Universidade do Minho

Mestrado Integrado em Engenharia Informática

Sistemas de Representação de Conhecimento e Raciocínio

Trabalho 1 - 1ª Fase

Grupo 34

Celso Rodrigues	Daniela Fernandes	Eduardo Araújo	José Gomes
A83655	A73768	A86012	A82418

9 de abril de 2021

Resumo

Este documento diz respeito à primeira fase do Trabalho Prático em Grupo da unidade curricular de Sistemas de Representação de Conhecimento e Raciocínio da Universidade do Minho.

Esta fase tem dois objetivos, utilização da linguagem de programação lógica em PROLOG e o recurso a invariantes. Ao longo do presente relatório serão apresentadas as formas de conhecimento adotadas e as características e funcionalidades do sistema, assim como serão descritas as estratégias utilizadas para a concretização das mesmas.

De uma perspectiva geral, consideramos que a realização deste projeto foi relativamente bem sucedida, visto que pensamos ter cumprido com todos os requisitos mínimos propostos.

Conteúdo

1 Introdução

Para a unidade curricular de Sistemas de Representação de Conhecimento e Raciocínio foi-nos proposto, como primeira fase do Trabalho Prático de grupo, o desenvolvimento de um sistema com capacidade para caracterizar um universo de discurso, na área da vacinação da Covid-19.

O sistema de representação de conhecimento e raciocínio a realizar deverá representar o conhecimento relativo pelo menos a utentes, staff, centro_saúde e vacinação_covid. Para além disto, deverá respeitar as necessidades de demonstração de variadas funcionalidades, no entanto, é indispensável o cumprimentos das seguintes:

- Definição de fases de vacinação, definindo critérios de inclusão de utentes nas diferentes fases;
- Identificar pessoas não vacinadas;
- Identificar pessoas vacinadas;
- Identificar pessoas vacinadas indevidamente;
- Identificar pessoas não vacinadas e que são candidatas a vacinação;
- Identificar pessoas a quem falta a segunda toma da vacina;
- Desenvolver um sistema de inferência capaz de implemetar os mecanismos inerentes a estes sistemas.

Os dois objetivos principais desta fase são nomeadamente a utilização da linguagem da programação em lógica PROLOG e o recursos a Invariantes.

2 Preliminares

Tal como dito na secção anterior, este trabalho vai ser realizado utilizando a linguagem de programação PROLOG, que é uma linguagem de progamação lógica. Um programa PROLOG consiste num conjunto de axiomas e regras de inferência que descrevem um problema. Para isto, é necessário adquirir conhecimentos na área da Lógica Matemática.

Começamos por perceber a distinção entres *Teoria dos Modelos* e *Teoria da Prova*. O primeiro examina a interpretação de relações lógicas através da associação de valores de verdade. Já, o segundo examina a derivabilidade a partir de outras fórmulas.

Outra matéria importante são as *Cláusulas de Horn*, que permitem representar os principais constituintes de um programa PROLOG : os **factos**, as **regras** e as **questões**. Enquanto os factos e as regras compõem a base de conhecimento do sistema, as questões representam as consultas à base de conhecimento que são respondidas por dedução lógica, recorrendo a um *Algoritmo de Resolução*.

Aplicando o *Algoritmo de Resolução* o motor de inferência permite que as questões que são colocados ao PROLOG sejam respondidas. Para a compreensão deste algoritmo temos que ter conhecimento no *Modus Ponens* e no *Modus Tollens* da Lógica Proposicional.

Para além de tudo o que já foi mencionado em cima, temos que saber os conceitos de *Negação por falha* na prova e do comando *cut*. Relativamente à procura de múltiplas soluções, é fundamental o entendimento do mecanismo de *backtracking*.

Vamos também utilizar uma característica PROLOG, a notação sintática *pred/2*. para representar um predicado e o seu número de argumentos, deste modo, o predicado *pred* tem 2 argumentos.

Para podermos testar o programa é necessário recorrer a interpretador que vai ser *SICStus Prolog*.

Por último, a utilização de *Invariantes* também vai ser abordada neste trabalho, pois têm um papel crucial no que toca à inserção e remoção de reconhecimento, permitindo garantir a coerência da informação contida na base de conhecimento, tanto a nível estrutura como referencial.

3 Descrição do Trabalho e Análise de Resultados

Finalmente será apresentado o trabalho realizado através da descrição do sistema, da construção da extensão de predicados, e até mesmo da aplicação de invariantes estruturais e referenciais. Tanto para os predicados como para os invariantes construídos, vão ser apresentador valores experimentais para comprovar a veracidade dos resultados obtidos.

Note-se que para a construção de alguma extensões de predicados e para a aplicação de invariantes serão utilizados os seguintes predicados:

- insere/1 que insere um termo, caso corra tudo bem, na base de conhecimento;
- remove/1 que remove um termo, caso corra tudo bem, na base de conhecimento;
- teste/1 que testa a veracidade de uma lista;
- solucoes/3 que calcula a lista de termos que atendem a uma questão;
- solucoesSRep/3 que calcula a lista de termos que atendem a uma questão, sem elementos repetidos;
- comprimento/1 que calcula o comprimento de uma lista.

3.1 Descrição do Sistema

O sistema também deverá representar pelo menos a informação relativa a *utente*, *staff*, *centro_saude* e *vacinacao_covid*. No entanto, decidiu estender o conhecimento do sistema, sendo o mesmo representado da seguinte forma:

- *utente: idUtente, nº segurança social, nome, data_nasc, email, telefone, morada, profissão, [doencas_cron], centro_saude -> V,F*
- *staff: idStaff, idCentro, Nome, email -> V,F*
- *centro_saude: idCentro, nome, morada, telefone, email -> V,F*
- *vacinacao_covid: staff, utente, data, vacina, toma -> V,F*

Além das funcionalidades básicas, escolhemos implementar funcionalidades adicionais, sendo que o sistema suportará qualquer um dos requisitos seguintes:

- Definição de fases de vacinação, definindo critérios de inclusão de utentes nas diferentes fases;
- Identificar pessoas não vacinadas;

- Identificar pessoas vacinadas;
- Identificar pessoas vacinadas indevidamente;
- Identificar pessoas não vacinadas e que são candidatas a vacinação;
- Identificar pessoas a quem falta a segunda toma da vacina;
- Desenvolver um sistema de inferência capaz de implemetar os mecanismos inerentes a estes sistemas.
- Identificar todos os staffs que administraram a vacina
- Listar as vacinas dadas por um dado *staff*
- Alterar o centro de saúde de um utente
- Listar todos os utentes
- Listar todos os centros de saúde
- Listar todos os *staff*
- Listas todas as ocorrências em vacinacao_covid
- Alterar o número de toma
- Alterar a morada de um utente

Para não se comprementer a veracidade nem a coerência do conhecimento já existente na base de conhecimento, precisamos de ter em conta:

- Não podem haver utentes com Id's iguais, o mesmo acontece para o *staff* e para os centros de saúde
- Todos os utentes, staff's e centros de saúde têm que se encontrar na base de conhecimento
- Não é permitido a remoção de um utente ou staff se estes estiver em vacinação covid
- Não é permitido inserir nenhum utente ou staff em vacinação covid se estes não estiverem na base de conhecimento

3.2 Predicados

Nesta subsecção vamos explicitar e explicar a construção dos diferentes predicados que caracterizam a base de conhecimento.

