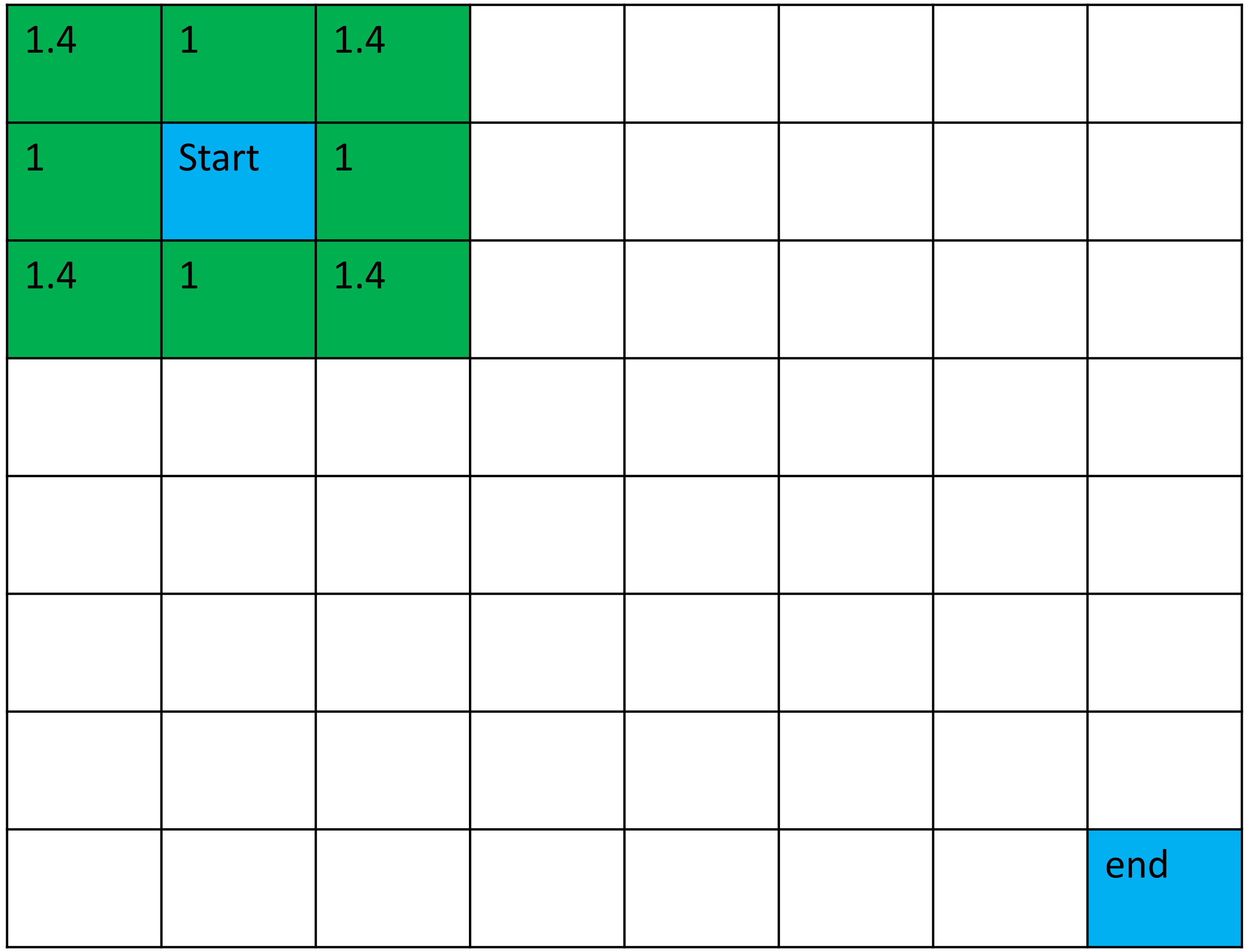
