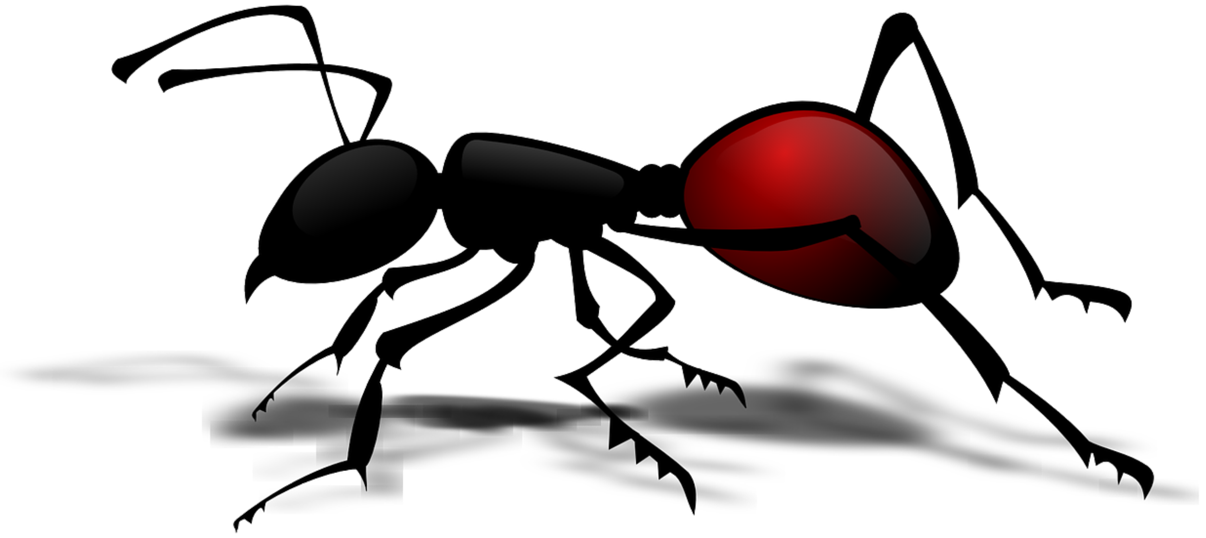
elindber & mlindhol
July 28, 2020

Lem_in

Elementary algorithmic project



Pathfinding algorithm

We look for available paths with double Breadth-first search (BFS) in the order of the parsed links. Followed by quadruple Depth-first search (DFS). Search stops when max path count or enough paths is fulfilled.

Start = level 0 End = level INT_MAX.

BFS

We traverse level by level until the shortest path has been found. After that we search for other paths in the same path set excluding the already found path.

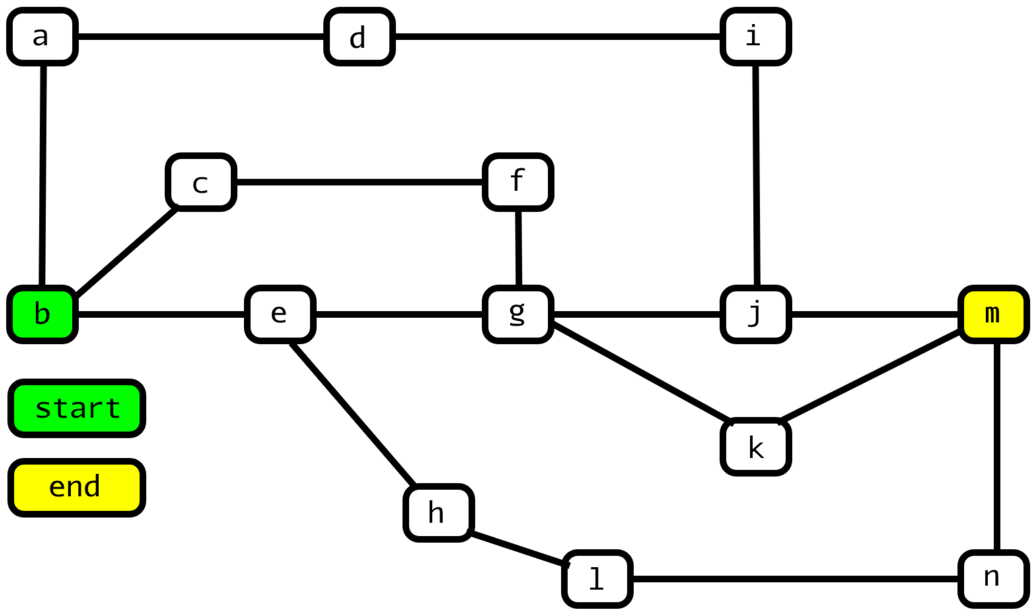


Figure 1. Example map before search.

