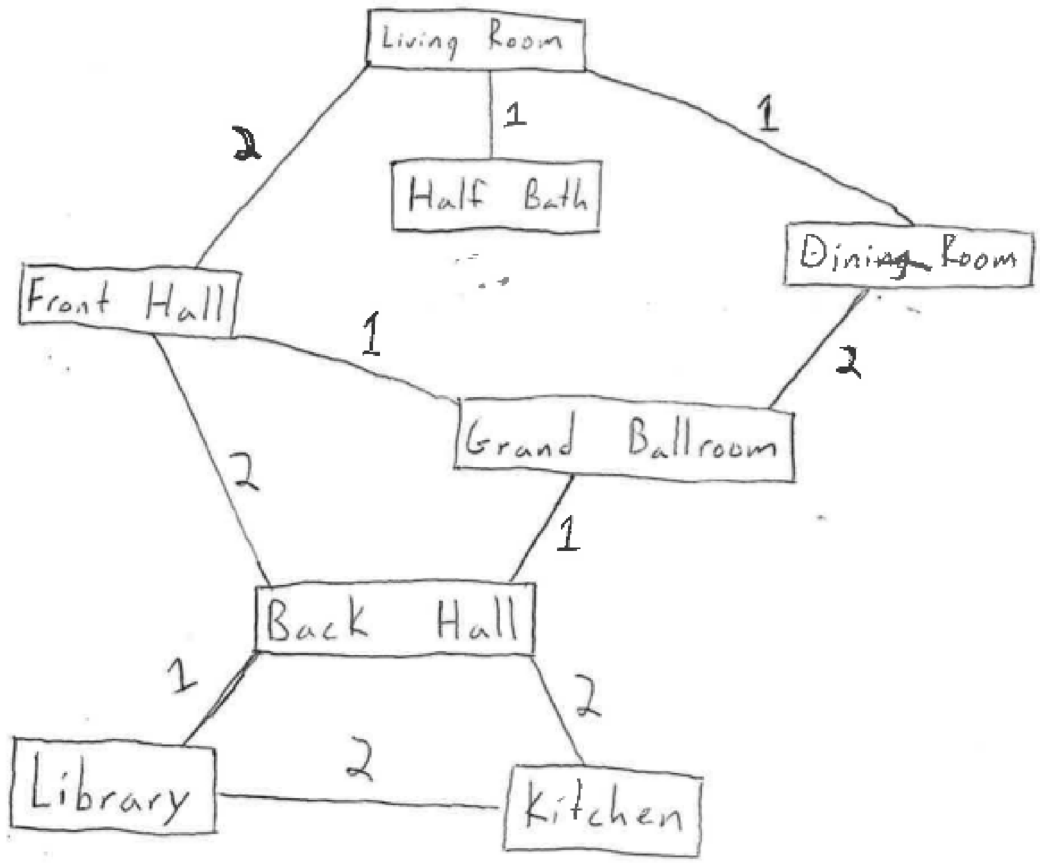
Nick Carey

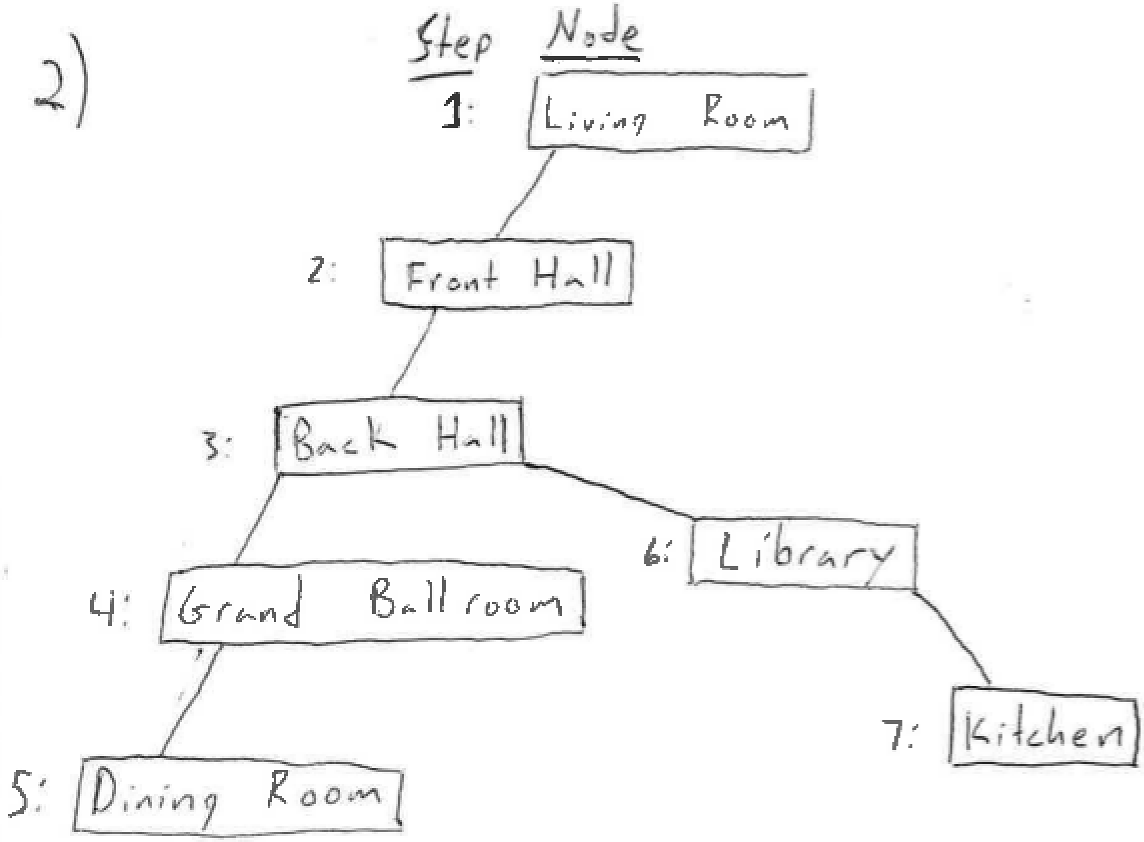
