UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia

Departamento de Computação

Inteligência Artificial

Atividade Tópico 6 - Algoritmo de Encadeamento para Trás em Prolog

Professora: Heloisa de Arruda Camargo

Autores

Leticia Bossatto Marchezi, 791003 - Ciência da Computação

Luís Augusto Simas do Nascimento, 790828 - Ciência da Computação

São Carlos, 12 de Março de 2022

1. Introdução

Os algoritmos de inferência são dispositivos da lógica usados para alcançar conclusões a partir de dados conhecidos por um sistema. As regras, os fatos e as consultas são as cláusulas que possibilitam a construção e verificação de novos conhecimentos. Tais conceitos são aplicados na linguagem de programação Prolog, uma linguagem do paradigma de Lógica Matemática.

No desenvolvimento do trabalho foi aplicado o algoritmo de encadeamento para trás com busca em profundidade e programação em Prolog usando SWI-Prolog.

2. Base de conhecimento

Remoção de quantificadores para transformação de sentenças em cláusulas de primeira ordem:

```
Preparo_fisico(Ruim) → Problema(Preparador_fisico)
Atritos(Constantes) ∧ Situação_psicológica(Ruim) → Problema
(Equipe_tecnica)
Preparo_fisico(Bom) ∧ Situação_de_gols(Ruim) →
Problema(Time)
Atritos(Constantes) ∧ Salarios(Atrasado) → Problema
(Insatisfação_financeira)
Jogador(x) ∧ Tecnico(y) ∧ Discute (x, y) →
Atritos(Constantes)
Jogador(x) ∧ Jogador(y) ∧ Discute (x, y) →
Atritos(Constantes)
Jogador(x) ∧ Suspenso(x) → Situação_psicologica(Ruim)
Jogador(x) ∧ Cortado(x) → Situação_psicologica(Ruim)
Gols_sofridos(x) ∧ Gols_feitos(y) ∧ Maior(x,y) →
Situação_de_gols(Ruim)
Cartão_vermelho(x) → Suspenso(x)
Jogador(x) ∧ Lento(x) → Preparo_fisico(Ruim)
Jogador(x) ∧ Lesão(x) → Preparo_fisico(Ruim)
```

Situação 2:

A remoção dos quantificadores existenciais originam novas variáveis, como o jogador Vanderson, Gabriel e Rafael.

```
jogador(lucas).
lento(lucas).
jogador(vanderson).
cartão_vermelho(vanderson).
jogador(gabriel).
jogador(rafael).
discute(gabriel, rafael).
gols_sofridos(3).
gols_feitos(5).
```

3. Aplicação do algoritmo

O algoritmo de encadeamento para trás com busca em profundida é iniciado pela conclusão, ou seja, pelo objetivo da busca. Assim, como o é desejado identificar quais são os problemas do time, a busca unifica os valores possíveis de x, como Preparador_fisico, Equipe_tecnica, Time e Insatisfação_financeira.

Foram construídas 4 árvores para a situação 2, uma para cada caso, com o intuito de permitir uma visualização adequada no documento, porém todas fazem parte de uma mesma árvore.