3.2.1 utente/10

- **Aquisição de conhecimentos**

O predicado *utente: idUtente, nº segurança social, nome, data _ nasc, email, telefone, morada, profissão, [doencas cron], centro saude - > V,F* tem como finalidade caracterizar um utente existente na base de conhecimento, através do seu número de utente, o número de segurança social, o nome, a data de nascimento, o email, o telefone, a morada, profissão, doenças crónicas se tiver e o centro de saúde onde está a ser seguido.

- **Apresentação dos conhecimentos adquiridos**

Para introduzir os factos existentes na base de conhecimento, utilizamos a seguinte formulação.

```
utente(1, 123477, eduardo, date(1988,01,30), 'eduardodf@gorj.pt', 932827487,
    braga, policia, [diabetes,hipertensao], centro_saude_gualtar).
utente(2, 623426, fernanda, date(1997,06,05), 'fernandinha@hotmail.com'
    ,918826356, braga, medico, [cancro], centro_saude_caranda).
utente(3, 632427, sofia, date(1972,10,02), 'sofsieh@gmail.com', 92937484, fafe
    , enfermeiro, [], centro_saude_fafe).
utente(4, 928323, marco,  date(2000,08,26), 'marcolol@outlook.pt', 918362758,
    vila_real, estudante, [], centro_saude_vilareal).
utente(5, 627848, alice, date(1968,12,26), 'alice1968@msn.pt', 912138366,
    vila_verde, professor, [diabetes], centro_saude_vilaverde ).
utente(6, 974837, carina, date(1980,10,04), 'carinaporto@hotmail.pt',
    913738633, esposende, lojista, [coracao], centro_saude_esposende).
utente(7, 353347, jose, date(1977,04,29), 'josejosejose@outlook.pt',
    917623527, vila_do_conde, gerente,[], centro_saude_viladoconde).
```

3.2.2 centro_saude/5

- **Aquisição de conhecimentos**

O predicado *centro _saude: idCentro, nome, morada, telefone, email - > V,F* tem como finalidade caracterizar o centro de saúde a que os utentes recorrem existente na base de conhecimento, atavés do seu número de identificação, nome, morada, telefone e email.

- **Apresentação dos conhecimentos adquiridos**

Para introduzir os factos existentes na base de conhecimento, utilizamos a seguinte formulação.

```
centro_saude(1, centro_saude_gualtar, braga, 253764648, 'gualtar@csb.pt').
centro_saude(2, centro_saude_caranda, braga, 253847495, 'caranda@csb.pt').
centro_saude(3, centro_saude_fafe, fafe, 252746433, 'fafe@csf.pt').
centro_saude(4, centro_saude_vilareal, vila_real, 251763389, 'vilareal@csvr.pt').
centro_saude(5, centro_saude_esposende, esposende, 253713748, 'esposende@cse.pt').
centro_saude(6, centro_saude_viladoconde, vila_do_conde, 254635327, 'viladoconde@csvdc.pt').
centro_saude(7, centro_saude_vilaverde, vila_verde, 254635327, 'vilaverde@csvv.pt').
```

3.2.3 staff/4

- **Aquisição de conhecimentos**

O predicado *staff: idStaff, idCentro, Nome, email -> V,F* tem como finalidade caracterizar o *staff* que trabalha num dado centro de saúde e adminitra as vacinas do covid existente na base de conhecimento, através do ser número de identificação, o número de identificação do centro de saúde, o seu nome e o email profissional.

- **Apresentação dos conhecimentos adquiridos**

Para introduzir os factos existentes na base de conhecimento, utilizamos a seguinte formulação.

```
staff(1,4, carla, 'carla_vilareal@csvr.pt').
staff(2,5, helena, 'helena_esposende@cse.pt').
staff(3,3, artur, 'artur_fafe@csf.pt' ).
staff(4,1, simao, 'simao_gualtar@csb.pt').
staff(5,2, elisabete, 'elisabete_caranda@csb.pt').
staff(6,7, carmo, 'carmo_vilaverde@csvv.pt').
```

3.2.4 vacinacao_covid/5

- **Aquisição de conhecimentos**

O predicado *vacinacao_covid: staff, utente, data, vacina, toma -> V,F* tem como finalidade caracterizar a vacinação contra a covid existente na base de conhecimento, através do nome do staff, do nome do utente, da data em que é adminitrada, o nome da vacina e qual o número de toma.

- **Apresentação dos conhecimentos adquiridos**

Para introduzir os factos existentes na base de conhecimento, utilizamos a seguinte formulação.

```
vacinacao_covid(4, 1, '10-03-2021', 'Pfizer', 1).
vacinacao_covid(5, 2, '28-12-2020', 'Moderna',2).
vacinacao_covid(3, 3, '05-04-2021', 'AstraZeneca',1).
vacinacao_covid(1, 4, '05-06-2021', 'Moderna', 0).
vacinacao_covid(6, 5, '10-05-2021', 'AstraZeneca',0).
vacinacao_covid(2, 6, '06-02-2021', 'Pfizer',2).
```

3.2.5 evolucao/1

- **Aquisição de conhecimentos**

O predicado *evolucao:Termo-> V,F* tem como objetivo adicionar informação à base de conhecimento, mais concretamentem registar utentes, *staff*, centros de saúde e vacinação covid, respeitando sempre os invariantes de inserção que vão ser referidos mais à frente.

A extensão deste predicado encontra-se de seguida:

```
evolucao(T) :- solucoes(I, +T :: I, S),
               insere(T),
               teste(S).
```

- **Apresentação dos conhecimentos adquiridos**

Ao tentar inserir na base de conhecimento um utente com número de identicação 2, que já se encontra atribuído a outro utente (invariante estrutural), não será possível.

Tal como seria de esperar, o novo utente com o número de identificação 2 não foi inserido na base de conhecimento, porque iria desrespeitar um invariante de inserção.

