Navigation of graph using BFS, DFS, GBFS and A-Star

Following is performance matrices for five paths:

Path : 1,20 to 20 20 // across the bottom

	BFS	DFS	GBFS	A-Star
Number of iterations	180	19	19	19
Maximum queue size	22	20	20	20
Vertices Visited	189	38	38	38
Path Length	19	19	19	19

Path 1,1 to 20,20 //corner to corner

	BFS	DFS	Greedy-Best First	A-Star
Number of iterations	264	43	31	116
Maximum queue size	21	43	36	35
Vertices Visited	270	85	66	150
Path Length	28	40	31	34

Path 1,10 to 20,10

	BFS	DFS	Greedy-Best First	A-Star
Number of iterations	244	164	53	121
Maximum queue size	25	93	42	43
Vertices Visited	257	222	94	163
Path Length	25	52	34	29

Path 10,1 to 10,20

	BFS	DFS	Greedy-Best First	A-Star
Number of iterations	264	170	34	65
Maximum queue size	24	87	43	26
Vertices Visited	270	238	76	90
Path Length	26	53	26	29

Path 13,6 to 7,6 //from under T on one side to the other

	BFS	DFS	Greedy-Best First	A-Star
Number of iterations	247	157	39	56
Maximum queue size	22	91	35	26
Vertices Visited	262	231	73	81
Path Length	28	68	28	28

Answer to the questions:

Questions are answered on the basis of **BFS, DFS and Greedy-Best First Search only**, for **A-Star the heuristic choosen is h(n) = g(n) + d(n);** where g(n) is distance from source and d(n) is distance from goal, to improve performance of A-Star better heuristic is required which will be explored in next assignment

Q. Which algorithm is the fastest?
A. Since for all cases Greedy-Best Best first search have minimum number of iterations, it is fastest algorithm.

Q. Which is the most memory efficient?
A. Since for most of the cases Maximum queue size for Breadth first search is minimum, it most memory efficient.

Q. Which visits fewest vertices?
A. In all cases Greedy-Best- First Search visits fewest vertices.

Q. Which generates the shortest path length?
A. Breadth-First-Search generates the shortest path length.

Q. Are the performance differences what you expected based on the theoretical complexity analysis?
A. The worst case time complexity of BFS is O(b^d), where d is depth of shallowest goal node and b is branching factor and complexity of DFS is O(b^m) , where m is maximum depth and for greedy is O(b^m) which can be reduced using good heuristic. In implementation if we look according to number of iterations the order comes to be BFS > DFS > Greed-Best-First-Search. And for the space complexity BFS have O(b^d), DFS O(b.m) and greedy have O(b^m) which is improved using better heruisitic, according to the implementation order for space complexity is DFS > Greed-Best-First-Search > BFS. In short BFS have maximum number of iterations and DFS have maximum queue size.

Q. Does BFS always find the shortest path? Does GBFS always go "straight" to the goal, or are there cases where it gets side-tracked?
A. Yes BFS always finds the shortest path. GBFS tries to take the path straight to the goal provided no obstacle comes in between I.e there is possible path straight to the goal or when the path assumed by heuristic is not the correct path then it gets side tracked.