Логическое программирование

Кевролетин В.В. группа c8503a(256)

12 November 2012

Содержание

1 Задание 5

1.1 Условие

During certain local floods five married couples found themselves surrounded by water, and had to escape from their unpleasant position in a boat that would only hold three persons at a time. Every husband was so jealous that he would not allow his wife to be in the boat or on either bank with another man (or with other men) unless he was himself present. Show the quickest way of getting these five men and their wives across into safety.

1.2 Решение

Для решения задачи используем метод поиска в ширину, т.к. он гарантированно находит кратчайшее решения. Воспользуемся фреймворком из книги "Искусство Пролога" для beast-first search и используя весовую фукнцию, возвращающую для всех состояний одно и то же значение поиск в ширинуую. Для фреймворка необходимо определить в каком виде будет храниться состояние, ходы, предикаты, проверяющие допустимость хода, допустимость состояния, переход из одного состояние в другое, весовую функцию, исходное и конечное состояния. Все это описано ниже.

1.2.1 Исходный код

- Фреймворк

  Фреймворк взят из книги "Искусство Пролога" почти без изменений. Единственное изменение - предикат insert был модифицирован так, чтобы при совпадании весов двух состояний новое состояние вставлялось в конец, а не в начало очереди. Это необходимо, для того, чтобы в случае весовой функции, возвращающей константу получить поиск в ширину.

```
solve_best([state(State,Path,Value)|Frontier],History,Moves) :-
        final_state(State),
        reverse(Path,Moves).
```

```
solve_best([state(State,Path,Value)|Frontier],History,FinalPath) :-
        findall(M,move(State,M),Moves),
        updates(Moves,Path,State,States),
        legals(States,States1),
        news(States1,History,States2),
        evaluates(States2,Values),
        inserts(Values,Frontier,Frontier1),
        solve_best(Frontier1,[State|History],FinalPath).

updates([Move|Moves],Path,State,[(State1,[Move|Path])|States]) :-
        update(State,Move,State1), updates(Moves,Path,State,States).
updates([],Path,State,[]).

legals([(S,P)|States],[(S,P)|States1]) :-
        legal(S), legals(States,States1).
legals([(S,P)|States],States1) :-
        \+ legal(S), legals(States,States1).
legals([],[]).

news([(State,Path)|States],History,States1) :-
        member(State,History), news(States,History,States1).
news([(State,Path)|States],History,[(State,Path)|States1]) :-
        \+ member(State,History), news(States,History,States1).
news([],History,[]).

evaluates([(State,Path)|States],[state(State,Path,Value)|Values]) :-
        value(State,Value),
        evaluates(States,Values).
evaluates([],[]).

inserts([Value|Values],Frontier,Frontier1) :-
        insert(Value,Frontier,Frontier0),
        inserts(Values,Frontier0,Frontier1).
inserts([],Frontier,Frontier).

insert(State,[],[State]).
insert(State,[State1|States],[State,State1|States]) :-
        less_value(State,State1).
insert(State,[State1|States],[State1|States1]) :-
        greatereq_value(State,State1), insert(State,States,States1).

equals(state(S,P,V),state(S,P1,V)).

less_value(state(S1,P1,V1),state(S2,P2,V2)) :- S1 \== S2, V1 =< V2.

greatereq_value(state(S1,P1,V1),state(S2,P2,V2)) :- V1 >= V2.

solve_bfs(Moves) :-
        initial_state(State),
        solve_best([state(State,[],0)],[],Moves).
```

```
% Program 20.6      Best first framework for problem solving
```

- Состояние и весовая функция

  Весовая функция любому состоянию сопоставляет одно и то же число.

  Состояние описывается термом вида:

  - p(BoatSide, LeftSide, RightSide)
    - BoatSide - сторона на которой находится лодка (isle, mainland)
    - LeftSide - список людей, находящихся на острове(отсортирован)
    - RightSide - список людей, находящихся на материке(отсортирован)

  Человек обозначается двузначным чилом, где первая цифра это пол(1 - мужчина, 2 - женщина), вторая цифра говорит о том, к какой семейной паре принадлежит человек.

```
value(_, 0).

initial_state(p(isle, [11, 12, 13, 14, 15, 21, 22, 23, 24, 25], [])).
final_state(p(mainland, [], [11, 12, 13, 14, 15, 21, 22, 23, 24, 25])).
```

- Переход между состояниями

  Описание предикатов, использованных в решении:

  - rest(Xs, X, Ys) - истина, если Ys - хвост списка Xs, расположенный сразу после элемента X
  - move(State, Move) - истина, если из текущего состояния State можно сделать ход Move. Ход описывается списком отсортированных элементов, люди, которые поедут с одного берега ну другой. Ход можно сделать, если люди находятся на том берегу, на котором сейчас лодки и если в лодке будут одни женщины, либо каждая женщина будет с мужем.
  - legal_content(Move) - истина, если Move содержит список людей, которые по правилам можно усадить в лодку(см. описание move).
  - couple(X, Y) - истина, если X, Y - номера, описывающие мужа и жену.
  - only_wives(Move) - истина, если список Move содержит только номера, описывающие женщин.
  - wives_with_husbands(List) - истина, если для каждой женщины из списка List в этом же списке найдётся её муж.
  - update(State, Move, NewState) - истина, если состояние NewState получется из состояния State после перемещения людей из списка Move с одного берега на другой
    - State - состояние
    - Move - отсортированный список, содержащий идентефикаторые людей
    - NewState - состояние
  - ordered_delete(A, B, C) - истина, если с получется после удаления из списка A всех элементов списка B. Подразумевается что елементы A, B отсортированы
  - $\text{ordered}_{\text{insert}}$(A, B, C) - истина, если отсортированный список C содержит элементы из списков A, B. Подразумевается, что A, B - отсортированы.
  - legal(State) - истина, если состояние State допустимо

