CECS 528, Learning Outcome Assessment 6, 10/16/2024, Dr. Ebert

Problem

LO2. Solve each of the following problems.

(a) Consider the following algorithm of Karatsuba called `multiply` for multiplying two n-bit binary numbers x and y. In what follows, we assume n is even. Let x_L and x_R be the leftmost $n/2$ and rightmost $n/2$ bits of x respectively. Define y_L and y_R similarly. Let P_1 be the result of calling `multiply` on inputs x_L and y_L, P_2 be the result of calling `multiply` on inputs x_R and y_R, and P_3 the result of calling `multiply` on inputs $x_L + x_R$ and $y_L + y_R$. Then return the value $P_1 \times 2^n + (P_3 - P_1 - P_2) \times 2^{n/2} + P_2$. Prove that Karatsuba's algorithm correctly computes xy. Hint: $xy = (x_L \times 2^{\frac{n}{2}} + x_R)(y_L \times 2^{\frac{n}{2}} + y_R)$.

(b) Draw the recursion tree that results when applying Mergesort to the array

$$a = 4, 16, 8, 20, 5, 6, 9, 3, 10, 17,$$

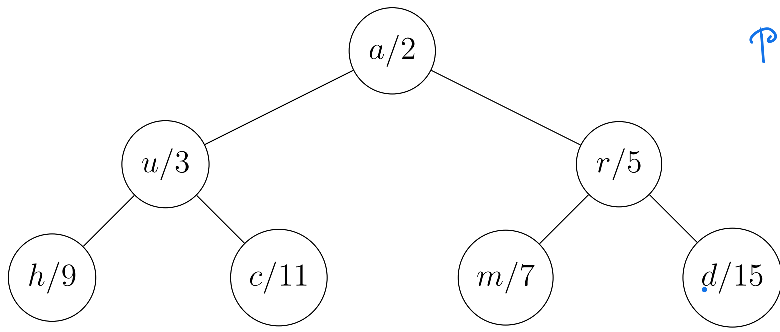
Label each node with the subproblem to be solved at that point of the recursion. Assume arrays of size 1 and 2 are base cases. Assume that odd-sized arrays are split so that the left subproblem has one more integer than the right. Next to each node, write the solution to its associated subproblem.

LO3. Solve each of the following problems.

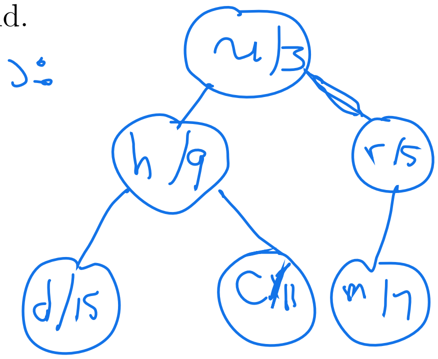
(a) The tree below shows the state of the binary min-heap at the beginning of some round of Prim's algorithm, applied to some weighted graph G. If G has edges

$$(r, d, 1), (a, m, 4), (c, a, 6), (a, p, 2), (a, h, 11), (p, d, 1),$$

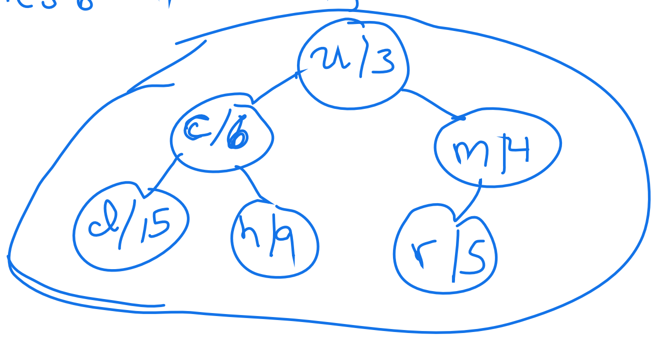
then draw a plausible state of the heap at the end of the round.



pop():



inc. Priorities: m: 7⇒4, c: 11⇒6



(b) Recall that an instance of the `Task Selection` problem is a finite set T of tasks, where each task t has a start time $s(t)$ and finish time $f(t)$ that indicate the interval for which the task should be completed by a single processor. The goal is to find a subset T_{opt} of T of maximum size whose tasks are pairwise non-overlapping, meaning that no two tasks in T_{opt} share a common time in which both are being executed. Note: a task with respective start and finish times 2 and 4 does *not* overlap with a task with respective start and finish times 4 and 7, but *does* overlap with a task with respective start and finish times 3 and 6. For the algorithm described in the lecture notes, **state the greedy choice that is being made in each step of the algorithm**. Select the task that finishes the earliest and starts at or after last finish time.

