

Trabalho de Programação Lógica

Inteligência Artificial – Mestrado Profissional em Computação Aplicada

Prof. Dr. Jefferson O. Andrade

Instituto Federal do Espírito Santo (Ifes) – Campus Serra

2022/1

1 Introdução

O objetivo deste trabalho é praticar a modelagem lógica de problemas e a implementação de soluções para os mesmos utilizando programação declarativa em Prolog. Para este trabalho, o código pode ser desenvolvido obrigatoriamente na linguagem Prolog. Mais especificamente no sistema SWI Prolog.

2 Problemas

O trabalho consistirá de quatro problemas a serem resolvidos individualmente.

2.1 Corrida

A recente corrida anual de veículos experimentais do PPComp acaba de acontecer. Os juízes desistiram de acompanhar quem ganhou, já que os resultados se perdem a cada ano. Usando as seguintes anotações dos espectadores, você pode determinar quem terminou em que colocação?

- *Helena Hypólito* terminou depois de *Fábio Freitas*, mas antes de *Bruna Barros*, de *Quincas Quaresma*, de *Eduardo Eugêncio* e de *Nauvia Novais*.
- *Tadeu Torres* terminou antes de *Renato Ramos* e de *Carla Correia*, mas depois de *Davi Dantas* e de *Kaio Kiefer*.
- *Gina Gomes* terminou antes de *Miguel Moraes*, de *Paula Prado* e de *Úrsula Uliana*, mas depois de *Fábio Freitas*, de *Nauvia Novais* e de *Tadeu Torres*.
- *Ivan Ignácio* terminou depois de *Fábio Freitas*, mas antes de *Joana Justo*, de *Yves Young*, de *Kaio Keifer* e de *Luana Lessa*.
- *Zaíra Zaia* terminou antes de *André Alves*, mas depois de *Renato Ramos* e de *Walter Watanabe*.
- *Bruna Barros* terminou depois de *Yves Young*, de *Eduardo Eugenio* e de *Paula Prado*, mas antes de *André Alves* e de *Zaíra Zaia*.

- *Davi Dantas* terminou antes de *Xena Xavier*, de *Joana Justo*, de *Renato Ramos* e de *Helena Hypólito*, mas depois de *Simone Souza*.
- *Valdo Villares* terminou antes de *Úrsula Uliana* e de *Otto Orelíio*, mas depois de *Walter Watanabe*, de *Kaio Kiefer*, de *Quincas Quaresma* e de *Paula Prado*.
- *Xena Xavier* terminou depois de *Kaio Keifer*, mas antes de *Renato Ramos*, de *Joana Justo* e de *Carla Correa*.
- *Luana Lessa* terminou depois de *Quincas Quaresma* e de *Nauvia Novai*, mas antes de *Walter Watanabe* e de *Otto Orelíio*.
- *Úrsula Uliana* terminou antes de *Otto Orelíio* e de *Zaíra Zaia*, mas depois de *Miguel Moraes* e de *Carla Correia*.

Escreva um programa em Prolog para resolver o quebra-cabeça lógico da corrida. Em particular, defina um predicado, `race(Ranking)`, que seja verdadeiro sse a lista, `Ranking`, contiver os competidores da corrida em uma ordem de chegada que não contradiga nenhum fato que foi dado acima.

Por exemplo, se o único fato fosse “Fulano chegou antes de Siclano, mas depois de Beltrano e Ziclano.”, seriam aceitáveis os seguintes resultados:

- `Ranking = [beltrano, ziclano, fulano, siclano]`
- `Ranking = [ziclano, feltrano, fulano, siclano]`

2.2 Labirinto

Você deve escrever um programa simples em Prolog para encontrar um caminho através de um labirinto. Sua entrada consistirá em fatos do formato `pway(X,Y,N)`, indicando que a interseção X está ligada à interseção Y por uma passagem de N metros.

Você deve escrever o predicado `solve(X, Y, P, N)`, onde:

X – é a interseção de origem.

Y – é a interseção de destino.

P – é o caminho (lista de interseções) que leva de X a Y.

N – é o custo total do caminho P.

A intenção é que o usuário invoque `solve` como uma consulta, especificando X e Y como constantes e P e N como variáveis.

Considere o seguinte exemplo concreto:

```

1 pway(a, b, 10).
2 pway(b, c, 15).
3 pway(d, c, 5).
4 pway(d, b, 10).
```

Para estes fatos, se for entrada a consulta:

```
?- solve(a, d, P, N).
```

O interpretador Prolog deverá responder:

```
P = [a, b, c, d]
N = 30
```

E, se outra resposta for solicitada, responderá:

```
P = [a, b, d]
N = 20
```

Que são dos dois únicos caminhos disponíveis.

Você deve escrever a sua solução, i.e., a implementação do predicado `solve(X, Y, P, N)`, em um arquivo contendo apenas ela, sem as definições do labirinto. As definições do labirinto estarão em um arquivo separado (é possível fazer `consult` de múltiplos arquivos no Prolog).

2.3 Quebra-cabeças

Considere o seguinte quebra-cabeças:

Primeiramente, desenhe um tabuleiro de xadrez, i.e., um “quadro” com 8 linhas e 8 colunas. Agora numere cada linha de 1 a 8, de cima para baixo, e numere cada coluna de 1 a 8 da esquerda para a direita. **Seu objetivo** é colocar um número inteiro em cada uma das 64 casas do tabuleiro seguindo as regras abaixo:

1. Não pode haver duas ou mais linhas iguais.
2. Cada linha deve ser igual a uma coluna (transposta), mas não pode ser igual à coluna que tem o mesmo número da linhas. Por exemplo, a linha 2 não pode ser igual à coluna 2 (transposta).
3. Se o maior número que você escrever no tabuleiro for N então você também tem que escrever os números $1, 2, \dots, N - 1$.

Considerando o quebra-cabeças proposto, resolva os itens abaixo.

- (a) Escreva um predicado `puzzle(Board)` em Prolog que seja verdadeiro sse `Board` for uma lista de listas (representando o tabuleiro) com uma solução do problema. Ou seja, `Board` deverá ser uma lista contendo 8 listas (as linhas do tabuleiro) com 8 números inteiros cada.
- (b) A soma dos sessenta e quatro números do tabuleiro é chamada de **total** do tabuleiro. Qual é o maior valor de **total** que você consegue obter para este quebra-cabeças?

3 Instruções

- O trabalho é de realização individual. Os estudantes podem debater ideias entre si, mas a elaboração do código deve ser um trabalho original de cada estudante.
- O trabalho deve ser entregue **exclusivamente** através do sistema AVA do Ifes na forma de um arquivo compactado em formato ZIP ou 7z na tarefa corresponde indicada na página da disciplina.

ATENÇÃO: Apenas os formatos de compactação ZIP e 7z serão aceitos. Trabalhos entregue em outros formato de compactação tais como RAR ou LHA serão **anulados**.

- O arquivo compactado entregue deve conter um diretório (pasta) com todo o código fonte desenvolvido para o trabalho. A estrutura do deve ser a seguinte:

```
tlog
├── Readme.md
├── source files
└── ...
```

O arquivo `Readme.md` é um arquivo em formato Markdown contendo ao menos: (a) o nome do autor do trabalho; (b) a descrição geral dos arquivos contidos no trabalho; (c) e os nomes de cada um dos predicados desenvolvidos.

- A distribuição de pontos no trabalho se dará da seguinte forma:

1. Maratona	30 pontos
2. Labirinto	30 pontos
3. Quebra-cabeça	40 pontos