DYNAMIC PROGRAMMING

TOP-DOWN: MEMOIZATION

BOTTOM-UP: SUBPROBLEMS
(TABULAR FORM)

DFS vs. BFS

WORST CASE $O(B^D)$
DFS: RECURSIVE vs. BFS: FIFO
(DFS MAY OVERFLOW THE STACK)
PRE-ORDER: VISIT ROOT
IN-ORDER: LEFT, ROOT, RIGHT
POST-ORDER: VISIT CHILDREN

HASHTABLES

FLOAT: $\frac{KEY-MIN}{MAX-MIN}M$

INT: $KEY \mod M$ where $\{M$: MERSENNE PRIME$\}$
% IMPL. CAN BE FAST

STRING: $\Sigma 128^i str[i] \mod M$
128 - DROP NONCODING BITS

RESOLUTION:

- SEPARATE CHAINING (LISTS)
  (A.K.A. OPEN HASHING, CLOSED ADDRESSING)
- LINEAR PROBING (ARRAY)
  (A.K.A. CLOSED HASHING, OPEN ADDRESSING)

HEAPS
(Java: PriorityQueue)

CHILD $=$ INDEX$'$ $\times 2$
PARENT $=$ INDEX$'/2$

BUILD: for each index from last parent to 0 (the root) traverse down picking the biggest/smallest child and correcting parent-child order
INSERT: append to the end of the array and traverse up correcting parent-child order $O(\log n)$"
DELETE: overwrite the node with the last and depending on comparing with its parent either traverse up or down $O(\log n)$"
SEARCH: linear scan through the array $O(n)$"

') correct index by 1 for 0-based array
") WORST/AVG

HEAPS (USAGE)

RUNNING MEDIAN - GREATER (SMALLER) HALF OF THE NUMBERS IN MIN (MAX) HEAP; INSERT INCOMING INTO HEAP DEPENDING ON COMPARISON WITH CURRENT MEDIAN; REBALANCE IF NECESSARY; MEDIAN IS ONE OF ROOTS OR THEIR AVERAGE; REMOVE OUTGOING SIMILAR

GRAPH REPRESENTATION

ADJACENCY MATRIX
PRO: LOOKUP TIME (MANY CONNECTIONS)
CON: SIZE (ALL POSSIBLE CONNECTIONS)
ADJACENCY LIST
PRO: SIZE, SPEED (FEW CONNECTIONS), SPARSE
SPACE TRADEOFF $X \times E > \frac{N^2}{8}$
(X depends on pointer size; matrix is packed)
OBJECTS & POINTERS

TRIE TREE

reTRIEval, PREFIX TREE (MARKOV CHAIN TXT GEN, AUTOCOMPLETE)

EXAMPLE: For autocomplete as you build the trie add a flag to each node, indicating if this character is a terminating one.