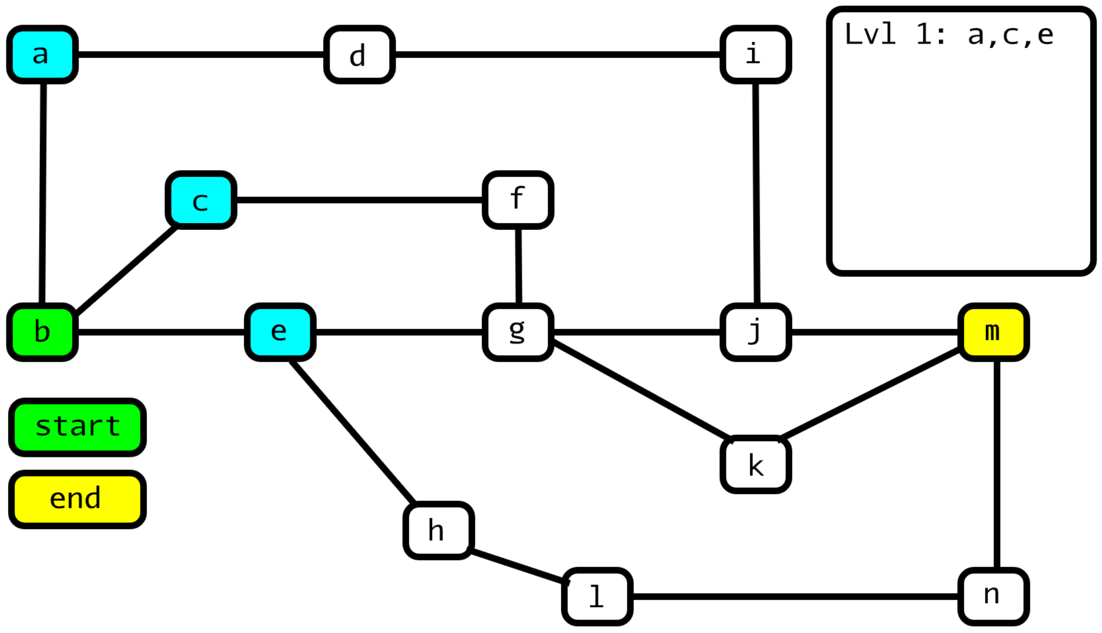


Figure 2. Search 1 - level 1

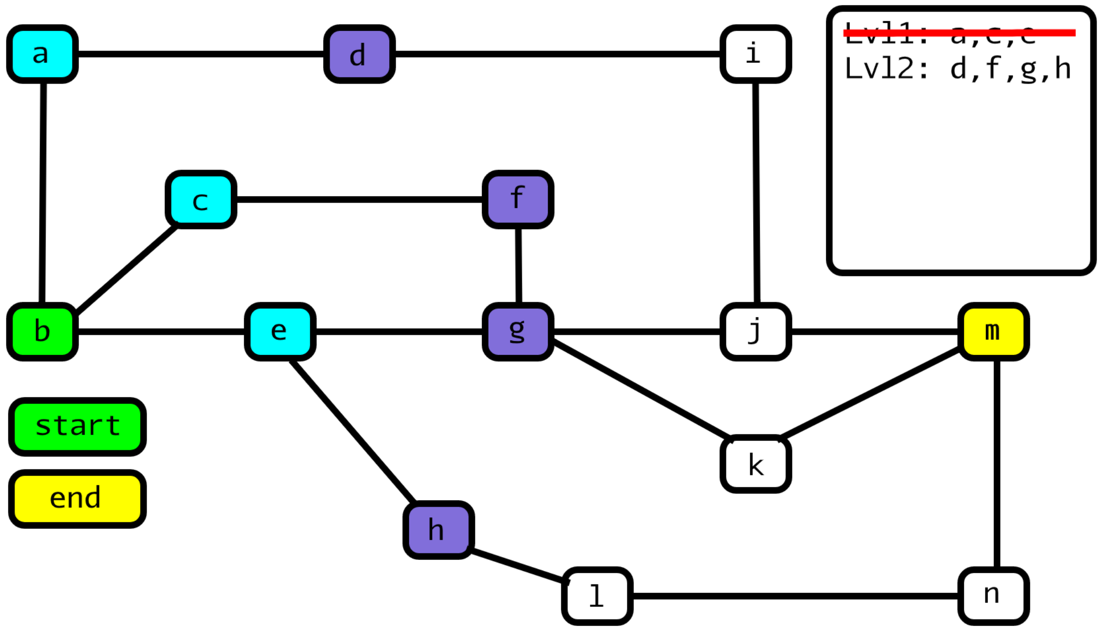


Figure 3. Search 1 - level 2

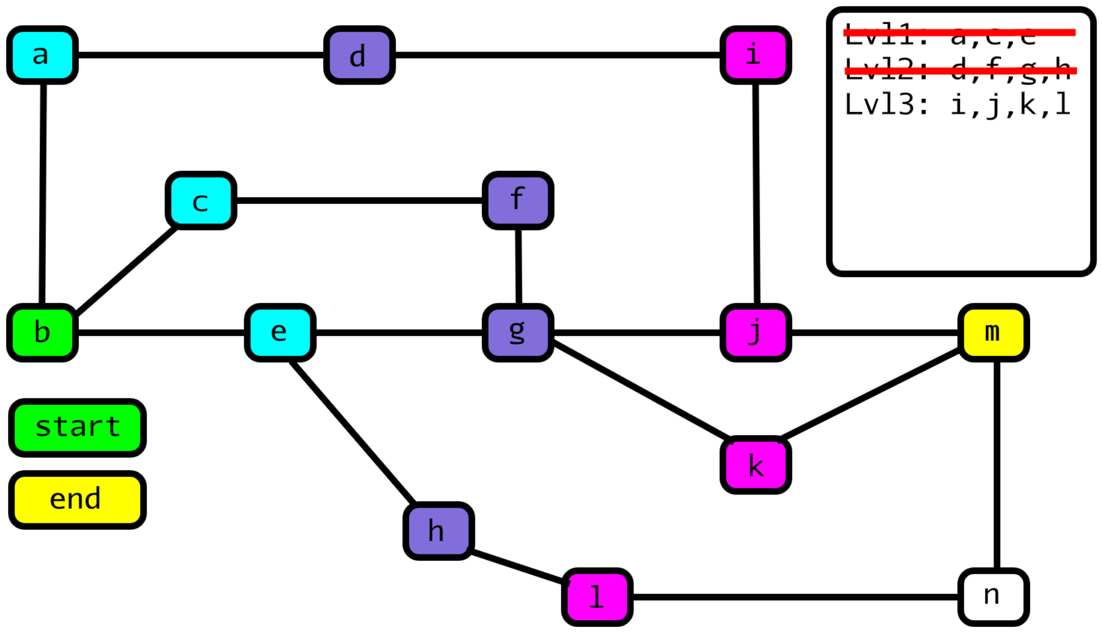


Figure 4. Search 1 - level 3

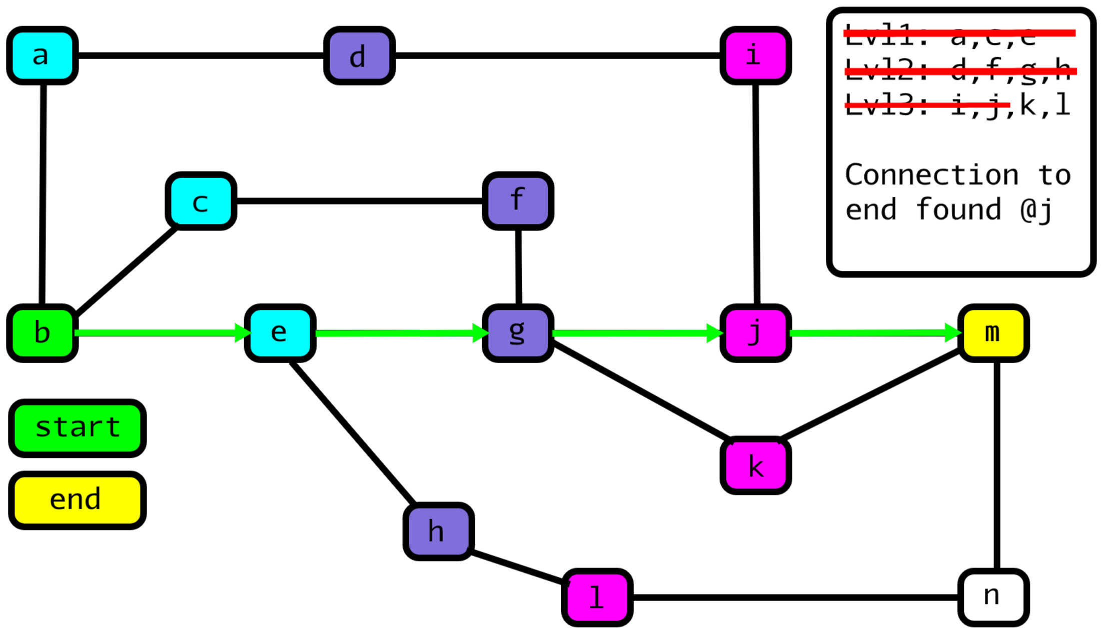


Figure 5. Search 1 - Shortest path found. Saved as path-set-1. This path blocks all other possible paths.

Now we move to search 2 (S2) which is the same as the first one with one difference. All non-start/-end rooms from path-set-1 (found during S1) get flow value of 1. During S2 we don't traverse from room with flow == 1 to another room with flow == 1. This means that moving from e to g and g to j is not possible during S2. This opens more path possibilities. This is the basis how big-superposition maps are built. One short path that blocks multiple decent length maps that would yield better result in the end.