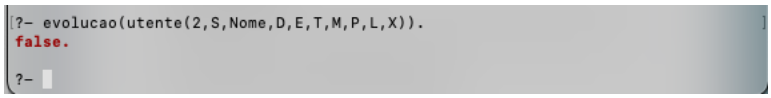

Figura 1: Resultados obtidos para o predicado evolucao/1

3.2.6 involucao/1

- **Aquisição de conhecimentos**

O predicado *involucao:Termo-> V,F* tem como objetivo retirar informação da base de conhecimento, mais concretamente, remover utentes, centros de saúde, *staff* e vacinaçao covid, respeitando os invariantes de remoção que vão ser referidos na próxima secção.

A extensão deste predicado encontra-se de seguida:

```
1 involucao(T) :- solucoes(I, -T::I, S),
                  remove(T),
3                 teste(S).
```

- **Apresentação dos conhecimentos adquiridos**

Ao tentar remover da base de conhecimento o utente com o número de identificação .., que está associada à vacinação covid (invariante referencial) tal não será possível.

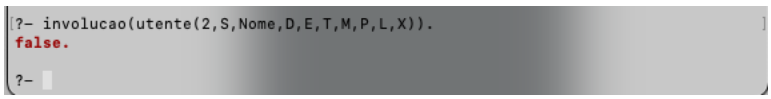

Figura 2: Resultados obtidos para o predicado involucao/1

Tal como seria de esperar, o utente com o número de identificação não foi removido porque iria desrespeitar um invariante de remoção.

3.2.7 fasesvacinacao/4

- **Aquisição de conhecimentos**

O predicado *fasesvacinacao:Criterio,Operador,Valor,Resultado -> V,F* tem a finalidade de identificar as fases de vacinação segundo um critério de seleção.

A extensão de fasesvacinacao/4 é o seguinte:

```
% Extensão do predicado que identifica utentes pela sua data de Nascimento
fasesvacinacao(1,<,V,R) :- solucoes((ID,SS,NM,D,E,T,M,P,L,CS), (utente(ID,SS,
    NM,D,E,T,M,P,L,CS), D < V), R).

% Extensão do predicado que identifica utentes pela sua profissão

fasesvacinacao(2,=,V,R) :- solucoes((ID,SS,NM,D,E,T,M,P,L,CS), (utente(ID,SS,
    NM,D,E,T,M,P,L,CS)), R).

% Extensão do predicado que identifica utentes pelas doenças crónicas

fasesvacinacao(3,=,V,R) :- solucoes((ID,SS,NM,D,E,T,M,P,L,CS), (utente(ID,SS,
    NM,D,E,T,M,P,L,CS)), R).

% Extensão da data de vacinação

fasesvacinacao(4,=,V,R) :-
  solucoesSRep((ID,SS,NM,D,E,T,M,P,L,CS), IDS^IDU^DT^V^T^(vacinacao_covid(IDS,
    IDU,DT,V,T), utente(ID,SS,NM,D,E,T,M,P,L,CS)), R).
```

Este predicado recebe como argumento um dos critérios:

1. Data de nascimento do utente
2. Profissão do utente
3. Doenças crónicas do utente
4. Data de vacinação

Para além disso o critério será comparado com um dado valor, segundo um operado (<, =).

3.2.8 naovacinadas/1

- **Aquisição de conhecimentos**

O predicado *naovacinados: IDu ,Resultado -> V,F* pretente identificar os utentes que não foram vacinados. Vai utilizar a negação no predicado vacinadas.

A extensão naovacinadas/2 é a seguinte:

```
naovacinados(R) :- solucoes(utente(ID, SS, NM, D, E, TLF, M, P, L, CS), (
    utente(ID, SS, NM, D, E, TLF, M, P, L, CS), nao(vacinacao_covid(_, ID, _,
    _, _))), R).
```

- **Apresentação dos conhecimentos adquiridos**

Ao utilizar identificar um utente, vamos obter a lista dos utentes que ainda não foram vacinamos.

```
?- naovacinados(R).
R = [utente(7, 353347, jose, date(1977, 4, 29), 'josejosejose@outlook.pt', 917623527, vila_do_conde, gerente, [
], centro_saude_viladoconde)].
```

Figura 3: Resultados obtidos para o predicado naovacinadas/1

Como podemos ver pela a imagem a cima, temos a lista de todos os utentes que ainda não tomaram a vacina, que neste caso é apenas o *jose*.

3.2.9 vacinadas/2

- **Aquisição de conhecimentos**

O predicado *vacinadas: Toma,Resultado -> V,F* pretende listar todos os utentes que foram vacinados.

A extensão vacinadas/2 é a seguinte:

```
vacinadas(T, R) :- solucoes(utente(ID, SS, NM, D, E, TLF, M, P, L, CS), (
    utente(ID, SS, NM, D, E, TLF, M, P, L, CS), vacinacao_covid(_, ID, _, _, T
    )), R).
```

- **Apresentação dos conhecimentos adquiridos**

Ao identificar o número de tomas é dada uma lista com todos os utentes que tomaram esse número de vacinas.

```
?- vacinadas(1, R).
R = [utente(1, 123477, eduardo, date(1988, 1, 30), 'eduardodf@gorj.pt', 932827487, braga, policia, [diabetes, h
ipertensao], centro_saude_gualtar), utente(3, 632427, sofia, date(1972, 10, 2), 'sofsieh@gmail.com', 92937484,
fafe, enfermeiro, [], centro_saude_fafe)].

?- vacinadas(2, R).
R = [utente(2, 623426, fernanda, date(1997, 6, 5), 'fernandinha@hotmail.com', 918826356, braga, medico, [cancro
], centro_saude_caranda), utente(6, 974837, carina, date(1980, 10, 4), 'carinaporto@hotmail.pt', 913738633, esp
osende, lojista, [coracao], centro_saude_esposende)].
```

Figura 4: Resultados obtidos para o predicado vacinadas/2

Como podemos verificar pela imagem em cima, são apresentadas duas listas para a toma = 1 e a toma = 2.

3.2.10 vacinadasInd/1

- **Aquisição de conhecimentos**

O predicado *vacinadasInd: Resultado -> V,F* pretende listar todas as pessoas que foram vacinadas indevidamente, isto é, pessoas que não têm profissões de risco ou doenças crónicas.

A extensão vacinadasInd/1 é a seguinte:

```
profissaoRisco(medico).
profissaoRisco(policia).
profissaoRisco(enfermeiro).
profissaoRisco(bombeiro).
profissaoRisco(auxiliar_saude).

vacinadasInd(R) :- solucoes((ID,SS,NM,D,E,T,M,P,[],CS), (utente(ID,SS,NM,D,E,T
    ,M,P,[],CS), vacinacao_covid(_, ID, _, _, _), nao(profissaoRisco(P))), R).
```

- **Apresentação dos conhecimentos adquiridos**

Ao identificar as pessoas que foram vacinadas indevidamente, **vacinadasInd(R)**, deveríamos obter uma lista de todas as pessoas que foram vacinadas sem terem condições para tal .

```
?- vacinadasInd(R).
R = [utente(4, 928323, marco, date(2000, 8, 26), 'marcolol@outlook.pt', 918362758, vila_real, estudante, [],
centro_saude_vilareal)].
```

Figura 5: Resultados obtidos para o predicado vacinadasInd/1

Tal como seria de esperar, ao realizar a consulta na figura em cima, obtiveram-se todas as pessoas que foram vacinadas indevidamente.

