Family Tree Problem

(CS-361 Artificial Intelligence)

FALL 2020

NAMAL INSTITUTE

Submitted to: Dr. Junaid Akhtar
Submitted by: Group-Alpha
Dated: March 15th, 2021.

Group Members:

No.	Name	Roll Number	Role
1	Shahzeb Khan	1802021	**Project Manager**
2	Muhammad Musawar Baig	1802003	Front-End Team
3	Abdullah Tahir	1802016	Front-End Team
4	Qazi Arslan Shah	1802030	Back-End Team
5	Manzoor Ul Haq	1802022	Back-End Team
6	Muhammad Rehan	1802008	Documentation-Team
7	Mahmood Yousaf	1802011	Documentation-Team

Abstract

Our program explores the inter-family relationships of the Khan Family (Figure 1). It can find a person's baap, beti, bahu, beta, pota, dada, nawasa, nana, sussar, dadi, chachataya, nani, khala, sala, or baapDada if that relationship exists. The front end of the program is written in Python and the backend in Prolog. Python is a general-purpose and high-level programming language. It is an interpreted language. On the other hand, Prolog is a logic programming language. It is primarily intended as a declarative programming language. It has applications in artificial intelligence and computational linguistics.

Introduction:

Predicate Logic:

Predicate logic is also called first-order logic. We can develop information about the objects in a much simpler way and the relationship between objects can be easily expressed. Charles Pierce and Gottlob Frege are the inventors of predicate logic. In artificial intelligence by using predicate logic, we can easily and efficiently represent our knowledge in computer. Predicate logic is used for knowledge representation. We can easily capture the actual meaning of statements by using predicate logic as compared to propositional logic.

Prolog:

Prolog was designed by Alain Colmerauer and a team of researchers with the thought In mind that using logic to represent knowledge and to write programs in the 1970s. Prolog uses a subset of predicate logic and draws its structures from the early work of logicians.

Features of prolog:

- **Facts:**

  Facts are statements that describe properties of objects or relationships between objects.

  For example:

  mianbiwi (chotekhan,chotirani).

  In the above example "**mianbiwi**" is a relation between two objects "**chotekhan**" and "**chotirani**".

  Collections of many facts and rules form a database. It represents the knowledge in some logical form.

- **Terms:**

  In prolog, all kind of data is called terms. e.g. **parent (abc, def),** that whole is considered as a complex term consisting of simple terms i.e. abc, def.

- **Queries:**

  A query is basically a request to retrieve information from the database. e.g.

  ? - *parent* **(X, kauser).**

  Ans: *chotekhan, chotirani*

- **Variables:**

  They are kind of prolog terms that starts with capital letters or with underscore "_".

  ? - *mianbiwi* **(X, chotirani).**

  "X" is a variable.

- **Rules:**

  Rules help us to derive new property from old defined facts from the knowledge base.

  beti(X, Y) : - parent(Y, X), gins (female, X).

Problem Statement:

In this project, we have to code khan's family tree by using provided facts, and then we must implement rules given in the requirements document. The application should be able to ask few questions like, "*Who is the father of X, who is khala of X",... etc.*

Khan Family Tree:

The khan family tree is given below:

