Adversarial Hierarchical-Task Network para Jogos em Tempo Real

Matheus de Souza Redecker

Orientador: Prof. Felipe Rech Meneguzzi

Curso de Ciência da Computação – Pontifícia Universidade Católica do Rio Grande do Sul

✉ matheus.redecker@acad.pucrs.br

Motivação

- As reações das jogadas, nos jogos de computador, devem ser quase que imediatas. Técnicas que tentam explorar todas as possibilidades de um jogo se tornam inviáveis para jogos de alta complexidade.
- Por exemplo, no xadrez a quantidade aproximada de estados possíveis é de 10^{40}, isso mostra que é preciso algoritmos eficientes para gerar uma ação de maneira rápida.
- O intuito deste trabalho é explorar eficientemente o espaço de ações disponíveis usando conhecimento de domínio, a fim de definir qual a próxima ação que deve ser executada. Para isso, propomos a utilização do algoritmo de Adversarial Hierarchical-Task Network (AHTN).
- O MicroRTS foi o jogo escolhido como plataforma para a implementação do algoritmo.

Background

Java Simple Hierarchical Ordered Planner 2

- Java Simple Hierarchical Ordered Planner 2 (JSHOP2) [1] é um sistema de planejamento independente de domínio baseado em HTN desenvolvido em Java.
- O planejamento HTN é feito a partir da decomposição de tarefas de alto nível para níveis mais baixos.
- O JSHOP2 precisa de uma descrição do domínio. A descrição do domínio contém a formalização das ações dos agentes, tarefas e métodos as decompõem.

Adversarial Hierarchical-Task Network

- O AHTN [3] é um algoritmo desenvolvido para lidar com o problema do grande fator de ramificação dos jogos em tempo real.
- Ele utiliza conhecimento de domínio no estilo de planejamento hierárquico (HTN).
- O algoritmo de AHTN combina técnicas de planejamento hierárquico com o algoritmo *minimax search*.
- O algoritmo assume jogos totalmente observáveis, baseados em turno e determinísticos.
- O pseudocódigo do algoritmo de AHTN é ilustrado no Código 1.

1: **function** AHTNMAX(s, N_+, N_-, t_+, t_-, d)
2: **if** $terminal(s) \vee d \leq 0$ **then**
3: **return** $(N_+, N_-, e(s))$
4: **end if**
5: **if** $nextAction(N_+, t_+) \neq \perp$ **then**
6: $t = nextAction(N_+, t_+)$
7: **return** AHTNMIN($(\gamma(s,t), N_+, N_-, t, t_-, d-1)$)
8: **end if**
9: $N_+^* = \perp, N_-^* = \perp, v^* = -\infty$
10: $\aleph = decompositions_+(s, N_+, N_-, t_+, t_-)$
11: **for all** $N \in \aleph$ **do**
12: $(N_+^{'}, N_-^{'}, v^{'}) = AHTNMax(s, N, N_-, t_+, t_-, d)$
13: **if** $v^{'} > v^*$ **then**
14: $N_+^* = N_+^{'}, N_-^* = N_-^{'}, v^* = v^{'}$
15: **end if**
16: **end for**
17: **return** (N_+^*, N_-^*, v^*)
18: **end function**

Código 1: Pseudocódigo do algoritmo de AHTN.

MicroRTS

- O MicroRTS [2] é um jogo de estratégia em tempo real (RTS), feito por Santiago Ontañón.
- Ele é uma simplificação de jogos como o Starcraft.
- O MicroRTS foi desenvolvido para fins acadêmicos, com o intuito de aplicar e desenvolver técnicas de Inteligência Artificial.
- A Figura 1 ilustra um exemplo de tela do jogo.

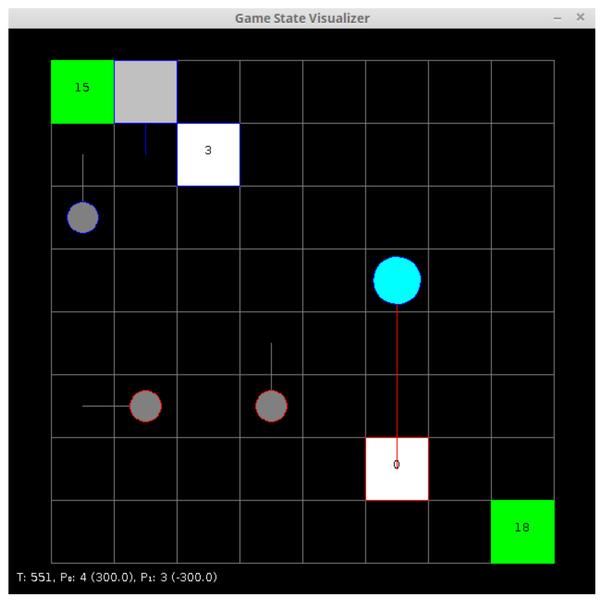

Figura 1: Exemplo de tela do MicroRTS.

Implementação

- A implementação do algoritmo foi feita na plataforma do MicroRTS, utilizando o SHOP2 para geração dos planos.
- Foram criados dois conhecimento de domínio pensando em cenários de jogo onde o jogador cria tropas para mandar destruir a base adversária.
- A primeira estratégia cria apenas uma unidade de ataque e manda atacar. Já a segunda estratégia enquanto está atacando também cria novas tropas.

Experimentos e resultados

- O MicroRTS possui técnicas de jogo implementadas para se jogar contra.
- As técnicas presentes no MicroRTS são usadas como adversário do algoritmo de AHTN.
- Os experimentos foram executados em um mapa 16 por 16 e com o algoritmo começando dos dois lados do jogo, o lado azul na parte superior, e o lado vermelho na parte inferior. Cada jogador inicia possuindo uma base, um trabalhador e dois recursos próximos a sua base.
- As técnicas do MicroRTS, usadas como adversário do algoritmo de AHTN, são determinísticas, fazendo com que o resultado do jogo seja sempre o mesmo.
- O lado vermelho possui vantagem, em relação ao azul, pois a construção que produz as tropas é posicionado em local propicio para tanto.
- A Tabela 1 ilustra os resultados obtidos.

Estratégia 1		
Adversário	**Lado Azul**	**Lado Vermelho**
RandomAI	100%	100%
RangedRush	0%	100%
HeavyRush	0%	100%
LightRush	0%	100%
Worker	0%	0%
Estratégia 2		
RandomAI	100%	100%
RangedRush	100%	100%
HeavyRush	0%	100%
LightRush	0%	100%
WorkerRush	0%	0%

Tabela 1: Porcentagem de vitórias do algoritmo AHTN contra as técnicas do MicroRTS.

Referências

[1] Okhtay Ilghami. Documentation for JSHOP2. *Department of Computer Science, University of Maryland, Tech. Rep*, 2006.

[2] Santiago Ontanón. The combinatorial multi-armed bandit problem and its application to real-time strategy games. In *Ninth Artificial Intelligence and Interactive Digital Entertainment Conference*, pages 58–64, 2013.

[3] Santiago Ontañón and Michael Buro. Adversarial hierarchical-task network planning for complex real-time games. In *Proceedings of the 24th International Conference on Artificial Intelligence*, pages 1652–1658, 2015.