3.2.11 candidatos/2

- **Aquisição de conhecimentos**

O predicado *candidatos: Toma, Resultado -> V,F* pretende identificar todas as pessoas que são candidatas a tomar a vacina mas que ainda não foram vacinadas.

A extensão de candidatos/2 é a seguinte:

```
candidatos(N, R) :- solucoes(utente(ID,SS,NM,D,E,T,M,P,L,CS), (utente(ID,SS,NM
    ,D,E,T,M,P,L,CS), vacinacao_covid(_, ID, _, _, N)), R).
```

- **Apresentação dos conhecimentos adquiridos**

Ao indicarmos o número da toma que queremos ver quais são os candidatos a ela, vamos obter uma lista com o nome de todas as pessoas que estão em vacinaca_covid com o mesmo número que demos.

```
?- candidatos(0, R).
R = [utente(4, 928323, marco, date(2000, 8, 26), 'marcolol@outlook.pt', 918362758, vila_real, estudante, [],
centro_saude_vilareal), utente(5, 627848, alice, date(1968, 12, 26), 'alice1968@msn.pt', 912138366, vila_verd
e, professor, [diabetes], centro_saude_vilaverde)].

?- candidatos(1, R).
R = [utente(1, 123477, eduardo, date(1988, 1, 30), 'eduardodf@gorj.pt', 932827487, braga, policia, [diabetes,
 hipertensao], centro_saude_gualtar), utente(3, 632427, sofia, date(1972, 10, 2), 'sofsieh@gmail.com', 929374
84, fafe, enfermeiro, [], centro_saude_fafe)].
```

Figura 6: Resultados obtidos para o predicado candidatos/2

Tal como seria de esperar, ao realizar a consulta na figuara em cima, obtemos todos os candidatos para tomar a vacina da covid.

3.2.12 prestadores/1

- **Aquisição de conhecimentos**

O predicado *prestadores: Resultado -> V,F* pretende identificar todos os staffs que administraram a vacina do covid, a qualquer utente.

A extensão de prestadores/1 é a seguinte:

```
1 prestadores(R) :- solucoesSRep(IDS, IDUˆDTˆVˆTˆvacinacao_covid(IDS,IDU,DT,V,T)
    ,R).
```

Note-se que é utilizado o meta-predicado solucoesSRep/3 dado que este não apresenta as ocorrências repetidas

- **Apresentação dos conhecimentos adquiridos**

Ao consultar quais são os id's dos satffs que realizaram a vacinação aos utentes, identificamos o 1, 2 ,3 ,4 ,5 e 6. Através da figura em cima podemos ver que os prestadores foram

```
?- prestadores(R).
R = [1, 2, 3, 4, 5, 6].
```

Figura 7: Resultados obtidos para o predicado prestadores/1

mesmo os que tinhamos previsto.

3.2.13 vacinasStaff/2

- **Aquisição de conhecimentos**

O predicado *vacinasStaff : IDs, Resultado -> V,F* pretende identificar as vacinadas dadas por um dado staff, tem como argumento o número de identidicação do staff.

A extensão de vacinasStaff/2 é a seguinte:

```
vacinasStaff(N,R) :- solucoes((N, IdU, D, V, T), vacinacao_covid(N, IdU, D, V, T), R).
```

- **Apresentação dos conhecimentos adquiridos**

Dado um ID de staff são listadas todas as doses que esse profissional administrou.

```
?- vacinasStaff(1, R).
R = [(1, 4, '05-06-2021', 'Moderna', 0)].

?- vacinasStaff(4, R).
R = [(4, 1, '10-03-2021', 'Pfizer', 1)].

?- vacinasStaff(6, R).
R = [(6, 5, '10-05-2021', 'AstraZeneca', 0)].
```

Figura 8: Resultados obtidos para o predicado vacinasStaff/2

Através da figura em cima podemos ver que o staff 4 administrou uma dose da vacina Pfiser no dia 10-03-2021 .

3.2.14 listaCentros/1

- **Aquisição de conhecimentos**

O predicado *listaCentros : Resultado -> V,F* pretende listar todos os centros de saúde definidos na base de conhecimento.

A extensão listaCentros/1 é a seguinte:

```
listaCentros(R) :- solucoesSRep((Id, N, M, T, E), centro_saude(Id, N, M, T, E), R).
```

- **Apresentação dos conhecimentos adquiridos**

Este predicado vai listar todos os centros de saúde presentes na base de conhecimento, que foram definidos no ínicio do projeto.

```
?- listaCentros(R).
R = [(1, centro_saude_gualtar, braga, 253764648, 'gualtar@csb.pt'),  (2, centro_saude_caranda, braga, 2538474
95, 'caranda@csb.pt'),  (3, centro_saude_fafe, fafe, 252746433, 'fafe@csf.pt'),  (4, centro_saude_vilareal, v
ila_real, 251763389, 'vilareal@csvr.pt'),  (5, centro_saude_esposende, esposende, 253713748, 'esposende@cse.p
t'),  (6, centro_saude_viladoconde, vila_do_conde, ..., ...),  (7, centro_saude_vilaverde, ..., ...)].
```

Figura 9: Resultados obtidos para o predicado listaCentros/1

Através da figura em cima podemos ver que os centros de saúde identificados foram exatamente os mesmos.

3.2.15 listaUtentes/1

- **Aquisição de conhecimentos**

O predicado *listaUtentes : Resultado -> V,F* pretende listar todos os utentes definidos na base de conhecimento.