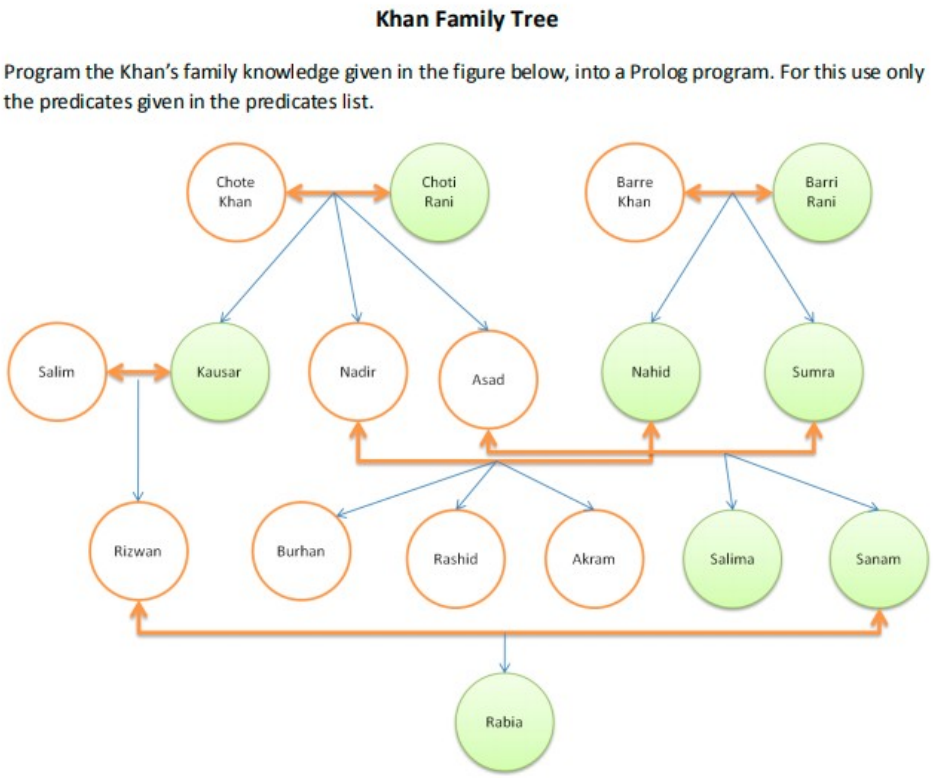

Figure 1: Khan Family Tree.

Implement the following rules:

- **baap**(Variable1, Variable2): - **parent**(Variable1, Variable2), **gins**(male, Variable1).
- **beti** (Variable1, Variable2)
- **beta** (Variable1, Variable2)
- **dada** (Variable1, Variable2)
- **nana** (Variable1, Variable2)
- **dadi** (Variable1, Variable2)
- **nani** (Variable1, Variable2)
- **sala** (Variable1, Variable2)

- **bahu** (Variable1, Variable2)
- **pota** (Variable1, Variable2)
- **nawasa** (Variable1, Variable2)
- **sussar** (Variable1, Variable2)
- **chachataya** (Variable1, Variable2)
- **khala** (Variable1, Variable2)
- **baapDada**(Variable1, Variable2)

Back-End Documentation:

Allowed Facts:

- mianbiwi (mian, biwi).
- parent (parent, child).
- gins (gender, person).

Main Coding Task:

We have to implement the given rules in the requirements document by using the above facts.

Rules:

- *beti (X, Y): - parent (Y, X), gins (female, X).* This rule finds the daughter if we provide parent to this rule. i.e., "X" is the daughter of "Y" and we can find this by using the fact "*parent (Y, X)"* and *gins (female, X).*
- *beta (X, Y): - parent (Y, X), gins (male, X). This rule finds the son if we provide parents to this rule. i.e., "X" is the son of "Y" and we can find this by using the fact "parent (Y, X)" and gins (male, X).*
- *dada (X, Y): - parent (X, Z), parent (Z, Y), gins (male, X), gins (male, Z). This rule finds the dada (Grandpapa) if we provide pota/poti to this rule. i.e., "X" is dada of "Y" and we can find this by getting parent of the parent of Y giving the condition that all parents are male.*
- *nana (X, Y): - parent (X, Z), parent (Z, Y), gins (male, X), gins (female, Z).* This rule finds the nana if we provide nawasaa/nawasii and vice versa. In simple words find the female parent ("Z") of child "Y" and then find the male parent of mama ("Z").
- *dadi (X, Y): - parent (X, Z), parent (Z, Y), gins (female, X), gins (male, Z).* This rule finds the dadi if we provide pota/poti and vice versa. In simple words find male parent ("Z") of child "Y" and then find the female parent of papa ("Z").
- *nani (X, Y): - parent (X, Z), parent (Z, Y), gins (female, X), gins (female, Z).* This rule finds the nani if we provide nawasaa/nawasii and vice versa. In simple words find the female parent ("Z") of child "Y" and then find the female parent of mama ("Z").
- *sala (X, Y): - mianbiwi (Y, Z), parent (A, Z), gins (female, Z), parent (A, X), gins (male, A), gins (male, X).* This rule finds the sala i.e., brother of the wife. Find the