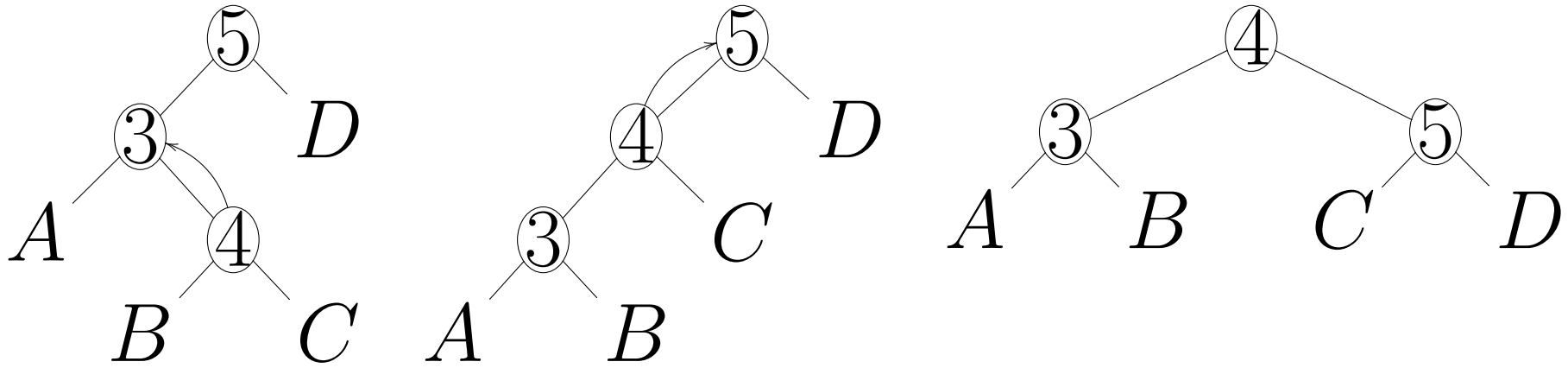
AVL TREE
(SELF-BALANCING BST)
BALANCING

(Java - red-black: TreeMap)

SEARCH/INSERT/DELETION - WORST/AVG:
$O(N \log N)$

DECORATE NODES WITH SUBTREE HEIGHT

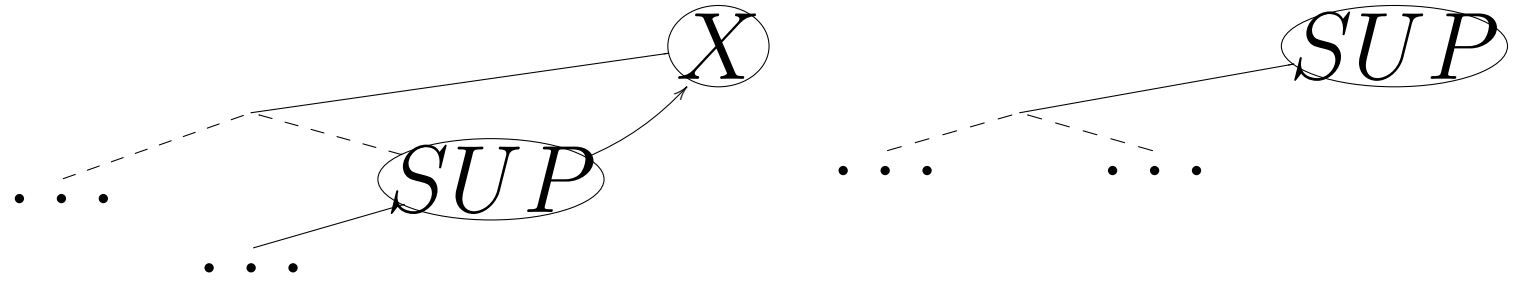
AFTER INSERTION/DELETION GO UPWARDS TO ROOT AND ROTATE IF HEIGHTS DIFFER BY > 1 (AND RECOMPUTE HEIGHTS), LEFT CASE:



AVL TREE (SELF-BALANCING BST) OPERATIONS

INSERTION: 1. NORMALLY; 2. BALANCE
DELETION: 1. IF LEAF THEN REMOVE; 2. IF ONLY CHILD THEN REMOVE AND BIND; 3. OTHERWISE RECONNECT SPLICING SUCCESSOR (OR PREDECESSOR); 3. BALANCE

SUCCESSOR CASE (SUP - SUPREMUM):



QUICK SORT

- (**shuffle** first or sample for pivot)
- PICK PIVOT
- REORDER W.R.T. PIVOT
- APPLY TO ARRAYS SEPARATED BY PIVOT

UNSTABLE
WORST $O(N^2)$ **!!!**
BEST $O(N \log N)$
AVERAGE $O(N \log N)$

MERGE SORT

- DIVIDE INTO SUBLISTS
- SORT SUBLISTS RECURSIVELY
- MERGE SUBLISTS

STABLE
WORST $O(N \log N)$ **!!!**
BEST $O(N \log N)$
AVERAGE $O(N \log N)$

MINIMUM SPANNING TREE

EXAMPLE **PRIM'S ALGORITHM**

1. CREATE TREE WITH ONE RANDOM VERTEX
2. CREATE SET OF ALL EDGES
3. LOOP TILL EVERY EDGE USED
   (a) USE AN EDGE WITH SMALLEST WEIGHT CONNECTING VERTEX IN THE TREE TO VERTEX NOT YET IN THE TREE