A extensão listaUtentes/1 é a seguinte:

```
listaUtentes(R) :- solucoes(utente(ID,SS,NM,D,E,T,M,P,L,CS), utente(ID,SS,NM,D
    ,E,T,M,P,L,CS), R).
```

- **Apresentação dos conhecimentos adquiridos**

Este predicado vai listar todos os utentes presentes na base de conhecimento, que foram definidos no ínicio do projeto.

```
?- listaUtentes(R).
R = [utente(1, 123477, eduardo, date(1988, 1, 30), 'eduardodf@gorj.pt', 932827487, braga, policia, [diabetes,
 hipertensao], centro_saude_gualtar), utente(2, 623426, fernanda, date(1997, 6, 5), 'fernandinha@hotmail.com'
, 918826356, braga, medico, [cancro], centro_saude_caranda), utente(3, 632427, sofia, date(1972, 10, 2), 'sof
sieh@gmail.com', 92937484, fafe, enfermeiro, [], centro_saude_fafe), utente(4, 928323, marco, date(2000, 8, 2
6), 'marcolol@outlook.pt', 918362758, vila_real, estudante, [], centro_saude_vilareal), utente(5, 627848, ali
ce, date(1968, 12, 26), 'alice1968@msn.pt', 912138366, vila_verde, professor, [diabetes], centro_saude_vilave
rde), utente(6, 974837, carina, date(1980, 10, 4), 'carinaporto@hotmail.pt', 913738633, esposende, lojista, [
coracao], centro_saude_esposende), utente(7, 353347, jose, date(1977, 4, 29), 'josejosejose@outlook.pt', 9176
23527, vila_do_conde, gerente, [], centro_saude_viladoconde)].
```

Figura 10: Resultados obtidos para o predicado listaUtentes/1

Através da figura em cima podemos ver que os centros de saúde identificados foram exatamente os mesmos.

3.2.16 listaStaff/1

- **Aquisição de conhecimentos**

O predicado *listaStaff : Resultado -> V,F* pretende listar todos os elementos de staff definidos na base de conhecimento.

A extensão listaStaff/1 é a seguinte:

```
listaStaff(R) :- solucoes((IdS, IdC, N, E), staff(IdS, IdC, N, E), R).
```

- **Apresentação dos conhecimentos adquiridos**

Este predicado vai listar todos elementos do staff presentes na base de conhecimento, que foram definidos no ínicio do projeto. Através da figura em cima podemos ver que os centros

```
?- listaStaff(R).
R = [(1, 4, carla, 'carla_vilareal@csvr.pt'), (2, 5, helena, 'helena_esposende@cse.pt'), (3, 3, artur, 'art
ur_fafe@csf.pt'), (4, 1, simao, 'simao_gualtar@csb.pt'), (5, 2, elisabete, 'elisabete_caranda@csb.pt'), (6
, 7, carmo, 'carmo_vilaverde@csvv.pt')].
```

Figura 11: Resultados obtidos para o predicado listaStaff/1

de saúde identificados foram exatamente os mesmos.

3.2.17 listaVac/1

- **Aquisição de conhecimentos**

O predicado *listaStaff : Resultado -> V,F* pretende listar todos os elementos de staff definidos na base de conhecimento.

A extensão listaStaff/1 é a seguinte:

```
listaVac(R) :- solucoes((IdS, IdU, D, Vac, T), vacinacao_covid(IdS, IdU, D,
    Vac, T), R).
```

- **Apresentação dos conhecimentos adquiridos**

Este predicado vai listar todos elementos da vacinacao_covid presentes na base de conhecimento, que foram definidos no ínicio do projeto. Através da figura em cima podemos ver que os centros de saúde identificados foram exatamente os mesmos.

```
?- listaVac(R).
R = [(4, 1, '10-03-2021', 'Pfizer', 1), (5, 2, '28-12-2020', 'Moderna', 2), (3, 3, '05-04-2021', 'AstraZ
ca', 1), (1, 4, '05-06-2021', 'Moderna', 0), (6, 5, '10-05-2021', 'AstraZeneca', 0), (2, 6, '06-02-2021
..., ...)].
```

Figura 12: Resultados obtidos para o predicado listaVac/1

3.2.18 atualizaToma/1

- **Aquisição de conhecimentos**

O predicado *atualizaToma : IDu -> V,F* pertente alterar o número da toma, se já tivesse tomado uma vez, o número da Toma ficaria igual a 2 .

A extensão atualizaToma/1 é a seguinte:

```
atualizaToma(IdU) :-
    solucoes((IdS, IdU, Data, Vac, Toma), vacinacao_covid(IdS, IdU, Data, Vac,
    Toma), [(IdSR, IdUR, DataR, VacR, TomaR)|T]),
    remove(vacinacao_covid(IdSR, IdUR, DataR, VacR, TomaR)),
    insere(vacinacao_covid(IdSR, IdUR, DataR, VacR, 2)).
```

- **Apresentação dos conhecimentos adquiridos**

Ao atualizar a toma da vacina que estava no número 1, atualizaToma(1), vai atualizar o primeiro elemento da vacinacao_covid para 2.

```
?- listaVac(R).
R = [(4, 1, '10-03-2021', 'Pfizer', 1), (5, 2, '28-12-2020', 'Moderna', 2), (3, 3, '05-04-2021', 'AstraZene
ca', 1), (1, 4, '05-06-2021', 'Moderna', 0), (6, 5, '10-05-2021', 'AstraZeneca', 0), (2, 6, '06-02-2021',
..., ...)].

?- atualizaToma(1).
true .

?- listaVac(R).
R = [(5, 2, '28-12-2020', 'Moderna', 2), (3, 3, '05-04-2021', 'AstraZeneca', 1), (1, 4, '05-06-2021', 'Mode
rna', 0), (6, 5, '10-05-2021', 'AstraZeneca', 0), (2, 6, '06-02-2021', 'Pfizer', 2), (4, 1, '10-03-2021',
..., ...)].
```

Figura 13: Resultados obtidos para o predicado atualizaToma/1

Através da figura em cima, podemos ver que ocorreu de acordo com o previsto.

3.2.19 atualizaMorada/2

- **Aquisição de conhecimentos**

O predicado *atualizaMorada : IDu, Morada -> V,F* pertente alterar a morada de um utente .

A extensão atualizaMorada/2 é a seguinte:

```
atualizaMorada(ID, Morada) :-
    solucoes((ID,SS,NM,D,E,Toma,M,P,L,CS), utente(ID,SS,NM,D,E,Toma,M,P,L,CS),
    [(IDR,SSR,NMR,DR,ER,TomaR,MR,PR,LR,CSR)|T]),
    remove(utente(IDR,SSR,NMR,DR,ER,TomaR,MR,PR,LR,CSR)),
    insere(utente(ID,SSR,NMR,DR,ER,TomaR,Morada,PR,LR,CSR)).
```

