ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

Bài Toán: Người Nông Dân Sang Sông
Môn Học: Nguyên Lý và Phương Pháp Lập Trình

Giáo viên hướng dẫn: thầy Nguyễn Tuấn Đăng
Sinh viên thực hiện: Nguyễn Hoàng Trung
MSSV 17521176

1. Bài Toán

Một người nông dân có 1 con sói, 1 con dê và 1 bắp cải ở bên bờ sông. Ông muốn tất cả cùng sang bờ bên kia sông. Người nông dân chỉ có 1 chiếc thuyền, và trên thuyền ngoài người nông dân ra chỉ có thể chở được 1 trong 3 vật trên. Không được để con sói và con dê ở cùng 1 bờ sông khi không có người nông dân, tương tự với dê và bắp cải. Làm thế nào để tất cả có thể cùng sang sông.

2. Code Prolog

```
transition(S,G):- operation(S,G).
%F is for Farmer
%W is for Wolf
%G is for Goat
%C is for Cabbage

%invalid
invalid([0,1,1,1]).
invalid([0,0,1,1]).

invalid([1,1,0,0]).
invalid([1,0,0,0]).

invalid([0,1,1,0]).
invalid([1,0,0,1]).
%goal
goal([1,1,1,1]).
%Move both
operation([1,W,1,C],[0,W,0,C]).
operation([0,W,0,C],[1,W,1,C]).

operation([0,0,G,C],[1,1,G,C]).
operation([1,1,G,C],[0,0,G,C]).

operation([0,W,G,0],[1,W,G,1]).
operation([1,W,G,1],[0,W,G,0]).
%Farmer moves
operation([0,W,G,C],[1,W,G,C]).
operation([1,W,G,C],[0,W,G,C]).
dfs(CurrentNode, CurrentPath, [CurrentNode, CurrentPath]):- goal(CurrentNode).
dfs(CurrentNode,CurrentPath,Path):-transition(CurrentNode,NextNode),
CurrentNode \= NextNode,
\+member(NextNode,CurrentPath),
\+invalid(CurrentNode),
dfs(NextNode,[CurrentNode|CurrentPath],Path).
```

Đoạn code để giải bài toán người nông dân sang sông.

```
?- dfs([0,0,0,0],[],X).
X = [[1, 1, 1, 1], [[0, 1, 0, 1], [1, 1, 0, 1], [0, 1, 0, 0], [1, 1, 1|...], [0, 0|...], [1|...], [...|...]]] ;
X = [[1, 1, 1, 1], [[0, 1, 0, 1], [1, 1, 0, 1], [0, 0, 0, 1], [1, 0, 1|...], [0, 0|...], [1|...], [...|...]]] ;
false
```

Kết quả chương trình.

3. Diễn Giải

Quy ước:

- S,G là 2 list để diễn tả trạng thái bắt đầu(hiện tại) và trạng thái đích(tiếp theo) của bài toán, thể hiện số nước của người nông dân và 3 vật, được truyền vào hàm transition để xử lý.

- Trạng thái 0 biểu diễn vật đang ở bờ sông bên trái, trạng thái 1 biểu diễn vật đang ở bờ sông bên phải.
- F,W,G,C đại diện cho trạng thái của người nông dân, con sói, dê và bắp cải.
- Trạng thái bắt đầu là [0,0,0,0], khi mà cả bốn đều ở bờ bên trái.
- Trạng thái kết thúc là [1,1,1,1], khi mà cả bốn đề ở bờ bên phải.

Các hành động được thử hiện:

- Người nông dân đi 1 mình sang sông:

```
operation([0,W,G,C],[1,W,G,C]).
operation([1,W,G,C],[0,W,G,C]).
```

- Người nông dân chở 1 vật theo:

```
operation([1,W,1,C],[0,W,0,C]).
operation([0,W,0,C],[1,W,1,C]).

operation([0,0,G,C],[1,1,G,C]).
operation([1,1,G,C],[0,0,G,C]).

operation([0,W,G,0],[1,W,G,1]).
operation([1,W,G,1],[0,W,G,0]).
```

- Áp dụng thuật toán DFS để có thể tìm ra tất cả lời giải của bài toán.

```
dfs(CurrentNode, CurrentPath, [CurrentNode, CurrentPath]):- goal(CurrentNode).
dfs(CurrentNode,CurrentPath,Path):-transition(CurrentNode,NextNode),
CurrentNode \= NextNode,
\+member(NextNode,CurrentPath),
dfs(NextNode,[CurrentNode|CurrentPath],Path).
```

Trạng thái kết thúc của bài toán:

```
goal([_,Y,_]) :- Y = 5.
goal([_,_,Z]) :- Z = 5.
```

Trạng thái bị cấm: người nông dân không được để sói và dê ở riêng, cũng như dê và bắp cải.

```
 %invalid
invalid([0,1,1,1]).
invalid([0,0,1,1]).

invalid([1,1,0,0]).
invalid([1,0,0,0]).

invalid([0,1,1,0]).
invalid([1,0,0,1]).
```

Diễn giải:

+ CurrentNode là trạng thái hiện tại, CurrentPath là list gồm các trạng thái đã duyệt (bao gồm Current Node).

+ Nếu CurrentNode không phải là trạng thái đích, ta chuyển sang tìm tất cả các trạng thái có thể nảy sinh từ trạng thái hiện tại; với các ràng buộc như trạng thái kế tiếp (NextNode) phải khác CurrentNode, NextNode không thuộc các trường hợp đã duyệt qua từ CurrentNode và NextNode không thuộc các trường hợp bị cấm.

+ Cho CurrentNode vào list các trạng thái đã duyệt. Tiếp tục gọi hàm dfs bắt đầu từ NextNode. Thực hiện đệ quy cho đến khi tìm được trạng thái kết thúc của bài toán.