de forma que é possível produzir várias unidades ao mesmo tempo.
- *Terran*: é uma raça parecida com os Protoss, porém mais versátil. Um meio termo entre Protoss e Zerg, já que possui unidades baratas e frágeis, mas com sua produção limitada a estruturas de produção semelhantes às dos Protoss. Os Terran possuem dois tipos de estratégias consolidados, um baseado nas unidades feitas nas *barracks* e outro baseado nas unidades feitas na *factory*. As estratégias baseadas em *barracks*

²Estrutura de produção da raça Terran que produz unidades de ataque terrestres mecânicas.

³Estrutura de produção da raça Terran que produz soldados.

⁴Unidade de ataque terrestre da raça Protoss que tem a capacidade de atingir unidades aéreas e terrestres.

⁵Estrutura de pesquisa da raça Protoss que se encarrega das tecnologias de ataque e defesa das unidades aéreas.

se chamam *Bio Terran* ou somente *Bio*, e as baseadas em *factory* se chamam *Mech Terran* ou somente *Mech*.

2.1 Competição em StarCraft

StarCraft: Brood War é um dos jogos mais bem sucedidos da história tanto em popularidade, tendo vendido mais de 11 milhões de cópias até 2009 (GRAFT, 2009), como em tempo de competição profissional. Por mais de dez anos, este jogo foi jogado profissionalmente na Coreia do Sul em várias ligas, como a OSL (OnGameNet Starleague)⁶, bem como no evento internacional de esportes eletrônicos World Cyber Games, desde do ano 2000 na WCG Challenge⁷ até a WCG 2010⁸.

Durante esse período, as competições acompanharam a evolução das estratégias de cada uma das raças. Um exemplo dessa evolução é o conjunto das estratégias usadas pelos jogadores Zerg contra jogadores Terran nas competições. Durante um certo período as estratégias mais usadas pelos Terran dominaram as estratégias usadas pelos Zerg. Com a criação e popularização de uma abertura chamada *3 Hatch Muta* pelo jogador sAviOr⁹, que é a criação de 3 *hatcheries* e sua transição para a criação de *mutalisks*¹⁰ para atingir a economia do oponente, foi constatado um aumento na taxa de vitórias Zerg nas competições, levando essa raça a ter uma vantagem contra os Terran.

O estado atual das estratégias mais usadas é chamado de *metagame*, ou metajogo, e é influenciado tanto pelos mapas que estão sendo jogados nas competições bem como pelas estratégias usadas pelos potenciais adversários, como visto anteriormente. Um exemplo de mudança de metajogo devido aos mapas aconteceu com o mapa Destination¹¹, no confronto Protoss versus Zerg (PvZ). Por um tempo, os Protoss estiveram com uma porcentagem favorável de vitórias no mapa, porém os jogadores Zerg descobriram uma ordem de construção que melhor se adaptava ao mapa e conseguiram reverter a situação, vencendo 13 partidas consecutivas. Porém, após essas derrotas os Protoss conseguiram criar uma *build order* que “contragolpeava” essa nova *build* dos Zerg e venceram 11 partidas das 15 posteriores.

⁶[http://wiki.teamliquid.net/starcraft/OnGameNet_Starleague_\(OSL\)](http://wiki.teamliquid.net/starcraft/OnGameNet_Starleague_(OSL))

⁷<https://web.archive.org/web/20020709193219/http://www.worldcybergames.org/challenge/summary.asp>

⁸http://wiki.teamliquid.net/starcraft/WCG_2010

⁹<http://wiki.teamliquid.net/starcraft/Metagame>

¹⁰Unidade aérea de assédio *zerg* capaz de atingir alvos aéreos e terrestres.

¹¹<http://wiki.teamliquid.net/starcraft/Destination>

2.1.1 Mapas Competitivos

Nas competições de StarCraft, dois jogadores se enfrentam em um ou mais mapas criados previamente. Estes mapas são utilizados durante toda a competição, de forma que jogadores têm a possibilidade de treinar e criar estratégias específicas para os mapas. Estratégias específicas podem aparecer em alguns mapas porque eles podem possuir algumas características únicas, que acabam influenciando as decisões estratégicas do jogador no decorrer da partida.

Ao longo de mais de uma década de competição de StarCraft, a técnica de criação de mapas foi aprimorada de forma a deixar os mapas usados nos campeonatos equilibrados para todas as partidas. Devido às características de cada raça, os criadores de mapas tiveram que adotar determinados padrões para manter a taxa de vitórias de cada raça balanceadas entre si.

Cada mapa possui de duas a quatro localizações iniciais, onde ficam as *bases principais* do mapa, e cada base principal possui uma expansão secundária ou *natural*, que é a expansão mais próxima (Figura 3). Bases principais estão em regiões que possuem uma maior área e mais minérios que as regiões de outras expansões, além de ter um gargalo pequeno, longe da posição inicial, ligando-a à sua natural. Expansões naturais possuem uma área menor e um outro gargalo, maior e mais próximo à base, que a conecta ao restante do mapa.

Todas as expansões normalmente possuem um grupo de minérios que descrevem uma curva ao redor da base. Esse grupo é chamado comumente de *linha de minérios*. As linhas de minérios das bases normalmente estão muito próximas às fronteiras da região, de forma que o espaço que há serve apenas para construção de estruturas de defesa pequenas.

Em alguns mapas são encontradas expansões que não possuem conexão por terra com o resto do mapa, sendo possível conquistá-las apenas através de um transporte aéreo que leve um trabalhador para construir a base. Esse tipo de expansão, chamado de expansão-ilha ou somente ilha, não é levado em conta nesse trabalho, devido à complexidade envolvida na hora de criar um processo de decisão que inclua as ilhas.



Figura 3: Indicação de base principal e natural, e indicação do significado dos principais termos no mapa Destination.

2.2 Processo de decisão nas partidas de StarCraft

StarCraft é um jogo que fornece pouca informação ao jogador sobre o estado atual da partida. O jogador consegue ver apenas aquilo que está dentro do campo de visão de suas unidades. O que está fora desse campo de visão fica coberto pelo que é conhecido como *névoa de guerra*, em inglês *fog of war* ou FOW. Essa névoa é responsável por inserir a incerteza que existe em um ambiente de guerra. O terreno que está coberto pela FOW pode ter dois estados: desconhecido ou visto recentemente. Posições do mapa desconhecidas ficam completamente obscuras, sendo impossível para o jogador saber dentro da partida quais são as características do terreno. O terreno já visitado fica visível para o jogador, porém com uma tonalidade mais escura para diferenciá-lo do terreno dentro do seu campo de visão. Nesse terreno, também é possível ver a posição das estruturas inimigas na última vez que o jogador passou pela região (Figura 4).



Figura 4: Névoa de guerra em StarCraft. No canto esquerdo, em uma tonalidade mais clara, está uma área visível ao jogador. No canto superior direito está uma área recentemente visitada pelo jogador. É possível ver estruturas e a tonalidade é mais escura que a área visível. No canto inferior direito está uma área inexplorada. Não é possível adquirir nenhum tipo de informação desta área até que uma unidade a explore.

Ao iniciar uma partida, o jogador deve se fazer uma série de perguntas a fim de tomar as melhores decisões:

1. Qual a raça que estou enfrentando?
2. Em que mapa estou jogando?
3. Quais aberturas posso utilizar contra essa raça nesse mapa?

4. Quais aberturas o oponente pode usar?

5. Onde ele está?

Respondendo essas perguntas, o jogador poderá traçar um plano estratégico inicial e começar a tentar executá-lo. Primeiramente, deve coletar recursos com seus trabalhadores. Em StarCraft, não há como mandar seus trabalhadores minerarem diretamente da base antes de criá-los, como acontece em jogos como Age of Empires. Por isso, o jogador deve levá-los individualmente após sua criação até uma jazida de minérios para a coleta. Depois de coletar minérios e produzir trabalhadores, o jogador precisará construir estruturas. A posição de construção das estruturas é escolhida normalmente de acordo com “a posição mais próximo da base”, porém isso nem sempre é desejável, a depender da abertura. Por exemplo, a abertura Protoss de expansão rápida, *Forge Fast Expand*, só é possível com a criação de uma muralha de estruturas próxima à expansão (Figura 5), já que há uma grande chance do oponente tentar puni-lo por ter escolhido uma abertura *greedy*.



Figura 5: Estado da expansão na abertura Protoss *Forge Fast Expand*. As estruturas e unidades são organizadas de forma a dificultar a movimentação das unidades inimigas, que podem ser atingidas pelos dois canhões de fótons que estão próximos do pilar energético, a estrutura azul central. Esta organização serve para proteger tanto a base que está situada na natural como a base principal.

Enquanto constrói estruturas e cria trabalhadores, o jogador deve procurar o seu oponente pelo mapa, sendo importante, para isso, identificar as posições iniciais, para ter um direcionamento melhor na sua exploração. Após encontrar seu oponente, o jogador deve identificar sua abertura e tentar reagir apropriadamente. Por exemplo, caso o oponente tenha iniciado o jogo com um *cheese rush*, o jogador terá que se focar em defender-se. Ele pode fazer isso construindo estruturas de defesa e unidades militares para ser capaz de defletir o ataque inimigo. Cada abertura pode necessitar de uma reação diferente das outras.

No decorrer do jogo, conforme a partida vai evoluindo, perguntas mais complexas vão aparecendo, como:

1. Que abertura o oponente usou de fato?
2. Que transição faço, após ver a abertura de meu oponente?
3. Quando posso expandir?
4. Onde posso expandir?
5. Onde colocar estruturas de defesa?
6. Qual a composição de unidades que devo ter?
7. Quantas estruturas de produção devo ter no meio-jogo?
8. Onde o oponente está expandindo?
9. Qual a composição do seu exército?
10. Que tecnologias pesquisou?
11. Onde irá atacar e com que unidades?
12. Quais tecnologias devo pesquisar?
13. Quando devo atacar?
14. Onde devo atacar?

Além disso, há perguntas que não dependem do pensamento estratégico de mais alto nível, sendo mais relacionadas ao controle de exército:

1. Como devo atacar?
2. Qual deve ser o posicionamento das minhas unidades?
3. Que unidades inimigas devem ser focadas primeiro em um combate?
4. Quando devo recuar?

Respondendo essas perguntas de acordo com o metajogo vigente, os jogadores profissionais conseguem determinar as composições, reações, *timings* e transições mais eficientes. O intuito deste trabalho é olhar para o que já foi feito pelos jogadores profissionais como o resultado de um grande processo de aprendizado e tentar levar esses questionamentos para dentro de um modelo de *game AI*.

3 Modelo organizacional de *game AI* para RTS

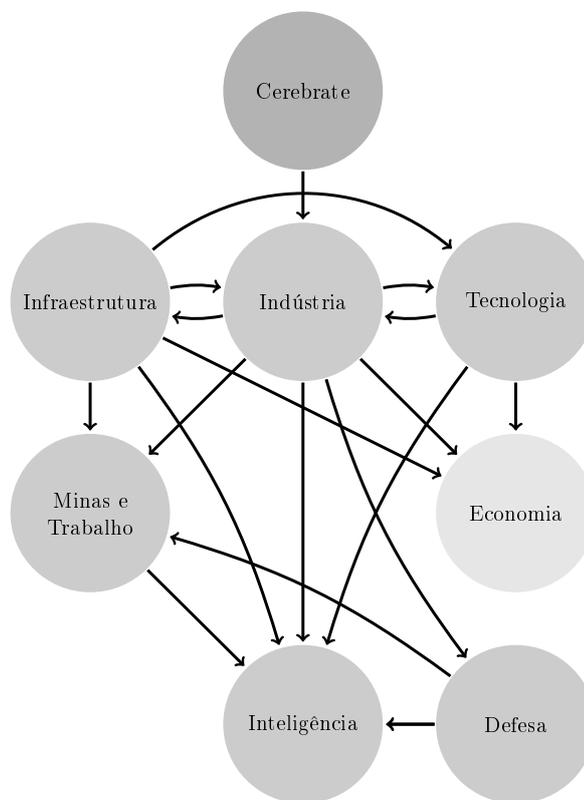


Figura 6: Fluxo de comunicação dos agentes. As setas apontam do requisitador ao seu alvo. Círculos escuros indicam os ministérios que controlam diretamente as unidades dentro do jogo. O círculo claro indica que o ministério da economia funciona apenas como sensor para os outros ministérios. O Cerebrate é o conjunto de todos os ministérios e funciona como o oráculo do ministério da indústria, dando a abertura a ser usada para ele.

Diversos trabalhos foram publicados com o foco em StarCraft e vários robôs foram criados para ele. Porém, poucos trabalhos apresentam um modelo completo de funcionamento de uma *game AI* voltada ao jogo. Um destes é o trabalho de Weber, Mateas e Jhala (2011a). Neste artigo, eles descrevem a arquitetura do EISBot, que fora baseado em um *framework* integrado de agentes para o jogo *Wargus*. Esta arquitetura é composta de cinco gerenciadores que dividem o *gameplay* em subproblemas. O gerenciador

de estratégias é responsável pela seleção de estratégias e pela determinação de tempos de ataque. O gerenciador de renda controla os trabalhadores, a coleta de recurso e a expansão econômica. O gerenciador de construções se responsabiliza pelos pedidos de construção de estruturas. O gerenciador tático executa as tarefas de combate e os comportamentos de microgerenciamento. E o gerenciador de reconhecimento implementa comportamentos de patrulha.

Foi observado que um agente fica sobrecarregado de tarefas distintas, sendo este o gerenciador de estratégias. Este agente deve modelar o oponente, inferir estratégias e definir os tempos de ataque da *game AI*. Para propor um novo modelo mais distribuído, observamos a divisão de tarefas de gerenciamento feita pelos governos atuais. Eles são compostos por ministérios, que se encarregam de gerenciar atividades distintas, porém interdependentes, de um país. Para este trabalho foram escolhidos seis ministérios e uma Agência de Inteligência, que apesar de não ser um ministério de fato, é uma entidade cuja área de trabalho é independente das áreas dos ministérios e que os serviços deles dependem da sua atuação. Os ministérios escolhidos estão brevemente descritos na Tabela 2 e suas interações estão expostas na Figura 6.

Departamento	Responsabilidade
Ministério de Minas e do Trabalho	Administração dos trabalhadores
Ministério da Economia	Monitoramento dos recursos
Ministério de Infraestrutura	Construção de estruturas
Ministério da Indústria	Administração da ordem de construção
Ministério da Defesa	Administração do exército
Ministério de Tecnologia	Pesquisa de novas tecnologias
Agência de Inteligência Militar	Coletar dados do jogador inimigo

Tabela 2: Agentes do sistema e suas responsabilidades

Como jogos de estratégia em tempo real possuem fases, este modelo também foi modelado para incorporar estas fases no desenrolar das partidas. O modelo proposto possui duas etapas de funcionamento: *ação* e *reação*. Elas são espelhadas nos conceitos existentes de abertura e meio-jogo. A ação é a fase em que uma abertura é escolhida e executada. A escolha de aberturas se dá por um produto da análise do mapa com um fator aleatório inicial, a passividade da inteligência artificial.

Na fase de ação, a *game AI* verifica as possibilidades referentes às aberturas que podem ser escolhidas no início do jogo. Essas aberturas são selecionadas previamente, a partir das estratégias mais conhecidas, e classificadas de acordo com o grau de agressividade e com a segurança que existe ao executá-las de acordo com um dos tipos de distâncias: (a) distância terrestre entre as principais, (b) distância aérea entre as principais.

Para classificar as estratégias desta maneira, cria-se duas funções de pertinência referentes a cada uma das aberturas, uma para a distância à qual está relacionada e outra para a agressividade. A seleção de quais aberturas poderão ser utilizadas pelo sistema e a criação de suas respectivas funções devem ser feitas previamente, no estudo de caso do modelo para o jogo escolhido. As funções relacionadas às aberturas devem ser modeladas com o auxílio de algum especialista no jogo. A escolha da abertura numa partida é feita baseando-se então nas combinações das funções de pertinência e, a partir delas, escolhe-se aleatoriamente uma entre as duas maiores.

Durante a fase de ação, o sistema apenas executará a abertura, que é uma ordem de construção. Assim, apenas as funcionalidades dos ministérios extritamente necessárias serão utilizadas durante essa fase.

Após a execução da abertura, a inteligência artificial passa para a outra fase da execução, a reação. Nesta fase, os ministérios ativam as funcionalidades que estavam dormentes durante a fase anterior. Os ministérios coletam dados referentes às suas responsabilidades e procuram agir com o foco de expandir-se economicamente e reagir aos movimentos do oponente. Essa expansão se dará com a construção de novas bases em locais estratégicos do mapa, além da produção de um exército e de estruturas de defesa que consigam proteger efetivamente todas as bases. Como todas as principais possuem uma natural, esta é a primeira escolha de expansão. A partir dela, o processo de decisão passa a ser mais complexo, sendo necessária a criação de uma função que consiga expressar as motivações de um jogador humano ao escolher tais expansões.