AI-HW1 Written Portion

---

1)

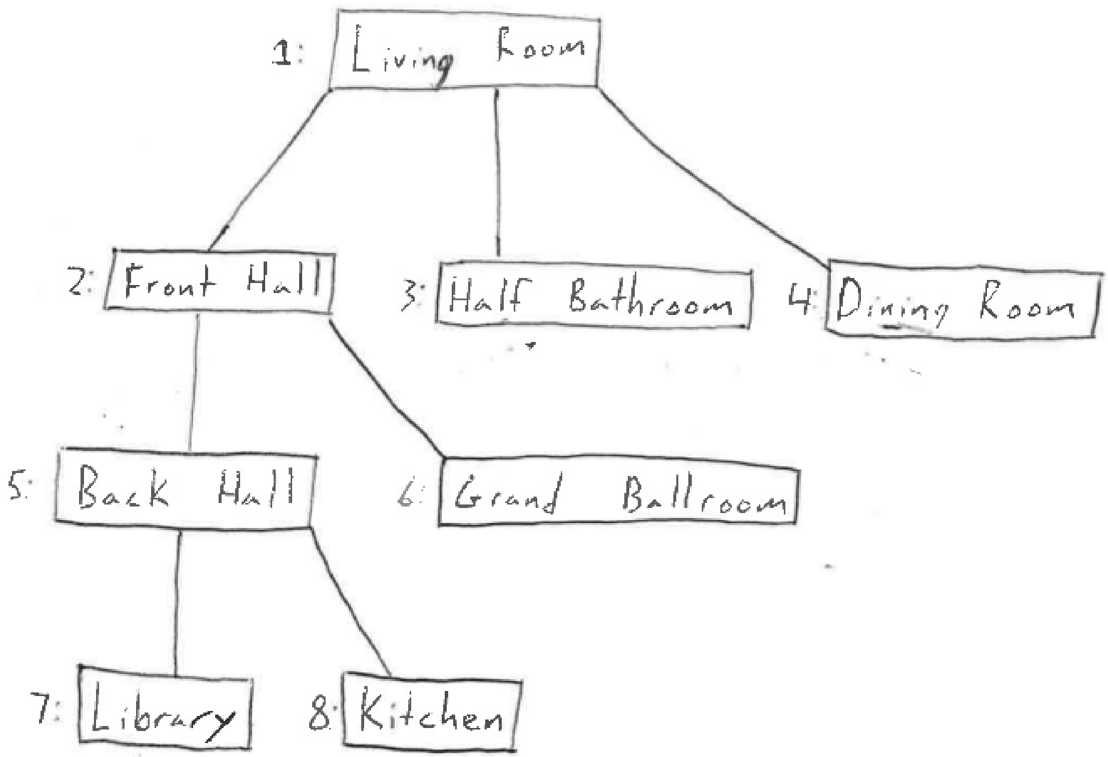


2)