- wife of the husband (“Y”) and then find the parents of the wife, after this gets all the male child of wife parent.
- *bahu (X, Y): - parent (Y, Z), gins (female, X), gins (male, Z), mianbiwi (Z, X).* Bahu is the wife of the son. To find bahu, we first son of “Y” and then find the wife of the son.
- *pota (X, Y): - parent (Y, Z), parent (Z, X), gins (male, X), gins (male, Z).* It is simple, find the son of the son.
- *nawasa (X, Y): - parent (Y, Z), parent (Z, X), gins (male, X), gins (female, Z).* Finds daughter’s son.
- *sussar (X, Y): - mianbiwi (Y, Z), parent (X, Z), gins (male, X), gins (female, Z), gins (male, Y).*
- *sussar (X, Y): - mianbiwi (Z, Y), parent (X, Z), gins (male, X), gins (male, Z), gins (female, Y).* This finds the husband’s father (or) wife’s father.
- *baapDada (X, Y): - parent (X, Y), gins (male, X).*
- *baapDada (X, Y): - parent (X, Z), baapDada (Z, Y), gins (male, Z), gins (male, X).* This finds ancestors of a given child. This is a recursive rule that goes up the tree till we reach the top.
- *khala (X, Y): - parent (Z, Y), gins (female, Z), parent (L, Y), gins (male, L), parent (A, Z), gins (male, A), parent (A, X), gins (female, X), not (mianbiwi (L, X)).* This rule finds khala “sister(s) of mother” of a child. The main logic in this rule is that find female children of nana and then exclude son’s / daughter’s mother by using “*not (mianbiwi (papa, khala))”* i.e., all-female children of nana who are not the wife of child’s (“Y”) father.
- *chachataya (X, Y): - parent (Z, Y), gins (male, Z), parent (Ma, Y), gins (female, Ma), parent (A, Z), gins (male, A), parent (A, X), gins (male, X), not (mianbiwi (X, Ma)).* This rule finds the chaha/Taya “brother (s) of the father” of a child. The main logic in this rule is that find male children of dada and then exclude son’s / daughter’s father by using “*not (mianbiwi (chacha, mama))”* i.e., all male children of dada who are not husbands of child’s (“Y”) mother.

Front End Documentation:

In this section, we will discuss how effectively we can use prolog logic using the interface. We integrated prolog and python for querying using the interface.

PySwip:

PySwip is a Python library - SWI-Prolog bridge enabling to query in python programs. It provides us a utility that makes it easy to query with the backend Prolog using a python interface.

Main functions used in Integrated environment:

- **Prolog.consult(“filename.pl”):**

This function links the backend prolog file with the python program. Backend prolog file contains facts and rules from which we can query for certain results derived from rules.

- **Prolog.query(“query”):**

Prolog.query("query") gives input to the backend prolog file as query and returns the result of that query. In our case we give input "Y" to that function and store returned result in the "val" array.

Description:

We have developed a prolog file for the backend and integrated it with the front end of python using PySwip. We have designed a simple console interface. By following instructions on the interface, A user can query on relations that are specified in the program. Users can check relations of "Khan Family". A ready interface for getting input is shown in Figure 2.

Figure 2: Interface for querying.

When specific relation is selected then a complete list of all the members of the khan family is displayed to help the user input a valid name for finding the relationship. Figure 3

```
---> Enter your Choice for the Relationship that you are interested in <---
1
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+                                                                +
+Members of Khan Family:                                         +
+                                                                +
+ChoteKhan, ChotiRani, BarreKhan, BarriRani                      +
+Salim, Kausar, Nadir, Asad, Nahid, Sumra                        +
+Rizwan, Burhan, Rashid, Akram, Salima, Sanam, Rabia             +
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++


Enter name of person whose Beti is required : sanam

============================================================
Ms.rabia is the beti of sanam.
============================================================
```

Figure 3: Query For Parent.

In Figure 3, you can see the user wanted to find beti of “sanam” so, he gave the parent name as input, and in return name of beti appeared on the interface. Now, in Figure 4, you can see another query and its respective result.

```
---> Enter your Choice for the Relationship that you are interested in <---
10
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+                                                                +
+Members of Khan Family:                                         +
+                                                                +
+ChoteKhan, ChotiRani, BarreKhan, BarriRani                      +
+Salim, Kausar, Nadir, Asad, Nahid, Sumra                        +
+Rizwan, Burhan, Rashid, Akram, Salima, Sanam, Rabia             +
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++


Enter name of person whose Nawasa is required : barrekhan

============================================================
Mr.burhan is the nawasa of barrekhan.
============================================================

============================================================
Mr.rashid is the nawasa of barrekhan.
============================================================

============================================================
Mr.akram is the nawasa of barrekhan.
============================================================
```

Figure 4: Query For Nawasa.