Para facilitar a criação de um método que reaja a um grupo de unidades inimigas, são designados papéis para as unidades (tabela 3) de forma que, quando a *game AI* detecta uma unidade que cumpra um determinado papel no exército inimigo, comece a produzir unidades que combatem efetivamente este papel. Os papéis das unidades são usados pelo ministério da defesa e pela agência de inteligência.

3.1 Ministério de Minas e do Trabalho

O ministério de minas tem como funções:

- Manter os trabalhadores minerando;
- Manter a saturação de trabalhadores das bases em níveis adequados.

Papel	Significado
<i>Large-scale</i>	Unidades que formam o núcleo do exército
<i>Anti-air</i>	Unidades que atacam unidades aéreas
<i>Harass</i>	Unidades rápidas e com poucos pontos de vida
<i>Area of Effect</i>	Unidades que causam dano em área
<i>Siege</i>	Unidades que possuem longo alcance
<i>Caster</i>	Unidades que possuem habilidades especiais ofensivas
<i>Stealth</i>	Unidades que podem ficar invisíveis
<i>Air</i>	Unidades aéreas
<i>Tank</i>	Unidades que possuem alta durabilidade
<i>Detector</i>	Unidades que expõem unidades invisíveis
<i>Transport</i>	Unidades de transporte
<i>No Ground Attack</i>	Unidades aéreas incapazes de atacar unidades terrestres

Tabela 3: Papéis das unidades

3.1.1 Atuação dos mineradores

Já que o principal foco do ministério é fazer os trabalhadores minerarem, o agente deve fazer uma relação dos recursos disponíveis nas bases ocupadas pelo jogador com os seus trabalhadores. Cada recurso deve possuir uma *flag* que ativa quando há um trabalhador minerando e desativa caso não haja. Esse atributo servirá para definir os estados dos outros trabalhadores.

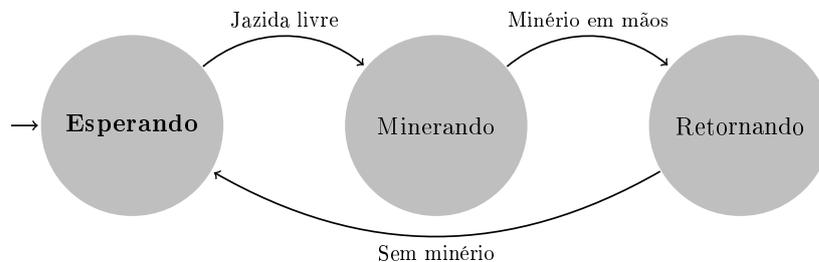


Figura 7: Máquina de estados dos mineradores

Além disso, cada campo de minérios deve ter no máximo um número S de trabalhadores, que revezam na execução da tarefa. Enquanto um deles retorna com o minério que coletou, o outro começa a minerar. Este trabalho é feito com uma máquina de três estados (Figura 7) para cada trabalhador:

- **Esperando:** estado inicial dos trabalhadores, no qual ele espera até que a jazida à qual ele está relacionado esteja livre.
- **Minerando:** estado no qual o trabalhador ativa a *flag* de ocupado do campo e começa a minerá-lo.

- **Retornando:** nesse estado, o trabalhador desativa a *flag* e retorna o minério para a estrutura de coleta de recursos.

3.1.2 Remanejamento dos trabalhadores

Em StarCraft, o jogador começa o jogo com uma base e quatro trabalhadores e, caso deseje partir para um jogo longo, deve progredir de forma a expandir sua economia com a construção de novas bases e mais trabalhadores. Cada uma das bases, por sua vez, possui uma quantidade de jazidas de minérios. Devido às características do jogo, essas jazidas podem ser mineradas de maneira ótima com 2 ou 3 trabalhadores por jazida.

Além disso, o jogador adversário possui um exército que pode estar se movimentando pelo mapa, podendo aproximar-se de alguma base. Esse movimento aumenta ou diminui o risco correspondente à mineração de uma base conforme o exército se aproxima ou se afasta dessa base, respectivamente.

Com a criação de uma nova base, a exaustão de uma jazida de minérios ou um risco elevado em alguma base já estabelecida, é necessário transferir trabalhadores entre as bases a fim de aumentar a renda de minérios. Porém essa transferência acarreta em uma perda temporária de lucro, sendo necessário, portanto, minimizar essa perda.

Esse problema foi modelado da seguinte forma: dados um ponto de saturação S ; um conjunto de bases $B = \{1, \dots, n\}$, onde cada base i possui uma quantidade de trabalhadores w_i , uma quantidade de jazidas de minérios m_i e um risco r_i ; e uma matriz de distâncias $D_{n,n}$ que contém as informações das distâncias entre as bases, a solução se dá em uma matriz $T_{n,n}$, tal que cada elemento t_{ij} informa quantos trabalhadores irão da base i à base j .

Essa solução deve obedecer a duas restrições: nenhuma base pode transferir mais trabalhadores que a quantidade inicial, isto é, a soma de uma linha qualquer T_i deve ser igual a w_i , e nenhuma base pode transferir trabalhadores para uma base que esteja trabalhando no seu limite de saturação, Sm_i . Isto significa que bases podem ficar com mais trabalhadores que podem comportar se não houver nenhuma opção para onde mandar o excesso. A partir disso, deve-se procurar minimizar a soma das distâncias percorridas por cada um dos trabalhadores, ao mesmo tempo que deve-se minimizar a soma dos riscos correspondentes à mineração em todas as bases após a transferência.

3.3 Ministério de Infraestrutura

As funções do ministério de infraestrutura são:

- Posicionar as futuras estruturas.
- Selecionar trabalhadores e mandar construir as estruturas necessárias.

Similarmente ao ministério de minas, o ministério de infraestrutura recebe o controle de trabalhadores que irão construir as estruturas solicitadas pela ordem de construção e atribui a eles uma máquina de estados finita, porém antes precisa posicionar a futura estrutura em um lugar propício. Após a construção, caso seja uma estrutura de criação de unidades, o ministério as guarda em um conjunto separado para cada tipo de estrutura.

3.3.1 Atuação dos construtores

O funcionamento dessa máquina de estados finita é semelhante à máquina dos mineiros. Porém, ao contrário destes, os construtores devem ter medo de unidades militares inimigas. Eles devem cancelar a construção quando virem que estão sob ataque, pois, na maioria dos RTS, cancelar a construção de uma estrutura retorna parte da quantia gasta.

A cada construtor são atribuídos uma estrutura e um local de construção. Essas informações são guardadas para que o construtor saiba aonde deve ir, e para que o ministério consiga saber quais estruturas foram canceladas ou destruídas e possa fazer um pedido de reconstrução ao ministério da indústria.

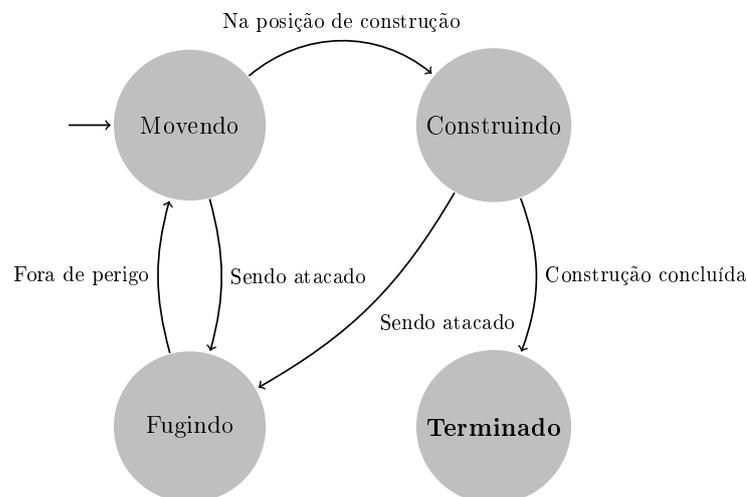


Figura 8: Máquina de estados dos construtores

Além disso, os construtores executam seu trabalho seguindo uma máquina de estados finita (Figura 8) com os seguintes estados:

- **Movendo:** estado inicial dos construtores, no qual ele move-se até o local de construção.
- **Construindo:** estado no qual o construtor está a construir sua estrutura.
- **Fugindo:** nesse estado, o construtor cancela qualquer ordem que tinha e procura fugir dos atacantes.
- **Terminado:** aqui, o construtor é dispensado do serviço, podendo voltar à coleta de recursos. O ministério de infraestrutura mantém, entretanto, a estrutura construída.

3.3.2 Posicionamento de estruturas

Os mapas da maior parte dos jogos de estratégia em tempo real são representados como grades, onde cada bloco dessa grade pode ser chamado de bloco de construção e estruturas possuem uma certa área que ocupam nessa grade. Além dessa grade, há também uma grade com maior precisão que modela o espaço de movimentação dos personagens no mapa. Essas duas representações podem ser observadas em jogos como Age of Empires e StarCraft.

Para avaliar o posicionamento de uma estrutura, será utilizado um novo método que utiliza o conceito de campos potenciais (*potential fields*), que foram usados com sucesso em vários jogos para resolver problemas de *pathfinding* (SAJJAD; ISLAM, 2011) (HAGELBÄCK, 2012). Essa técnica se baseia em atribuir *cargas* a alguns elementos do jogo de forma que esses elementos podem ser atrativos ou repulsivos. Ao colocar uma carga atrativa no destino de uma unidade e carga repulsiva nos obstáculos, por exemplo, é possível fazer um método simples que consegue chegar ao destino evitando os obstáculos. O uso dessa técnica para determinar o posicionamento de estruturas necessita de uma pequena mudança na mentalidade, pois o destino ainda não é conhecido.

Dependendo de cada jogo, os elementos que receberiam carga variam. Em jogos como Age of Empires e StarCraft, o posicionamento das estruturas vai depender se haverá ou não necessidade de criação de uma muralha de estruturas para impedir o movimento das unidades inimigas em algum gargalo. Em Age of Empires 2, é comum a criação de muralhas em gargalos criados por florestas, especialmente gargalos que funcionam como potenciais “portas dos fundos”. Em StarCraft, essa abordagem é usada também por cada

raça, mas em situações diferentes. Os Protoss fazem uma muralha em sua base a fim de impossibilitar a destruição de uma expansão rápida e a passagem de unidades inimigas pela sua natural sem que levem algum tipo de retaliação; os Terran o fazem para defender sua base principal de *rushes* Zerg, impedindo parcialmente ou completamente a passagem de unidades; e os Zerg o fazem com o mesmo intuito dos Protoss, apenas em um período mais tardio dentro do jogo.

Como visto nas figuras 9 e 10, a posição das estruturas em muralhas tende a se aproximar da região periférica da base, ou de algum gargalo, como é o caso das muralhas dos Terran. Isso implica em dizer que, para a criação de muralhas, as bordas são atrativas, então devem receber carga equivalente.

Porém, ao analisar o posicionamento de estruturas que não fazem parte de muralhas, é possível perceber que há uma repulsão por gargalos, já que elas devem oferecer espaço livre para as unidades navegarem. Algumas vezes, como o caso dos Terran em StarCraft, há uma repulsão até mesmo por outras estruturas, uma vez que o espaço entre elas pode impedir as unidades de navegar. Além disso, recursos naturais repelem estruturas, pois as estruturas não podem impedir os trabalhadores de coletá-lo. Essas características devem ser quantificadas e verificadas de acordo com o jogo em questão. Em Age of Empires 2, a distância entre um recurso e a estrutura de entrega de recurso (como uma *town center*) normalmente é mínima, sendo necessário um valor baixo para a carga de recursos, mas em StarCraft a distância mínima entre recurso e base (a única estrutura de entrega de recursos do jogo) é alta e pode comportar estruturas grandes dentro dessa brecha, o que implica um valor alto para a carga dos recursos.



Figura 9: Exemplo de muralha em Age of Empires



(a) Muralha Protoss



(b) Muralha Terran



(c) Muralha Zerg

Figura 10: Muralhas de cada raça de StarCraft

Algumas estruturas também podem fugir de todos esses conceitos, como estruturas de suprimentos. Elas, quando não estão em posições de muralha, estão em lugares distantes de gargalos e possíveis focos de ataque, já que representam também um outro recurso dentro do jogo. Isso implica que ao posicionar estruturas de suprimentos, deverão ser utilizados outros valores para cada um dos elementos que podem receber carga.

Como há duas maneiras distintas de posicionar estruturas, a opção escolhida procura preencher primeiramente as muralhas, e, somente em seguida, posiciona as estruturas de forma mais tradicional.

Além de tentar achar o posicionamento das estruturas da muralha, outra coisa que deve ser considerada é o espaçamento entre as estruturas. Em StarCraft, todas as estruturas possuem tamanho de construção e tamanho real, isto é, o tamanho que a estrutura ocupa na grade de construção, e outro tamanho na grade de movimentação. A partir desses tamanhos é possível calcular a brecha que existe em cada um dos lados de uma estrutura, como pode ser visto na Figura 11. Brechas negativas significam, entretanto, que a estrutura bloqueia a passagem de uma região além do seus limites.



(a) Construções Protoss



(b) Construções Terran



(c) Construções Zerg

Figura 11: Brechas dentro das estruturas de cada raça de StarCraft

Čertický (2013) propôs uma abordagem para a criação de muralhas usando *answer-set programming*, porém ela requer que sejam dadas de antemão as estruturas a serem usadas na muralha. Essa abordagem busca reduzir as brechas existentes entre as construções.

Além do posicionamento e do espaçamento entre estruturas, pode ser levado em conta o custo de cada estrutura.

3.4 Ministério da Indústria

As funções do ministério da indústria são:

- Executar a abertura escolhida.
- Criar ordens de construção no decorrer do jogo.
- Guardar todas as estruturas de criação de unidades.

O ministério da indústria é responsável pela manutenção da ordem de construção da IA. Ele inicialmente irá receber a ordem de construção inicial, isto é, a abertura, de um oráculo. Para a execução dessa ordem de construção, o ministério dá ordens aos ministérios de infraestrutura e de tecnologia de forma que eles criem as estruturas e pesquisem as tecnologias necessárias, bem como dá ordens às estruturas de produção, as quais fazem parte de suas responsabilidades.

Após o término da abertura, o ministério da indústria começa a pedir dos outros ministérios quais as necessidades que eles possuem no momento, e pondera sobre essas necessidades e as capacidades de produção da IA para determinar quais passos devem ser tomados a seguir. Ele deve ser capaz de identificar os requisitos de construção de unidades e tecnologias, para que não fique estagnado tanto tecnologicamente como militarmente.

Para a ordem de construção é usada uma fila de prioridades, de forma que o ministério tenha como avaliar e implantar mudanças nesta fila de acordo com as prioridades do estado atual do jogo.

3.5 Ministério da Defesa

As funções do ministério da defesa são:

- Controlar as unidades militares.
- Manter um controle das unidades que pode-se criar e definir uma composição ideal a partir disso.

3.5.1 Controle de unidades

O controle de unidades foi alvo de inúmeros trabalhos ao longo dos anos. Por exemplo, Uriarte e Ontañón (2012) mostraram como criar o comportamento de “atacar e correr”

usando mapas de influência. Cadena e Garrido (2011) e Synnaeve e Bessi ere (2012) usaram *case-based reasoning* para o planejamento tatico. Este agente, entao, deve utilizar alguma tecnica, ja pesquisada ou nao, para controlar de forma eficiente as unidades militares do jogador.

3.5.2 Composio de unidades

A composio das unidades deve ser planejada de acordo com o que e possivel produzir, levando em considerao a composio do exercito inimigo. Para atingir este feito, dois agentes vao cuidar de cada aspecto separadamente. A composio final vai ficar decidida pelo Ministerio da Industria, que fara uma negociao sobre quais unidades serao criadas.

O Ministerio da Defesa e responsavel por determinar quais unidades devem ser feitas, sob a otica do que e possivel fazer. A Agencia de Inteligencia e encarregada por verificar a composio que melhor enfrenta o exercito inimigo. Para esses agentes verificarem isso, eles devem observar os papeis das unidades e selecionar aquelas que melhor se encaixam na composio.

Por exemplo, um jogador Zerg ao enfrentar um oponente Terran pode recorrer a uma composio de fim de jogo com zerglings e *ultralisks*¹. O jogador Terran pode, entao, escolher uma das seguintes alternativas para enfrentar essa composio: (a) criar *siege tanks*² para causar uma grande quantidade de dano em area tanto nos zerglings, provavelmente matando varios de uma so vez, como nos ultralisks; (b) criar *vultures*³ que podem depositar *spider mines* no chao, e estas causam tambem dano em area; (c) criar *science vessels*⁴ que podem usar a habilidade *irradiate* e matar ultralisks, deixando o trabalho das unidades militares mais facil.