Код:

```
% rest

rest([X|Xs], X, Xs).
rest([_|Xs], Y, Zs) :- rest(Xs, Y, Zs).

% move

move(p(isle, I, _), [P1]):-
  rest(I, P1, _).
move(p(isle, I, _), [P1, P2]):-
  rest(I, P1, I1), rest(I1, P2, _),
  legal_content([P1, P2]).
move(p(isle, I, _), [P1, P2, P3]):-
  rest(I, P1, I1), rest(I1, P2, I2), rest(I2, P3, _),
  legal_content([P1, P2, P3]).
move(p(mainland, _, M), [P1]):-
  rest(M, P1, _).
move(p(mainland, _, M), [P1, P2]):-
  rest(M, P1, M1), rest(M1, P2, _),
  legal_content([P1, P2]).
move(p(mainland, _, M), [P1, P2, P3]):-
  rest(M, P1, M1), rest(M1, P2, M2), rest(M2, P3, _),
  legal_content([P1, P2, P3]).

% legal_content

legal_content(Xs) :- only_wives(Xs), !.
legal_content(Xs) :- wives_with_husbands(Xs, Xs).

% only_wives

only_wives([]).
only_wives([W|Xs]) :- couple(_, W), only_wives(Xs).

% wives_with_husbands

wives_with_husbands([], _).
wives_with_husbands([H|Xs], Ys):-
  couple(H, _), !, wives_with_husbands(Xs, Ys).
wives_with_husbands([W|Xs], Ys):-
  couple(H, W), rest(Ys, H, _), !, wives_with_husbands(Xs, Ys).

% couple

couple(11, 21).
couple(12, 22).
couple(13, 23).
couple(14, 24).
```

```
couple(15, 25).

% update

update(p(isle, I, M), Boat, p(mainland, I1, M1)) :-
  ordered_delete(Boat, I, I1),
  ordered_insert(Boat, M, M1).
update(p(mainland, I, M), Boat, p(isle, I1, M1)) :-
  ordered_delete(Boat, M, M1),
  ordered_insert(Boat, I, I1).

% ordered_delete

ordered_delete([], Ys, Ys).
ordered_delete([X|Xs], [X|Ys], Zs) :- !,
  ordered_delete(Xs, Ys, Zs).
ordered_delete([X|Xs], [Y|Ys], Zs) :-
  X > Y, !, Zs = [Y|Ws], ordered_delete([X|Xs], Ys, Ws). %Zs is [Y|Ws].
ordered_delete([_|Xs], [Y|Ys], [Y|Zs]) :-
  ordered_delete(Xs, Ys, Zs).

% ordered_insert

ordered_insert([], Ys, Ys).
ordered_insert(Xs, [], Xs).
ordered_insert([X|Xs], [Y|Ys], Zs) :-
  X >= Y, !, Zs = [Y|Ws], ordered_insert([X|Xs], Ys, Ws).
ordered_insert([X|Xs], Ys, [X|Zs]) :-
  ordered_insert(Xs, Ys, Zs).

% legal

legal(p(_, Xs, Ys)) :-
  legal_content(Xs), legal_content(Ys).
```

1.2.2 Тесты

Метод поиска в ширину позволяет найти решение длиной 45. Для сравнени приведу результаты полученные для этой задачи другими методами:

- beast-first search с весовой функцией сопоставляющей состоянию число людей на материке - длина решения 79.
- dfs - длина решения 97.

```
?- solve_bfs(X), print_ans(X), length(X, Len).
w3 w4 w5
w4 w5
w2 w4 w5
w3 w4 w5
w1 w4 w5
```

```
w2 w4 w5
w3 w4 w5
w1 w3 w5
w1 w2 w3
w3 w4
w3 w4 w5
w2 w4 w5
w2 w5
w5
h1 h2 h3
h3 w3
h3 h4 h5
w1 w2
w3 w4 w5
w4 w5
w2 w4 w5
w3 w4 w5
w1 w4 w5
w2 w4 w5
w3 w4 w5
w1 w3 w5
w1 w2 w3
h4 h5 w4
h4 h5 w5
w1 w2 w3
w2 w4
h1 h3
h3 w3
h5 w5
h1 w1
h3 w3
h5 w5
h4 w4
h3 w3
h2 w2
h4 w4
h5 w5
h2 h5
w4
w2 w4 w5
---
X = [[23, 24, 25], [24, 25], [22, 24, 25], [23, 24, 25], [21, 24, 25], [22, 24, 25], [23
Len = 45

 ?-
```