Московский авиационный институт

(государственный технический университет)

**Факультет прикладной математики и физики**

Кафедра вычислительной математики и программирования

**Лабораторная работа №2**

по курсу

«Логическое программирование»

|  |  |
| --- | --- |
| Выполнил: | Карпова В.А. |
| Группа: | 08-306 |
| № по списку: | 9 |
| Руководитель: | Левинская М.А. |
| Оценка: | |
| Дата: | |

Москва

2012 г.

**Тема: «Язык Пролог и его применение для решения логических задач».**

**Условие**

Написать и отладить Пролог-программу решения логической задачи, проанализировать эффективность, безопасность и непротиворечивость решения.

**Вариант 10**

В одном городе живут 5 человек. Их имена Леонид, Михаил, Николай, Олег и Петр. Их фамилии Атаров, Бартенев, Кленов, Данилин и Иванов. Бартенев знаком только с двумя из перечисленных мужчин. Петр знаком со всеми, кроме одного. Леонид знает только одного из всех. Данилин и Михаил незнакомы. Николай и Иванов знают друг друга. Михаил, Николай и Олег знакомы между собой. Атаров незнаком только с одним из всех. Только один из всех знаком с Кленовым. Назовите имена и фамилии каждого. С кем знаком каждый из них?

**Метод решения**

Перед выполнением задачи на SWI-Prolog, задача была решена «вручную», с помощью составления двух таблиц (соответствие фамилий и кто с кем знаком) и последовательным извлечением из них непротиворечивой информации.

Из условий задачи (Леонид знает одного из всех, только один из всех знаком с Кленовым), становится, очевидно, что у Леонида фамилия – Кленов. По условиям связи имен с фамилиями, становится понятным, что Николай не может быть с фамилией Иванов, у Михаила фамилия не Данилин. А Петр не может быть Бартневым, т.к. количество друзей у них не совпадает.

Это небольшие выводы позволяют увеличить эффективность выполнения программы, за счет отсечения некоторых условий.

Все условия задачи определяются в предикате info. Выполнения этого предиката были заданы еще одни предикаты know(знакомы) и count(количество друзей). Для окончательного ответа используется предикат solve, который выдает в качестве Х список фамилий, порядок которого определяется в соответствии с предикатом surname(списков всех фамилий). Также solve выдает список знакомых, в соответствии с предикатом people (списком всех имен).

Для еще большего повышения эффективности, были введены следующие ускорения:

После получения возможного списка знакомых, производились повторные проверки условий на принадлежность к списку очевидных условий. Например, Михаил, Николай и Олег знакомы между собой, следовательно, каждый из них должен входить в свой список R2, R3 или R4.

Для форматированного вывода на экран имен и фамилий, использовался предикат print.

Целостность задачи характеризуется единственным решением. Решение существует и единственное и совпадает со сделанными ранее таблицами и выводами (вручную).

*Реа,изация.*

% Устанавливаем имена и фамилии

people(X):-

X = [leonid, michael, nikolay, oleg, petr].

surname(Y):-

Y = [atarov, bartenev, klenov, danilin, ivanov].

%Возвращает друзей

get\_friends(X, Name, Solve) :-

people(Tmp),

nth0(Pos, Tmp, Name), % на каком месте стоит человек

nth0(Pos, Solve, X).

% Количество друзей (Res) у Name

count(Name, Res, Solve) :-

get\_friends(Friends, Name, Solve),

length(Friends, Res).

% знакомые -- т.е. у Name1 одним из друзей является Name2. Наоборот тоже верно.

know(Name1, Name2, Solve) :-

get\_friends(Friends1, Name1, Solve),

get\_friends(Friends2, Name2, Solve),

member(Name2, Friends1),

member(Name1, Friends2).

%ускорение -- накладывает условия на фамилии которые очевидны

speedup(Who) :-

[Atarov, Bartenev, Klenov, Danilin, Ivanov] = Who,

% Klenov \= nikolay,

% Klenov \= oleg,

% Klenov \= michael,

% Klenov \= petr,

% =>

Klenov == leonid, % Только один из всех знаком с Кленовым, аналогично тому, что Кленов знаком только с одним из всех (А у нас только Леонид знаком с одним из всех)

Ivanov \= nikolay, % Иванов и Николай знакомы (т.е. разные люди)

Danilin \= michael, %Данилин и Михаил незнакомы (т.е. разные люди)

Bartenev \= petr. %Кол-во друзей у Бартенва не совпадает с колв-ом друзей у Петра

% Возвращает список имен на нужном месте в соответсвии с фамилиями

info(Who, Solve) :-

[Atarov, Bartenev, Klenov, Danilin, Ivanov] = Who,

count(Bartenev, 2, Solve),

count(petr, 3, Solve),

count(leonid, 1, Solve),

not(know(Danilin, michael, Solve)),

know(nikolay, Ivanov, Solve),

know(nikolay, michael, Solve),

know(nikolay, oleg, Solve),

know(michael, oleg, Solve),

count(Atarov, 3, Solve),

count(Klenov, 1, Solve).

%Все комбинации без повторений

combinations(\_,[]).

combinations([X|T],[X|Comb]) :- combinations(T,Comb).

combinations([\_|T],[X|Comb]) :- combinations(T,[X|Comb]).

% список друзей для одного человека

get\_friendlist(Res, Man) :-

people(P0),

delete(P0, Man, P1),

combinations(P1, Res).

print([],[]).

print([H1|T1],[H2|T2]) :- write(H1), write(' '), write(H2), write('\n'), print(T1,T2).

solve(X) :-

people(S0), % S0 - начальный порядок имен, WhoIsWho -- переставленный

permutation(S0, WhoIsWho),

speedup(WhoIsWho),

people([X1,X2,X3,X4,X5]),

get\_friendlist(R1, X1),

length(R1, 1), % leonide

get\_friendlist(R5, X5), % petr

length(R5, 3),

get\_friendlist(R2, X2), % michael

member(oleg, R2),

member(nikolay, R2),

get\_friendlist(R3, X3), % nikolay

member(oleg, R3),

member(michael, R3),

get\_friendlist(R4, X4), % oleg

member(michael, R4),

member(nikolay, R4),

Solve = [R1, R2, R3, R4, R5],

info(WhoIsWho, Solve),

foreach\_friend(R1, X1, Solve),

foreach\_friend(R2, X2, Solve),

foreach\_friend(R3, X3, Solve),

foreach\_friend(R4, X4, Solve),

foreach\_friend(R5, X5, Solve),

write('SOLVE:\n'),

surname(Sn),

print(WhoIsWho,Sn),

write([R1, R2, R3, R4, R5]),

write('\n'),

X = WhoIsWho.

foreach\_friend([],\_,\_).

foreach\_friend([H|T], Name, Solve) :- know(H, Name, Solve), foreach\_friend(T, Name, Solve).

*Запрос.*

Solve(X).

*Результат.*

SOLVE:

nikolay atarov

michael bartenev

leonid klenov

petr danilin

oleg ivanov

[[petr],[nikolay,oleg],[michael,oleg,petr],[michael,nikolay,petr],[leonid,nikolay,oleg]]

X = [nikolay, michael, leonid, petr, oleg]

**Выводы**

Программа была написана не совсем в декларативном стиле. Встречались переборы (combination, permutation) и повторения условий задачи. Однако, именно Пролог позволяет легко задавать предикаты, как например предикат info, который содержит полное условие задачи.

Сложность задачи за счет переборов – факториальная. Но т.к. используются дополнительные проверки и условия, то сложность немного уменьшается и работает за приемлемое время.