Neste exemplo, suponhamos que o jogador terran nao possui a capacidade de produzir *siege tanks* ou *science vessels*, e pode produzir tanto *vultures* como *marines*⁵. Entao, durante a fase de negociao, o Ministerio da Defesa iria propor a criao de *vultures* e *marines*, pois os primeiros possuem um bom ataque terrestre contra unidades leves e a habilidade de causar dano em area, enquanto os ultimos possuem maior dano por

¹Unidade zerg do mais alto nivel tecnologico. Unidade militar corpo-a-corpo, com alta durabilidade e dano.

²Unidade militar mecanica terran que possui dois modos de ataque, um com baixo dano contra uma unica unidade e outro com alto dano em area.

³Unidade militar mecanica terran de assedio. Possui, apos pesquisa tecnologica, habilidade de depositar *spider mines* que causam grande dano em area.

⁴Unidade terran conjuradora e voadora. Amplamente utilizada nos confrontos Terran vs Protoss e Terran vs Zerg.

⁵Primeira unidade militar biologica terran disponivel.

unidade de tempo, além de atacarem unidades aéreas. A Agência de Inteligência proporia tanto *vultures* como as unidades que o jogador é incapaz de construir, *siege tanks* e *science vessels*, pois elas são as que melhor combatem a composição inimiga. Isso levaria a inteligência artificial a produzir mais *vultures* do que *marines*, tendendo a ter uma composição boa contra as unidades inimigas e consistente entre si.

3.6 Ministério de Tecnologia

A função do ministério é tirar o jogador de uma estagnação tecnológica, verificando com a Agência de Inteligência quais tecnologias o oponente já pesquisou e guardando informações sobre as unidades que estão sendo usadas pelo jogador.

Tecnologias são classificadas de acordo com o papel que eles desbloqueiam nas unidades, seguindo o mesmo exemplo de unidades. A diferença sendo que tecnologias podem aprimorar a efetividade de algumas unidades, e isso requer um novo papel, além daqueles descritos para as unidades.

Além disso, o ministério terá que identificar pré-requisitos tecnológicos para poder guiar corretamente o andamento da *game AI*. Assim, o Ministério da Indústria poderá perceber que um determinado pré-requisito está sendo mais necessário do que o esperado e adiantar, se possível, sua posição na fila de produção.

3.7 Agência de Inteligência Militar

A agência de inteligência tem como funções:

- Analisar o mapa;
- Verificar as movimentações do exército inimigo e prever ataques iminentes;
- Verificar e guardar informações sobre o desenvolvimento estratégico do inimigo.

3.7.1 Análise do mapa

A análise do mapa é feita de maneira estática criando um multigrafo direcionado de todas as bases principais, onde elas são ligadas por arestas que representam as distâncias terrestres e aéreas (Figura 12). Também ligadas às bases estão suas naturais, a partir

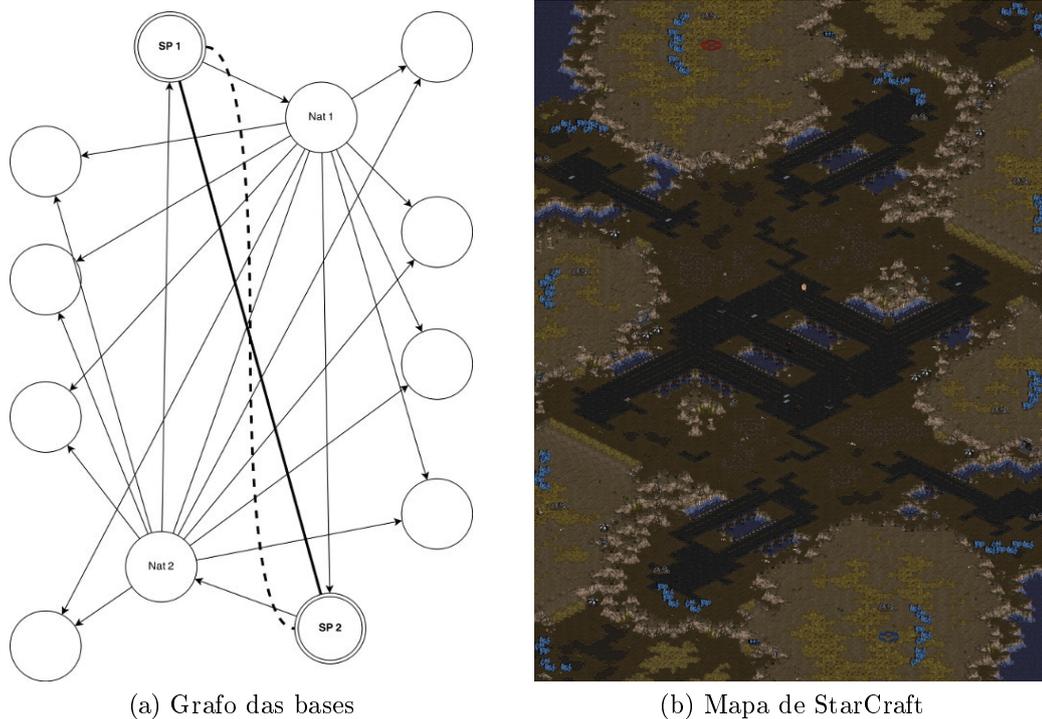


Figura 12: Comparação de mapa com o seu grafo correspondente

delas pode-se chegar a todas as outras bases. A ligação das outras bases às naturais é feita somente pela distância terrestre.

3.7.2 Verificação da movimentação e predição de ataques

A verificação da movimentação se dará através do uso de unidades terrestres de baixo custo e/ou de unidades aéreas que servirão de unidades de reconhecimento. Elas ficarão navegando pelo mapa à procura de pontos estratégicos para observar o exército inimigo, até que haja perigo iminente, quando o exército do jogador precisará se reagrupar.

Após a primeira vista da movimentação, o agente fará uma aproximação da direção do movimento das unidades avistadas com alguns pontos de controle. Estes pontos de controle serão:

- Expansões, ocupadas ou não;
- Gargalos.

Weber, Mateas e Jhala (2011b) propuseram um modelo para estimar o movimento das unidades inimigas em StarCraft. Como esse modelo foi feito para receber treinamento

vindo de *replays* de partidas, uma proposta mais simples é utilizar algum conhecimento especialista para determinar os parâmetros e eliminar o treinamento.

3.7.3 Verificação do desenvolvimento estratégico do inimigo

Além de unidades monitorando o exército inimigo, um grupo menor de unidades deve verificar as expansões não ocupadas do mapa periodicamente, a fim de determinar quando uma expansão foi tomada pelo inimigo. Espera-se que esse tipo de informação sirva para ativar algum tipo de resposta da inteligência artificial, seja expandir ou atacar o inimigo.

Para verificar outros aspectos do desenvolvimento estratégico, o ministério deve verificar quais tecnologias foram pesquisadas e quais unidades foram criadas pelo inimigo, analisando os combates. Espera-se que essa informação seja usada para modelar um plano para combater o inimigo mais eficientemente.

3.8 Comunicação dos agentes

Os agentes irão trocar informações em momentos diferentes do processo de atuação da *game AI*. Essas comunicações podem ser categorizadas em comunicações de dependência e comunicações de negociação.

As comunicações de dependência ocorrerão quando, para realizar alguma ação, um ministério precise de informações provenientes de outro. Esse tipo de comunicação é a mais comum, já que pode ocorrer a qualquer momento da execução da inteligência artificial.

Já as comunicações de negociação se darão quando a *game AI* estiver no modo de reação e o ministério da indústria precisar atualizar a ordem de construção. Uma vez que cada agente terá suas preferências quanto ao que se deve ser feito, a indústria recolherá o que cada um prefere e fará um leilão a fim de decidir qual a melhor opção para todos. Como entrada para esse leilão, os agentes passarão os resultados de funções de pertinência referentes às unidades que eles estão responsáveis por observar.

3.8.1 Comunicações de Dependência

Comunicações de dependência ocorrem quando agentes precisam de informações ou ações de outros agentes para atingir seu objetivo. Na Figura 6, os fluxos de comunicação estão demonstrados, do requisitador ao seu alvo. Abaixo, cada uma dessas requisições

é explicada e categorizada de acordo com a ação. Além destas, a negociação feita pelo Ministério da Indústria também é considerada uma comunicação de dependência, mas devido às suas características, ela será explicada mais à frente.

- Seleção de aberturas

- Um oráculo insere a ordem de construção da abertura escolhida no Ministério da Indústria. Este oráculo é representado no gráfico como Cerebrate, que é o conjunto de todos os ministérios.

- Execução da ordem de construção

1. Antes de executar a ordem de construção, o Ministério da Indústria envia o orçamento da produção ao Ministério de Economia.
2. Uma referência para o orçamento é repassada ao Ministério da Indústria, que a envia para o ministério responsável pela criação do item do topo da fila.
3. Quando o item começa a ser feito, o ministério responsável por ele manda o aviso que o orçamento foi cumprido para o Ministério de Economia.

- Construção de estruturas

1. O Ministério da Indústria ordena ao Ministério de Infraestrutura a construção da estrutura que está no topo da fila de produção.
2. O Ministério de Infraestrutura procura o melhor posicionamento para tal estrutura, e pede ao Ministério de Minas o trabalhador livre mais próximo da posição final escolhida.
3. Após o término da construção, o Ministério de Infraestrutura envia a estrutura para o ministério correspondente e o trabalhador usado de volta ao Ministério de Minas. Caso seja uma estrutura de produção, ela irá para o Ministério da Indústria. Caso seja uma estrutura de pesquisa, o Ministério de Tecnologia se torna responsável por ela.

- Criação de expansões

- O Ministério de Infraestrutura, quando for construir uma expansão, pede à Agência de Inteligência qual a melhor posição para a próxima expansão.

- Defesa de bases
 - O Ministério da Defesa pede ao Ministério de Minas trabalhadores, quando não há unidades suficientes para defender uma base.
- Atualização de informações importantes
 - O Ministério da Defesa constantemente verifica com a Agência de Inteligência quais são os pontos que precisam ser defendidos e quais precisam ser atacados.
 - O Ministério da Tecnologia verifica com a Agência de Inteligência quais tecnologias foram pesquisadas pelo oponente, para dar uma boa resposta à situação.
- Destruição de estruturas
 1. Quando uma estrutura é destruída, o ministério responsável informa ao Ministério da Indústria.
 2. Então, ele busca saber se existe a necessidade de reconstruir a estrutura através de uma negociação extraordinária.

3.8.2 Comunicações de Negociação

Agente	Unidades
Ministério de Minas	Trabalhadores, bases
Ministério da Indústria	Suprimentos
Ministério da Economia	Bases, estruturas de produção de unidades
Ministério da Infraestrutura	Trabalhadores, Estruturas de defesa
Ministério da Tecnologia	Estruturas de pesquisa, tecnologias
Ministério de Defesa	Trabalhadores, unidades de combate, estruturas de produção de unidades
Agência de Inteligência	Estruturas de defesa, unidades de combate, bases

Tabela 4: Relação entre agentes e unidades

Embora existam vários agentes que monitoram a necessidade de um mesmo tipo de unidade, eles observam sob diferentes perspectivas (tabela 4). Abaixo estão descritas as motivações dos agentes com relação a cada tipo de unidade.

- Trabalhadores:
 - Minas: Procura obter a saturação das bases.
 - Defesa: Procura repor trabalhadores que morreram em combate.

- Bases:
 - Minas: Procura expandir quando há mais trabalhadores que o necessário para operação das bases.
 - Economia: Procura manter as estruturas de produção sempre funcionando.
 - Inteligência: Procura expandir depois de vencer uma batalha recente, pois o inimigo terá que gastar tempo e recursos repondo tropas.
- Suprimentos:
 - Indústria: Procura não ficar com bloqueio de suprimentos.
- Estruturas de produção:
 - Economia: Procura aumentar o número de estruturas, caso a renda dê suporte a isso, não importando que tipo de estrutura seja.
 - Defesa: Procura criar estruturas que ajudem a construir um exército com uma determinada composição.
- Estruturas de defesa:
 - Infraestrutura: Procura deixar todas as bases com alta capacidade de defesa.
 - Inteligência: Procura aumentar o número de estruturas de defesa ao perceber um ataque se aproximando.
- Estruturas de pesquisa e tecnologias:
 - Tecnologia: Procura construir estruturas que possam melhorar as unidades que estão sendo produzidas.
- Unidades de combate:
 - Defesa: Procura fazer as melhores unidades, considerando o que é factível e a composição desejada.
 - Inteligência: Procura fazer as melhores unidades, considerando o que é factível e a composição inimiga.

4 Estudo de Caso em StarCraft

O objetivo do estudo de caso deste trabalho é testar o modelo Cerebrate, proposto no capítulo anterior, no ambiente do jogo StarCraft. Devido ao problema de StarCraft ser bastante complexo, neste trabalho faremos testes em um contexto reduzido deste jogo com o intuito de comparar as aberturas no confronto Zerg versus Terran. Como o trabalho se focará na raça Zerg, segue uma descrição mais detalhada desta raça.

4.1 A Raça Zerg

A força dos *Zerg* está nos números, e não em poucas unidades fortes. É a única raça que possui unidades que consomem menos que um suprimento: *zerglings*¹ e *scourges*² são produzidos aos pares e cada unidade consome meio suprimento. Suprimentos são aumentados com a produção de *overlords* e bases. A base zerg fornece apenas um de suprimento, ao contrário dos 10 suprimentos do *command center* dos Terran e dos 9 suprimentos do *nexus* dos Protoss. A fonte padrão de suprimentos, o overlord, além de aumentar a capacidade populacional também é a unidade *detectora*³ e o *transporte*⁴.

Suas estruturas de produção de unidades são também as estruturas de coleta de recurso. Elas produzem *larvas*, de onde é possível fazer qualquer unidade. Cada estrutura pode sustentar até 3 larvas simultaneamente.

Já que não possuem estruturas dedicadas somente à produção de unidades, os Zerg possuem apenas estruturas que desbloqueiam unidades. Além disso, dependem da *gosma* para construção dessas estruturas. Ela é espalhada ao redor de bases e de estruturas de defesa, *creep colony*, *spore colony* e *sunken colony*.

Sua capacidade tecnológica depende do tipo de estrutura que serve de base, começando

¹Primeira unidade de ataque *zerg*. Só é capaz de atacar unidades terrestres, por ataques corpo-a-corpo.

²Uma das primeiras unidades voadoras *zerg*. Só é capaz de atacar unidades voadoras, por ataques corpo-a-corpo.

³Unidade capaz de expor unidades invisíveis.

⁴Unidade voadora que pode carregar outras unidades dentro de si.

no primeiro nível tecnológico com a *hatchery*, e seguindo com *lair* e *hive*, no segundo e terceiro níveis, respectivamente. Isto significa que para desbloquear a possibilidade de construção de estruturas de segundo nível, é preciso evoluir a *hatchery* para uma *lair*.

4.2 Simplificações do modelo

Devido às características da raça Zerg e à proposta do trabalho, que é de avaliar as aberturas que existem no confronto Zerg versus Terran, alguns aspectos do modelo Cerebrate foram alterados ou retirados.

Aberturas são ordens de construção que contém apenas estruturas e unidades. Não há, dentro do escopo delas, a possibilidade de alterar a ordem de construção ou de fazer alguma pesquisa tecnológica. Devido a essas características, constatamos que não há a necessidade de controlar unidades militares ou de fazer pesquisas tecnológicas.

Portanto, as seguintes simplificações foram feitas:

- A segunda fase do modelo, a reação, não será tratada. Como apenas as aberturas serão testadas, a parte do modelo que cuida do meio-jogo não é necessária.
- O Ministério de Tecnologia não será utilizado, pois as aberturas não requerem a pesquisa de nenhuma tecnologia.
- O Ministério de Defesa não será utilizado, pois as unidades militares não deverão ser controladas.
- Como não haverá controle de unidades militares, a Agência de Inteligência não se encarregará de fazer tarefas de patrulhamento.
- O Ministério de Infraestrutura não liberará trabalhadores, pois eles se tornam a própria estrutura a ser construída, devido às características específicas da raça Zerg,.

As próximas seções descrevem detalhes da aplicação do modelo Cerebrate em Star-Craft.

4.3 Estrutura do modelo

Cada ministério do Cerebrate é estruturado hierarquicamente. Eles são representados como *namespaces* que definem estruturas de controle e atuação nas áreas específicas de

cada ministério. Em todos eles, há uma estrutura que controla instâncias das outras estruturas que pertencem ao ministério. Esta estrutura controladora é o agente responsável pelo ministério. Estes agentes possuem as funções:

- *update*: Atualiza as suas informações do controlador a cada quadro do jogo.
- *act*: Caso o agente possua unidades para controlar, esta função é usada para dar ordens a essas unidades.

A seguir estão descritas as estruturas que compõem os ministérios.

