Representation of graph edges, E, : M = matrix, LL = Linked List

Algorithm		Time	Space	Application	Other
BFS	LL	O(V+E)	Θ(V) (color, d, p array, Queue max size)	Flight - fewest connections	Undirected (Ok directed)
	M	$O(V^2)$			
DFS	LL	O(V+E)	Θ(V) (color, st, finish, p arrays rec stack max size)	Graph traversal or search Detect cycles Strongly Connected Components Topological sort	Directed (ok Undirected) Recursion
	M	$O(V^2)$			
Topological sort		Same as DFS	Same as DFS	Order of items in a production line Order of finishing dependent tasks	Directed only No cycles allowed
(Strongly) Connected components		Same as DFS	Θ(DFS) + Θ(G) (needs G^T)	Groups with 2-way communication between any pair within the group	Directed (for undirected - BFS+ restart also works)
MST	LL	O(V+ElgV) *	Θ(V)	Network layout (e.g. cables for electrical network)	Undirected Greedy, Optimal If priority queue is based on edges, space is Θ(E)
Prim	M	$O(V^2)$	Θ(V) our method		
SPST (Dijkstra)	LL	O(V+ElgV) * Same as PRIM	Θ(V) Same as PRIM	Flight - cheapest cost Driving directions (time, or distance)	Directed Greedy, Optimal
	M	$O(V^2)$			
All pairs SPST		Θ(V)*Θ(SPST)	$\Theta(V^2)$ (row i = SPST(G,i))	Same as SPST	Same as SPST. Do SPST(G,v) for every vertex, v

* : if connected: Θ(ElgE)=Θ(ElgV) ** : Assume the graph is connected => E≥(V-1)