- **Apresentação dos conhecimentos adquiridos**

Ao atualizar a morada do utente com número de identificação 1 para Porto, deveríamos verificar que a morada a si associada deixaria de ser Braga e passaria a ser Porto.

```
?- listaUtentes(R).
R = [utente(1, 123477, eduardo, date(1988, 1, 30), 'eduardodf@gorj.pt', 932827487, braga, policia, [diabetes, h
ipertensao], centro_saude_gualtar), utente(2, 623426, fernanda, date(1997, 6, 5), 'fernandinha@hotmail.com', 91
8826356, braga, medico, [cancro], centro_saude_caranda), utente(3, 632427, sofia, date(1972, 10, 2), 'sofsieh@g
mail.com', 92937484, fafe, enfermeiro, [], centro_saude_fafe), utente(4, 928323, marco, date(2000, 8, 26), 'mar
colol@outlook.pt', 918362758, vila_real, estudante, [], centro_saude_vilareal), utente(5, 627848, alice, date(1
968, 12, 26), 'alice1968@msn.pt', 912138366, vila_verde, professor, [diabetes], centro_saude_vilaverde), utente
(6, 974837, carina, date(1980, 10, 4), 'carinaporto@hotmail.pt', 913738633, esposende, lojista, [coracao], cent
ro_saude_esposende), utente(7, 353347, jose, date(1977, 4, 29), 'josejosejose@outlook.pt', 917623527, vila_do_c
onde, gerente, [], centro_saude_viladoconde)].

?- atualizaMorada(1, porto).
true .

?- listaUtentes(R).
R = [utente(2, 623426, fernanda, date(1997, 6, 5), 'fernandinha@hotmail.com', 918826356, braga, medico, [cancro
], centro_saude_caranda), utente(3, 632427, sofia, date(1972, 10, 2), 'sofsieh@gmail.com', 92937484, fafe, enfe
rmeiro, [], centro_saude_fafe), utente(4, 928323, marco, date(2000, 8, 26), 'marcolol@outlook.pt', 918362758, v
ila_real, estudante, [], centro_saude_vilareal), utente(5, 627848, alice, date(1968, 12, 26), 'alice1968@msn.pt
', 912138366, vila_verde, professor, [diabetes], centro_saude_vilaverde), utente(6, 974837, carina, date(1980,
10, 4), 'carinaporto@hotmail.pt', 913738633, esposende, lojista, [coracao], centro_saude_esposende), utente(7,
353347, jose, date(1977, 4, 29), 'josejosejose@outlook.pt', 917623527, vila_do_conde, gerente, [], centro_saude
_viladoconde), utente(1, 123477, eduardo, date(1988, 1, 30), 'eduardodf@gorj.pt', 932827487, porto, policia, [d
iabetes|...], centro_saude_gualtar)].
```

Figura 14: Resultados obtidos para o predicado atualizaMorada/2

Através da figura em cima, conseguimos comprovar que a morada do utente *eduardo* passou a ser o porto.

3.2.20 atualizaCentro/2

- **Aquisição de conhecimentos**

O predicado *atualizaCentro : IDu, CentroS -> V,F* pertente alterar o centro de saúde a que um utente utiliza os serviços de saúde.

A extensão atualizaCentro/2 é a seguinte:

```
atualizaCentro(ID, CentroS) :-
    solucoes((ID,SS,NM,D,E,Toma,M,P,L,CS), utente(ID,SS,NM,D,E,Toma,M,P,L,CS),
    [(IDR,SSR,NMR,DR,ER,TomaR,MR,PR,LR,CSR)|T]),
    remove(utente(IDR,SSR,NMR,DR,ER,TomaR,MR,PR,LR,CSR)),
    insere(utente(ID,SSR,NMR,DR,ER,TomaR,MR,PR,LR,CentroS)).
```

- **Apresentação dos conhecimentos adquiridos**

Ao atualizar o centro de saude do utente com número de identificação 1 para centro_saude_povoa, deveríamos verificar que o centro a si associado deixaria de ser centro_saude_gualtar e passaria a ser centro_saude_povoa.

```
?- listaUtentes(R).
R = [utente(1, 123477, eduardo, date(1988, 1, 30), 'eduardodf@gorj.pt', 932827487, braga, policia, [diabetes, h
ipertensao], centro_saude_gualtar), utente(2, 623426, fernanda, date(1997, 6, 5), 'fernandinha@hotmail.com', 91
8826356, braga, medico, [cancro], centro_saude_caranda), utente(3, 632427, sofia, date(1972, 10, 2), 'sofsieh@g
mail.com', 92937484, fafe, enfermeiro, [], centro_saude_fafe), utente(4, 928323, marco, date(2000, 8, 26), 'mar
colol@outlook.pt', 918362758, vila_real, estudante, [], centro_saude_vilareal), utente(5, 627848, alice, date(1
968, 12, 26), 'alice1968@msn.pt', 912138366, vila_verde, professor, [diabetes], centro_saude_vilaverde), utente
(6, 974837, carina, date(1980, 10, 4), 'carinaporto@hotmail.pt', 913738633, esposende, lojista, [coracao], cent
ro_saude_esposende), utente(7, 353347, jose, date(1977, 4, 29), 'josejosejose@outlook.pt', 917623527, vila_do_c
onde, gerente, [], centro_saude_viladoconde)].

?- atualizaCentro(1, centro_saude_povoa).
true .

?- listaUtentes(R).
R = [utente(2, 623426, fernanda, date(1997, 6, 5), 'fernandinha@hotmail.com', 918826356, braga, medico, [cancro
], centro_saude_caranda), utente(3, 632427, sofia, date(1972, 10, 2), 'sofsieh@gmail.com', 92937484, fafe, enfe
rmeiro, [], centro_saude_fafe), utente(4, 928323, marco, date(2000, 8, 26), 'marcolol@outlook.pt', 918362758, v
ila_real, estudante, [], centro_saude_vilareal), utente(5, 627848, alice, date(1968, 12, 26), 'alice1968@msn.pt
', 912138366, vila_verde, professor, [diabetes], centro_saude_vilaverde), utente(6, 974837, carina, date(1980,
10, 4), 'carinaporto@hotmail.pt', 913738633, esposende, lojista, [coracao], centro_saude_esposende), utente(7,
353347, jose, date(1977, 4, 29), 'josejosejose@outlook.pt', 917623527, vila_do_conde, gerente, [], centro_saude
_viladoconde), utente(1, 123477, eduardo, date(1988, 1, 30), 'eduardodf@gorj.pt', 932827487, braga, policia, [d
iabetes|...], centro_saude_povoa)].
```

Figura 15: Resultados obtidos para o predicado atualizaCentro/2

Através da figura em cima, conseguimos comprovar que o centro de saude do utente *eduardo* passou a ser o centro_saude_povoa.

3.3 Invariantes

Na construção dos invariantes, existem dois tipos principais a ter em conta, a estrutura do conhecimento e a referência a informação contida na base do conhecimento. São utilizados para a manutenção da verdade, tanto na inserção como na remoção de conhecimento.

Nesta seccção vamos apresentar a construção dos diferentes invariantes estruturais e referenciais, utilizados neste trabalho.

3.3.1 Estruturais

Os invariantes estruturais de inserção permitem que seja adicionado à base conhecimento um utente, staff e centro de saúde cujo número de identificação seja único, não pode haver ID's repetidos. Logo, a quantidade de cada um com esse número de ID tem que ser menor ou igual a 1.