4.3.1 Ministério de Minas e do Trabalho

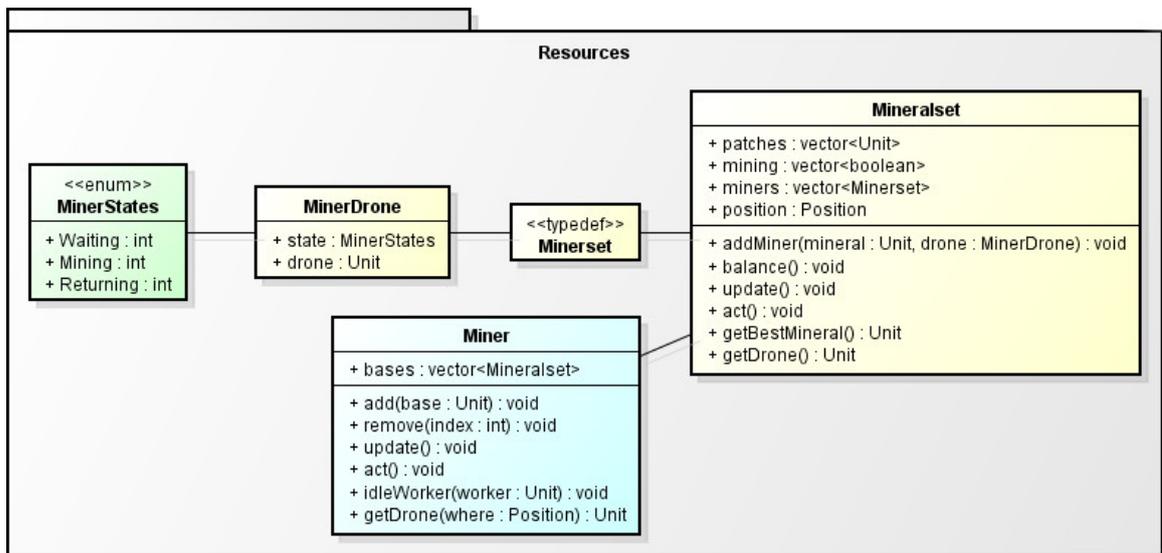


Figura 13: Diagrama de Classes das estruturas do Ministério de Minas e do Trabalho.

A Figura 13 mostra o diagrama das estruturas que compõem o ministério de minas. Ele possui uma estrutura para ligar trabalhadores a seus estados, a *MinerDrone*. Esta estrutura é agrupada em um *vector*, chamado *Minerset*. Desta forma, existe uma estrutura que comporte todos os trabalhadores ligados a uma jazida de minérios. Porém, para a mineração efetiva de uma base, são necessários outras informações e funcionalidades. A estrutura *Mineralset* comporta as informações sobre as jazidas de minérios (*patches*), o estado destas jazidas (*mining*), os trabalhadores ligados a elas (*miners*) e a posição da base (*position*). Além disso, possui as funções:

- *addMiner*: adiciona um novo trabalhador a uma das jazidas da base;

- *balance*: equilibra a quantidade de trabalhadores por jazida de minério, para uma mineração mais eficiente;
- *getBestMineral*: retorna a jazida de minério mais próxima da posição da base e com menos trabalhadores por minério;
- *getDrone*: retorna um trabalhador que não esteja minerando, caso exista.

A partir desta estrutura, podemos criar um conjunto de bases e fazê-las atuar como um conjunto. Para isso, foi feita a estrutura *Miner*, que além de reunir as bases em um *vector*, possui as funcionalidades:

- *add*: adiciona uma base recém criada;
- *remove*: remove uma base, a partir do índice que esta base possui no *vector* de bases;
- *idleWorker*: esta função é chamada ao identificar um trabalhador ocioso que ainda não faça parte do conjunto de mineradores do ministério;
- *getDrone*: procura a base mais próxima do ponto dado e retorna o trabalhador dado pela função *getDrone* do *Mineralset* que descreve esta base.

4.3.2 Ministério da Economia

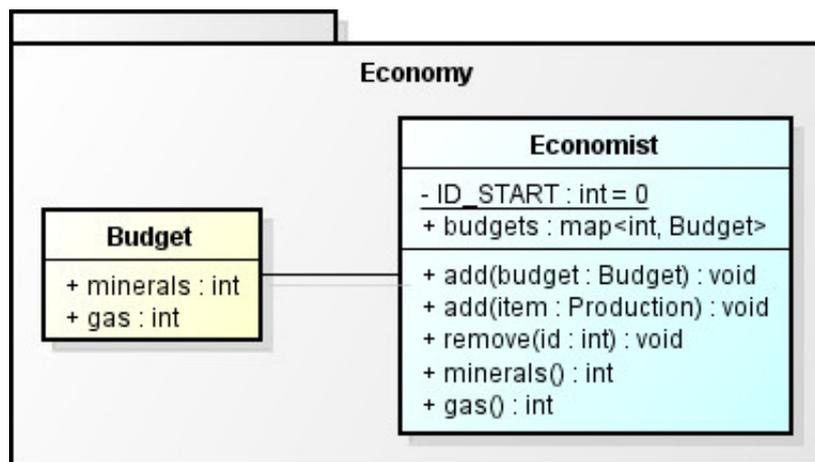


Figura 14: Diagrama de Classes das estruturas do Ministério da Economia.

O ministério da economia oferece um meio de gerenciar os itens cujo gasto não é imediato. Mostradas na Figura 14, as estruturas auxiliares são: *Budget*, que descreve o valor dos itens, e *Economist* que armazena os orçamentos atuais. Esta última contém

uma lista de orçamentos identificados por números únicos (*budgets*) e provê as seguintes funções:

- *add*: adiciona um orçamento ou um item à lista do ministério;
- *remove*: remove o orçamento que possui a identificação dada;
- *minerals*: retorna a quantidade de minérios que o jogador pode gastar, isto é, os minérios totais menos o valor em minérios de todos os orçamentos;
- *gas*: retorna a quantidade de gás vespeno que o jogador pode gastar, utilizando-se da mesma lógica.

4.3.3 Agência de Inteligência

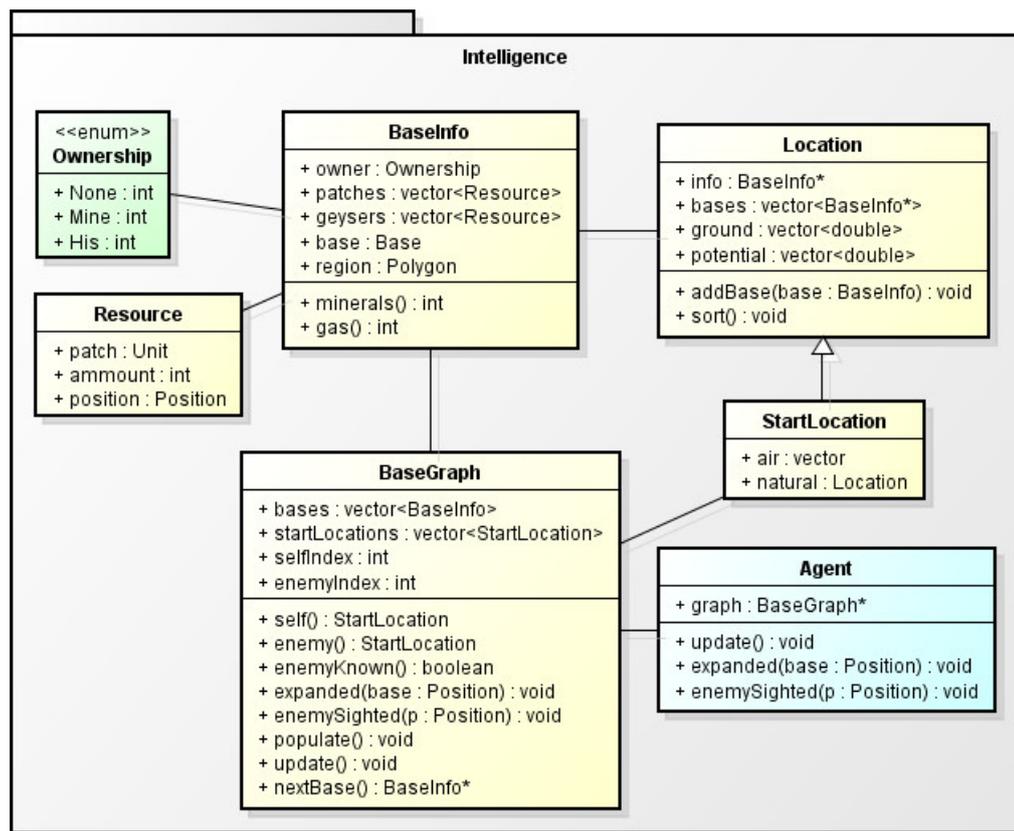


Figura 15: Diagrama de Classes das estruturas da Agência de Inteligência.

A agência de inteligência ficou responsável por manter e atualizar o grafo de bases do mapa da partida. As estruturas e suas relações podem ser vistas na Figura 15. Para descrever as bases, precisaremos de estruturas auxiliares que indiquem a posse da base (*Ownership*) e o último estado conhecido dos recursos (*Resource*). As bases (*BaseInfo*)

possuem estados de posse (*owner*), jazidas de minérios (*patches*), gêiseres de gás vespeno (*geysers*), um polígono que descreve a região andável (*region*) e as informações dadas pela biblioteca de análise de terreno (*base*). As funções *minerals* e *gas* retornam a quantidade total de recursos de uma base.

A partir destas informações sobre as bases, podemos criar um grafo. O grafo descreve as posições iniciais e as distâncias delas entre si, as naturais destas posições e as distâncias das naturais até todas as outras bases. As naturais deste grafo são descritos pela estrutura *Location*. Eles possuem a informação sobre a base (*info*), os nós adjacentes (*bases*), a distância até estes nós (*ground*) e o valor que cada nó adjacente possui (*potential*). Para identificar a ordem de expansão, esta estrutura possui a função *sort*, que ordena as bases pelo potencial e pela posse, deixando as bases com maior potencial e sem dono nas primeiras posições, e as bases com dono e menor potencial nas últimas posições.

Cada natural está ligada a uma posição inicial, que é descrita pela estrutura *StartLocation*. Além de conter informações sobre as distâncias terrestres entre as localizações iniciais do mapa, esta estrutura também possui as distâncias aéreas entre elas e a informação sobre a natural.

A estrutura *BaseGraph* contém todas as informações necessárias para o grafo. Ela guarda as informações de todas as bases do mapa (*bases*), informações dos nós do grafo (*startLocations*) e a referência das posições iniciais do jogador e do inimigo (*selfIndex* e *enemyIndex*, respectivamente). Além disso, esta estrutura possui as funções:

- *self*: retorna a posição inicial do jogador;
- *enemy*: retorna a posição inicial do inimigo;
- *enemyKnown*: retorna a posição inicial do inimigo é conhecida;
- *expanded*: altera a posse da base que fica na posição indicada, tornando-a do jogador;
- *enemySighted*: altera a posse da base que fica na posição indicada, tornando-a do inimigo;
- *populate*: chamada após a análise de terreno, esta função cria as informações da base e o grafo;
- *nextBase*: retorna a próxima base que o jogador deve tomar.

Como a agência de inteligência ficou responsável apenas pelo grafo de bases, a estrutura *Agent* serve como fachada para a *BaseGraph*. Ele possui um ponteiro para ela, pois o grafo é alocado dinamicamente.

4.3.4 Ministério da Infraestrutura

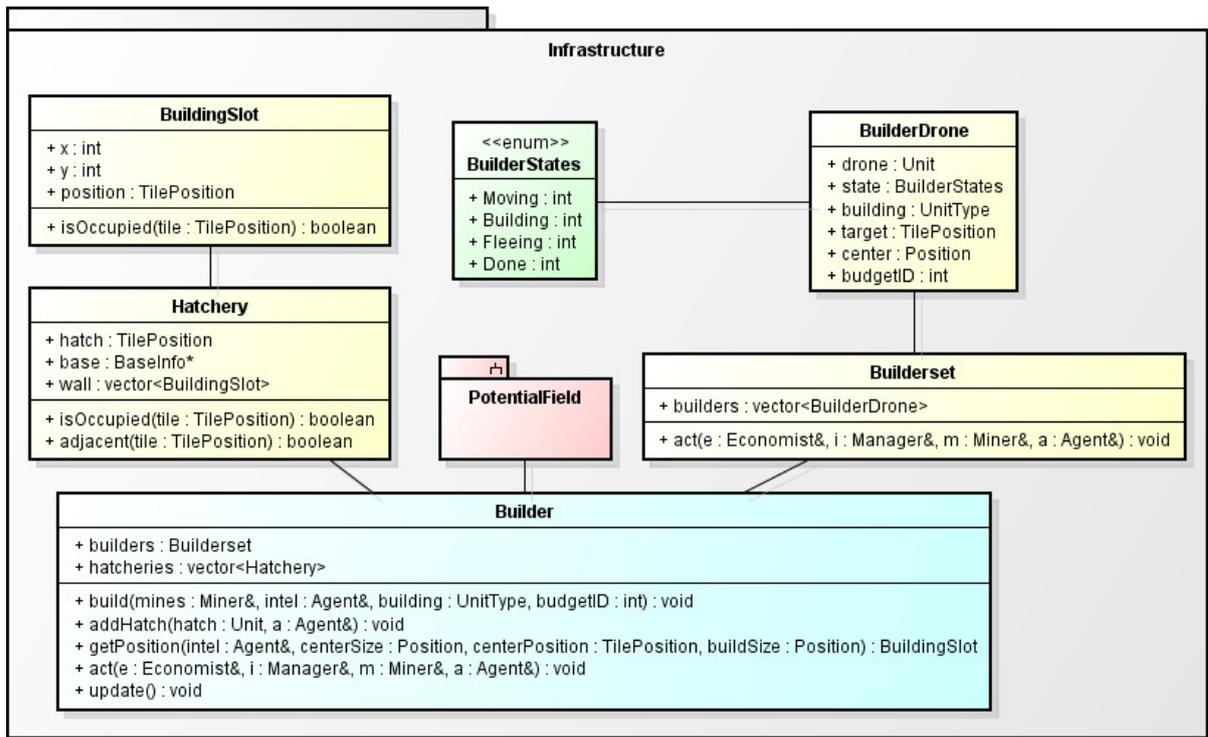


Figura 16: Diagrama de Classes das estruturas do Ministério de Infraestrutura.

O ministério de infraestrutura precisa de estruturas para controle de unidades, posicionamento prévio de estruturas e um conjunto de funções para o cálculo de *potential fields* (Figura 16). As estruturas de controle de construtores se assemelham às estruturas do ministério de minas. *BuilderStates* descreve quais estados comportamentais existem. *BuilderDrone* faz a ligação da unidade (*drone*) com seu estado atual (*state*), a construção que deve erguer (*building*), a localização da construção (*target*), a posição que a unidade deve atingir para que possa construir (*center*) e a identificação do orçamento no ministério da economia, para que se possa apagar o orçamento após a construção.

O conjunto de construtores é organizado na estrutura *Builderset*, que, além de possuir a lista destas unidades (*builders*), possui a função *act*. Esta função, como todas as outras com este nome, dá ordens às unidades que estão sob o controle da estrutura. Porém, além disso, ela faz uso de outros ministérios para atualização de algumas informações:

- *Economist*: retira o orçamento que foi alocado para uma construção que tenha sido começada;
- *Manager*: agente responsável pelo ministério da indústria, ele é chamado quando uma construção tenha sido cancelada;
- *Miner*: quando uma nova base é conquistada, sua localização é adicionada à lista de bases do ministério;
- *Agent*: quando uma base é conquistada, seu estado de posse é alterado para ser do jogador.

Para alocar muralhas, foi necessária a criação de uma estrutura que descreva uma “vaga” para as construções. Esta estrutura, *BuildingSlot*, possui o tamanho da construção em blocos de construção (x e y) e a posição da “vaga” (*position*). Ela também oferece a função *isOccupied*, que retorna um booleano indicando se um determinado bloco de construção está dentro da área de sua “vaga”.

Como as muralhas são feitas a partir de uma base, elas foram descritas a partir disto. As bases estão detalhadas na estrutura *Hatchery*. Ela possui a posição da base (*hatch*), as informações sobre ela (*base*) e uma lista de alocações de estruturas para a muralha (*wall*). Ela oferece as funções:

- *isOccupied*: informa se o bloco de construção está dentro das áreas da muralha ou da base;
- *adjacent*: informa se o bloco de construção é adjacente à base.

Finalmente, a estrutura que une todos estes conceitos é a *Builder*. Ela possui as informações dos construtores (*builders*) e das bases (*hatcheries*). Suas funções são:

- *build*: solicita um trabalhador do ministério de minas (*Miner*) e adiciona este trabalhador na lista de construtores;
- *addHatch*: adiciona uma base à lista de bases da estrutura, além de alocar uma muralha para ela;
- *getPosition*: retorna o alocamento de uma construção.

