

CSE 4082 – Assignment 1

- **Describe the problem formally and classify its environment.**

States: A knight's tour path includes board squares. First node is (1,1), and its first child node is (1,1)→(2,3), second child node is (1,1)→(3,2)

Initial State: Knight starts at square (1,1)

Actions: Knight can be moved 2 forward/backward and 1 left/right squares anywhere it is in board squares. So totally maximum 8 actions.

Transition Model: Knight's moves returns an added square into the frontier and visited list and a changed position of Knight.


Goal Test: All squares are visited, no unvisited squares left and knight has visited all of the squares only one time. (If a node has $N*N$ path elements, this node is goal. Each path elements is unique.)

In Knight's Tour Problem, environment is **fully observable, deterministic, sequential, static and discrete**. There is only one agent in this task. So it is **single agent** problem.

- **Implementation Details**

LocationMatrix

Defines possible action of a knight can take.

	6		5	
7				4
				
0				3
	1		2	

Init

We first create root node and its path with (1,1).

Bfs

Firstly we control whether root node is goal or not. If $N=1$ root would be goal. Then we add root to the frontier. Frontier is a linked list created with nodes. At the beginning of program it is NULL. If frontier is not null we pop the node which is in the beginning of the frontier. This popped node will be expanded. We need to calculate all children of it. So we call findAllChildren function. We firstly calculate location of the popped node by calling getLastAction. For example if node's is pathHead → (1,1) → (2,3) then getLastLocation returns (2,3). We consider totally 8 possible actions. So by adding locationMatrix's each value with getLastLocation return value we get next location. If next location is greater than 0 and smaller than or equal to N it is valid. Also next location should not be in the path of parentNode(popped node) in order to be valid. For example when the pathHead → (1,1) → (2,3) we can not consider (1,1) as next location. If childNode is valid and is not goal we add it to the frontier. We always add new node to the end of the frontier. If childNode is

valid and is goal it means we do not expand more nodes and return solutionFound. After finding all children of the popped node we pop next node from frontier. This pop operation always gets the first node in the frontier (popFromBeginningFrontier). By adding at the end of the frontier and popping from the beginning of the frontier, we are searching depth by depth. If frontier is empty(NULL) at any point we will return failure, since all nodes are expanded.

Dfs

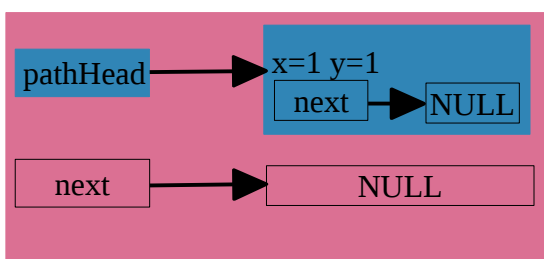
While bfs gets the node from the beginning of the frontier, dfs gets the node from the end of the frontier. By doing this, dfs expands lastly added child. It goes through one way from the root to the N-1th depth.

DfsH

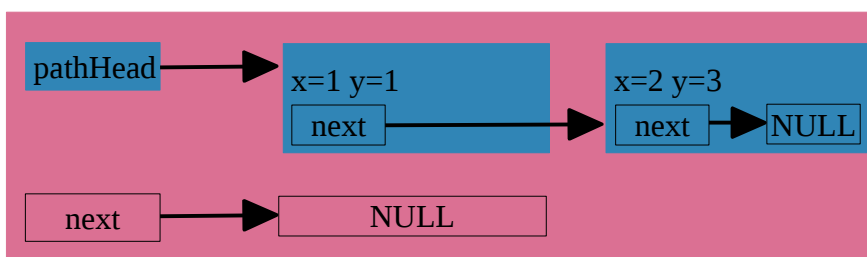
In dfsH we pop from end of the frontier like we did in dfs. But the difference is that we add to the frontier by priority. This priority is defined by heuristic value. Heuristic value is calculated by counting number of possible actions of a popped node from frontier. Children are sorted in descending order by looking their number of possible of actions. If two children have same number of possible actions then we calculate their distances to any of the corners. Whichever has min distance will come later in the sorted list. Then we add this children list (sorted by looking their priority value) to frontier. Which node has the least number of children and closer to the corner will be added at the end of the frontier. So when we pop from the end of the frontier we get the node with the most priority and expand it.