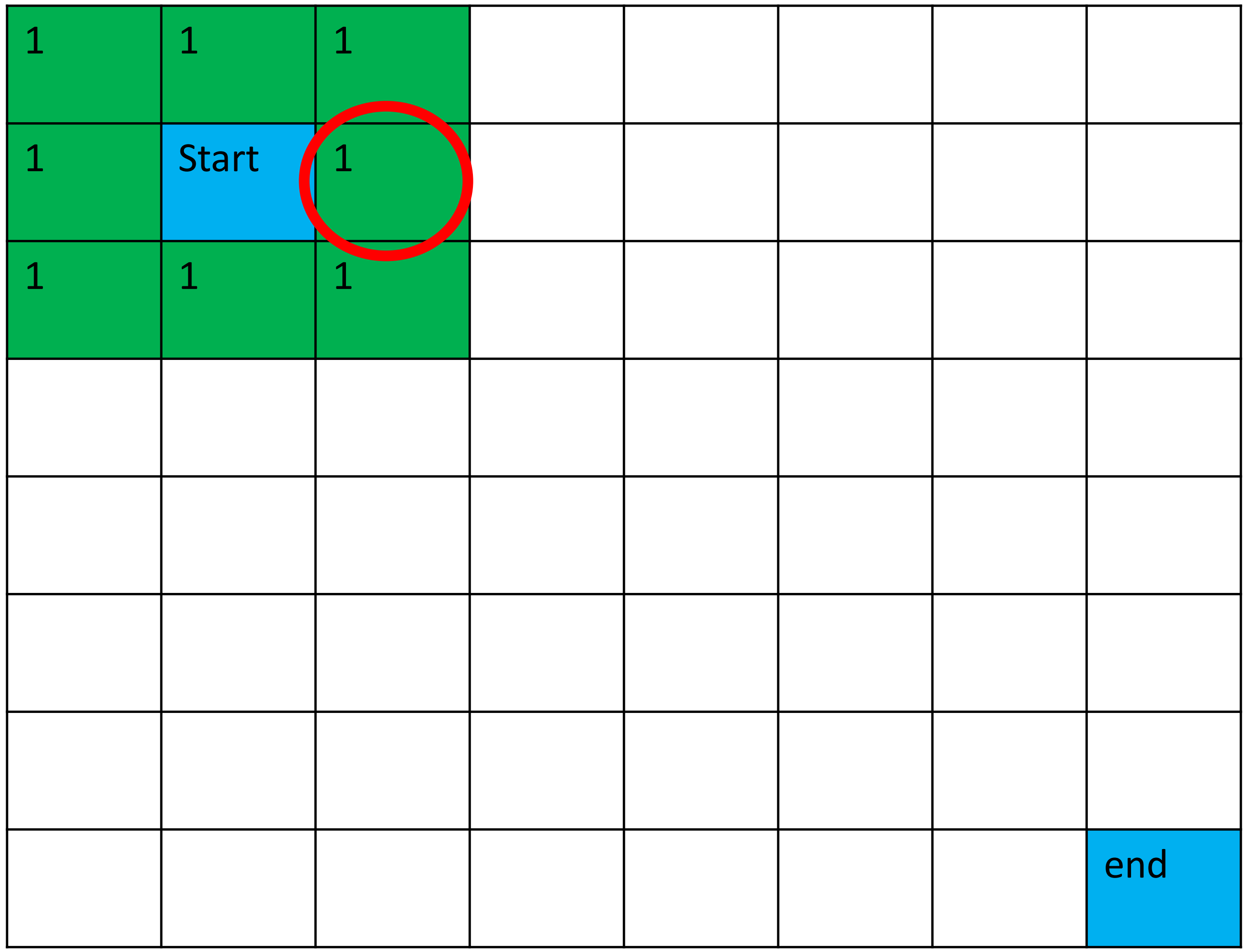