4.3.5 Ministério da Indústria

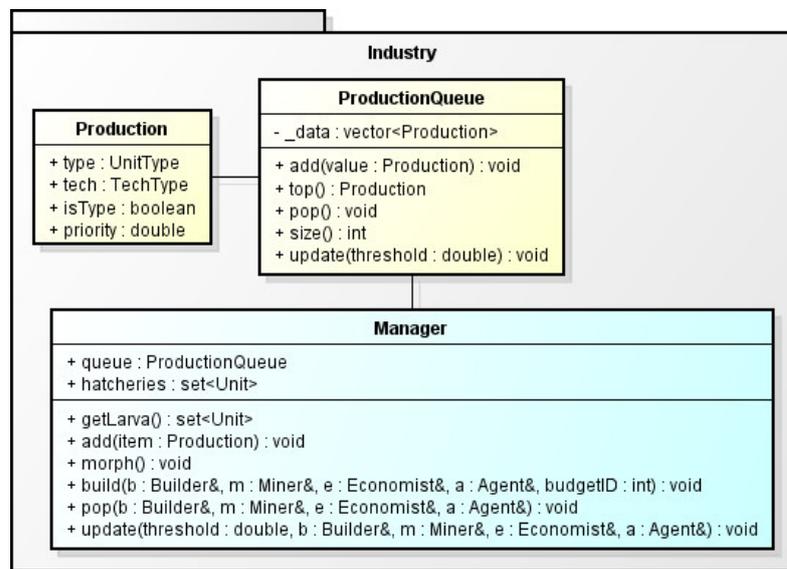


Figura 17: Diagrama de Classes das estruturas do Ministério da Indústria.

O ministério da indústria possui uma fila de prioridades (*ProductionQueue*) para os itens da ordem de construção (*Production*). Cada item pode ser ou uma unidade (*type*), o que inclui construções também, ou uma pesquisa (*tech*), além de possuir um valor que indica a sua prioridade (*priority*). As filas atualizam todo quadro para retirar os elementos que possuem um valor de prioridade menor que um determinado limiar (*threshold*).

A estrutura controladora da fila, a *Manager*, possui tanto a fila (*queue*) como as estruturas de produção (*hatcheries*). Para facilitar o acesso das larvas que existem em todas as *hatcheries*, a função *getLarva* junta todas as larvas em um único conjunto. Para criação das aberturas, a função *add* foi criada, recebendo o item a ser inserido. As outras funções existem para auxiliar a função *update*:

- *morph*: caso o primeiro item seja uma unidade, esta função seleciona uma larva e cria a unidade;
- *build*: caso o item seja uma construção, esta função chama a função *build* da *Miner*;
- *pop*: esta função que avalia os itens e chama as funções necessárias para o cumprimento da ação;

4.3.6 Cerebrate



Figura 18: Estrutura do Cerebrate.

O Cerebrate contém uma instância de cada agente dos ministérios. Eles são atualizados na ordem: (a) Agência de Inteligência, (b) Ministério de Infraestrutura, (c) Ministério de Minas e do Trabalho e (d) Ministério da Indústria. Além das funções comuns a todos os agentes, o Cerebrate também possui:

- *start*: o jogo começa no quadro 0, e neste quadro esta função especial de inicialização, responsável pela análise de terreno e a configuração inicial de todos os agentes, é chamada;
- *setOpening*: durante esta fase inicial, a escolha da abertura a ser utilizada é feita através desta função.

4.4 Escolha de aberturas

Para decidir qual abertura tomar, a *game AI* verificará as distâncias terrestres entre as principais. Caso haja mais de duas posições iniciais e não se saiba a posição do oponente a tempo de fazer uma escolha, ele assume que cada valor será a média das distâncias correspondentes entre todas as principais e naturais. Essas distâncias serão, então, passadas para funções de pertinência referentes às aberturas escolhidas para esse trabalho. As distâncias são dadas em uma unidade de distância D , que é a distância correspondente à largura de um pixel no mapa, e passadas para as funções em unidades de kD , 10^3D . As funções de pertinência para as distâncias, cujo gráfico está mostrado na Figura 19, são as seguintes:

$$5 \text{ pool}_d = \left(1 - \frac{\tanh(2(\text{terrestre} - 5)) + 1}{2}\right)^2 \quad (4.1)$$

$$9 \text{ pool}_d = e^{-\frac{(\text{terrestre} - 4.5)^2}{0.32}} \quad (4.2)$$

$$\text{Overpool}_d = e^{-\frac{(\text{terrestre} - 5)^2}{0.8}} \quad (4.3)$$

$$12 \text{ pool}_d = e^{-\frac{(\text{terrestre} - 5.5)^2}{0.64}} \quad (4.4)$$

$$12 \text{ hatch}_d = \frac{\tanh(1.5(\text{terrestre} - 5)) + 1}{2} \quad (4.5)$$

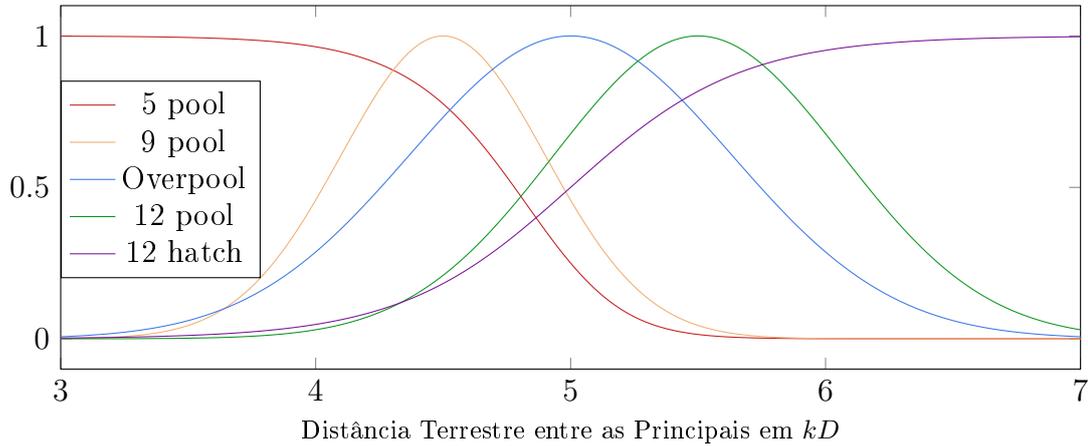


Figura 19: Gráfico das funções de pertinência das distâncias

Logo após, são calculadas as pertinências referentes à passividade do Cerebrate. As funções foram modeladas de forma que quanto maior a passividade, menor a pertinência nas aberturas agressivas (*rushes* e *cheeses*) e maior a pertinência nas aberturas mais tradicionais. O gráfico dessas funções está plotado na Figura 20.

$$5 \text{ pool}_p = \left(1 - \frac{\tanh(5(\text{passividade} - 0.5)) + 1}{2}\right)^2 \quad (4.6)$$

$$9 \text{ pool}_p = e^{-\frac{(\text{passividade} - 0.2)^2}{0.08}} \quad (4.7)$$

$$\text{Overpool}_p = e^{-\frac{(\text{passividade} - 0.3)^2}{0.12}} \quad (4.8)$$

$$12 \text{ pool}_p = e^{-\frac{(\text{passividade} - 0.43)^2}{0.06}} \quad (4.9)$$

$$12 \text{ hatch}_p = \frac{\tanh(2(\text{passividade} - 0.1)) + 1}{2} \quad (4.10)$$

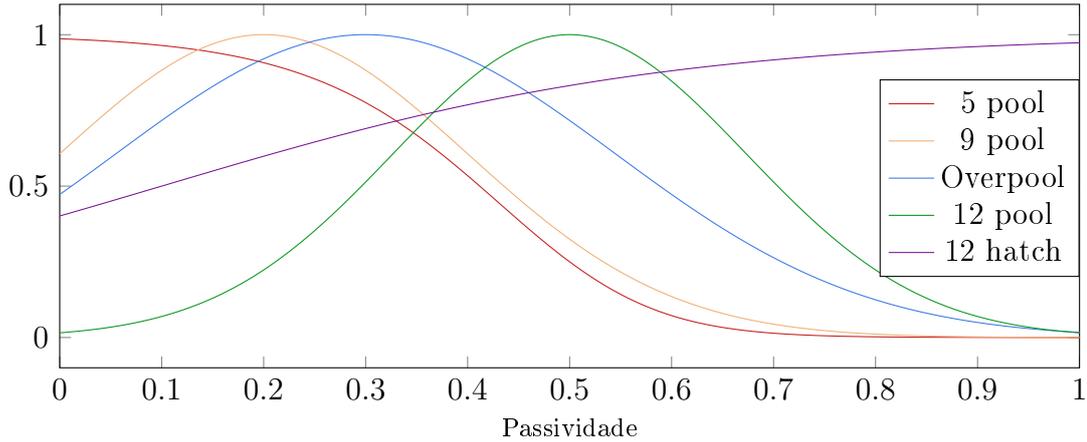


Figura 20: Gráfico das funções de pertinência da passividade

A partir das pertinências de distância e passividade, é possível calcular o valor da pertinência final das aberturas, que se dará ao utilizar o operador fuzzy *ou*⁵ sobre os dois valores. Isto é:

$$5 \text{ pool} = 5 \text{ pool}_d \vee 5 \text{ pool}_p \quad (4.11)$$

$$9 \text{ pool} = 9 \text{ pool}_d \vee 9 \text{ pool}_p \quad (4.12)$$

$$\text{Overpool} = \text{Overpool}_d \vee \text{Overpool}_p \quad (4.13)$$

$$12 \text{ pool} = 12 \text{ pool}_d \vee 12 \text{ pool}_p \quad (4.14)$$

$$12 \text{ hatch} = 12 \text{ hatch}_d \vee 12 \text{ hatch}_p \quad (4.15)$$

Após o cálculo final, a escolha é feita aleatoriamente entre as duas aberturas com maiores pertinências. Cada uma delas possui chance proporcional à sua pertinência em

⁵O operador fuzzy *ou* utilizado tem como significado uma operação de união: $\mu_{a \cup b} = \mu(a) + \mu(b) - \mu(a)\mu(b)$.

relação à soma das duas maiores, ou seja:

$$chance_a = \frac{pertinencia_a}{pertinencia_a + pertinencia_b} \quad (4.16)$$

$$chance_b = \frac{pertinencia_b}{pertinencia_a + pertinencia_b} \quad (4.17)$$

4.5 Expansão econômica

Para a criação do grafo de bases, primeiro são achadas as bases correspondentes às posições iniciais do mapa, isto é, todas as principais do mapa. Então, procura-se a base mais próxima a cada uma das posições iniciais. Estas bases são as naturais e são atribuídas a suas respectivas principais. Após isso, o grafo é completado com as distâncias terrestres das naturais para todas as outras bases, que não são suas principais. As bases atribuídas à natural do jogador são ordenadas de acordo com uma função que avalia o potencial de mineração de uma base e de acordo com o dono dessa base. Bases que possuem dono são as últimas, enquanto bases livres e com alto potencial são as primeiras da lista de bases.

A função de potencial das bases é calculada baseada nas seguintes propriedades:

- Quantidade de jazidas de minério
- Quantidade total de minérios disponíveis⁶
- Quantidade total de gás vespeno disponível⁷
- Distância até a base principal
- Distância até a base do oponente

Cada uma dessas propriedades é passada para uma função de pertinência específica para ela, e então passadas para uma função que calcula o potencial da expansão. As distâncias são passadas para a função *dist* em kD , a quantidade de minérios nas jazidas e a quantidade de vespeno nos gêiseres são passadas em unidades de 10^3 . O gráfico da função *dist* e das outras funções de pertinências podem ser vistos na Figura 21.

A função *dist* retorna um valor fuzzy referente a quão distante um ponto está do outro. Ela foi modelada de acordo com os mapas usados nas competições existentes de

⁶Como a priori não sabe-se quantos minérios existem em cada jazida, utiliza-se o valor padrão de 1500.

⁷Da mesma forma que os minérios, utiliza-se o valor padrão de 5000 para expressar o valor desconhecido.

game AI para StarCraft. As distâncias entre as principais de cada mapa foram verificadas e uma função sigmóide foi modelada baseando-se nas distâncias mais curtas e mais longas encontradas nos mapas. Esta função é usada nas funções referentes à distância para a base do jogador e à distância para a base do oponente.

A função que avalia a proximidade de bases para a principal do jogador retorna o valor 0 caso a base avaliada seja uma ilha. Caso contrário, retorna a negação⁸ da função *dist* aplicada à distância entre as duas bases. Desta forma, a função tende a retornar 1 à medida em que a distância entre as bases tende a 0.

A função que avalia a proximidade de bases para a principal do oponente segue uma lógica similar, porém com valores opostos. Caso o oponente não seja conhecido, as distâncias entre as bases não podem ser verificadas, então o retorno é 1. Isto é, não há efeitos negativos no potencial da base avaliada. Caso a base avaliada esteja em uma ilha, o retorno também é 1. Caso a base seja a natural ou a principal do oponente, o retorno é 0, pois elas são as bases mais próximas da base principal do oponente. No caso padrão, a função retorna o valor de *dist* aplicada à distância entre a base avaliada e a principal do oponente. Desta forma, quanto menor a distância da base avaliada para a base inimiga, menor o potencial dela.

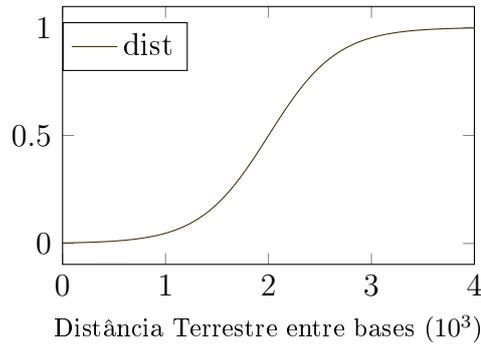
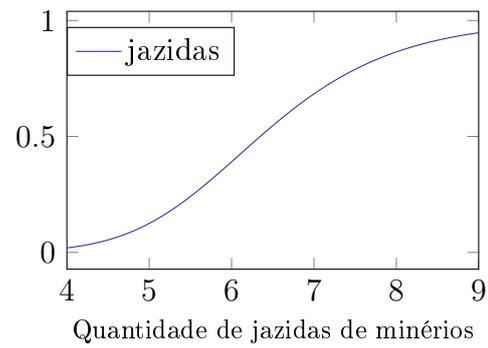
A função de pertinência relativa à quantidade de jazidas foi modelada de acordo com o que foi encontrado nos mapas das competições. Na maioria dos mapas, as bases principais possuem 9 jazidas de minérios, suas naturais possuem 8 e as outras bases possuem 7 jazidas. Já que, quanto maior o número de jazidas, mais trabalhadores podem trabalhar paralelamente para recoltamento de minérios, a função descreve uma sigmóide cujo valor tende a 1 conforme o número de jazidas aumenta.

A quantidade de minérios é avaliada de maneira similar, tendo em vista que a maior parte das bases principais possuem 13.500 minérios espalhados em 9 jazidas, cada uma com 1.500 minérios. Ela também descreve uma sigmóide que tende a 1 à medida que a quantidade de minérios numa base aumenta.

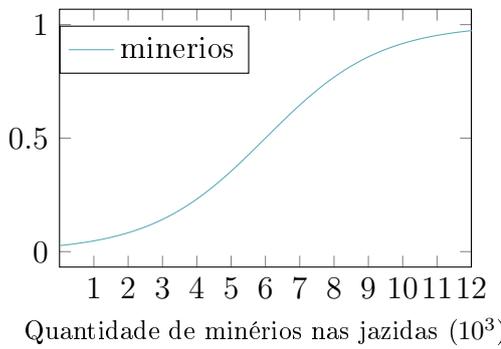
A quantidade de gás vespeno numa base é avaliada linearmente com um mínimo de 0,5, pois a sua influência na escolha de bases é baixa. Isto se deve pelo fato da coleta de gás vespeno ser muito alta em relação à coleta de minérios e à taxa com a qual o gás é usado durante uma partida.

As funções de pertinência são:

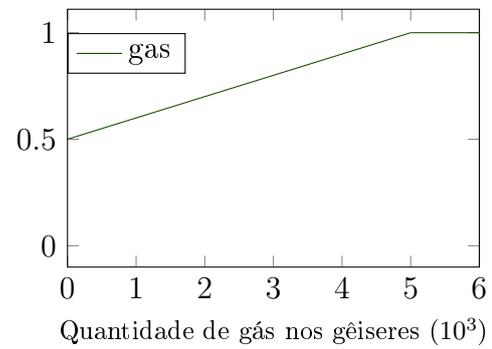
⁸A negação fuzzy se dá pela função: $\neg(x) = 1 - x$.