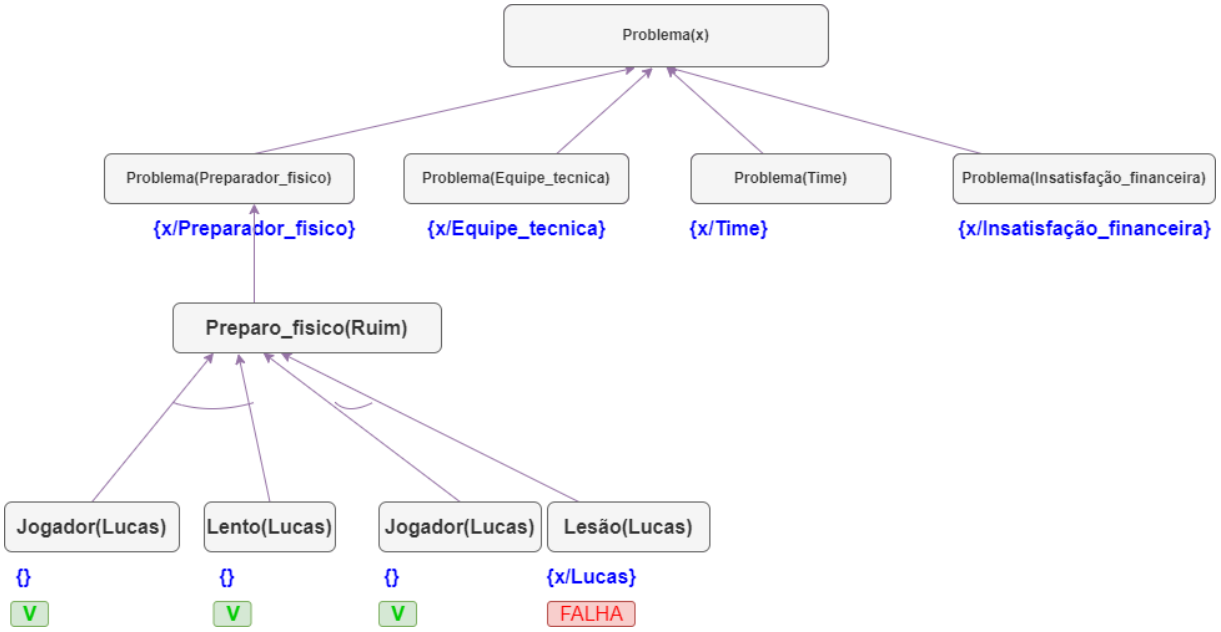
3.1. Preparador físico

O problema Preparador_físico pode ser concluído usando 2 regras:

$$\text{Jogador}(x) \land \text{Lento}(x) \rightarrow \text{Preparo\_fisico}(\text{Ruim})$$

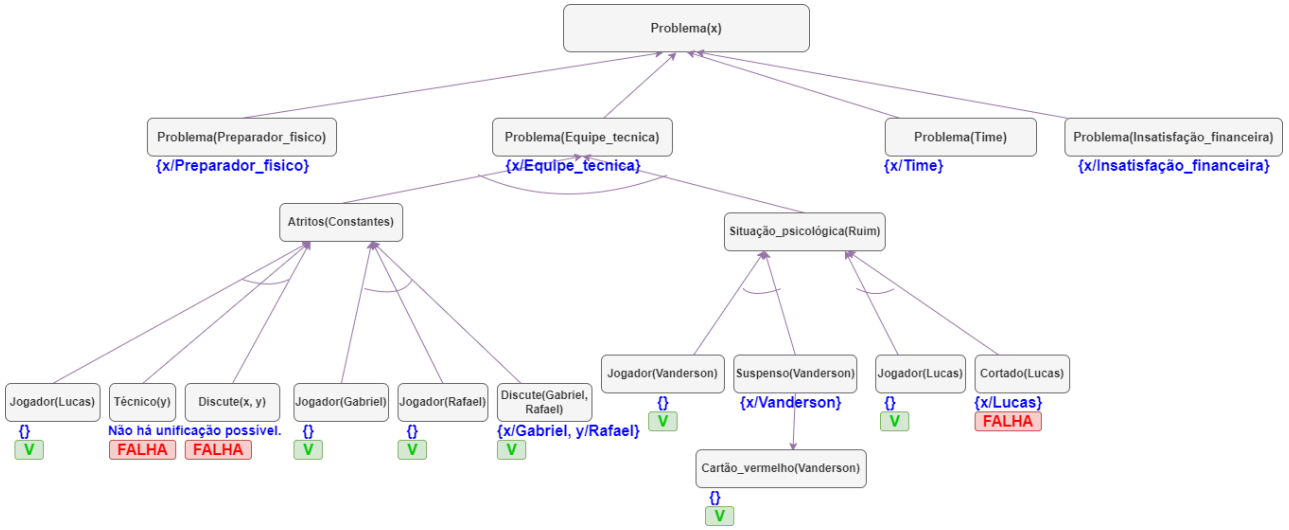
$$\text{Jogador}(x) \land \text{Lesão}(x) \rightarrow \text{Preparo\_fisico}(\text{Ruim})$$

Ou seja, caso alguma das regras(ou ambas) sejam satisfeitas, um problema do time será o preparador físico. Na árvore ocorre a unificação {x/Lucas}, em que Lucas é um jogador e é lento, logo preparo físico é ruim, e por consequência preparador_fisico é um problema.



