Instituto Tecnológico de Aeronáutica

Divisão de Ciência da Computação
Departamento de Teoria da Computação
CTC-15: Inteligência Artificial

Trabalho de Exame

Planet Wars

Alunos
Cesar R. Kawakami
Guilherme R. N. Souza
Leonardo Ribeiro Carvalho
Rodrigo Simões Almeida

Professor
Carlos Henrique Ribeiro

02 de Dezembro de 2010

1 Objetivos

Fazer um jogador autônomo para Planet Wars, jogo usado na competição do Google Ai-Contest do ano de 2010. Esse jogador deve aplicar os conceitos e algoritmos vistos na matéria de Inteligência Artificial.

2 Introdução

Planet Wars é um jogo entre dois ou mais jogadores, cujo objetivo final é ter a maior frota de naves dentre todos os participantes. Apesar de uma partida poder ter vários jogadores, todas as partidas realizadas, tanto nos testes como na competição em si, foram realizadas com dois adversários.

O jogo é dividido em turnos no qual ambos jogadores tem todas informações relevantes disponíveis para tomar suas decisões e enviar frotas para combate. O combate entre frotas ocorre nos planetas, cada nave destrói exatamente uma nave inimiga e também é destruída. Ao final do combate quem tiver o maior número de naves sobreviventes no planeta o domina. Se restarem 0 naves o dono do planeta não muda. O jogador com planetas sob seu domínio tem sua frota local aumentada de acordo com a taxa de crescimento específica, a qual depende do tamanho do planeta dominado.

A abordagem para construção do jogador autônomo (a partir de agora denominado *bot*) dividiu-se em duas partes. A primeira foi o desenvolvimento de um *bot* que sabe se comunicar com o juíz/servidor e implementa heurísticas (estratégias básicas relacionados ao entendimento do funcionamento geral do jogo). Exemplos de fatores levados em consideração na heurística são: tentar defender suas naves, destruir naves inimigas, benefício trazido por conquistar planetas, entre outros. Mais informações na seção 3.

Essa versão inicial foi colocada no servidor do Google Ai-Contest e já apresentou bons resultados. A heurística, contudo, dependia de um conjunto de trinta e dois pesos associados a diferentes situações de jogo. Esses pesos haviam sido escolhidos sem muito critério, apenas seguiram um lógica geral relacionada principalmente ao sinal e módulo do peso associado. Para conseguir um melhor conjunto de pesos foi utilizado um algoritmo genético de evolução também detalhado na seção 3.

3 Implementação

3.1 *Bot* com estratégia básica

Inicialmete foi implementado um *bot* com estratégia básica. Toda a avaliação do estado do jogo considera um certo número de jogadas a frente, chamado horizonte. Esse horizonte funciona como uma poda de considerações muito distantes que poderiam atrasar o jogador e evita considerações que não fazem sentido, pois a previsão não considera que haverão jogadas, i.e., essa aboradagem avalia o jogo daqui a um número de jogadas se todos os jogadores tivessem como suas próximas ações não fazer nada. Esse horizonte é calculado levando em consideração

a distância média entre planetas a serem defendidos pelo *bot* e os planetas dominados pelo adversário.

A estratégia de jogo pode ser definida pelos passos:

- Para cada planeta acha-se quantas naves podem ser mandadas para o ataque sem que o planeta fique muito desprotegido e possa ser facilmente conquistado. Para isso, analisa-se as frotas que o estão atacando, achando quantas naves de defesa ele perderia a cada turno até o horizonte. O valor restante de naves ao atingir o horizonte é o valor que é considerado disponível para ataque e defesa, um valor negativo significa perda do domínio.
- Cada planeta com naves de crédito tenta defender outros planetas. Cada planeta pertencente ao jogador busca outros planetas que serão consquistados pelo inimigo (previsão negativa no item anterior) tentando fornecer naves suficientes para evitar essa perda. Essa ajuda a planetas vizinhos está condicionada à manutenção do domínio do planeta de origem das naves, caso a ajuda signifique perder o planeta que ajudou a ajuda não acontecerá. A ajuda também considera o tempo que a ajuda demorará para chegar ao planeta ajudado, se ele for conquistado antes de ser possível chegar a ajuda ela não será enviada.
- Com as naves restantes, após defender a si mesmo e fornecer ajuda a planetas aliados, cada planeta busca entre os planetas neutros e do inimigo aquele que apresenta o maior ganho esperado e realiza o ataque. Para cada planeta, é calculado um valor que representa o lucro esperado por atacá-lo. Dentro desse lucro é levado em consideração naves perdidas no ataque, naves perdidas pelo inimigo, quantas naves seriam ganhas e quantas naves o inimigo deixaria de ganhar devido à taxa de crescimento fornecida pelo planeta.