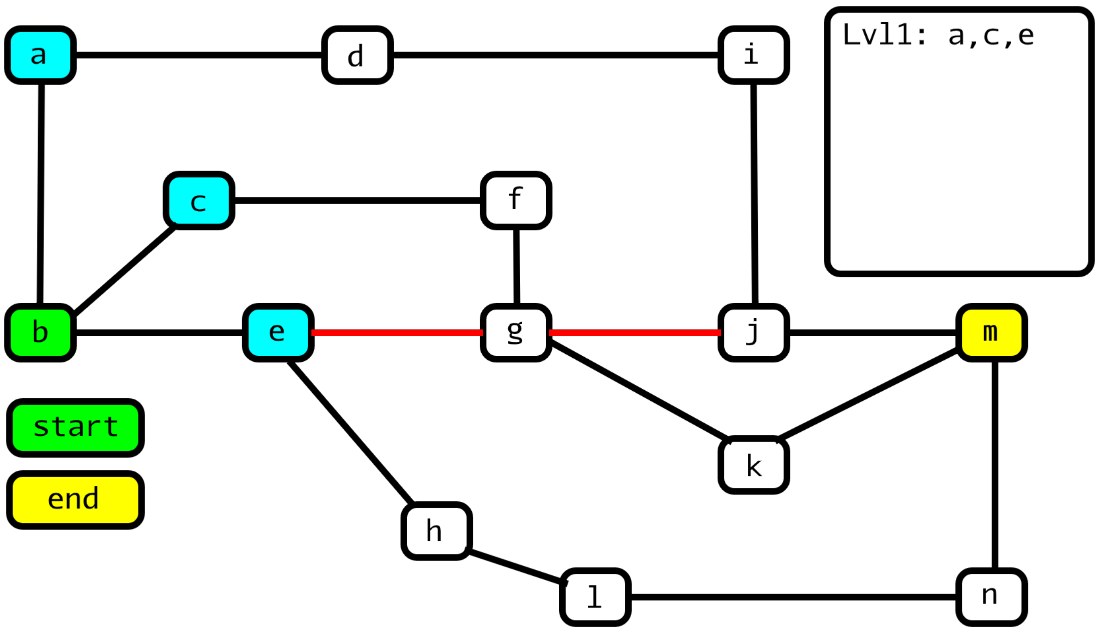


Figure 6. Search 2 - level 1

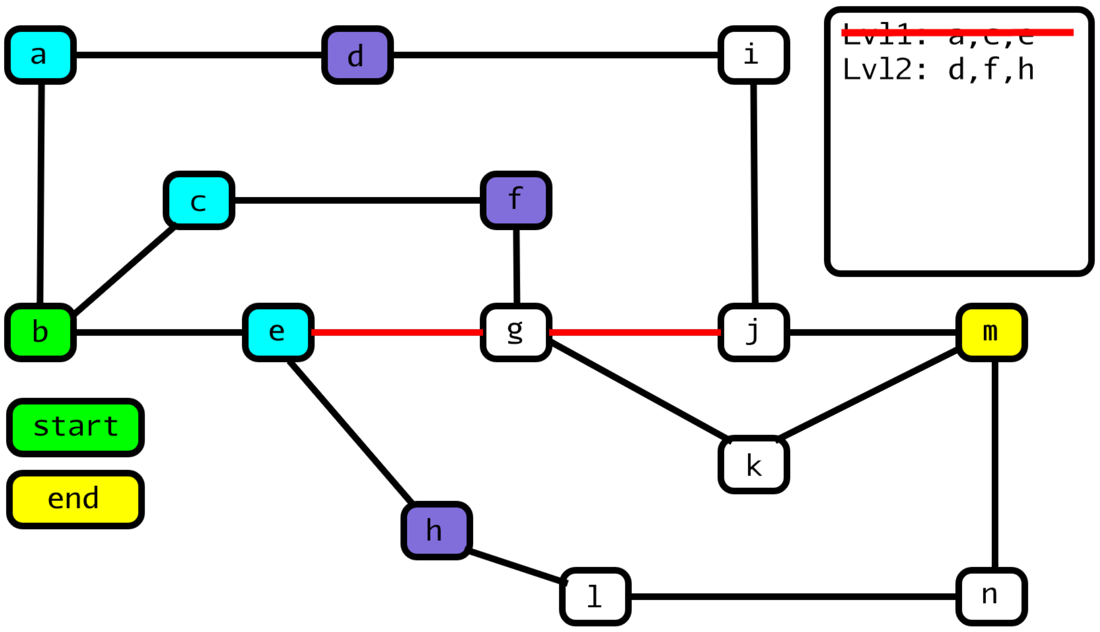


Figure 7. Search 2 - level 2

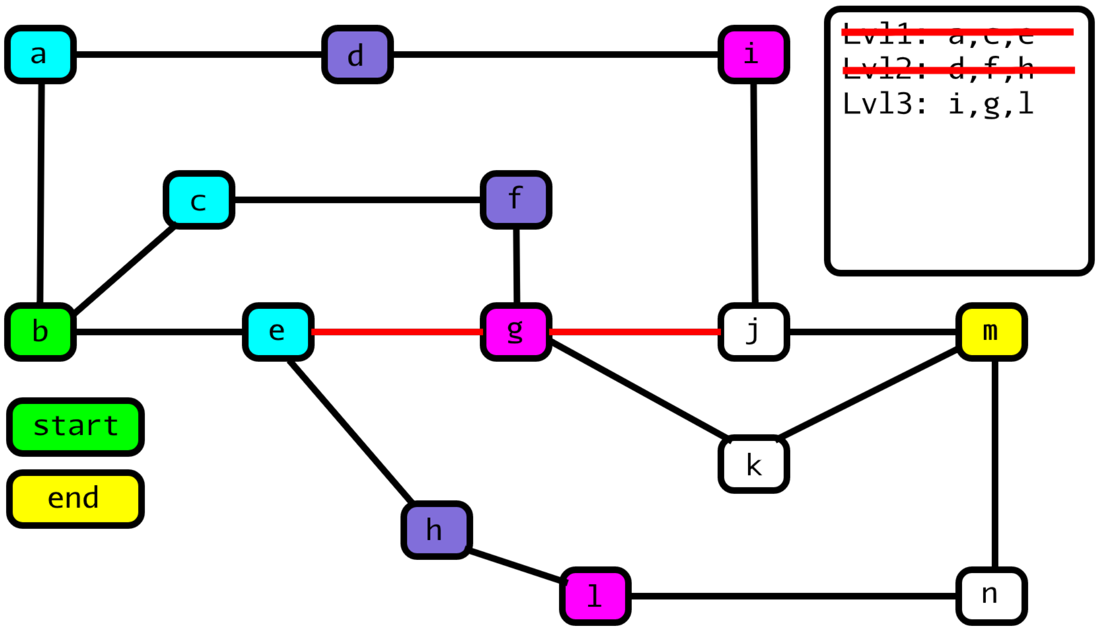


Figure 8. Search 2 - level 3

