Логическое программирование

Кевролетин В.В. группа c8503a(256)

12 November 2012

Содержание

1 Задание 7

1.1 Условие

Применить подход, реализующий метод подъема на вершину, к решению задачи о волке, козе и капусте. Сравнить результаты со случаем, когда используется поиск в глубину.

1.2 Решение

Метод подъема на вершину является обобщением метода поиска в глубину, поэтому код, используемый для поиска решения в глубину можно легко использовать(при правильном структурировании кода) для метода подъема в глубину. Единственное отличие - необходимо реализовать весовую функцию. В качестве весовой функции возьмем количество животных на правом берегу:

```
value(wgc(_, _, Right), Res) :- length(Right, Res).
```

1.2.1 Исходный код

- Фреймворк

```
solve_hill_climb(State,History,[]) :-
        final_state(State).
solve_hill_climb(State,History,[Move|Moves]) :-
        hill_climb(State,Move),
        update(State,Move,State1),
        legal(State1),
        \+ member(State1,History),
        solve_hill_climb(State1,[State1|History],Moves).

hill_climb(State,Move) :-
```

```
        findall(M,move(State,M),Moves),
        evaluate_and_order(Moves,State,[],MVs),
        member((Move,Value),MVs).

evaluate_and_order([Move|Moves],State,MVs,OrderedMVs) :-
        update(State,Move,State1),
        value(State1,Value),
        insert((Move,Value),MVs,MVs1),
        evaluate_and_order(Moves,State,MVs1,OrderedMVs).
evaluate_and_order([],State,MVs,MVs).

insert(MV,[],[MV]).
insert((M,V),[(M1,V1)|MVs],[(M,V),(M1,V1)|MVs]) :-
        V >= V1.
insert((M,V),[(M1,V1)|MVs],[(M1,V1)|MVs1]) :-
        V < V1, insert((M,V),MVs,MVs1).

/*      Testing the Framework */

test_hill_climb(Problem,Moves) :-
        initial_state(Problem,State),
        solve_hill_climb(State,[State],Moves).
```

- Задача о волке, козе и капусте

```
initial_state(wgc,wgc(left,[wolf,goat,cabbage],[])).

value(wgc(_, _, Right), Res) :- length(Right, A).

final_state(wgc(right,[],[wolf,goat,cabbage])).

move(wgc(left,L,R),Cargo) :- member(Cargo,L).
move(wgc(right,L,R),Cargo) :- member(Cargo,R).
move(wgc(B,L,R),alone).

update(wgc(B,L,R),Cargo,wgc(B1,L1,R1)) :-
        update_boat(B,B1), update_banks(Cargo,B,L,R,L1,R1).

update_boat(left,right).
update_boat(right,left).

update_banks(alone,B,L,R,L,R).
update_banks(Cargo,left,L,R,L1,R1) :-
        select(Cargo,L,L1), insert(Cargo,R,R1).
update_banks(Cargo,right,L,R,L1,R1) :-
        select(Cargo,R,R1), insert(Cargo,L,L1).

insert(X,[Y|Ys],[X,Y|Ys]) :-
        precedes(X,Y).
insert(X,[Y|Ys],[Y|Zs]) :-
```

```
        precedes(Y,X), insert(X,Ys,Zs).
insert(X,[],[X]).

precedes(wolf,X).
precedes(X,cabbage).

legal(wgc(left,L,R)) :- \+ illegal(R).
legal(wgc(right,L,R)) :- \+ illegal(L).

illegal(Bank) :- member(wolf,Bank), member(goat,Bank).
illegal(Bank) :- member(goat,Bank), member(cabbage,Bank).

select(X,[X|Xs],Xs).
select(X,[Y|Ys],[Y|Zs]) :-
        select(X,Ys,Zs).
```

1.2.2 Тесты

Метод подъёма на вершину позволяет найти решение задачи. Решение отлично от решения, полученного поиском в глубину но состоит из такого же числа действий:

```
?- test_dfs(wgc, X).
X = [goat, alone, wolf, goat, cabbage, alone, goat]

?- test_hill_climb(wgc, X).
X = [goat, alone, cabbage, goat, wolf, alone, goat]

?-
```