Essa estratégia considera os pontos tidos como essenciais para o bom desempenho do jogador e leva em consideração diversos jogos assistidos. A partir desta versão é possível fazer variantes jogarem entre si e selecionar a melhor dentre as opções geradas.

3.1.1 Detalhamento em algoritmo da estratégia utilizada

A estratégia geral de jogo, em cada turno, é dada por

procedimento FaçaTurno()
para todo planeta meu p **faça**
  DefendeOutrosPlanetas(p)
  AtacaOutroPlaneta(p)

procedimento DefendeOutrosPlanetas(planeta meu p)
para todo planeta meu q **faça**
  se existir $1 \leq t \leq$ MAX_TURN com Previsão(q, t) < 0 **então**
    $n \leftarrow -$Previsão(q, t)
    se naves sobrando em $p > n$ **e** Distância(p, q) $< t$ **então**
      EnviarFrota(p, q, n)

procedimento AtacaOutroPlaneta(planeta meu p)
para todo planeta não-meu q **faça**
  $horizonte \leftarrow$ Média($\{$Distância(p, q) : $p \in$ meus planetas $\wedge$ $q \in$ não-meus planetas$\}$)
  se Distância(p, q) $\leq horizonte$ **então**
    $residuo \leftarrow -$Previsão(q, $horizonte$)
    se $residuo \geq 0$ **e** $residuo \leq$ naves sobrando em p **então**

3.2 Algoritmo genético

Para a execução do algoritmo foi definido como cromossomo o conjunto de trinta e dois pesos, sendo cada peso um alelo. Uma população é o conjunto de cinquenta cromossomos. A população inicial foi gerada com quarenta e nove *bots* com cormossomos aleatórios mais o *bot* original.

Após gerar a população inicial, o programa entra em um loop infinito de treinamento no qual repete as seguintes ações: salva a geração, promove a seleção natural e obtém a próxima geração. Salvar a geração signiica simplesmente salvar os pesos de todos os elementos que a compõe em um arquivo. Esses arquivos funcionam como uma espécie de controle de versão.

A seleção natural dos indivíduos e a obtenção da próxima geração são realizadas através do processo:

- Seleciona, dentro dos cinquenta elementos da população, seis jogadores que irão participar de um campeonato
- Realiza o campeonato, que consiste fazer todas as combinações dois a dois desses jogadores e para cada dupla realizar jogos em 10 mapas aleatórios dentre os 100 existentes
- Os jogadores são ordenados por número de vitórias (contando uma por mapa) e a seleção de quem produzirá descendentes segue o procedimento conhecido como *Tournment Selection* com probalidade de escolher o campeão de 70%.
- É produzida a nova geração. Dado o conjunto de indivíduos que produzirão descendentes são escolhidos dois aleatoriamente e é feito um crossover uniforme gerando dois indivíduos. Isso é feito cinquenta vezes, até gerar duas vezes o tamanho da população.
- Esses indivíduos sofrem mutação com probabilidade de 3%
- Desses cem indivíduos são escolhidos cinquenta aleatoriamente que comporão a próxima geração

O programa foi feito de modo que uma geração pode ser evoluída de maneira paralela, dividindo jogos entre diversos computadores. Essa funcionalidade, junto às técnicas probabilísticas de efetuar campeonatos e seleção de população, foi muito importante, já que a evolução é um gargalo na obtanção de um bom *bot*, justamente por ser um processo demorado.

4 Resultados

O *bot* que participou da competição oficial no Google Ai-Contest foi a versão original, sem evolução, obtendo a posição 524 de um total de 4617 participantes válidos, ou seja, que o programa rodava adequadamente. O resultado obtido pode ser considerado satisfatório frente ao alto nível dos competidores. Ao final da evolução o bot obtido foi capaz de ganhar do original com relativa facilidade, mostrando a validade do processo.

A evolução atingiu a geração XXXXXXX e teve a seguinte relação de pesos:

Valor	Descrição
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	
0	

5 Possíveis melhorias

O programa ainda poderia

6 Conclusão

foi bom