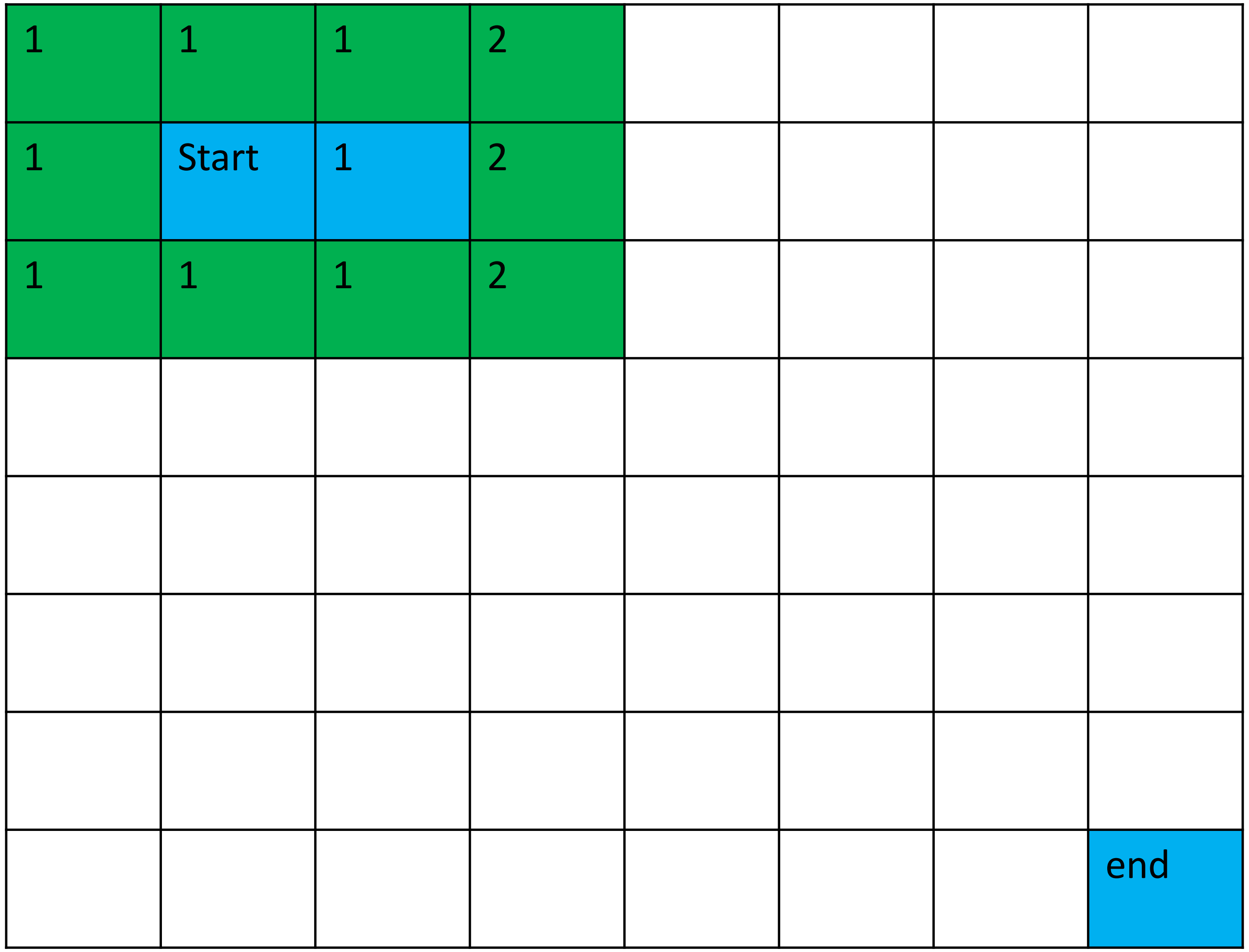
Cost

Cost = distance from parent node



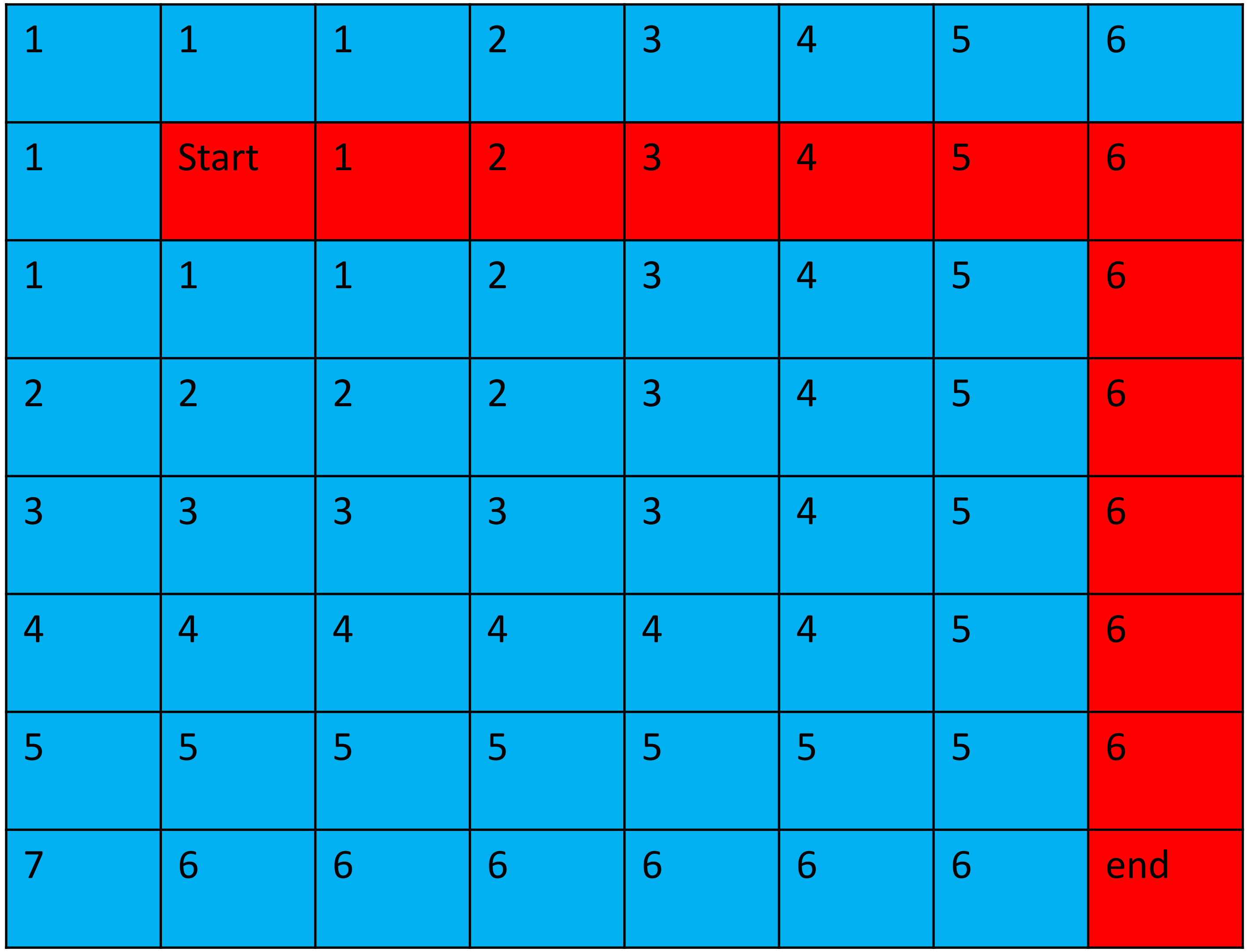
Not visited list: green cell
Visited list: blue cell
Path: Red cell

- Find the lowest cost cell in Not visited list
- Choose it as the father node in next iteration
- Remove current cell from Not visited list
- Add current cell to Visited list



Not visited list: green cell
Visited list: blue cell
Path: Red cell

- Check whether the adjacent cell is over the maze limit
- Also, calculate those node cost, which overlay the father node cost
- If adjacent cell exist in visited list cell, compare their cost, and update the new cost if the new cost is lower than it
- Add adjacent cell from father node to Not visited list if there are not exist in it and not over the limit



1	1	1	2	3	4	5	6
1	Start	1	2	3	4	5	6
1	1	1	2	3	4	5	6
2	2	2	2	3	4	5	6
3	3	3	3	3	4	5	6
4	4	4	4	4	4	5	6
5	5	5	5	5	5	5	6
7	6	6	6	6	6	6	end

Not visited list: green cell
Visited list: blue cell
Path: Red cell

- Repeat the process, until the adjacent cell include the end node
- Once found the end node, keep tracing back the parent node, so we can get the shortest path