$O((|E| + |V|) \log |V|)$ WITH BINARY HEAP & ADJACENCY

DIJKSTRA'S ALGORITHM (BUT FLOYD ON NEGATIVE)

SHORTEST PATH $O(|V|^2)$:

1. FOREACH 0 IF INITIAL, ∞ OTHERWISE (DISTANCE)
2. MARK ALL UNVISITED; INITIAL AS CURRENT
3. BREADTH FIRST; FOR EACH CHILD ITS DISTANCE AND OVERWRITE IF LESS
4. LOWEST DISTANCE UNVISITED NODE AS CURRENT
5. END WHEN ALL VISITED

$O((|E| + |V|) \log |V|)$ WITH A PRIORITY QUEUE (SELF-BALANCING BST OR BINARY HEAP)

LEADER ELECTION

LELANN-CHANG-ROBERTS

ONLY FORWARD LARGER ID IF HAVE NOT SENT OWN

TIME $O(N)$

AVG. MESSAGE $O(N \log N)$

WORST MESSAGE $O(N^2)$

HIRSCHBERG-SINCLAIR

ELECTION IN NEIGHBORHOOD

WORST MESSAGE $O(N \log N)$

Big O

O bounded above (up to constant factor) asymptotically

o dominated asymptotically (note: every constant)

Θ bounded both above and below asymptotically

Ω bounded below by g asymptotically

P VS. NP COMPLETE

P: SOLUTION FOUND IN POLYNOMIAL TIME
NP: SOLUTION VERIFIABLE IN POLYNOMIAL TIME
COMPLETE: IF ANY PROBLEM IN THAT CLASS CAN BE REDUCED TO IT
HARD: IF PROBLEM ALLOWS QUICKLY SOLVE ANY PROBLEM IN THE CLASS

$P \neq NP$:
$P \subset NP \wedge NP - COMPLETE \equiv NP \cap NP - HARD$
((P) NP (NP-COMPLETE) NP-HARD)

NP-COMPLETE PROBLEMS

GRAPH COLORING
TRAVELING SALESMAN
KNAPSACK PROBLEM
BOOLEAN SATISFIABILITY
GRAPH ISOMORPHISM
SET COVER PROBLEM

KNAPSACK PROBLEM

NUMBER OF EACH SO WEIGHT<=W & VALUE MAX

BOTTOM-UP DYN. PROG:

W - INTEGERS: UNBOUNDED (SAME ITEM > ONCE): $max_{wi<=w}(m[w-1], max(vi, m[w-wi])$

PSEUDO POLYNOMIAL TIME (LIKE TESTING IF N IS PRIME): - EXP. IN LENGTH (NUMBER OF DIGITS) OF INPUT; POLY - IN NUMERIC VALUE OF INPUT

SET COVER PROBLEM

FROM SET OF SETS SELECT SMALLEST NUMBER OF SETS COVERING UNION

GREEDY (CHOOSE ONE COVERING MOST)

TRAVELING SALESMAN PROBLEM

EXACT ALGORITHM HELD-KARP
$O(N^2 2^N)$
APPROXIMATE ALGORITHM

- **CONSTRUCTIVE HEURISTICS** GREEDY (NN - 25% of optimal), MINIMUM SPANNING TREE BASED ALGORITHMS (CHRISTOFIDES ALG.)
- **ITERATIVE IMPROVEMENT**
- **RANDOM IMPROVEMENT** ANT COLONY OPTIMIZATION

