University of Massachusetts in Lowell
COMP4200 – Artificial Intelligence
Project Proposal
Students: Phong Vo (SID #01790283) – Giang Tran (SID #01670778)

Project: The Ant's Nest Building

I. Introduction:

The A* algorithm was first described in 1986 by Peter Hart, Nils Nilso and Bertram Raphael. In their report, the algorithm is called algorithm A, when using this algorithm with an appropriate heuristic evaluation function will obtain the optimal operation, hence the name A*.

In computer science, A* is a search algorithm in graphs. This algorithm finds a path from the originating node to a given destination node (or to a node that satisfies the target condition). This algorithm uses a heuristic rating to rank each node according to an estimate of the best route through that node. This algorithm browses nodes in the order of this heuristic evaluation. Therefore, the algorithm A* is an example of a best-first search.

Consider the problem of finding the way - the problem that A* is often used to solve. A* builds ascending all the routes from the starting point until it finds a path that reaches the destination. However, just as all search algorithms have information, it only builds routes that appear to be gradually reaching their destination.

To determine which routes are likely to lead to the destination, A* uses a heuristic evaluation function of the distance from any point to the destination. In the case of finding a route, this rating can be the distance the crow flies - a rough rating that is used for the distance of a road.

The difference of A* from the best option search is that it also takes into account the distance traveled. That makes A* complete and optimal, meaning that A* will always find the shortest path if such a path exists. A* does not guarantee to run faster than simpler search algorithms. In a labyrinth-like environment, the only way to get there is to first go to the far end and finally return. In that case, trying the buttons in the order "closer to the destination, first tried" can be time consuming.

II. Algorithm:

Suppose n is an attainable state (there is a path from the initial state 0 to n). We define the evaluation function: f (n) = g (n) + h (n)

+ g (n) is the cost from the original node n0 to the current node n

+ h (n) estimated cost from current node n to destination

+ f (n) the estimated total cost of the path through the current node n to the destination

A heuristic estimate of h (n) is considered to be acceptable if for every node n:

$$0 < h(n) < h^*(n)$$

Where h * (n) is the actual (actual) cost to go from node n to destination.

III. State Machine:

- Open: a set of states that have been born but not taken into account.
- Close: set of states that are considered.

Cost (p, q): is the distance between p, q.
g (p): distance from initial state to current state p.
h (p): value evaluated from the current state to the target state.
f (p) = g (p) + h (p).

Step 1:

Open: = {s}

Close: = {};

Step 2: while (Open! = {})

2.1 Choose the best state (top) p in Open (remove p from Open).

2.2 If p is the end state then exit.

2.3 Switch p to Close and create successive states q after p

a) If q is already in Open

If (g (q)> g (p) + cost (p, q))

g (q) = g (p) + cost (p, q)

f (q) = g (q) + h (q);

prev (q) = p // vertex of q is p

b) If q is not already in Open

g (q) = g (p) + cost (p, q);

f (q) = g (q) + h (q);

prev (q) = p;

Add q to Open.

c) If q is in Close

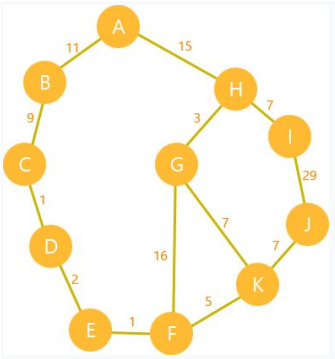
If (g (q)> g (p) + cost (p, q))

Remove q from Close;

Add q to Open.

Step 3: Can't find it.

IV. Graph:



h(A) = 60 , h(B) = 53 , h(C) = 36 , h(D) = 35 , h(E) = 35 , h(F) = 19 , h(G) = 16 , h(H) = 38 , h(I) = 23 , h(J) = 0 , h(K) = 7

Peak begins A.

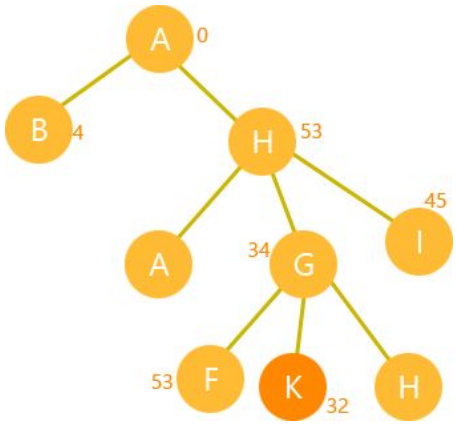
The ending ends K.

Estimate the distance from the current vertex to the ending vertex f (x) = g (x) + h (x) where g is the shortest distance from the current vertex to the destination.

Example f (A) = 0 + 60.