(a) Gráfico da função $dist$ 

(b) Gráfico da função de pertinência das jazidas



(c) Gráfico da função de pertinência dos minérios



(d) Gráfico da função de pertinência do gás

Figura 21: Gráficos das funções de pertinência que auxiliam a classificação das bases.

$$dist(x) = \frac{\tanh(1.5x - 3) + 1}{2}$$

$$dist_{principal}(b) = \begin{cases} 0 & , b.ilha() \\ 1 - dist(jogador.natural.distancia(b)) & , \text{ caso contrário} \end{cases}$$

$$dist_{oponente}(b) = \begin{cases} 1 & , b.ilha() \vee \\ & , \neg \text{oponente.conhecido()} \\ 0 & , b = \text{oponente.principal}() \vee \\ & , b = \text{oponente.natural}() \\ dist(\text{oponente.natural.distancia}(b)) & , \text{ caso contrário} \end{cases}$$

$$\text{jazidas}(b) = \left(\frac{\tanh\left(\frac{b.\text{jazidas}()-5}{2}\right) + 1}{2} \right)^3$$

$$\text{minerios}(b) = \frac{\tanh(0.3(b.\text{minerios}() - 6)) + 1}{2}$$

$$\text{gas}(b) = \min\left\{\frac{b.\text{gas}()}{10} + 0.5, 1\right\}$$

$$\text{potencial}(b) = (\text{dist}_{\text{principal}}(b) \times \text{dist}_{\text{inimigo}}(b))(\text{minerios}(b) \times \text{jazidas}(b) \times \text{gas}(b))$$

Como podem haver mapas com mais de duas posições iniciais, utiliza-se a média das distâncias entre a base em questão e as posições iniciais em que o inimigo pode estar.

4.6 Ministério de Minas e Trabalho

O Ministério de Minas e Trabalho foi organizado de forma que ele seja uma coleção de agentes, relacionados a cada trabalhador. Cada agente está agrupado de acordo com a jazida que está coletando. Estas jazidas, por sua vez, estão agrupadas de acordo com a base que está mais próxima a elas.

Dessa forma, existem duas formas de otimizar a mineração. Uma no que diz respeito à quantidade de trabalhadores em cada jazida de uma base, e a outra na quantidade de trabalhadores por base que o jogador possui.

No caso de teste, nos focaremos na otimização de trabalhadores por minérios, já que não passaremos da fase de abertura do jogo.

4.7 Ministério de Infraestrutura

O Ministério de Infraestrutura tem como um de seus objetivos escolher a posição onde as estruturas serão construídas. Para calcular o posicionamento, escolhemos o método *potential fields* que atribui cargas a alguns elementos do jogo, como obstáculos ou unidades, e calcula o valor potencial dos blocos a partir da influência destes elementos no bloco analisado.

Inicialmente, é necessário definir quais elementos receberão carga. Aqui, usaremos cargas positivas como atrativas e cargas negativas como repulsivas. Como visto no capítulo anterior, os pontos no perímetro da região devem receber carga atrativa, pois o objetivo é criar uma muralha de estruturas em uma determinada base. Esse perímetro é calculado segundo a técnica de Perkins (2010). A influência do perímetro sobre um bloco de construção é determinada pela distância do perímetro ao bloco. Como o perímetro é dado em segmentos de reta, essa distância é calculada de acordo com a projeção do ponto correspondente ao bloco em cada um dos segmentos de reta (Figura ??) e escolhendo menor distância entre o ponto projetado e o ponto correspondente ao bloco. Após esse cálculo, o valor da influência do perímetro é determinado por uma função de potencial. A função de potencial usada por este trabalho é:

$$potencial(q, p_c, p_b) = \frac{q}{dist(p_c, p_b)^2} \quad (4.18)$$

Onde q é a carga do objeto, p_c é a posição do objeto carregado no mapa e p_b , a posição do bloco em questão. A carga dos pontos do perímetro é igual a 400. Como um bloco de construção ocupa um quadrado de 32×32 pixels, utiliza-se a média dos valores correspondentes a cada um dos quatro cantos que esse bloco ocupa no mapa. Essa técnica de utilizar a média das influências sobre os quatro pontos será usada por todos os outros elementos com carga.

Outros elementos que receberão carga são os recursos. Para não colocar construções que bloqueiem sua coleta, eles recebem carga repulsiva. Como cada recurso possui áreas diferentes, sua carga será proporcional à sua área. Ao contrário do perímetro, cada recurso possui apenas um ponto fixo no mapa, de forma que para calcular a influência, basta verificar a distância até o recurso. Cada recurso da região é levado em consideração, ao contrário do mais próximo. A carga das jazidas de minérios é igual a -13000 , e a carga dos gêiseres de gás vespeno é igual a -52000 .

Além desses, os gargalos também receberão carga repulsiva. Gargalos possuem uma posição e uma largura, dada em pixels. A influência deles irá depender desses dois fatores, de forma que a carga para os gargalos é igual a $-1000l$, onde l é a largura do gargalo.

Além de levar em conta esses fatores, o valor do bloco de construção é limitado pelo intervalo $[-80, 80]$ inicialmente e pode receber penalidades que o façam superar esse intervalo:

- -160 , caso o modo de busca por posicionamento seja tradicional, ao invés de criação



Figura 22: Muralhas Zerg em vários mapas. Aqui é possível observar a existência de duas *hatcheries* em todas as muralhas. Há também uma tendência de utilizar estruturas de defesa para preencher as brechas existentes na muralha.

de muralhas, e o bloco esteja fora da gosma;

- -80 , caso o bloco não seja construível ou esteja dentro de alguma alocação feita anteriormente;

Sabe-se que toda muralha zerg necessita de duas *hatcheries*, uma servindo como base e outra servindo como *macro hatch*⁹, como mostrado na Figura 22. Tendo em vista isso, e que a maior parte das *macro hatches* ficam na região mais distante, o que se propõe é iniciar a alocação da muralha com o posicionamento de uma *hatchery* utilizando todas essas cargas e funções. Em seguida, utilizar uma carga atrativa para base ao invés do perímetro, cujo valor é igual a 20000. Utiliza-se o ponto central da estrutura como referência. Além disso, todas as outras cargas também são utilizadas e somadas às devidas penalidades por construtibilidade, para que o algoritmo complete a muralha.

Para calcular o valor do posicionamento de uma estrutura, será utilizada a soma dos valores de influência de cada um dos blocos dentro da área da estrutura. A partir disso, deve-se seguir o seguinte algoritmo para a alocação de muralhas:

1. Alocar uma *hatchery*, ou estrutura 4×3 , no melhor posicionamento dentro da região da base escolhida, utilizando as funções mencionadas acima.
2. Até que a base esteja adjacente a alguma estrutura alocada na muralha, repita:

⁹*Hatchery* criada apenas para produção de unidades, ao invés de servir como depósito de recursos.

- (a) Para cada área que estruturas podem ocupar (para Zergs, isto significa, efetivamente, $\{2 \times 2, 3 \times 2\}$):
 - i. Achar o melhor posicionamento possível para a área que seja adjacente à última estrutura alocada, usando apenas a base como elemento carregado.
- (b) Selecionar o melhor valor de todas as áreas e alocar na muralha.

Deixando a alocação de estruturas mais genérica, o sistema de tomada de decisões poderá tentar otimizar as brechas, o custo ou o tempo de construção da muralha, tendo em vista as áreas alocadas.

Através dos direcionamentos tomados, conseguimos aplicar o modelo Cerebrate no jogo StarCraft, utilizando a raça Zerg como foco do trabalho. No próximo capítulo, apresentaremos experimentos práticos para validação do modelo.

5 Experimentos

5.1 Metodologia

Cerebrate foi implementado em C++, utilizando o Visual C++ 2008 Express. O código-fonte da implementação deste modelo está armazenado no site: <https://github.com/caiofreitaso/Cerebrate>.

Para implementá-lo, foi utilizada a BWAPI 3.7.6¹. Esta API é o *framework* usado para interagir com StarCraft. Ela provê várias funcionalidades que permitem estudantes e pesquisadores criar *game AIs* para esse jogo. Além das funcionalidades, a API expõe ao programador todas as informações necessárias para o controle e verificação do estado atual do jogo. Os módulos criados com esta API devem ser injetados dentro do processo do jogo através de um programa chamado Chaoslauncher², utilizando um *plug-in* específico para a BWAPI. Dessa forma, o Chaoslauncher dá acesso às informações do jogo para o módulo criado.

Os ministérios do Cerebrate necessitam de dados de análise de terreno como a localização das bases e as regiões construíveis. Estes dados são adquiridos com o uso da BWTA 1.7.1³. A BWTA é uma biblioteca de análise de terreno para a BWAPI. Ela implementa uma técnica de identificação de regiões no mapa e as separa através de gargalos. Esta técnica é descrita por Perkins (2010), e a biblioteca foi criada pelo mesmo autor. Além dessa técnica, a biblioteca implementa uma busca pelas localizações das bases no mapa que está sendo jogado. Estas localizações são usadas pela Agência de Inteligência para decidir a ordem de expansões.

Os experimentos foram realizados nos mapas Destination⁴ e Fighting Spirit⁵ (Figura 23). Estes foram os mapas mais jogados em competições, com 1196 e 1052 partidas

¹<http://bwapi.github.io>

²<http://wiki.teamliquid.net/starcraft/Chaoslauncher>

³<https://code.google.com/p/bwta/>

⁴<http://wiki.teamliquid.net/starcraft/Destination>

⁵http://wiki.teamliquid.net/starcraft/Fighting_Spirit

em competições oficiais, respectivamente. Eles também são amplamente utilizados em ligas de jogadores amadores como na liga iCCup⁶. Por serem mapas competitivos, o seu objetivo original é a destruição de todas as estruturas inimigas. Porém, este objetivo não será utilizado devido ao foco da pesquisa.

Apenas um experimento foi realizado para cada caso de teste. Como o posicionamento das unidades e dos recursos é estática, e a ordem dada às unidades é determinística, vários experimentos feitos nas mesmas condições apresentaram os mesmos resultados.

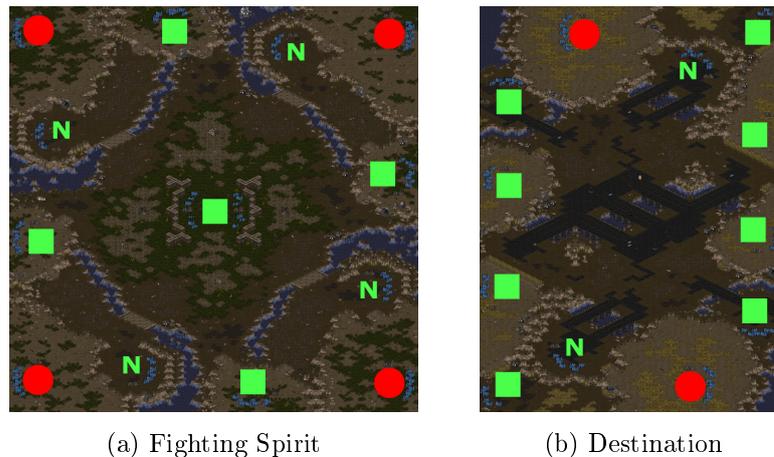


Figura 23: Mapas usados nos experimentos. Os círculos vermelhos apontam as posições iniciais em cada mapa. As naturais estão marcadas com a letra N. Os quadrados indicam as outras bases.

Para todos os experimentos, foi usada a trapaça de invencibilidade⁷. Isto se deve ao fato da inteligência artificial inimiga atacar o jogador. Caso os ataques fossem bem sucedidos, a perda de trabalhadores ou estruturas criaria ruídos nos resultados.

Os mapas foram jogados de maneira diferente para verificar cada um dos objetivos deste trabalho. Para testar o sistema de exploração de recursos, o Cerebrate jogou no mapa Destination e foram criados 17 trabalhadores. A coleta de recursos foi contabilizada durante um período de 5150 quadros, ou aproximadamente três minutos e trinta segundos. Como métrica, foi utilizada a quantidade total de minérios coletados após o tempo determinado. O sistema foi comparado com o método padrão de coleta de recursos do jogo, que é utilizado pelos jogadores profissionais. Desta forma, há um meio de verificar se o sistema proposto obteve uma melhora tanto na coleta total de recursos como na taxa de minérios coletados por que o sistema padrão.

⁶<http://iccup.com/en/starcraft/>

⁷Esta trapaça impede que unidades do jogador recebam qualquer tipo de dano proveniente de outros jogadores.

O sistema de posicionamento e construção de estruturas foi testado em ambos os mapas. A avaliação do posicionamento das estruturas se deu através da criação de muralhas nas naturais de todas as posições iniciais dos mapas. Utilizaremos a soma de todas as brechas existentes entre as estruturas e do final da muralha até o ponto não navegável mais próximo, como medida de avaliação. As muralhas alocadas pelo Cerebrate foram comparadas com as muralhas propostas pelo site TeamLiquid⁸. Este site reúne jogadores de diversos níveis, desde amadores até jogadores profissionais, para discutir os eventos que acontecem sobre StarCraft. As recomendações de muralhas publicadas neste site foram escolhidas por jogadores profissionais e jogadores amadores de alto nível, com o objetivo de reduzir as brechas existentes. Além de comparar quantitativamente, as muralhas também foram comparadas qualitativamente.

O sistema de definição da sequência de expansão territorial foi avaliado tanto em Destination como em Fighting Spirit. Ele foi comparado com as ordens de expansão usadas por jogadores profissionais nos mesmos mapas. Foram usados, para comparação, jogos Zerg versus Terran que ocorreram em quatro competições:

- WCG Korea 2009⁹,
- WCG Korea 2010¹⁰,
- GOM Classic Season 4¹¹,
- 2010 BigFile MBCGame StarCraft League¹².

Por fim, as aberturas foram testadas em todos os mapas. Foram escolhidas cinco aberturas sugeridas por um jogador de alto nível em entrevista. Elas são *5 pool*, *9 pool*, *overpool*, *12 pool* e *12 hatch*. As aberturas foram comparadas de acordo com a quantidade de minérios coletados e o tempo de término da ordem de construção. Também são mostrados os tempos em que os soldados inimigos chegaram à base. O Cerebrate enfrentou em todos os experimentos a *game AI* nativa do jogo. As aberturas estão descritas através das ordens de construção expostas na Tabela 5.

⁸<http://teamliquid.net/>

⁹http://www.teamliquid.net/tlpd/korean/leagues/219_WCG2009_Korea/main

¹⁰http://www.teamliquid.net/tlpd/korean/leagues/596_WCG2010_Korea/main

¹¹http://www.teamliquid.net/tlpd/korean/leagues/4329_GOMTV_Classic_Season_4/main

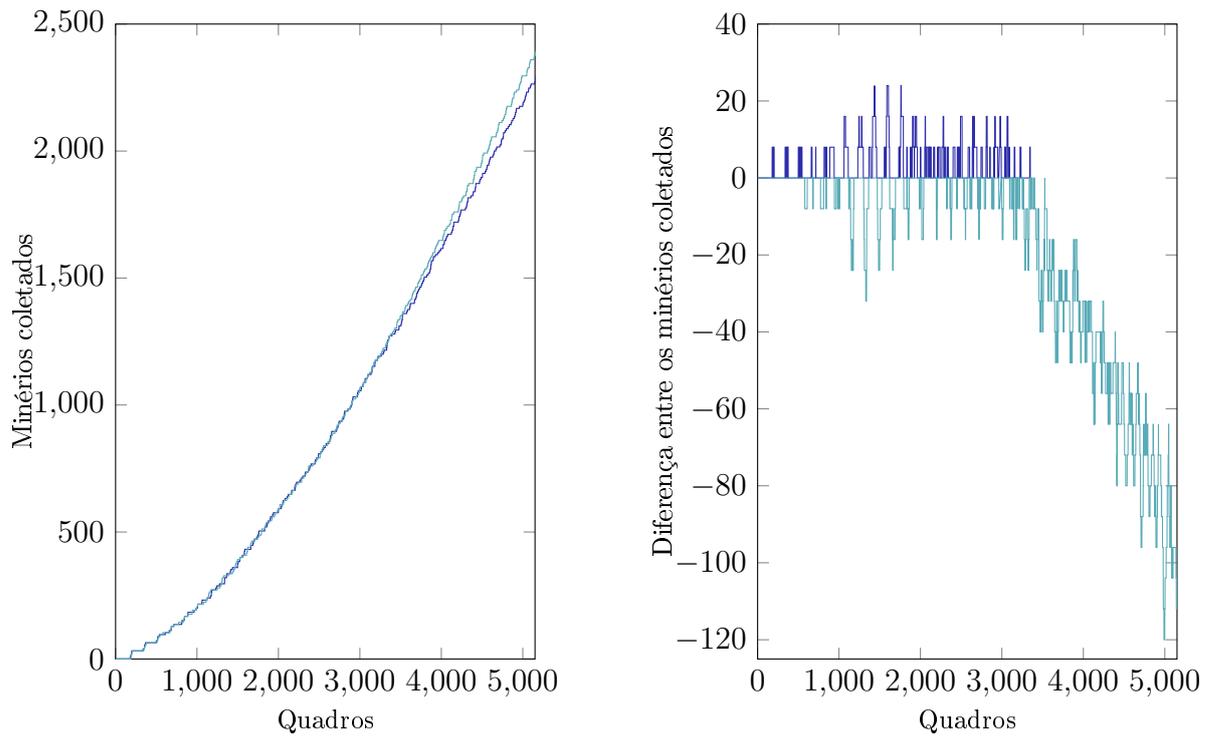
¹²http://www.teamliquid.net/tlpd/korean/leagues/530_Bigfile_MSL/main