- **Não podem existir utentes com o mesmo ID**

```
+utente(ID,S,No,D,E,T,M,P,L,C) :: (solucoes((ID),(utente(ID,A,B,C,F,G,H,I,J,K)
    ),S),
2                                   comprimento(S,N), N =< 1).
```

Como podemos ver na figura abaixo, ao tentar inserir um utente com número de identificação *1* e nome *flavio*, que já se encontra outro utente com o mesmo número de identificação, este não vai ser possível ser adicionado à base de conhecimento.

```
?- listaUtente(R).
Correct to: "listaUtentes(R)"? yes
R = [utente(1, 123477, eduardo, date(1988, 1, 30), 'eduardodf@gorj.pt', 932827487, braga, policia, [diabetes,
 hipertensao], centro_saude_gualtar), utente(2, 623426, fernanda, date(1997, 6, 5), 'fernandinha@hotmail.com'
, 918826356, braga, medico, [cancro], centro_saude_caranda), utente(3, 632427, sofia, date(1972, 10, 2), 'sof
sieh@gmail.com', 92937484, fafe, enfermeiro, [], centro_saude_fafe), utente(4, 928323, marco, date(2000, 8, 2
6), 'marcolol@outlook.pt', 918362758, vila_real, estudante, [], centro_saude_vilareal), utente(5, 627848, ali
ce, date(1968, 12, 26), 'alice1968@msn.pt', 912138366, vila_verde, professor, [diabetes], centro_saude_vilave
rde), utente(6, 974837, carina, date(1980, 10, 4), 'carinaporto@hotmail.pt', 913738633, esposende, lojista, [
coracao], centro_saude_esposende), utente(7, 353347, jose, date(1977, 4, 29), 'josejosejose@outlook.pt', 9176
23527, vila_do_conde, gerente, [], centro_saude_viladoconde)].

?- evolucao(utente(1, 124123, flavio, date(1990,02,04), 'flav@gmail.com', 911333222, braga, arrumador, [], ce
ntro_saude_fafe)).
false.

?- listaUtente(R).Correct to: "listaUtentes(R)"? yes
R = [utente(1, 123477, eduardo, date(1988, 1, 30), 'eduardodf@gorj.pt', 932827487, braga, policia, [diabetes,
 hipertensao], centro_saude_gualtar), utente(2, 623426, fernanda, date(1997, 6, 5), 'fernandinha@hotmail.com'
, 918826356, braga, medico, [cancro], centro_saude_caranda), utente(3, 632427, sofia, date(1972, 10, 2), 'sof
sieh@gmail.com', 92937484, fafe, enfermeiro, [], centro_saude_fafe), utente(4, 928323, marco, date(2000, 8, 2
6), 'marcolol@outlook.pt', 918362758, vila_real, estudante, [], centro_saude_vilareal), utente(5, 627848, ali
ce, date(1968, 12, 26), 'alice1968@msn.pt', 912138366, vila_verde, professor, [diabetes], centro_saude_vilave
rde), utente(6, 974837, carina, date(1980, 10, 4), 'carinaporto@hotmail.pt', 913738633, esposende, lojista, [
coracao], centro_saude_esposende), utente(7, 353347, jose, date(1977, 4, 29), 'josejosejose@outlook.pt', 9176
23527, vila_do_conde, gerente, [], centro_saude_viladoconde)].
```

Figura 16: Resultados obtidos

- **Não podem existir staffs com o mesmo ID**

```
+staff(ID,I,No,E) :: (solucoes((ID), (staff(ID,X,Y,Z)), S),
                      comprimento(S,N), N =< 1).
```

Como podemos ver na figura abaixo, ao tentar inserir um staff com número de identificação *2* e nome *joao*, que já se encontra outro staff com o mesmo número de identificação, este não vai ser possível ser adicionado à base de conhecimento.

```
?- listaStaff(R).
R = [(1, 4, carla, 'carla_vilareal@csvr.pt'), (2, 5, helena, 'helena_esposende@cse.pt'), (3, 3, artur, 'artur
_fafe@csf.pt'), (4, 1, simao, 'simao_gualtar@csb.pt'), (5, 2, elisabete, 'elisabete_caranda@csb.pt'), (6, 7,
 carmo, 'carmo_vilaverde@csvv.pt')].

?- staff(2, 1, joao, 'joao@csc.pt').
false.

?- listaStaff(R).
R = [(1, 4, carla, 'carla_vilareal@csvr.pt'), (2, 5, helena, 'helena_esposende@cse.pt'), (3, 3, artur, 'artur
_fafe@csf.pt'), (4, 1, simao, 'simao_gualtar@csb.pt'), (5, 2, elisabete, 'elisabete_caranda@csb.pt'), (6, 7,
 carmo, 'carmo_vilaverde@csvv.pt')].
```

Figura 17: Resultados obtidos

- **Não podem existir centros de saúde com o mesmo ID**

```
+centro_saude(ID,No,M,T,E) :: (solucoes((ID), (centro_saude(ID,X,Y,Z,W)),S)
                               comprimento(S,N), N =< 1).
```

Como podemos ver na figura abaixo, ao tentar inserir um centro de saúde com número de identificação *1* e nome *centro_saude_cila*, já se encontra outro centro de saúde com o mesmo número de identificação logo este não vai ser possível ser adicionado à base de conhecimento.

```
?- listaCentro(R).
Correct to: "listaCentros(R)"? yes
R = [(1, centro_saude_gualtar, braga, 253764648, 'gualtar@csb.pt'), (2, centro_saude_caranda, braga, 253847495
, 'caranda@csb.pt'), (3, centro_saude_fafe, fafe, 252746433, 'fafe@csf.pt'), (4, centro_saude_vilareal, vila_
real, 251763389, 'vilareal@csvr.pt'), (5, centro_saude_esposende, esposende, 253713748, 'esposende@cse.pt'),
(6, centro_saude_viladoconde, vila_do_conde, ..., ...), (7, centro_saude_vilaverde, ..., ...)].

?- centro_saude(1, centro_saude_cila, braga, 253789029, 'cila@csc.pt').
false.

?- listaCentro(R).
Correct to: "listaCentros(R)"? yes
R = [(1, centro_saude_gualtar, braga, 253764648, 'gualtar@csb.pt'), (2, centro_saude_caranda, braga, 253847495
, 'caranda@csb.pt'), (3, centro_saude_fafe, fafe, 252746433, 'fafe@csf.pt'), (4, centro_saude_vilareal, vila_
real, 251763389, 'vilareal@csvr.pt'), (5, centro_saude_esposende, esposende, 253713748, 'esposende@cse.pt'),
(6, centro_saude_viladoconde, vila_do_conde, ..., ...), (7, centro_saude_vilaverde, ..., ...)].
```

