Manual Técnico

UC: Inteligência Artificial

Alunos:

- João Fernandes - 202100718
- Rodrigo Santos - 202100722

Docente: Joaquim Filipe

Índice

1. Introdução

No âmbito do segundo projeto da Unidade Curricular (UC) de Inteligência Artificial (IA), do 1º semestre do 3º ano da Licenciatura em Engenharia Informática, foi solicitado aos alunos que implementassem um programa para resolução do problema Adji-boto, recorrendo um algoritmo de teoria de jogos.

O presente manual técnico tem como objetivo descrever a solução desenvolvida pelos alunos, bem como os detalhes da sua implementação.

2. Algoritmo

De forma a resolver o problema proposto, os alunos implementaram o algoritmo Negamax com cortes alfa-beta, usando uma abordagem funcional, recursiva e não-destrutiva. Foi ainda aplicada memoização de forma a evitar a análise de estados repetidos.

```
(defun negamax (node depth jogador &optional (alpha most-negative-fixnum) (beta
most-positive-fixnum) (color 1))
  "Implementa o algoritmo Negamax com memorização e poda alfa-beta."
  (let ((cache-key (list node depth jogador alpha beta color)))
    (or (gethash cache-key *negamax-cache*)
        (setf (gethash cache-key *negamax-cache*)
              (if (or (zerop depth) (terminalp node))
                  (* color (evaluate node jogador))
                (labels ((negamax-recursivo (children alpha beta)
                           (if (null children)
                               alpha
                             (progn
```

```
                                (incrementar-nos)
                                (let* ((new-alpha (max alpha (- (negamax (car
children)
                                                                (1- depth)
                                                                (alternar-jogador
jogador)
                                                                (- beta)
                                                                (- alpha)
                                                                (- color))))))
                                  (if (>= new-alpha beta)
                                      (progn
                                        (incrementar-cortes)
                                        beta)
                                    (negamax-recursivo (cdr children) new-alpha
beta)))))))
                    (negamax-recursivo (or (remove nil (sucessores node jogador))
(list node)) alpha beta)))))))
```

3. Tipos Abstratos Utilizados

No presente projeto foi utilizado um tipo abstrato de dados para representar um nó, constituído pelo estado do problema em cada momento, juntamente com a pontuação de cada jogador. Este tipo abstrato possui o seguinte formato: `(((1 2 3 4 5 6)(1 2 3 4 5 6)) (0 0))`.

4. Limitações e Opções Técnicas

A solução desenvolvida pelos alunos possui uma limitação relacionada com memória do programa. Ao fornecer uma profundidade ao algoritmo superior a 4, é atingido o limite de memória stack durante o jogo.

5. Resultados

O algoritmo implementado pelos alunos resolve de forma eficaz e eficiente o jogo proposto. Os cortes realizados são todos do tipo beta tendo em conta que é usado o algoritmo Negamax.

No entanto, poderia existir uma árvore de procura com maior profundidade e, por conseguinte, apresentar melhores resultados, se não existisse a limitação de memória do IDE LispWorks.

O tempo gasto nas jogadas do computador é bastante baixo, devido à utilização de memoização no algoritmo Negamax.

6. Análise Estatística

O algoritmo Negamax desenvolvido pelos alunos, possui as seguintes médias numa execução contra um adversário humano:

- Número de nós analisados - entre 300 e 600
- Número de cortes (beta) - entre 100 a 200
- Tempo gasto - até 0.5 segundos

Para cada jogada, consultar o ficheiro log.dat

7. Conclusão

Ao longo deste projeto, foi possível aplicar na prática os conhecimentos teóricos adquiridos na UC de IA, no que diz respeito aos algoritmos de procura em espaços de estados.

A implementação da solução para o problema Adji-boto em LISP proporcionou uma valiosa experiência de programação e um aprofundamento da compreensão dos conceitos fundamentais de IA. O desenvolvimento deste programa não só permitiu consolidar os conhecimentos sobre os algoritmos de procura, mas também possibilitou a melhoria das competências de resolução de problemas e de pensamento lógico.

Além disso, este projeto evidenciou a importância da eficiência computacional na implementação de soluções para problemas complexos. Os alunos foram confrontados com a necessidade de otimizar os seus algoritmos para lidar com a explosão combinatória dos estados do problema Adji-boto, o que resultou numa compreensão mais profunda das implicações práticas das escolhas algorítmicas.

Em suma, os alunos implementaram com sucesso uma solução eficaz para o problema proposto, aplicando os algoritmos em procura de espaços de estados lecionados na UC de IA.