Apply the algorithm to the following set of tasks, where each triple in set T represents the id, start time, and finish time.

$$T = \{(1, 50, 100), (2, 60, 120), (3, 120, 190), (4, 210, 250), (5, 80, 220), (6, 100, 200), (7, 200, 250),$$

$$(8, 120, 140), (9, 140, 200), (10, 150, 200), (11, 210, 230), (13, 90, 140), (14, 140, 180), (15, 190, 220)\}.$$

Provide a table that, for each round of the algorithm, shows the value of any statistic that is used to make the greedy choice, and also the task that is selected for that round.

Last Finish	Task Selected	Task Finish Time
0	1	100
100	8	140
140	14	180
180	15	220

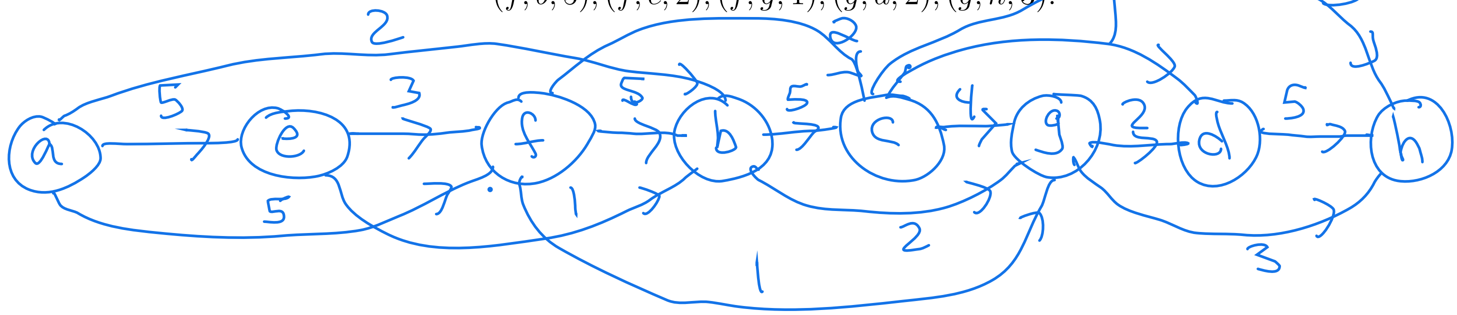
LO4. Answer the following.

(a) Provide the dynamic-programming recurrence for computing the distance $\mathrm{d}(u,v)$, from a single-source vertex u to a vertex v in a directed acyclic graph (DAG) $G = (V, E, c)$, where $c(x,y)$ gives the cost of edge $e = (x,y)$, for each $e \in E$. Hint: step *backward* from v, rather than forward from u.

$$d(u,v) = \begin{cases} 0 & \text{if } u = v \\ \infty & \text{if } \deg^{+}(v) = 0 \\ \min\limits_{w \in E^{+}(v)} \left(d(u,w) + c(w,v)\right) \end{cases}$$

(b) Draw the vertices of the following DAG G in a linear left-to-right manner so that the vertices are topologically sorted, meaning, if (u,v) is an edge of G, then u appears to the left of v. The vertices of G are a-h, while the weighted edges of G are

$$(a,b,2), (a,e,5), (a,f,5), (b,c,5), (b,g,2), (c,d,1), (c,g,4), (c,h,5), (d,h,5), (e,b,1), (e,f,3),$$
$$(f,b,5), (f,c,2), (f,g,1), (g,d,2), (g,h,3).$$



(c) Starting from left to right in topological order, use the recurrence to compute

$$d(a,a), \ldots, d(a,h).$$

$d(a,a) = 0$ $\quad d(a,e) = 5$ $\quad d(a,f) = \min(5, 3 + d(a,e)) = 5$

$d(a,b) = \min(2, 5 + d(a,f), 1 + d(a,e)) = 2$ $\quad d(a,c) = \min(5 + d(a,b), 2 + d(a,f)) = 7$