y=6						
y=5						
y=4						
y=3		②				
y=2						
y=1	①					
	x=1	x=2	x=3	x=4	x=5	x=6

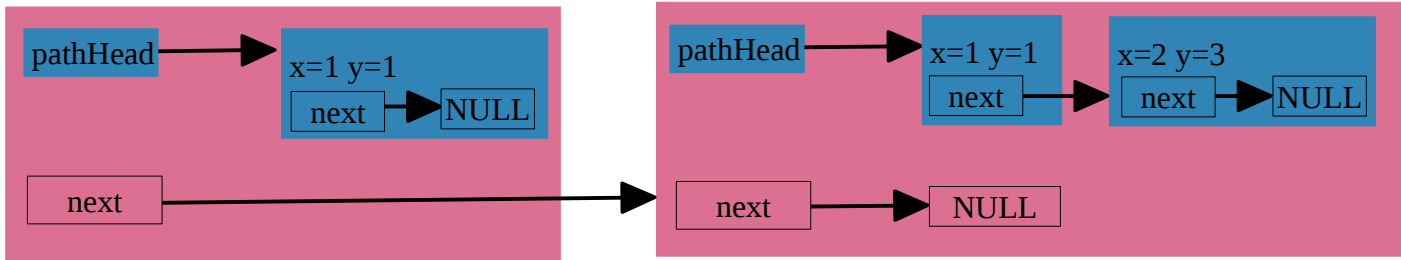
In this project we have used nodes and paths as in the following.



1 node with 1 path inside.



1 node with 2 path inside.



2 node with paths inside.

Table 1. (Time limit is 30 min.)

	n=6	n=8	n=12	n=16	n=20
bfs	Timeout.	Timeout.	Timeout.	Timeout.	Timeout.
dfs	A solution found.	A solution found.	Out of Memory.	Out of Memory.	Out of Memory.
dfsH	A solution found.	A solution found.	A solution found.	A solution found.	A solution found.

For compiling

gcc project.c -o project -lm (For Linux)

gcc project.c -o project (For Windows)

For running

project n searchMethod(a=bfs, b=dfs, c=dfsH) timeLimit(min)

./project n searchMethod(a=bfs, b=dfs, c=dfsH) timeLimit(min)

ex: ./project 6 a 30(For Linux)

ex: project 6 a 30(For Windows)s

bfs

```
C:\Users\Emre\Desktop>project 8 a 30
Program is running with N=8 searchMethod(a=bfs, b=dfs, c=dfsH)=a timeLimit(min)=30...
Timeout.
time(sec)=1800.039000 noOfNodeExpanded=82352

C:\Users\Emre\Desktop>project 12 a 30
Program is running with N=12 searchMethod(a=bfs, b=dfs, c=dfsH)=a timeLimit(min)=30...
Timeout.
time(sec)=1800.006000 noOfNodeExpanded=63249

C:\Users\Emre\Desktop>project 16 a 30
Program is running with N=16 searchMethod(a=bfs, b=dfs, c=dfsH)=a timeLimit(min)=10...
Timeout.
time(sec)=1800.015000 noOfNodeExpanded=113001
```

```
C:\Users\Emre\Desktop>project 20 a 30
Program is running with N=20 searchMethod(a=bfs, b=dfs, c=dfsH)=a timeLimit(min)=30...
Timeout.
time(sec)=1800.068000 noOfNodeExpanded=56371

C:\Users\Emre\Desktop>_
```