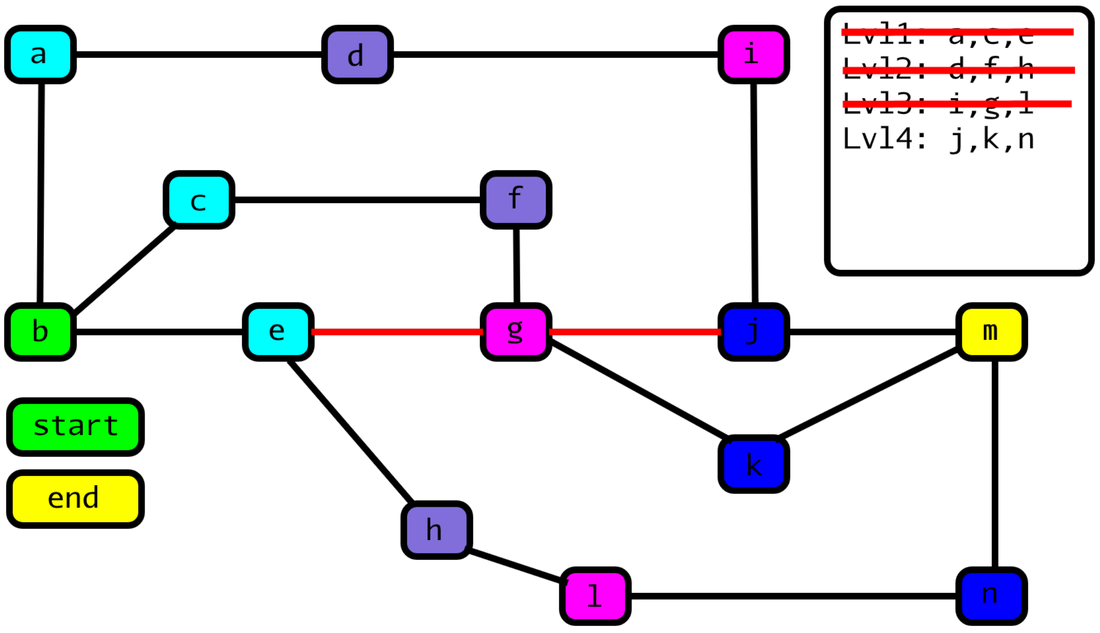


Figure 9. Search 2 - level 4

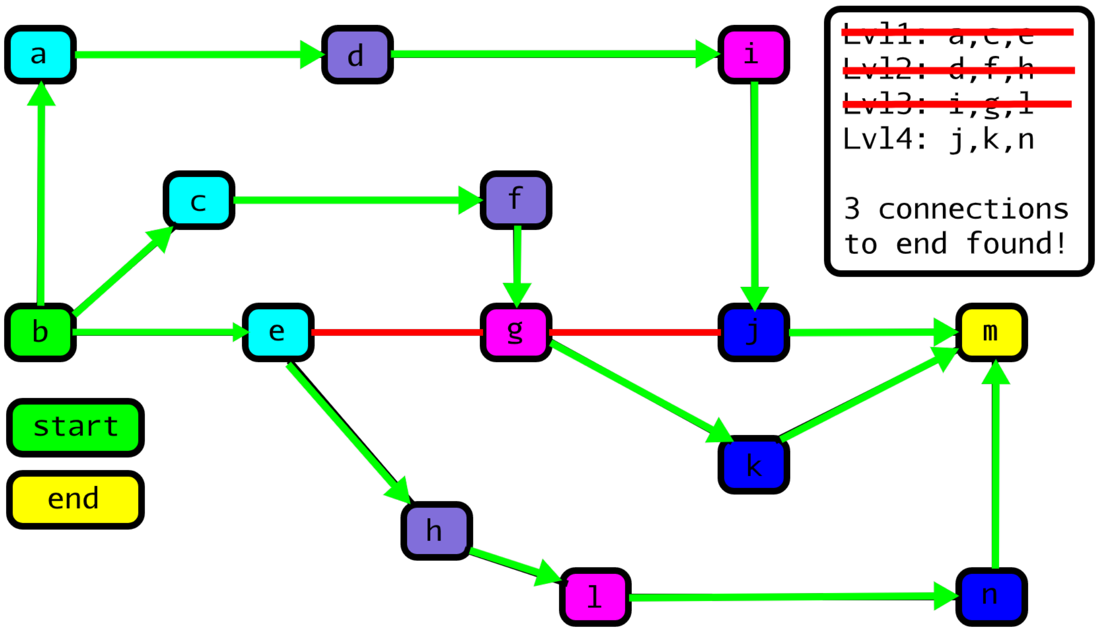


Figure 10. Search 2 - Shortest paths found. Saved as path-set-2.

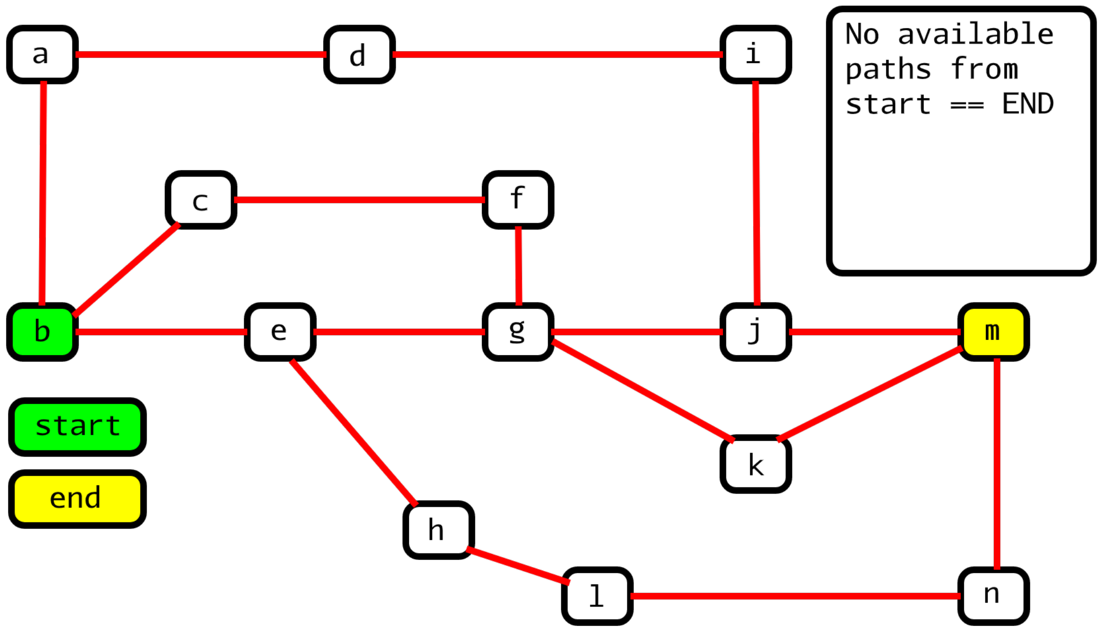


Figure 11. Search 2 - END, no available paths left.

DFS

DFS1: Starts from 1st node of level 1 and traverses level 1 nodes in regular order.

DFS2: Starts from last node of level 1 and traverses level 1 nodes in reverse order.

DFS3: Starts from (level 1 node count / 2) and traverses level 1 nodes in regular order

DFS4: Starts from (level 1 node count / 2 + 1) and traverses level 1 nodes in reverse order.