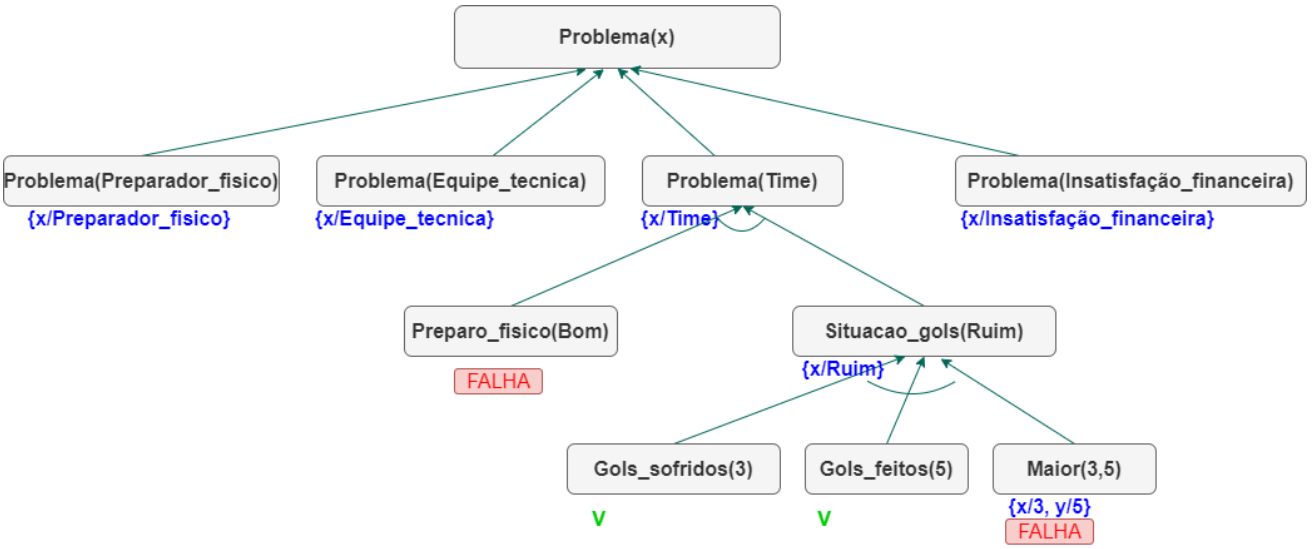
3.2. Equipe técnica

Já para o problema Equipe_tecnica há uma conjunção entre as expressões Atritos(Constantes) e Situação_psicológica(Ruim). A primeira cláusula é satisfeita ao unificar {x/Gabriel, y/Rafael}. A segunda se torna verdade para {x/Vanderson}, jogador que recebeu um cartão vermelho, logo foi suspenso. Então conclui-se que equipe técnica é um problema.



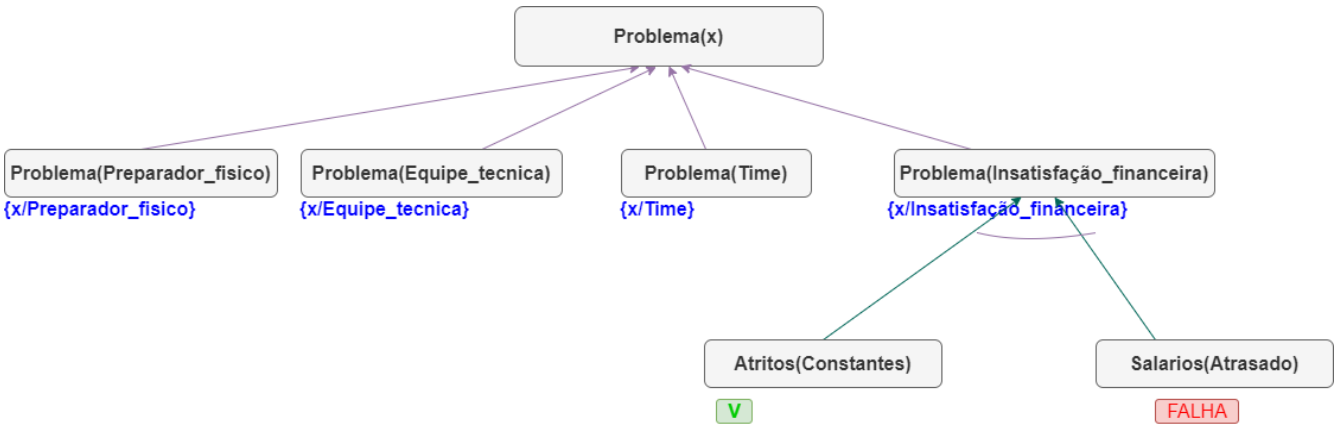
3.3. Time

Para o problema de Time, o preparo físico do time deve ser bom, entretanto tal fato não foi adicionado à base de dados. A situação de gols também não é satisfeita, pois foram feitos mais gols do que sofreram.