PROCESS VS. THREAD CONTEXT SWITCH

THREAD: PART OF PROCESS,
SHARED MEMORY,
ONLY STACK & REGISTERS SAVED, CTX SWITCH - SMALLER CACHE IMPACT
PROCESS: INDEPENDENT,
SEPARATE MEMORY (MAP), STATE / IPC (FILE HANDLES, DEVICE HANDLES, SOCKETS), CTX SWITCH - MEMORY MAP SWITCH, BIGGER CACHE IMPACT (SOME ARCH. MUST FLUSH), POSSIBLY STATE SWITCH, PAGING

DEADLOCK
VS.
LIVELOCK

TWO OR MORE PROCESSES WAIT FOR THE OTHER(S) TO RELEASE THE RESOURCE

THE STATE DOES CHANGE BUT STILL NO PROGRESS
(COURTESY RELEASE)
AVOIDANCE BY RANDOMIZATION

LOCKS

SLEEPLOCK - SPINLOCK

TEST-AND-SET - FETCH-AND-ADD - CMP-AND-SWAP

MUTEX: 2 STATE
SEMAPHORE: COUNTS NO OF UNITS OF RESOURCE AVAILABLE
MONITOR: OBJ. WHERE METHODS ARE MUTUALLY EXCLUSIVE

DINING PHILOSOPHERS

- SEMAPHORES / MUTEXES (GRAB FIRST THEN SECOND; RISK: CAN STARVE)
- CENTRAL MONITOR/WAITER/CONDUCTOR (CAN EAT IF NEITHER NEIGHBOR EATS; RISK: REDUCED PARALLELISM)
- RESOURCE HIERARCHY (NUMBER FORKS; PICK AND PUT DOWN IN ORDER; RISK: LOW EFFICIENCY)

LOCK (JAVA, PYTHON)

threading.Lock.acquire([blocking])
threading.Lock.release([blocking])
synchronized

SCHEDULING

- FIFO
- FIXED PRIORITY PREEMPTIVE (NOT COOPERATIVE)
- ROUND ROBIN (NO INTERRUPTS)
- MULTILEVEL FEEDBACK QUEUE: 1. PREFERENCE TO SHORT OR I/O, 2. MULTIPLE FIFO, 3. STARTS AT END OF TOP FIFO, 4. (NORMALLY) TO THE END OF QUEUE (VOLUNTARY), 5. USES UP QUANTUM - PREEMPT AND LOWER QUEUE

STRING (JAVA)

compareTo - lexi, ignoreCase, starts, endsWith, equals, indexOf, lastIndexOf, matches - regex, replaceAll - regex, replaceFirst - regex, split, substring, toUpperCase, trim

SORT & BINARY SEARCH (JAVA)

Arrays
Collections.sort(List[, Comparator])
int compare(T, T)
binarySearch(T[] array, T key)

LISTS (JAVA)

Vector
Collections.synchronizedList(List)
Where List: ArrayList or LinkedList

$-\infty$ (JAVA, PYTHON)

Double.NEGATIVE_INFINITY
float('-inf')

SORT (PYTHON)

```
sorted([...], key=lambda item: item[1],
reverse=True)
```

LIST COMPREHENSION (PYTHON)

$$[x + 1 \text{ for } x \text{ in } X \text{ if } x > 0]$$

CLOSURE

```
# closure:  function w/ an env.
def counter():
  x = 0
  def increment(y):
    nonlocal x # ← free variable, to be closed
    x += y
    print(x)
  return increment # ← closure
```

TCP HANDSHAKE

- OPEN
  1. $\rightarrow$ SYN
  2. $\leftarrow$ SYN-ACK
  3. $\rightarrow$ ACK
- CLOSE
  1. $\rightarrow$ FIN
  2. $\leftarrow$ ACK
  3. $\leftarrow$ FIN
  4. $\rightarrow$ ACK

COMBINATORIAL

PERMUTATION

$N! \leftarrow$ factorial

VARIATION WITHOUT REP.

$$\frac{N!}{(N-K)!}$$

VARIATION WITH REP. (EQUIVALENT GROUPS REPEAT)

$$N^K$$

COMBINATION

$$\binom{N}{K}$$