Compiled by Tian Xiao

CS2040 Data Structures and Algorithms

AY2020/21 Semester 2

Table 1: Data structures and algorithms associated with common time complexities

Time Complexity	**Data Structures and Algorithms**
$O(1)$	• Insert/Delete/Contain in hashtable • Push/Pop/Peek in stack • Enqueue/Dequeue/Peek in queue
$O(\alpha(n))$	• FindSet/IsSameSet/UnionSet in UFDS
$O(\log n)$	• Insert value to heap • Extract max/min from heap (is there a need to SORT?) • Enqueue/Dequeue in priority queue • Search/Insert/Delete in AVL tree
$O(n)$	• Quick select (is there a need to SORT?) • Radix sort • Build max/min heap from array • Build UFDS • Create AVL tree from sorted list • In-order/Pre-order/Post-order traversal of BST
$O(V+E)$	• BFS/DFS (reachability test, unweighted shortest path, counting CC) • Topological sort • Kosaraju's (counting SCC)
$O(E)$	• One-pass Bellman Fod (for DAG)
$O(n \log n)$	• Merge sort • Quick sort • Heap sort • Create AVL tree
$O((V+E) \log V)$	• Dijkstra's (no negative weighted edge) • Modified Dijkstra's (no negative weighted cycle)
$O(E \log V)$	• Prim's/Kruskal's (extra DFS to find MINIMAX in MST)
DANGER ZONE	
$O(n^2)$	• Selection sort • Bubble sort • Insertion sort
$O(VE)$	• Bellman Ford's
$O(V^3)$	• Floyd Warshall's (for APSP)

Table 2: Time complexities associated with common data structures and algorithms

Data Structures and Algorithms	**Methods and Time Complexities**
Sorting	• Selection sort: $O(n^2)$ • Bubble sort: $O(n^2)$ • Insertion sort: $O(n^2)$ • Merge sort: $O(n\log n)$ • Quick sort: $O(n\log n)$ • Quick select: $O(n)$ • Radix sort: $O(dn)$ • Heap sort: $O(n\log n)$ • Topological sort: $O(V+E)$
ArrayList	• Insert: $O(1)$ • Remove: $O(n)$ • Access: $O(1)$
LinkedList	• Insert: $O(n)$ • Remove: $O(n)$ • Access: $O(n)$
Stack	• Push: $O(1)$ • Pop: $O(1)$ • Peek: $O(1)$
Queue	• Offer: $O(1)$ • Poll: $O(1)$ • Peek: $O(1)$
HashTable	• Insert: $O(1)$ • Access: $O(1)$ • Delete: $O(1)$
PriorityQueue	• Build heap from array: $O(n)$ • Enqueue: $O(\log n)$ • Dequeue: $O(\log n)$ • Heap sort: $O(n\log n)$
UFDS	• Initialise: $O(n)$ • FindSet: $O(\alpha(n))$ • IsSameSet: $O(\alpha(n))$ • UnionSet: $O(\alpha(n))$
BinarySearchTree	• Search: $O(h)$ • Insert: $O(h)$

	• Delete: $O(h)$ • FindMax/FindMin: $O(h)$
AVL Tree	• Search: $O(\log n)$ • Insert: $O(\log n)$ • Delete: $O(\log n)$ • FindMax/FindMin: $O(\log n)$
Graph	• BFS: $O(V + E)$ • DFS: $O(V + E)$ • Reachability test: $O(V + E)$ • Count components: $O(V + E)$ • Topological sort (Kahn's/DFS): $O(V + E)$ • Count SCCs: $O(V + E)$ • MST (Prim's/Kruskal's): $O(E \log V)$ • SSSP in unweighted graph (BFS): $O(V + E)$ • SSSP in tree (BFS/DFS): $O(V)$ • SSSP in graph without negative edge (Dijkstra's): $O((V + E) \log V)$ • SSSP in graph with negative edges but without negative cycle (modified Dijkstra's): $O((V + E) \log V)$ • SSSP in directed acyclic graphs (one-pass Bellman Ford's): $O(E)$ • SSSP in other weighted graphs (Bellman Ford's): $O(VE)$ • APSP (Floyd Warshall's): $O(V^3)$

Table 3: Orders of graph traversal

Name	**Order**
In-order traversal	left – root – right
Pre-order traversal	root – left – right
Post-order traversal	left – right – root
BFS	breadth-first
DFS	depth-first

Table 4: Minimum and maximum numbers of vertices of an AVL tree

Height	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Min	2	4	7	12	20	33	54	88	143	232	376	609	986	1596
Max	3	7	15	31	63	127	255	511	1023	2047	4095	8191	16383	32767

Table 5: Types of graph data structures

Type	**Scenario**
AdjacencyMatrix	• Dense graphs • Floyd Warshall’s
AdjacencyList	• Sparse graphs • BFS • DFS • Dijkstra’s
EdgeList	• Prim’s • Kruskal’s