dfs

```
feyza@feyza-Latitude-E7470:~/Desktop/Artificial Intelligence$ gcc project.c -o project -ln
feyza@feyza-Latitude-E7470:~/Desktop/Artificial Intelligence$ ./project 6 b 30
Program is running with N=6 searchMethod(a=bfs, b=dfs, c=dfsH)=b timeLimit(min)=30...
Solution Path->(x:1,y:1)->(x:2,y:3)->(x:1,y:5)->(x:3,y:6)->(x:5,y:5)->(x:6,y:3)->(x:4,y:4)->(x:2,y:5)->(x:4,y:6)->(x:6,y:5)->(x:5,y:3)->(x:3,y:4)->(x:2,y:6)->(x:4,y:5)->(x:6,y:6)->(x:5,y:4)->(x:6,y:2)->(x:4,y:1)->(x:2,y:2)->(x:1,y:4)->(x:3,y:3)->(x:5,y:2)->(x:6,y:4)->(x:5,y:6)->(x:3,y:5)->(x:1,y:6)->(x:2,y:4)->(x:1,y:2)->(x:3,y:1)->(x:4,y:3)->(x:5,y:1)->(x:3,y:2)->(x:1,y:3)->(x:2,y:1)->(x:4,y:2)->(x:6,y:1)
A solution found.
time(sec)=0.013198 noOfNodeExpanded=5422 memoryUsed(Bytes)=2752664
feyza@feyza-Latitude-E7470:~/Desktop/Artificial Intelligence$ ./project 8 b 30
Program is running with N=8 searchMethod(a=bfs, b=dfs, c=dfsH)=b timeLimit(min)=30...
Solution Path->(x:1,y:1)->(x:2,y:3)->(x:1,y:5)->(x:2,y:7)->(x:4,y:8)->(x:6,y:7)->(x:8,y:8)->(x:7,y:6)->(x:5,y:7)->(x:3,y:8)->(x:4,y:6)->(x:5,y:8)->(x:7,y:7)->(x:8,y:5)->(x:6,y:6)->(x:4,y:7)->(x:2,y:8)->(x:3,y:6)->(x:1,y:7)->(x:2,y:5)->(x:1,y:7)->(x:1,y:8)->(x:2,y:6)->(x:4,y:5)->(x:6,y:4)->(x:5,y:6)->(x:6,y:8)->(x:8,y:7)->(x:7,y:5)->(x:8,y:3)->(x:7,y:1)->(x:5,y:2)->(x:3,y:3)->(x:1,y:4)->(x:3,y:5)->(x:1,y:6)->(x:2,y:4)->(x:1,y:2)->(x:3,y:1)->(x:4,y:3)->(x:5,y:5)->(x:7,y:4)->(x:8,y:2)->(x:6,y:1)->(x:5,y:3)->(x:3,y:4)->(x:2,y:2)->(x:4,y:1)->(x:6,y:2)->(x:8,y:1)->(x:7,y:3)->(x:5,y:4)->(x:4,y:2)->(x:2,y:1)->(x:1,y:3)->(x:3,y:2)->(x:4,y:4)->(x:6,y:3)->(x:5,y:1)->(x:7,y:2)->(x:8,y:4)->(x:6,y:5)->(x:8,y:6)->(x:7,y:8)
A solution found.
time(sec)=6.819800 noOfNodeExpanded=3242065 memoryUsed(Bytes)=3083916048
feyza@feyza-Latitude-E7470:~/Desktop/Artificial Intelligence$ ./project 12 b 30
Program is running with N=12 searchMethod(a=bfs, b=dfs, c=dfsH)=b timeLimit(min)=30...
Out of Memory.
time(sec)=15.200445 noOfNodeExpanded=2503237 memoryUsed(Bytes)=5368710560
feyza@feyza-Latitude-E7470:~/Desktop/Artificial Intelligence$ ./project 16 b 30
Program is running with N=16 searchMethod(a=bfs, b=dfs, c=dfsH)=b timeLimit(min)=30...
Out of Memory.
time(sec)=17.095859 noOfNodeExpanded=1362264 memoryUsed(Bytes)=5368713048
feyza@feyza-Latitude-E7470:~/Desktop/Artificial Intelligence$ ./project 20 b 30
Program is running with N=20 searchMethod(a=bfs, b=dfs, c=dfsH)=b timeLimit(min)=30...
Out of Memory.
time(sec)=18.054143 noOfNodeExpanded=864531 memoryUsed(Bytes)=5368726096
```

[illegible]

We tested our program with the requested parameters and,

- RAM
 - 8 GB (1 x 8 GB)
- Max Supported Size
 - 16 GB
- Technology
 - DDR4 SDRAM
- Speed
 - 2133 MHz / PC4-17000 - 2133 MHz

For Windows PC:

Windows sürümü

Windows 10 Home Single Language

© 2017 Microsoft Corporation. Tüm hakları saklıdır.



Sistem

Üretici:	Samsung Electronics
İşlemci:	Intel(R) Core(TM) i5-3230M CPU @ 2.60GHz 2.60 GHz
Yüklü bellek (RAM):	8,00 GB (kullanılabilir miktar: 7,88 GB)
Sistem türü:	64 bit İşletim Sistemi, x64 tabanlı işlemci
Kalem ve Dokunma:	Bu Görüntü Biriminde Kalem Girdisi veya Dokunarak Giriş yok