5 pool	9 pool	Overpool	12 pool	12 hatch
<i>Drone</i>	<i>5 drones</i>	<i>5 drones</i>	<i>5 drones</i>	<i>5 drones</i>
<i>Spawning Pool</i>	<i>Spawning Pool</i>	<i>Overlord</i>	<i>Overlord</i>	<i>Overlord</i>
<i>2 drones</i>	<i>Drone</i>	<i>Spawning Pool</i>	<i>3 drones</i>	<i>3 drones</i>
<i>3 zerglings</i>	<i>Overlord</i>	<i>Drone</i>	<i>Spawning Pool</i>	<i>Hatchery</i>
	<i>3 zerglings</i>	<i>3 zerglings</i>	<i>Hatchery</i>	<i>Spawning Pool</i>
			<i>3 zerglings</i>	<i>3 zerglings</i>

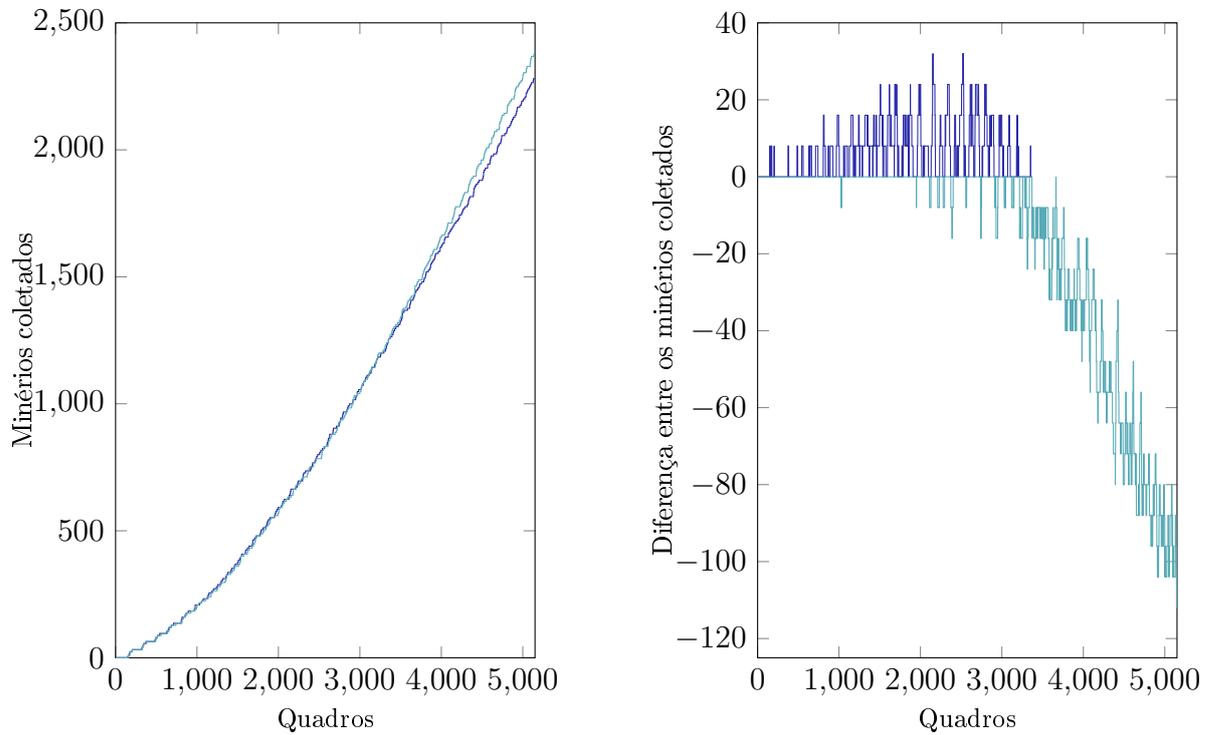
Tabela 5: Ordens de construção usadas no confronto Zerg versus Terran.

5.2 Coleta de Minérios

A Figura 24 mostra os resultados referentes aos minérios coletados através da máquina de estados do Cerebrate bem como a técnica implementada pelo jogo. A princípio, não existem grandes diferenças entre a quantidade de minérios coletados, pois existe apenas um trabalhador por minério. Esses trabalhadores trabalham de forma idêntica em ambos os casos. Mas conforme o número de trabalhadores por minério aumenta, maior fica a diferença entre os minérios coletados, mostrando que o comportamento conhecido como “trabalhadores andarilhos” torna a mineração mais ineficiente que poderia ser. Esse comportamento acontece quando um trabalhador encontra uma jazida já sendo minerada. Ele, então, busca outra jazida que esteja livre no quadro atual. Ao final do experimento foram coletados mais de 100 minérios a mais que o método nativo, em cada caso. Isto mostra que o tempo perdido ao ir de uma jazida à outra pode ser usado esperando a mesma jazida ocupada ficar livre. Para efeito de comparação, com esses 100 minérios é possível construir um *overlord*, que aumentaria a capacidade populacional, dois trabalhadores ou quatro *zerglings*.



(a) Destination (posição superior).



(b) Destination (posição inferior).

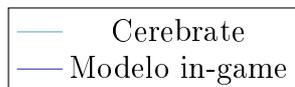


Figura 24: Comparação entre técnicas de mineração no mapa Destination.

5.3 Ordem de Expansão

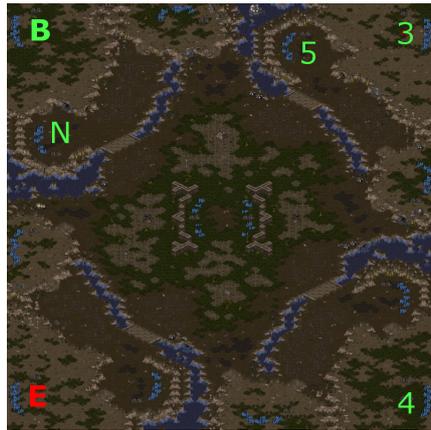


Figura 25: Ordem de expansão visto em partida profissional no mapa Destination. B indica a posição inicial, N indica a natural e os números indicam as expansões além da natural, na ordem que serão tomadas.

A ordem vista no mapa Destination (Figura 25) foi seguida fielmente pelo Cerebrate. O mesmo não pode ser dito do que ocorreu nos outros mapas, onde a lógica por trás da ordem de expansão não é a mesma.

No mapa Fighting Spirit, as expansões tendem a ser em outras posições iniciais, como pode ser visto na Figura 26. Isto se deve ao fato de a raça Zerg possuir a estrutura *Nydus Canal* (Figura 27.a). Esta estrutura permite o transporte instantâneo de unidades de um extremo a outro do canal. Isso permite ao jogador defender em pouco tempo uma base que esteja longe de sua principal. O modelo proposto não leva em consideração a possibilidade de usar essa estrutura, de forma que a ordem escolhida foi com as bases mais próximas. Outro ponto importante na escolha das expansões, para o jogador humano, é a facilidade de defender um ataque terrestre. Uma vez que as bases possuem apenas um gargalo que as ligam com o resto do mapa, é possível utilizar as unidades *lurkers*¹³ (Figura 27.b) para defender o gargalo. Esta configuração de principal, com apenas um gargalo longe da localização da base, se mostra mais fácil de defender que bases com vários gargalos e com uma área pequena.

¹³Unidade zerg que causa dano em área com seus ataques. Precisa enterrar-se para atacar, tornando-se uma unidade invisível no processo.



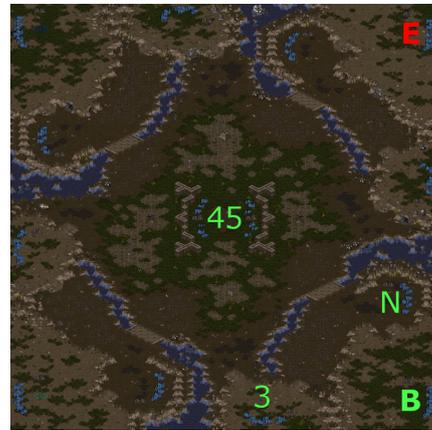
(a) Ordem feita por um jogador profissional (Canto superior esquerdo)



(b) Ordem feita pelo Cerebrate (Canto superior esquerdo)



(c) Ordem feita por um jogador profissional (Canto inferior direito)



(d) Ordem feita pelo Cerebrate (Canto inferior direito)

Figura 26: Comparação de ordens de expansão no mapa Fighting Spirit. B indica a posição inicial, N indica a natural e os números indicam as expansões além da natural, na ordem que serão tomadas.



(a) Nydus Canal



(b) Lurker

Figura 27: Elementos desconsiderados na criação da função de avaliação de bases.

5.4 Posicionamento de Estruturas

A Tabela 6 mostra a soma de todas as brechas das muralhas em pixels. Como o objetivo deste sistema é a minimização desta soma, quanto maior o número, pior o posicionamento das estruturas. Pelos resultados mostrados nesta tabela, as muralhas propostas pelo Cerebrate ficam aquém das recomendações de jogadores experientes de StarCraft. Apenas em um caso, o Cerebrate conseguiu atingir a expectativa, na posição superior do mapa Destination.

Caso	Brechas (Cerebrate)	Brechas (TeamLiquid.net)
Destination (superior)	75	75
Destination (inferior)	97	82
Fighting Spirit (superior direita)	75	51
Fighting Spirit (superior esquerda)	61	49
Fighting Spirit (inferior direita)	62	52
Fighting Spirit (inferior esquerda)	76	61

Tabela 6: Brechas dos modelos de muralha propostos pelo Cerebrate e pelo site TeamLiquid.



(a) TeamLiquid (Posição superior)



(b) Cerebrate (Posição superior)



(c) TeamLiquid (Posição inferior)



(d) Cerebrate (Posição inferior)

Figura 28: Comparação de muralhas no mapa Destination.



(a) TeamLiquid (Canto inferior direito)



(b) Cerebrate (Canto inferior direito)



(c) TeamLiquid (Canto inferior esquerdo)



(d) Cerebrate (Canto inferior esquerdo)



(e) TeamLiquid (Canto superior direito)



(f) Cerebrate (Canto superior direito)



(g) TeamLiquid (Canto superior esquerdo)



(h) Cerebrate (Canto superior esquerdo)

Figura 29: Comparação de muralhas nas posições iniciais de Fighting Spirit.

As muralhas alocadas em Destination (Figura 28) ficaram muito próximas das recomendações, com uma das construções mais à frente ou atrás do que deveria ser. A diferença do resultado da posição inferior se dá porque não há uma comparação de quantos blocos estão tocando a alocação anterior. Já a diferença do resultado da posição superior se dá pelo fato de que todos os blocos estão mais próximos do perímetro da região na sugestão do Cerebrate do que na posição sugerida pelo fórum.

A figura 29 mostra a comparação entre as muralhas do mapa Fighting Spirit. Destas imagens duas informações importantes podem ser tiradas. A primeira é que o algoritmo não leva em consideração a quantidade de blocos de movimentação que existem em um bloco de construção. Desta forma, a muralha acaba por ser atraída blocos de construção que possibilitam a passagem de unidades, como é possível observar na Figura 29.d e na Figura 29.f.

A segunda informação importante é a distância que existe entre as muralhas e os gargalos. As muralhas propostas possuem uma distância muito pequena para o maior gargalo, procurando deixar a muralha com a menor brecha possível. Já as muralhas alocadas pelo Cerebrate possuem uma distância grande, pois o valor da carga repulsiva é proporcional à largura do gargalo.

5.5 Aberturas

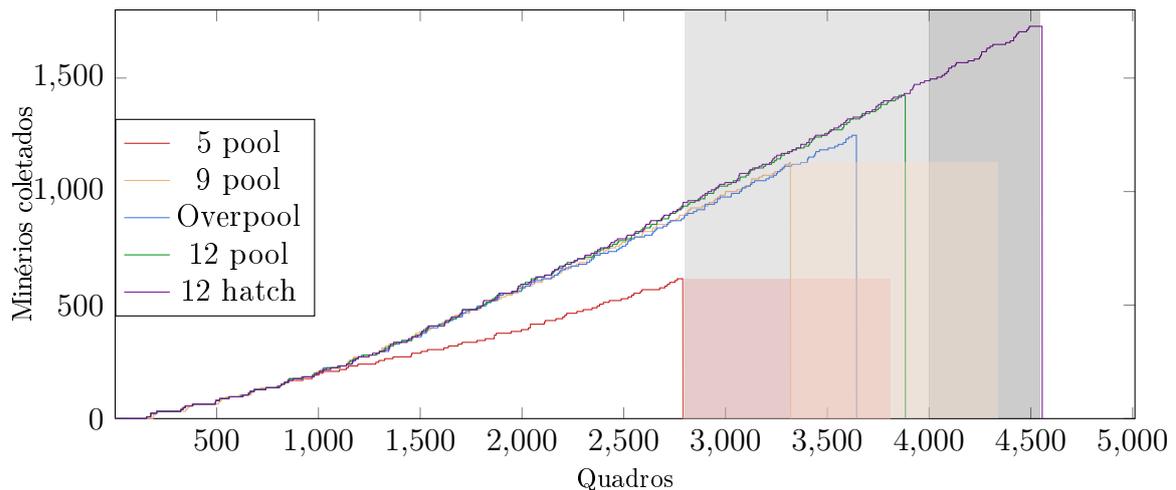


Figura 30: Comparação dos minérios coletados em cada abertura no mapa Destination. A área cinza mais clara representa o período de tempo em que as unidades inimigas estão no mapa movimentando-se até a base do Cerebrate, baseado no ataque que chega mais cedo. A área cinza mais escura mostra quando os ataques inimigos aconteceram. As áreas coloridas mostram o tempo que leva da criação das unidades até elas chegarem na base inimiga. As linhas perpendiculares mostram o tempo em que a abertura foi terminada.

A Figura 30 mostra a coleta de minérios no mapa Destination em cada abertura. Pode-se perceber uma enorme diferença entre a abertura *5 pool* e as outras. Embora ela consiga criar *zerglings* cerca de 530 quadros (aproximadamente 22 segundos) antes que a *9 pool*, a segunda abertura mais agressiva, a diferença econômica e a possibilidade de existir uma muralha de estruturas no terreno inimigo tornam esta estratégia muito arriscada. Ela consegue atacar as unidades inimigas que estão a caminho, caso o oponente opte por um *rush*, como visto pela sobreposição da área vermelha (tempo do percurso dos *zerglings* até a base inimiga) com a área cinza mais clara (tempo de percurso das unidades inimigas até a base do Cerebrate). Porém, caso o inimigo não possua uma muralha de defesa bem feita na sua base, esta abertura também tem a capacidade de causar danos significativos, pois os *zerglings* chegam em um tempo muito diferente da *9 pool*.

As outras aberturas possuem um poder ofensivo muito reduzido, visto que as unidades militares são criadas quando o ataque inimigo já está na iminência de acontecer. Destas, as aberturas *overpool* e *12 pool* possuem uma boa chance de sucesso, principalmente a última por ter uma economia mais solidificada, tanto por ter mais trabalhadores como por ter coletado mais minérios. A abertura *12 hatch* é muito arriscada, pois não possui uma defesa muito grande contra ataques e tem a mesma taxa de crescimento econômico que a *12 pool*.