Figura 18: Resultados obtidos

- **Não permite inserir utente em vacinacao_covid se não estiver na base de conhecimento**

```
+vacinacao_covid(IDs,IDu,D,V,T) :: (solucoes((IDu), (utente(IDu,Ss,No,D,E,T,M,
    P,L,Cs)),S),
                                    comprimento(S,N), N == 1).
```

Como podemos ver na figura abaixo, inserir um utente que não pertence à base de conhecimento na vacinacao_covid, não é possível .

```
?- vacinacao_covid(1, 8, '2021-05-02', 'Pfizer', 1).
false.
```

Figura 19: Resultados obtidos

- **Não permite inserir staff em vacinacao_covid se não estiver na base de conhecimento**

```
+vacinacao_covid(IDs,IDu,D,V,T) :: (solucoes((IDs), (staff(IDs,IDu,No,E)),S),
                                    comprimento(S,N), N == 1).
```

Como podemos ver na figura abaixo, inserir um staff que não pertence à base de conhecimento na vacinacao_covid, não é possível .

```
?- vacinacao_covid(7, 1, '2021-05-02', 'Pfizer', 1).
false.
```

Figura 20: Resultados obtidos

3.3.2 Referenciais

Os invariantes referenciais de remoção não permitem retirar utentes e staff que estejam na realizar a vacinação do covid. Seguem a mesma lógica que os anteriores, no sentido em que se qualque vacinacao_covid tiver qualquer id, esse elemento já não pode ser removido.

- **Não permite remover utente se estiver em vacinacao_covid**

```
-utente(ID,SS,N,D,E,T,M,P,L,C) :: (solucoes((ID), (vacinacao_covid(IDs, ID, DT
    , V, T)),S),
                                   comprimento(S,N), N==0).
```

```
R = [utente(2, 623426, fernanda, date(1997, 6, 5), 'fernandinha@hotmail.com', 918826356, braga, medico, [canc
ro], centro_saude_caranda), utente(3, 632427, sofia, date(1972, 10, 2), 'sofsieh@gmail.com', 92937484, fafe,
enfermeiro, [], centro_saude_fafe), utente(4, 928323, marco, date(2000, 8, 26), 'marcolol@outlook.pt', 918362
758, vila_real, estudante, [], centro_saude_vilareal), utente(5, 627848, alice, date(1968, 12, 26), 'alice196
8@msn.pt', 912138366, vila_verde, professor, [diabetes], centro_saude_vilaverde), utente(6, 974837, carina, d
ate(1980, 10, 4), 'carinaporto@hotmail.pt', 913738633, esposende, lojista, [coracao], centro_saude_esposende)
, utente(7, 353347, jose, date(1977, 4, 29), 'josejosejose@outlook.pt', 917623527, vila_do_conde, gerente, []
, centro_saude_viladoconde)].

?- involucao(utente(1, 123477, eduardo, date(1988,01,30), 'eduardodf@gorj.pt', 932827487, braga, policia, [di
abetes,hipertensao], centro_saude_gualtar)).
false.

?- listaUtente(R).
Correct to: "listaUtentes(R)"? yes
R = [utente(2, 623426, fernanda, date(1997, 6, 5), 'fernandinha@hotmail.com', 918826356, braga, medico, [canc
ro], centro_saude_caranda), utente(3, 632427, sofia, date(1972, 10, 2), 'sofsieh@gmail.com', 92937484, fafe,
enfermeiro, [], centro_saude_fafe), utente(4, 928323, marco, date(2000, 8, 26), 'marcolol@outlook.pt', 918362
758, vila_real, estudante, [], centro_saude_vilareal), utente(5, 627848, alice, date(1968, 12, 26), 'alice196
8@msn.pt', 912138366, vila_verde, professor, [diabetes], centro_saude_vilaverde), utente(6, 974837, carina, d
ate(1980, 10, 4), 'carinaporto@hotmail.pt', 913738633, esposende, lojista, [coracao], centro_saude_esposende)
, utente(7, 353347, jose, date(1977, 4, 29), 'josejosejose@outlook.pt', 917623527, vila_do_conde, gerente, []
, centro_saude_viladoconde), utente(1, 123477, eduardo, date(1988, 1, 30), 'eduardodf@gorj.pt', 932827487, br
aga, policia, [diabetes|...], centro_saude_gualtar)].
```

Figura 21: Resultados obtidos

Tal como podemos ver na figura, ao tentar remover o utente com o número de identificação 1, que está contido na vacinacao_covid com o número de identificação 1 a si associado, este não vai ser retirado da base de conhecimento.

- **Não permite remover staff se estiver em vacinacao_covid**

```
-staff(ID,IDu,N,E) :: (solucoes((ID), (vacinacao_covid(ID, IDu, DT, V, T)),S),
                       comprimento(S,N), N==0).
```

Tal como podemos ver na figura abaixo, ao tentar remover o staff com o número de identificação 4, que está contido na vacinacao_covid com o número de identificação 4 a si associado, este não vai ser retirado da base de conhecimento.

```
?- listaStaff(R).
R = [(1, 4, carla, 'carla_vilareal@csvr.pt'), (2, 5, helena, 'helena_esposende@cse.pt'), (3, 3, artur, 'artur
_fafe@csf.pt'), (4, 1, simao, 'simao_gualtar@csb.pt'), (5, 2, elisabete, 'elisabete_caranda@csb.pt'), (6, 7,
 carmo, 'carmo_vilaverde@csvv.pt')].

?- involucao(staff(4,1,'10-03-2021', 'Pfizer', 1)).
false.

?- listaStaff(R).
R = [(1, 4, carla, 'carla_vilareal@csvr.pt'), (2, 5, helena, 'helena_esposende@cse.pt'), (3, 3, artur, 'artur
_fafe@csf.pt'), (4, 1, simao, 'simao_gualtar@csb.pt'), (5, 2, elisabete, 'elisabete_caranda@csb.pt'), (6, 7,
 carmo, 'carmo_vilaverde@csvv.pt')].
```

Figura 22: Resultados obtidos

4 Conclusão

Depois da realização desta fase do Trabalho Prático consideramos que foi relativamente bem sucedida, visto que pensamos que cumprimos com todos os requisitos.

O maior desafio que encontramos foi conseguir desenvolver o raciocínio lógico necessário para começar a desdobrar os problemas que foram surgindo. A falta de conhecimento de alguns meta-predicados que poderiam ter auxiliado a construção da extensão dos predicados a desenvolver, foi também um entrave na resolução deste exercício.

É notável que a realização deste exercício foi imprescindível para a consolidação dos conhecimentos até agora adquiridos, dado que nos permitiu adquirir um maior à-vontade ao trabalhar com a linguagem de programação em lógica PROLOG e com a ferramenta SICStus Prolog, e maior facilidade em construir Invariantes.