Once DFS stops we compare best path-set of DFS to best path-set of BFS and pick the best set for ants.

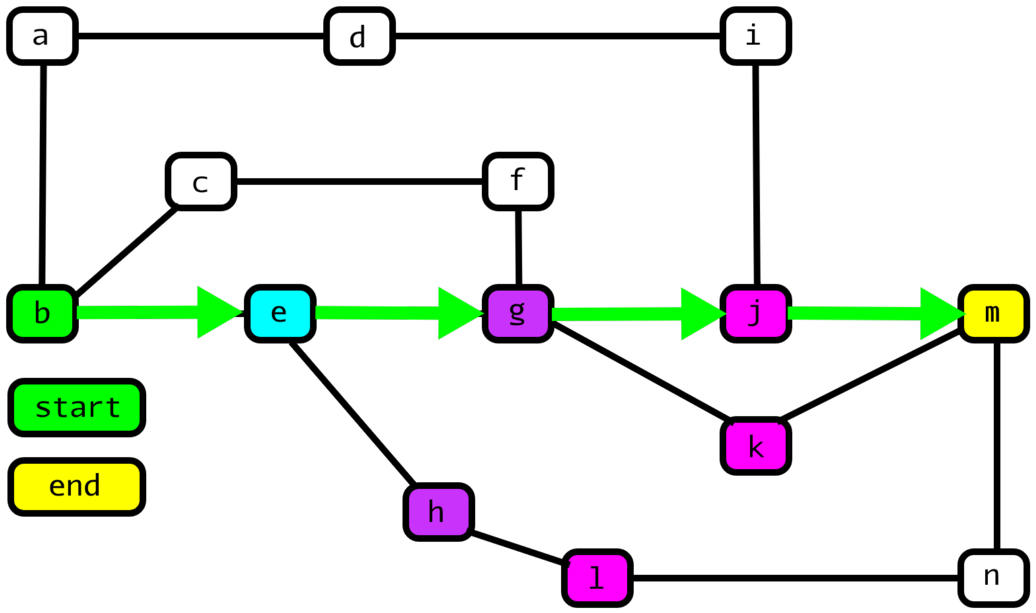


Figure 12. DFS S1 completed, blocks all other paths so moving to DFS S2.

Ant flow

Now we calculate the optimal amount of paths between path-set-1 and -2. Calculation is based on the number of ants. For the example map, when ants > 2 it is more cost-effective to use the path-set-2 as you can move more ants in one turn.

First we add ants to queue of the shortest path until path len is equal to len of 2nd shortest path. After that possible extra ants are divided on all of the shortest paths from initially shortest path toward the longest, level by level. Ex. with 5 ants on map of 3 paths (1,3,5):

```
                                        x

                    x                   x

x   _   _           x   _   _           x   _   _

1   3   5   ||      1   3   5   ||      1   3   5   ||
                        x
x                       x
x                       x
x     x     _           x     x     _
1     3     5     ||    1     3     5     ||
```

Maps

Following maps are available in the evaluation-form-provided generator:

./generator --flow-one

generates an ant farm with distinctive path and [1] ant in it

./generator --flow-ten

generates an ant farm with distinctive path and approximately [10] ants in it

./generator --thousand

generates an ant farm with distinctive path and approximately [100] ants in it

./generator --big

generates a big map (approximately [1000] rooms) to test the time complexity

./generator --big-superposition

generates a big map with overlapping paths

Run_map.sh

Custom script to run lem-in multiple times. Shows average lines compared to the required. Default map is big-superposition and default amount of runs is 10. Change map by commenting in/out lines 21-25. Flag -x overrides default amount of runs. How to run: ./run_map.sh -x 100

Bonus flags

Available bonus flags:

-a:	Insert number of ants manually. Number of ants must be after all flags. ./lem-in -e -ax 100.
-e:	Improved error management.
-h:	Help with running options.
-l:	Print amount of lines used.
-p:	Print used path-set.
-v:	Verbose mode. Prints all found path-sets and deleted dead-end-rooms plus adds colouring to paths and ant flow.
-x:	Check for leaks.