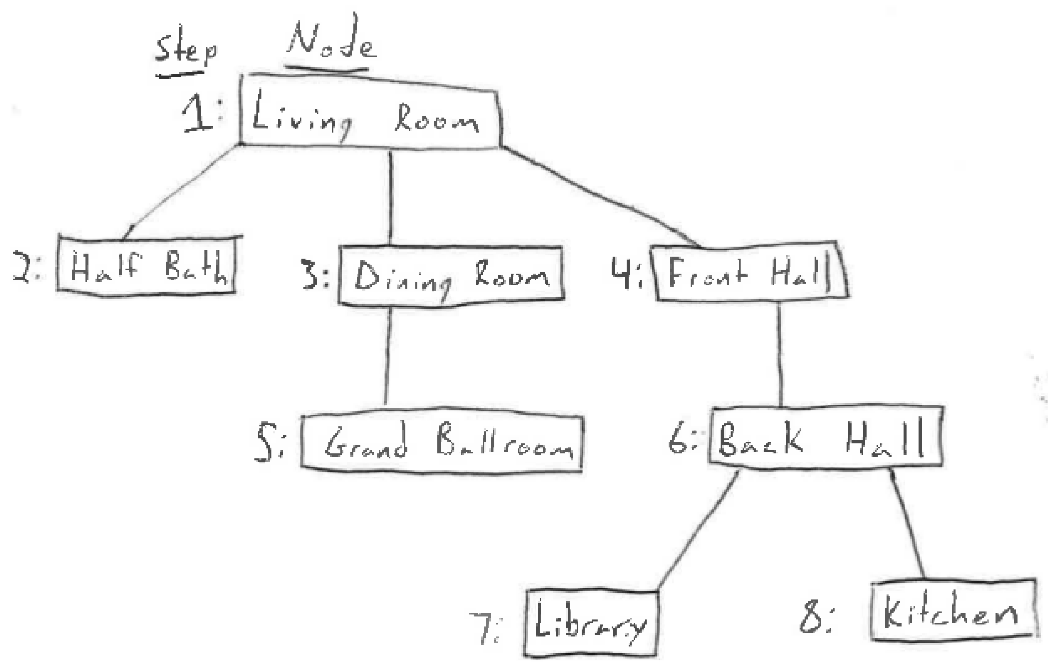


Step Node
1: Living Room
2: Front Hall
3: Back Hall
4: Grand Ballroom
5: Dining Room
6: Library
7: Kitchen

3)


1: Living Room
2: Front Hall
3: Half Bathroom
4: Dining Room
5: Back Hall
6: Grand Ballroom
7: Library
8: Kitchen

4)


Step
Node
1: Living Room
2: Half Bath
3: Dining Room
4: Front Hall
5: Grand Ballroom
6: Back Hall
7: Library
8: Kitchen

5) Time is in seconds

	BFS	node	time	cost	DFS	node	time	cost	A*	node	time	cost
mp1		8	.104	7		8	.104	7		7	.096	7
mp2		15	.092	14		15	.092	14		14	.092	14
mp3		32	.088	14		34	.076	14		25	.088	
mp4		45	.096	22		35	.100	34		8461	2.78	
mp5		289	.084	21		43	.092	42		500	.280	
mp6												
mp7		45	.104	inf		39	.088	inf				
mp8		29	.084	21		16	.104	15		13	.096	

6) I was only able to use the Manhattan distance from current node to goal node. It is admissible because the actual distance could only be equal to or longer than the Manhattan distance, since the Manhattan distance is the shortest path in this setting. However, it is not a very useful heuristic. It does not take into account the cost of ‘,’ nodes versus ‘.’ nodes.

7) One such problem is where the map resembles a wheel with many spokes. The start is at the center of the wheel. Each spoke is a path in the map. Near the beginning of one of the spokes is the goal state. All the other spokes are very long dead end paths. BFS would perform well here because it would examine each layer of of nodes emanating out from the center start point before actually going completely down a spoke. DFS would not perform too well because it would travel down each long dead end spoke until it ended up happening upon the goal.

8) Same map as above, but now the goal is at the very end of one of the spokes/paths. BFS would have to expand nearly every single node until it got to the goal at the end of one of the spokes. At least with DFS, the goal will be found after expanding, on average, half the nodes.

9) One such example is map1.txt. The only path to explore is the optimal path, and A* will show no improved performance over DFS or BFS since they all tie.

10)

- Euclidean distance from current node to goal node is admissible, since it will always be shorter than the actual distance. In our setting we can only go up/down/left/right, and not in any diagonal, so the actual path will always be longer than the Euclidean distance. This heuristic will be slightly useful, since in an open map with little obstacles the Euclidean distance will correlate to the actual distance. However in a map with many obstacles and pitfalls this heuristic will not help.

- This is not admissible, since it is possible for twice the Euclidean distance to be larger than the shortest path to the goal.

- This is admissible since the Manhattan distance will be the shortest possible path to the goal. If there are no obstacles, the Manhattan distance will be the actual distance. Otherwise the actual distance will be slightly larger. This heuristic is useful on maps with little obstacles, but a simple barrier between the start and goal state will make this heuristic inaccurate.

- This is not admissible, since anything longer than the Manhattan distance has the possibility of being longer than the actual distance.

- This is not admissible since the goal could be less than 2 steps away.

- This is admissible since it will always be less than or equal to the actual distance to the goal, but it is just about useless as it gives no information on the distance of the goal to the current state.

11) This is not admissible because of the wind and weather conditions on Earth. A strong tailwind could mean the actual cost of flying from A to B is less than the cost predicted by Euclidean distance alone. A better heuristic would be similar to Euclidean distance but also take into account weather effects on the aircraft velocity.