Step	P	The vertices connected to P	Open	Close
0			A60	
1	A	B, H	B64, H53	A
2	H	G, I, A	B64, G34, I45	A, H
3	G	H, K, F	B64, I45, K32, F53	A, H, G
4	K	G, F, J	B64, J32, F49, I45	A, H, G
5	K (stop)			

Search tree corresponding to the above graph.



V. Game' conception:

The project is developed with A* algorithm which is optimized for path finding in order to gain the reaching out of the leaves. There is an ant going to build its nest, literally marked by the UML logo, which is located at the top-left corner of the field. The purpose of the program is to optimize the path of the ant to be least walks as it can. The material for building up the nest are ten (10) leaves which are laying around the map. Every leaf might be blocked by ten (10) bricks. Hence, the ant must wisely reach out the leaves without hitting the walls. Once it gets a leaf, it brings it back to the nest and then continues heading to the other remaining leaves. The loop of collecting the leaves is finished once all of the leaves are gathered in the nest. The size of the field can be changed rather than 10x10. The number of leaves on the field can be also changed.

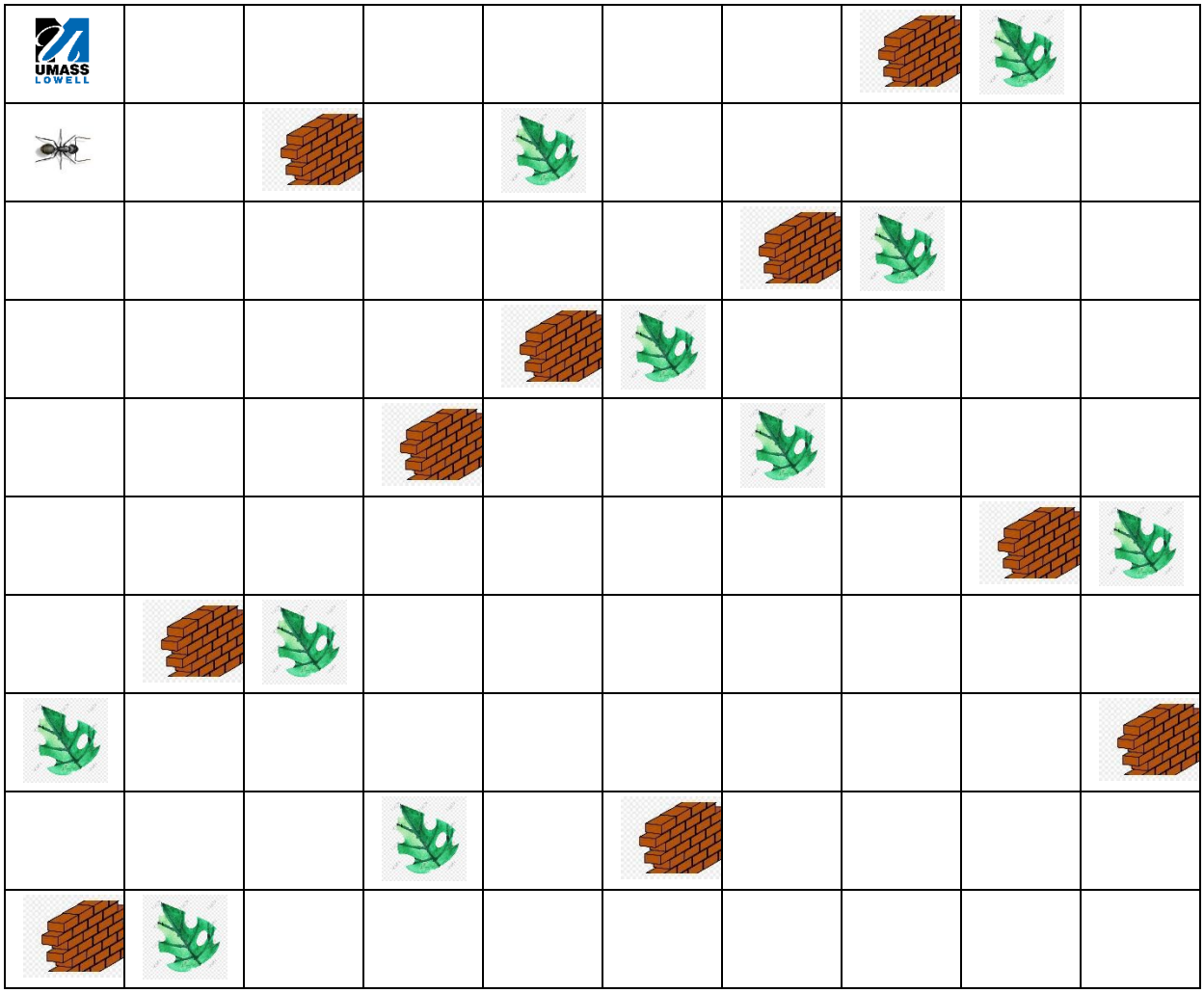


Fig. 1: The initial state of the program

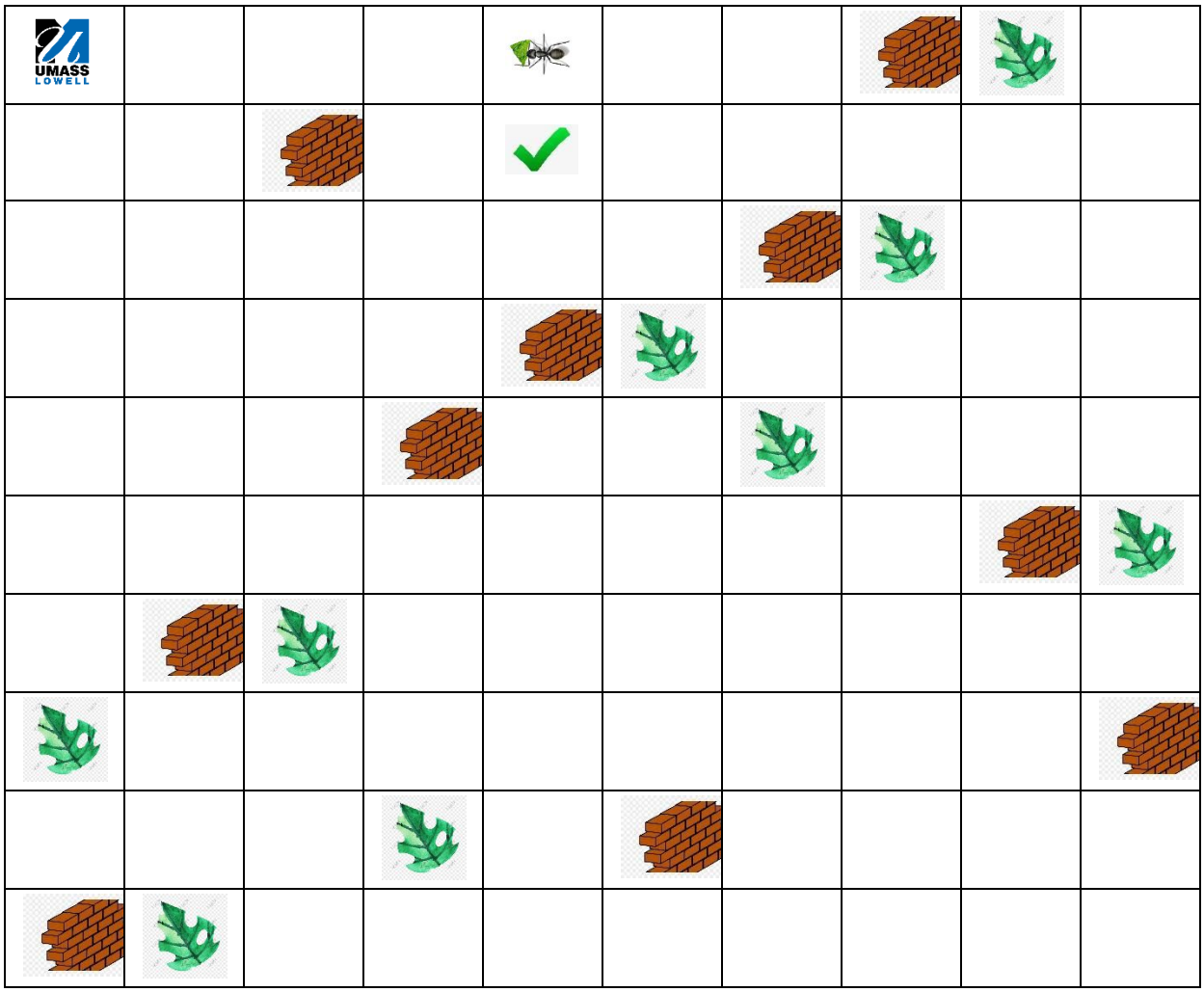


Fig. 2: The ant has successfully collected a leave, then bring back to the nest.

The game would finish when the ant collects all the leaves on the map (final state reached) => successful. The path that the ant goes to collect the leaves would be the shortest path (takes the least cost with both heuristic and expected costs) to reach the final state.

Three major components to complete the project:

1. The algorithm that is best fitted to the project. We use A* as the core algorithm. (11/7)
2. The programming language and the platform that helps to develop the project. We use C# runs on Visual Studio 2019. (11/7)
3. The goal that will be achieved from using of the A* algorithm to the project. (TBA)

REFERENCES

[1] Russell, Stuart J., et al. Artificial Intelligence: A Modern Approach. Prentice Hall, Boston, 2010.

[2] Lester, Patrick (2005): A* Path-finding for beginners.

[3] Sharma, Shrawan and Pal, B.L. (2015): Shortest Path Searching for Road Network using A* Algorithm.