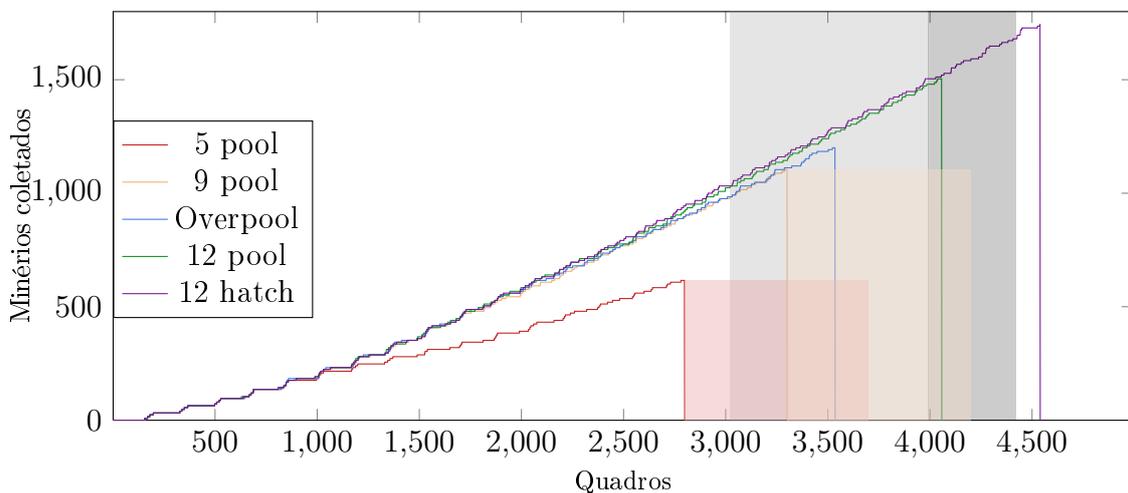


Figura 31: Comparação dos minérios coletados em cada abertura no mapa Fighting Spirit. Nestes experimentos, o Cerebrate começou na posição superior direita e o oponente na posição inferior direita do mapa. A área cinza mais clara representa o período de tempo em que as unidades inimigas estão no mapa movimentando-se até a base do Cerebrate, baseado no ataque que chega mais cedo. A área cinza mais escura mostra quando os ataques inimigos aconteceram. As áreas coloridas mostram o tempo que leva da criação das unidades até elas chegarem na base inimiga. As linhas perpendiculares mostram o tempo em que a abertura foi terminada.

No mapa Fighting Spirit (Figura 31), em posições próximas, a abertura *5 pool* possui um *timing* mais agressivo. Os *zerglings* chegam à base inimiga em cerca de 900 quadros ou 37 segundos, em comparação com os 1020 quadros (42 segundos) necessários em Destination. Os mesmos problemas existem neste mapa com relação ao posicionamento de estruturas inimigas. Porém, como o ataque dos *zerglings* chega mais cedo à base inimiga, a possibilidade do oponente não ter concluído sua muralha é maior.

Mesmo com essa diferença de *timing*, optar pela *9 pool* permanece uma opção economicamente melhor. Embora o ataque chegue atrasado em relação à *5 pool*, a curta distância entre as bases torna esta abertura mais viável que em Destination. Em situações reais, jogadores tiveram êxito ao executar esta abertura nas mesmas condições, como visto nas finais da WCG Korea 2010¹⁴ onde Lee Jae Dong (Jaedong) venceu Lee Young Ho (Flash).

Pela distância, as outras aberturas se tornam mais vulneráveis. Tanto *12 pool* como *12 hatch* possuem tempo de conclusão dentro do intervalo de tempo dos ataques inimigos. É possível defletir o ataque, ao custo de um dano econômico grande. Isto se deve tanto por desviar recursos para a construção de unidades militares fora do plano, como pelo fato de que trabalhadores poderão ser mortos durante o ataque.

A abertura *overpool* oferece, então, uma alternativa defensiva mais segura nestas condições. Seus *zerglings* terminam em um período mais tardio que *9 pool* e possui um crescimento econômico menos acentuado que *12 pool* e *12 hatch*. Tanto ela como a *9 pool* são boas escolhas neste mapa.

Estes experimentos serviram para mostrar os aspectos necessários para execução das aberturas da raça Zerg em StarCraft. Eles mostraram uma melhora no desempenho da coleta de recursos em relação ao método nativo, quando a quantidade de trabalhadores é superior ao número de jazidas de uma base. Além disso, mostramos o resultado do processo de posicionamento de estruturas na criação de muralhas. Este processo apresentou resultados aquém do necessário, pois não avaliou brechas em posições extremas da muralha. Este método é um dos trabalhos sugeridos a partir deste trabalho.

No próximo capítulo, abordaremos mais a fundo as conclusões referentes aos resultados encontrados e os trabalhos futuros.

¹⁴<http://www.youtube.com/watch?v=JtJd-br89z8>

6 Considerações finais

Este trabalho apresentou um modelo para *game AI* de jogos de estratégia em tempo real. Este modelo foi baseado na separação de tarefas dos ministérios de governos atuais, e possui duas fases de funcionamento. Na primeira fase, a *game AI* se encarrega de seguir uma abertura selecionada através de métodos de análise de terreno. Na segunda fase, o sistema reage ao que o oponente está fazendo através de negociações entre os ministérios, onde eles debatem sobre quais itens devem ir para a ordem de construção. Cada ministério e suas responsabilidades foram descritos em detalhes, bem como a comunicação que ocorre entre eles.

Este modelo, chamado Cerebrate, foi aplicado ao jogo StarCraft, utilizando a raça Zerg como objeto de estudo. Para delimitar o escopo da pesquisa, apenas as aberturas Zerg utilizadas no confronto Zerg versus Terran foram consideradas. Elas foram pesquisadas através de uma entrevista com um jogador de alto nível. Estas aberturas foram classificadas de acordo com sua agressividade e com sua capacidade de sobrevivência em mapas variados. O objetivo deste trabalho foi comparar os comportamentos das aberturas de um ponto de vista militar e do ponto de vista econômico.

Para a implementação deste modelo, foram utilizadas as bibliotecas padrões de interação com o jogo, a BWAPI e a BWTA. Devido à complexidade do jogo e do modelo, a implementação se focou em apresentar as diferenças de mineração e tempo de criação de *zerglings* de cada abertura no confronto Zerg versus Terran. Assim, a segunda fase do sistema não foi necessária, pois apenas aberturas foram testadas. Cada ministério criado foi testado em alguma funcionalidade, porém eles apresentaram resultados subótimos, com a exceção do Ministério de Minas que conseguiu melhorar a capacidade de mineração de um grande número de trabalhadores.

As aberturas apresentaram três tipos comportamentos. A *5 pool* possui a menor taxa de crescimento econômico, pois para de produzir trabalhadores em determinado ponto. Desta forma, a diferença de minérios coletados, em comparação com as outras aberturas,

crece com o tempo. As aberturas *9 pool* e *overpool* não apresentam diferença de crescimento econômico significativa. A diferença entre essas aberturas está no tempo de criação dos *zerglings*, onde a *9 pool* apresenta um tempo menor que a *overpool*. Também com crescimento similar entre si, as aberturas *12 pool* e *12 hatch* apresentam uma diferença maior no tempo de criação das unidades militares.

Além deste objetivo principal, comparar as aberturas, três objetivos específicos foram determinados buscando otimizar os aspectos essenciais da execução das aberturas. O primeiro objetivo é a criação de um sistema que maximize a quantidade de recursos explorados. O sistema de coleta de recursos se mostrou melhor que o método nativo quando o número de trabalhadores é maior que o número de jazidas de minérios. O método sugerido impede o comportamento “trabalhador andarilho” que existe no método nativo, sendo possível evitar a perda de tempo de mineração produzida por este comportamento. Desta forma, o sistema conseguiu atingir seu objetivo.

O segundo objetivo foi a criação de um sistema que dificulte o acesso dos inimigos à base. O posicionamento de estruturas conseguiu encontrar soluções boas no mapa *Destination*, porém encontrou soluções sobótimas no mapa *Fighting Spirit*. Isto se deu pelo fato do algoritmo não levar em consideração as brechas que podem ser geradas pelos blocos na fronteira da região da base. Além disso, a influência dos gargalos estava grande a ponto de desviar a posição mais extrema da muralha da posição ótima. Assim, o sistema conseguiu atingir seu objetivo, mesmo que não tenha sido de maneira efetiva.

O último objetivo é a criação de um sistema que aponte a sequência de expansões que o jogador deve tomar. Este sistema conseguiu reconhecer a natural de uma base principal, porém não seguiu a lógica por trás da escolha de expansões feita por jogadores profissionais. Não foram considerados na criação da função de avaliação a possibilidade de usar *nydus canal* e *lurkers*. O sistema conseguiu definir uma ordem de expansão coerente, portanto mesmo que não tenha chegado às mesmas ordens dos profissionais, ele cumpriu seu objetivo.

Assim, este trabalho validou o uso *potential fields* para alocação de muralhas em jogos de estratégia em tempo real. Embora não tenha produzido resultados ótimos, novas pesquisas podem ser realizadas usando *potential fields* para posicionamento de estruturas. Ele também propôs um novo modelo para *game AI* voltada a RTS, baseada em ministérios de governos atuais. Uma vez que o modelo não foi implementado em sua completude, é necessária uma investigação a fundo do comportamento dos outros ministérios. Também é necessária a investigação de sua segunda fase, a reação, para avaliar o desempenho de

todos os ministérios atuando em conjunto. Mesmo os ministérios já implementados podem ser investigados com a avaliação ou a mudança dos métodos que estão relacionados a eles.

O Ministério de Minas pode ser investigado através do método de redistribuição de trabalhadores, para validar o modelo de transferência proposto. Isto é, se o modelo proposto realmente produz os melhores custo-benefício para a situação.

O Ministério de Infraestrutura pode ser investigado através da mudança do método de alocação de estruturas. Usando do mesmo conceito, pode-se atribuir valores distintos a cada um dos blocos que estão no perímetro da região, inversamente proporcionais a quantidade de blocos de movimentação que existem dentro deles. Esta mudança substituiria a projeção feita no polígono que descreve os limites da região de uma base. Outra mudança pode ser feita na carga dos gargalos, deixando-a independente da largura. Outros experimentos podem ser feitos ao tentar prever qual lado da estrutura vai ficar adjacente à parede da região e alocar a estrutura com menor brecha neste lado. Também pode-se adotar outro modelo que não utilize *potential fields*, similar ao usado por Čertický (2013).

A Agência de Inteligência pode ser investigada no que diz respeito à ordem de expansão. Uma vez que não foram levados em conta a quantidade de gargalos de uma região ou a possibilidade de usar um *nydus canal*, a fórmula de potencial de base deve ser atualizada e verificada. Também pode-se avaliar a investigação de movimentação inimiga e a investigação do desenvolvimento tecnológico inimigo.

Este trabalho apresentou um modelo conceitual genérico para jogos de estratégia em tempo real, que procura distribuir as responsabilidades de tomadas decisão mais alto nível para todos os seus componentes. Esperamos que este trabalho sirva para estimular a pesquisa na área de inteligência artificial voltada a jogos, de forma que não só sejam investigados mais resultados desta aplicação do modelo, mas também que sejam pesquisados outros modelos que procurem imitar o processo de decisão de um jogador profissional. Esperamos também pesquisas envolvendo o modelo em sua completude, aplicado a algum jogo de estratégia em tempo real. Embora o jogo usado como ambiente de pesquisa tenha sido o StarCraft, a aplicação do modelo proposto em outros jogos de estratégia em tempo real.

Aqui também foram propostos modelos de resolução de problemas mais específicos de RTS, como o posicionamento de estruturas e a transferência de trabalhadores. Devido ao baixo número de trabalhos voltados a esses temas, esperamos que esta exposição dos problemas estimule a busca por soluções e/ou modelos mais eficazes para estes problemas.

Referências

- BOUZY, B.; CAZENAVE, T. Computer go: an ai oriented survey. *Artificial Intelligence*, Elsevier, v. 132, n. 1, p. 39–103, 2001.
- BURO, M. Real-time strategy games: A new ai research challenge. In: *2003 International Joint Conference on Artificial Intelligence Proceedings*. [S.l.: s.n.], 2003. p. 1534–1535.
- CADENA, P.; GARRIDO, L. Fuzzy case-based reasoning for managing strategic and tactical reasoning in starcraft. *MICAI (1)*, Springer, v. 7094, p. 113–124, 2011.
- DERESZYNSKI, E. W. et al. Learning probabilistic behavior models in real-time strategy games. In: *Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*. [S.l.: s.n.], 2011.
- GRAFT, K. Blizzard confirms one “frontline release” for ’09. fev. 2009. Disponível em: <<http://www.edge-online.com/news/blizzard-confirms-one-frontline-release-09/>>.
- HAGELBÄCK, J. *A Multiagent Potential Field-Based Bot for Real-Time Strategy Game*. Tese (Doutorado) — Blekinge Institute of Technology, Karlskrona, Suécia, jan. 2012.
- HSIEH, J. L.; SUN, C. T. Building a player strategy model by analyzing replays of real-time strategy games. In: *Proceedings of the International Joint Conference on Neural Networks*. [S.l.: s.n.], 2008. p. 3106–3111.
- ONTAÑÓN, S. et al. A survey of real-time strategy game ai research and competition in starcraft. *IEEE Transactions on Computational Intelligence and AI in Games*, IEEE, v. 5, n. 4, p. 293–311, 2013.
- PERKINS, L. Terrain analysis in real-time strategy games: An integrated approach to choke point detection and region decomposition. In: AAAI PRESS. *Proceedings of the Sixth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE-10)*. Stanford, Palo Alto, California: AAAI Press, 2010. p. 168–173. Disponível em: <<http://aaai.org/ocs/index.php/AIIDE/AIIDE10/paper/view/2114>>.
- RUSSEL, S.; NORVIG, P. *Artificial Intelligence: A Modern Approach*. 3th. ed. [S.l.]: Pearson Education, 2010.
- SAJJAD, M.; ISLAM, M. M. ul. *Asymmetric Potential Fields: Implementation of Asymmetric Potential Fields in Real Time Strategy Game*. Karlskrona, Suécia: Blekinge Institute of Technology, jan. 2011.
- SYNNAEVE, G.; BESSIÈRE, P. A bayesian model for opening prediction in rts games with application to starcraft. In: *2011 IEEE Symposium on Computational Intelligence and Games*. [S.l.]: IEEE Press, 2011. p. 281–288.

- SYNNAEVE, G.; BESSIÈRE, P. Special tactics: a bayesian approach to tactical decision-making. In: *2012 IEEE Symposium on Computational Intelligence and Games*. [S.l.]: IEEE Press, 2012. p. 409–416.
- URIARTE, A.; ONTAÑÓN, S. Kiting in rts games using influence maps. In: AAAI PRESS. *Artificial Intelligence in Adversarial Real-Time Games: Papers from the 2012 AIIDE Workshop*. Stanford, Palo Alto, California: AAAI Press, 2012. p. 31–36.
- ČERTICKÝ, M. Implementing a wall-in building placement in starcraft with declarative programming. *arXiv preprint arXiv:1306.4460*, 2013.
- WEBER, B. G.; MATEAS, M. A data mining approach to strategy prediction. In: *2009 IEEE Symposium on Computational Intelligence and Games*. [S.l.]: IEEE Press, 2009. p. 140–147.
- WEBER, B. G.; MATEAS, M.; JHALA, A. Building human-level ai for real-time strategy games. In: AAAI PRESS. *Proceedings of the AAAI Fall Symposium on Advances in Cognitive Systems*. Stanford, Palo Alto, California: AAAI Press, 2011.
- WEBER, B. G.; MATEAS, M.; JHALA, A. A particle model for state estimation in real-time strategy games. In: AAAI PRESS. *Proceedings of the Seventh AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE-11)*. Stanford, Palo Alto, California: AAAI Press, 2011. p. 103–108.
- WENDER, S.; WATSON, I. Applying reinforcement learning to small scale combat in the real-time strategy game starcraft: Broodwar. In: *2012 IEEE Symposium on Computational Intelligence and Games*. [S.l.]: IEEE Press, 2012. p. 402–408.
- YOUNG, J.; HAWES, N. Evolutionary learning of goal priorities in a real-time strategy game. In: *Proceedings of the Eighth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE-12)*. [S.l.: s.n.], 2012. p. 87–92.

APÊNDICE A - Papéis das Unidades em StarCraft

Papéis marcados com X estão nas unidades desde sua criação, e papéis marcados com P precisam de uma pesquisa tecnológica para desbloquear.

Unidade	LS	AA	Harass	AoE	Siege	Caster	Stealth	Air	Tank	Detector	Transport	NGA
Overlord										X	P	
Zergling	X		P									
Hydralisk	P	X										
Lurker				X			X					
Mutalisk		X	X					X				
Scourge		X						X				X
Queen						X		X				
Guardian					X			X				
Devourer		X						X				X
Ultralisk	P			X								
Defiler		X				X			X			

Papéis das unidades da raça Zerg.

Unidade	LS	AA	Harass	AoE	Siege	Caster	Stealth	Air	Tank	Detector	Transport	NGA
Marine	X	X										
Firebat	X				X							
Medic	X					X						
Vulture	X		P									
Siege Tank	X			P	P							
Goliath		X										
Wraith		X	X				P					
Dropship											X	
Valkyrie		X			X							X
Science Vessel						X				X		
Battlecruiser		X				X						
Ghost		X		P		X	P					
Spider Mine				X			X					

Papéis das unidades da raça Terran.

Unidade	LS	AA	Harass	AoE	Siege	Caster	Stealth	Air	Tank	Detector	Transport	NGA
Zealot	X								X			
Dragon	X	X										
High Templar		X		X		X						
Dark Templar			X				X					
Archon		X		X		X			X			
Dark Archon									X			
Shuttle							X					
Observer										X		
Reaver				X	X							
Scout		X						X				
Corsair		X	X	X				X				X
Carrier	X	X			X			X				
Arbiter						X						

Papéis das unidades da raça Protoss.