$d(a,g) = \min(d(a,f) + 1, \underline{d(a,b) + 2}, d(a,c) + 4) = 4$

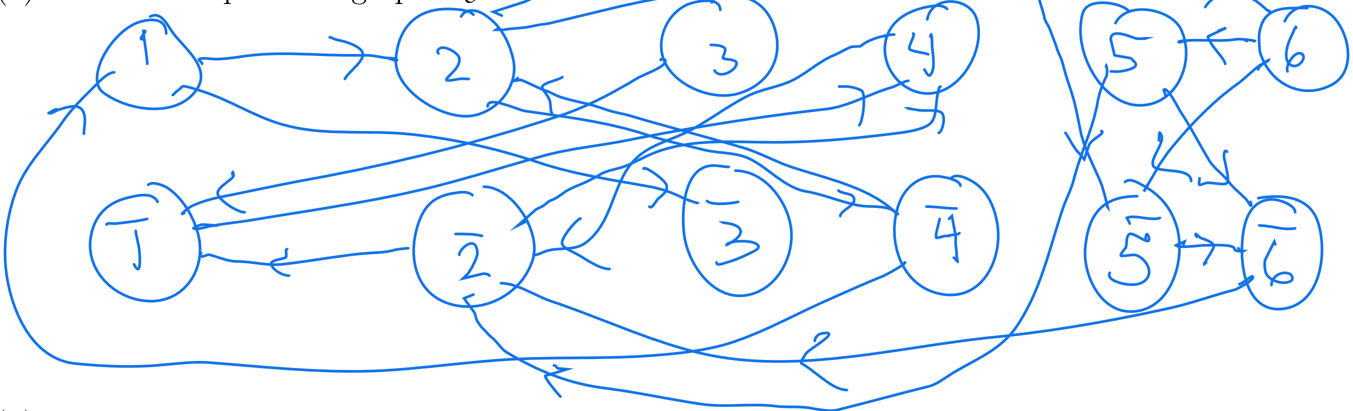
$d(a,d) = \min(d(a,c) + 1, \underline{d(a,g) + 2}) = 6$

$d(a,h) = \min(d(a,c) + 5, d(a,d) + 5, \underline{d(a,g) + 3}) = 7$

LO5. Consider the 2SAT instance

$$\mathcal{C} = \{(\overline{x}_1, x_2), (\overline{x}_1, \overline{x}_3), (x_1, x_4), (x_2, x_4), (\overline{x}_2, x_6), (\overline{x}_2, \overline{x}_4), (\overline{x}_2, \overline{x}_5), (x_5, \overline{x}_6), (\overline{x}_5, \overline{x}_6)\}.$$

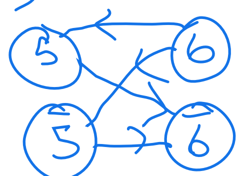
(a) Draw the implication graph $G_{\mathcal{C}}$.



(b) Perform the `Improved 2SAT` algorithm by computing the necessary reachability sets. Use numerical order (in terms of the variable index) and positive literal before negative literal when choosing the reachability set to compute next. Draw the resulting reduced `2SAT` instance whenever a consistent reachability set is computed. Either provide a final satisfying assignment for $\mathcal{C}$ or indicate why $\mathcal{C}$ is unsatisfiable.

$R_{x_1} = \{x_1, x_2, \bar{x}_4, \bar{x}_5, \bar{x}_6, \bar{x}_2, \bar{x}_1, x_4, x_6, x_5\}$

$R_{\bar{x}_1} = \{\bar{x}_1, x_4, \bar{x}_2\}$ consistent



$R_{x_5} = \{x_5, \bar{x}_6\}$ is consistent.

$\therefore \alpha = \alpha_{R_{\bar{x}_1}} \cup \alpha_{R_{x_5}} = (x_1 = 0, x_2 = 0, x_4 = 1, x_5 = 1, x_6 = 0)$ Satisfies $\mathcal{C}$

Note: x_3 may be assigned 0 or 1. (Check!)

(c) Suppose `2SAT` instance $\mathcal{C}$ is unsatisfiable and uses 336 variables and 615 clauses. Using the original `2SAT` algorithm, what is the *least* number of queries to a `Reachability` oracle that is needed in order to establish $\mathcal{C}$'s unsatisfiability. In other words, if we make fewer than this number of queries then it is possible that the `2SAT` instance $\mathcal{C}$ may be satisfiable. Explain.

At least 2 queries are needed:

$\text{Reachable}(G_{\mathcal{C}}, x_1, \bar{x}_1) = 1$ and

$\text{Reachable}(G_{\mathcal{C}}, \bar{x}_1, x_1) = 1 \Rightarrow$ Unsatisfiable.

One or fewer queries cannot establish the existence of an inconsistent cycle in $G_{\mathcal{C}}$.

LO6. Answer the following.

(a) Provide the definition of what it means to be a mapping reduction from problem A to problem B.

See Definition From Lecture

(b) Consider the mapping reduction $f(n) = 3n^2 + 2n + 9$ from the `Even` decision problem to the `Odd` decision problem. Is $f(n)$ a valid mapping reduction? If no, then provide an example of why it's invalid. If yes, then establish its validity by stating and applying basic arithmetic properties of even and odd numbers.

Arithmetic properties of even and odd numbers

		+	×
Even	Even	Even	Even
Even	Odd	Odd	Even
Odd	Even	Odd	Even
Odd	Odd	Even	Odd

+ acts like $\oplus$

× acts like $\wedge$

$\therefore$ f(Even) = 3 × Even × Even + 2 × Even + odd = odd

f(Odd) = 3 × odd × Odd + 2 × Odd + Odd = Even

and f maps positive (resp. negative) instances of Even to negative (resp. positive) instances of Odd