Olivia Fang
UWNetIDs: olivfwm

Assignment 2, Part 1, in CSE 415, Winter 2024

This file contains my blind search solutions

Part 1 Step 4 (c)

if h_remaining > 0 and h_remaining < r_remaining: return False

This line checks if the remaining humans on the side that the ferry leaves from is greater than 0, and outnumbered by robots. If it is, then the robots will rebel and the function returns false for can't move.

h_at_arrival = p[H][1-side]+h

r_at_arrival = p[R][1-side]+r

Then the function calculates the number of humans and robots on the side that the ferry will arrive at, if the current move was made. *h* is added to the number of humans on the arrival side, and stored in *h_at_arrival*. Same with *r_at_arrival.* Then these variables can be compared.

Part 1 Step 8

BFS:

	Humans, Robots and Ferry	Farmer, Fox, Chicken, and Grain	Disk Towers of Hanoi
path from start to goal	H on left:3 R on left:3 H on right:0 R on right:0 ferry is on the left. H on left:2 R on left:2 H on right:1 R on right:1 ferry is on the right. H on left:3	Left bank:['fox', 'chicken', 'grain'] Right bank:[] farmer is on the left. Left bank:['fox', 'grain'] Right bank:['chicken'] farmer is on the right. Left bank:['fox', 'grain'] Right bank:['chicken']	[[4, 3, 2, 1] ,[] ,[]] [[4, 3, 2] ,[1] ,[]] [[4, 3] ,[1] ,[2]] [[4, 3] ,[] ,[2, 1]] [[4] ,[3] ,[2, 1]] [[4, 1] ,[3] ,[2]]

	R on left:2
	  H on right:0
	  R on right:1
	ferry is on the left.
	
	H on left:0
	R on left:2
	  H on right:3
	  R on right:1
	ferry is on the right.
	
	H on left:2
	R on left:2
	  H on right:1
	  R on right:1
	ferry is on the left.
	
	H on left:0
	R on left:1
	  H on right:3
	  R on right:2
	ferry is on the right.
	
	H on left:1
	R on left:1
	  H on right:2
	  R on right:2
	ferry is on the left.
	
	H on left:0
	R on left:0
	  H on right:3
	  R on right:3
	ferry is on the right.
		farmer is on the left.
		
		Left bank:['fox']
		Right bank:['chicken', 'grain']
		farmer is on the right.
		
		Left bank:['fox', 'chicken']
		Right bank:['grain']
		farmer is on the left.
		
		Left bank:['chicken']
		Right bank:['grain', 'fox']
		farmer is on the right.
		
		Left bank:['chicken']
		Right bank:['grain', 'fox']
		farmer is on the left.
		
		Left bank:[]
		Right bank:['grain', 'fox', 'chicken']
		farmer is on the right.
			[[4, 1] ,[3, 2] ,[]]
			
			[[4] ,[3, 2, 1] ,[]]
			
			[[] ,[3, 2, 1] ,[4]]
			
			[[] ,[3, 2] ,[4, 1]]
			
			[[2] ,[3] ,[4, 1]]
			
			[[2, 1] ,[3] ,[4]]
			
			[[2, 1] ,[] ,[4, 3]]
			
			[[2] ,[1] ,[4, 3]]
			
			[[] ,[1] ,[4, 3, 2]]
			
			[[] ,[] ,[4, 3, 2, 1]]

length	7 edges	7 edges	15 edges
nodes expanded	10 nodes	9 nodes	70 nodes

DFS:

	Humans, Robots and	Farmer, Fox, Chicken,	Disk Towers of

	Ferry	and Grain	Hanoi
path from start to goal	H on left:3 R on left:3 H on right:0 R on right:0 ferry is on the left. H on left:2 R on left:2 H on right:1 R on right:1 ferry is on the right. H on left:3 R on left:2 H on right:0 R on right:1 ferry is on the left. H on left:0 R on left:2 H on right:3 R on right:1 ferry is on the right. H on left:2 R on left:2 H on right:1 R on right:1 ferry is on the left. H on left:1 R on left:1 H on right:2 R on right:2 ferry is on the right. H on left:3 R on left:1 H on right:0 R on right:2 ferry is on the left. H on left:0 R on left:1	Left bank:['fox', 'chicken', 'grain'] Right bank:[] farmer is on the left. Left bank:['fox', 'grain'] Right bank:['chicken'] farmer is on the right. Left bank:['fox', 'grain'] Right bank:['chicken'] farmer is on the left. Left bank:['grain'] Right bank:['chicken', 'fox'] farmer is on the right. Left bank:['grain', 'chicken'] Right bank:['fox'] farmer is on the left. Left bank:['chicken'] Right bank:['fox', 'grain'] farmer is on the right. Left bank:['chicken'] Right bank:['fox', 'grain'] farmer is on the left. Left bank:[] Right bank:['fox', 'grain', 'chicken'] farmer is on the right.	[[4, 3, 2, 1] ,[] ,[]] [[4, 3, 2] ,[1] ,[]] [[4, 3] ,[1] ,[2]] [[4, 3, 1] ,[] ,[2]] [[4, 3] ,[] ,[2, 1]] [[4] ,[3] ,[2, 1]] [[4, 1] ,[3] ,[2]] [[4] ,[3, 1] ,[2]] [[4, 2] ,[3, 1] ,[]] [[4, 2, 1] ,[3] ,[]] [[4, 2] ,[3] ,[1]] [[4] ,[3, 2] ,[1]] [[4, 1] ,[3, 2] ,[]] [[4] ,[3, 2, 1] ,[]] [[] ,[3, 2, 1] ,[4]] [[1] ,[3, 2] ,[4]] [[] ,[3, 2] ,[4, 1]] [[2] ,[3] ,[4, 1]] [[2, 1] ,[3] ,[4]] [[2] ,[3, 1] ,[4]] [[] ,[3, 1] ,[4, 2]] [[1] ,[3] ,[4, 2]] [[] ,[3] ,[4, 2, 1]] [[3] ,[] ,[4, 2, 1]] [[3, 1] ,[] ,[4, 2]] [[3] ,[1] ,[4, 2]] [[3, 2] ,[1] ,[4]] [[3, 2, 1] ,[] ,[4]] [[3, 2] ,[] ,[4, 1]] [[3] ,[2] ,[4, 1]] [[3, 1] ,[2] ,[4]] [[3] ,[2, 1] ,[4]] [[] ,[2, 1] ,[4, 3]] [[1] ,[2] ,[4, 3]] [[] ,[2] ,[4, 3, 1]] [[2] ,[] ,[4, 3, 1]] [[2, 1] ,[] ,[4, 3]] [[2] ,[1] ,[4, 3]] [[] ,[1] ,[4, 3, 2]] [[1] ,[] ,[4, 3, 2]]

	H on right:3 R on right:2 ferry is on the right. H on left:1 R on left:1 H on right:2 R on right:2 ferry is on the left. H on left:0 R on left:0 H on right:3 R on right:3 ferry is on the right.		[[] ,[] ,[4, 3, 2, 1]]
length	9 edges	7 edges	40 edges
nodes expanded	10 nodes	7 nodes	40 nodes

Observations:

For the Towers of Hanoi, the maximum length of the OPEN list is more for BFS than DFS. This is because BFS searches by “layers”, while DFS finishes searching into one path before going to other paths. BFS will keep on adding to the *end* of the OPEN list when it finds more possible states to search. DFS is more likely to finish searching most (but not all) of the nodes in OPEN list before adding more to it.

The solution PATH length is different for one algorithm from that of the other is because DFS searches through each path until there are no more successors before it moves on to other paths. DFS will stop once it finds a solution, and usually at that point, it has not searched much of the other paths. So this solution path that it found is not likely to be the most efficient. While BFS searches in “layers”, it usually reaches the goal state with longer time than DFS. So it searches more nodes than DFS does, and might sometimes find a more efficient path. Therefore, BFS and DFS’ solution paths have different lengths.