Resolução de Problema de Decisão usando Programação em Lógica com Restrições: Chess-Num

Pedro Coelho up201806802

Tomás Mendes up201806522

FEUP-PLOG 3MIEIC04 Chess-Num_4

Faculdade de Engenharia da Universidade do Porto,
Rua Dr. Roberto Frias, 4200-465, Porto, Portugal

Abstract. Este projeto foi desenvolvido no âmbito da unidade curricular de Programação Lógica, no Sistema de Desenvolvimento SICStus Prolog. O objetivo é resolver um problema de decisão com implementação de restrições. O problema que selecionamos foi o Chess_Num, que consiste em fornecer ao programa um tabuleiro de xadrez com o número de ataques em cada quadrado e o programa tem como objetivo descobrir como posicionar peças de xadrez de forma a atacar esses quadrados esse número de vezes. Através da linguagem Prolog foi criado um solucionador destes puzzles.

1 Introdução

O projeto foi desenvolvido no âmbito da unidade curricular de Programação em Lógica do 3º ano do curso Mestrado Integrado em Engenharia Informática e de Computação. Foi necessário implementar uma possível resolução para um problema de decisão ou otimização em Prolog, com restrições. O grupo escolheu o problema de decisão Chess_Num.

O problema escolhido consiste na adição de 6 peças de xadrez a um tabuleiro de forma a ser congruente com as restrições de ataques a células impostas por um utilizador. Este artigo tem a seguinte estrutura:

- **Descrição do Problema:** descrição com detalhe o problema de otimização ou decisão em análise, incluindo todas as restrições envolvidas.

- **Abordagem:** descrição da modelação do problema como um PSR / POR:

  Variáveis de Decisão: descrição das variáveis de decisão e respectivos domínios, assim como o seu significado no contexto do problema em análise.

  Restrições: descrição das restrições rígidas e flexíveis do problema e a sua implementação utilizando o SICStus Prolog.

- **Visualização da Solução:** explicação dos predicados que permitem visualizar a solução em modo de texto.

- **Conclusões e Trabalho Futuro:** conclusões retiradas deste projeto, resultados obtidos, vantagens e limitações da solução proposta e aspetos a melhorar.

- **Bibliografia:** fontes bibliográficas usadas, incluindo livros, artigos, páginas Web, entre outros, utilizados para desenvolver o trabalho.

- **Anexo:** ficheiros de dados e resultados, entre outros.

2 Descrição do Problema

Chess_Num é um problema de decisão. Este problema consiste num tabuleiro de xadrez em que são impostas restrições pelo utilizador, as restrições são o número de ataques a cada célula à escolha do tabuleiro. O programa de seguida vai tentar colocar 6 peças de xadrez, a Rainha, a Torre, o Cavalo, o Peão, o Rei e o Bispo no tabuleiro de forma a descobrir qual a disposição que fica congruente com o pedido do utilizador

3 Abordagem ao problema

Para a resolução deste problema utilizamos listas de listas, neste caso, uma matriz 2D em que inicialmente todas as células são zeros. Ao adicionar peças vai se atualizando o número de ataques a cada célula de forma a procurar a configuração correta.

3.1 Variáveis de Decisão

As varáveis de decisão são as coordenadas das peças de xadrez no tabuleiro. Estas variáveis são calculadas pelo nosso programa através das restrições que mencionaremos em baixo

3.2 Restrições

Nos problemas de Chess-Num, duas peças não podem estar na mesma posição, nem podem estar em cima de nenhuma das casas com um número de ataques imposto.

4 Visualização da Solução

O nosso programa permite resolver puzzles Chess-Num e para conseguir visualizar a sua resolução implementámos 3 predicados:

- **printBoard:** Desenha o cimo do quadro e chama o printLines para a impressão do resto do board
- **printLines:** Desenha recursivamente a estrutura do tabuleiro e chama a função printLinesAux que preenche as células de cada linha
- **printLinesAux:** Escreve cada célula

```
printBoard(B):-
    nl,
    write('|-----|-----|-----|-----|-----|-----|-----|-----|\n'),
    printLines(B, 1).

printLines([], 9).
printLines([H|T], X):-
    write('|  '),
    printLinesAux(H),
    nl,
    write('|-----|-----|-----|-----|-----|-----|-----|-----|\n'),
    X1 is X + 1,
    printLines(T, X1).

printLinesAux([]).
printLinesAux([H|T]):-
    write(H),
    write('  |  '),
    printLinesAux(T).
```

5 Resultados

Para avaliar os resultados obtidos fizemos algumas tentativas de tabuleiros e o programa conseguiu resolver. Infelizmente, se o tabuleiro apenas tiver uma solução este programa não a consegue descobrir em tempo útil. Testámos durante 15 minutos sem conseguir obter a solução nestes casos.

6 Conclusões e Trabalho Futuro

Este projeto teve como objetivo aplicar o conhecimento da linguagem Prolog adquirido nas aulas teóricas e práticas da unidade curricular de Programação Lógica, mais concretamente, o módulo de restrições, que se provou útil para a resolução de problemas de decisão e otimização.

Tivemos algumas dificuldades no desenvolvimento do projeto devido á dimensão do tabuleiro de xadrez, ao número de possíveis distribuições de peças e á complexidade do problema em si. No entanto foi um desafio interessante e um bom progresso de aprendizagem numa linguagem que nos era desconhecida previamente a este semestre.

Poder-se-ia melhorar certos aspetos do programa como a eficiência temporal e espacial mas em geral o projeto foi concluído com sucesso parcial dado que consegue resolver corretamente os problemas com algumas soluções, mas não os problemas com apenas uma solução.