3.4. Insatisfação financeira

Como provado anteriormente no caso Equipe_técnica, Atritos(Constantes) é uma conclusão da base de dados. Entretanto, Salarios(Atrasado) não faz parte da base de dados, logo não pode-se concluir que insatisfação financeira é um problema.



4. Implementação em Prolog

As sentenças da base de conhecimento foram implementadas em um programa Prolog na forma das seguintes regras:

```
maior(X, Y) :- X > Y.
problema(preparador_fisico) :- preparo_fisico(ruim).
problema(equipe_tecnica) :- atritos(constantes),
situação_psicológica(ruim).
problema(time) :- preparo_fisico(bom), situação_de_gols(ruim).
problema(insatisfação_financeira) :- atritos(constantes),
salarios(atrasado).
atritos(constantes) :- jogador(X), tecnico(Y), discute(X, Y).
atritos(constantes) :- jogador(X), jogador(Y), discute(X, Y).
situação_psicológica(ruim) :- jogador(X), suspenso(X).
situação_psicológica(ruim) :- jogador(X), cortado(X).
situação_de_gols(ruim) :- gols_sofridos(X), gols_feitos(Y),
maior(X,Y).
suspenso(X) :- cartão_vermelho(X).
preparo_fisico(ruim) :- jogador(X), lento(X).
preparo_fisico(ruim) :- jogador(X), lesão(X).
```

Além da definição da base, é necessário utilizar a diretiva `dynamic` para declarar que alguns dos predicados são definidos dinamicamente em tempo de execução. Isso permite a execução de consultas em situações iniciais nas quais alguns dos predicados não são definidos sem que sejam geradas exceções no processo de unificação.

```
:- dynamic jogador/1, tecnico/1, discute/2, atritos/1,
salarios/1, suspenso/1, cortado/1, gols_sofridos/1,
gols_feitos/1, lento/1, lesão/1, cartao_vermelho/1.
```

Com isso, é possível definir cada situação inicial através de fatos adicionados à base de conhecimento e verificar o resultado das consultas.

4.1. Situação 1

A primeira situação inicial foi definida através dos seguintes fatos:

```
jogador(jorge).
lesão(jorge).
gols_sofridos(2).
gols_feitos(1).
preparo_fisico(bom).
```

Dada essa situação inicial, o resultado da consulta pode ser verificado a seguir:

```
?- problema(X).
X = preparador_fisico;
X = time;
false.
```

Com isso, observa-se que o algoritmo foi capaz de encontrar dois problemas para a consulta.

4.2. Situação 2

A segunda situação foi definida através dos seguintes fatos:

```
jogador(joaquim).
suspenso(joaquim).
tecnico(zico).
jogador(nilson).
discute(nilson, zico).
preparo_fisico(bom).
salarios(atrasado).
```

Dada essa situação inicial, o resultado da consulta pode ser verificado a seguir:

```
?- problema(X).
X = equipe_tecnica;
X = 'insatisfação_financeira';
false.
```

Com isso, observa-se que o algoritmo foi capaz de encontrar outros dois problemas para a nova consulta.

4.3. Situação 3

A terceira situação foi definida através dos seguintes fatos:

```
gols_feitos(7).
gols_sofridos(2).
jogador(romario).
jogador(ronaldo).
tecnico(felipao).
cartão_vermelho(romario).
```

Dada essa situação inicial, o resultado da consulta pode ser verificado a seguir:

```
?- problema(X).
false.
```

Dessa vez observa-se que a consulta produziu apenas o resultado `false`, ou seja, nenhum problema foi identificado.

4.4. Situação 4

A quarta e última situação foi definida através dos seguintes fatos:

```
jogador(carlos).
cartão_vermelho(carlos).
```

```
tecnico(eduardo).
discute(carlos, eduardo).
gols_feitos(7).
gols_sofridos(2).
preparo_fisico(bom).
```

Dada essa situação inicial, o resultado da consulta pode ser verificado a seguir:

```
?- problema(X).
X = equipe_tecnica;
false.
```

Observa-se que a consulta resulta em apenas uma unificação possível, ou seja, apenas um problema foi detectado.

5. Conclusão

Dessa forma, foi possível observar que o algoritmo de encadeamento para trás com busca em profundidade, tanto na execução manual quanto na implementação em Prolog, é capaz de fornecer inferências a partir de uma base de conhecimento aliada ao conhecimento de um conjunto de fatos específicos.