Midterm Review

Divide & Conquer, Trees, Binary Trees, Heap & Heap Sort

Shuyue Wang Nov 8

Claim

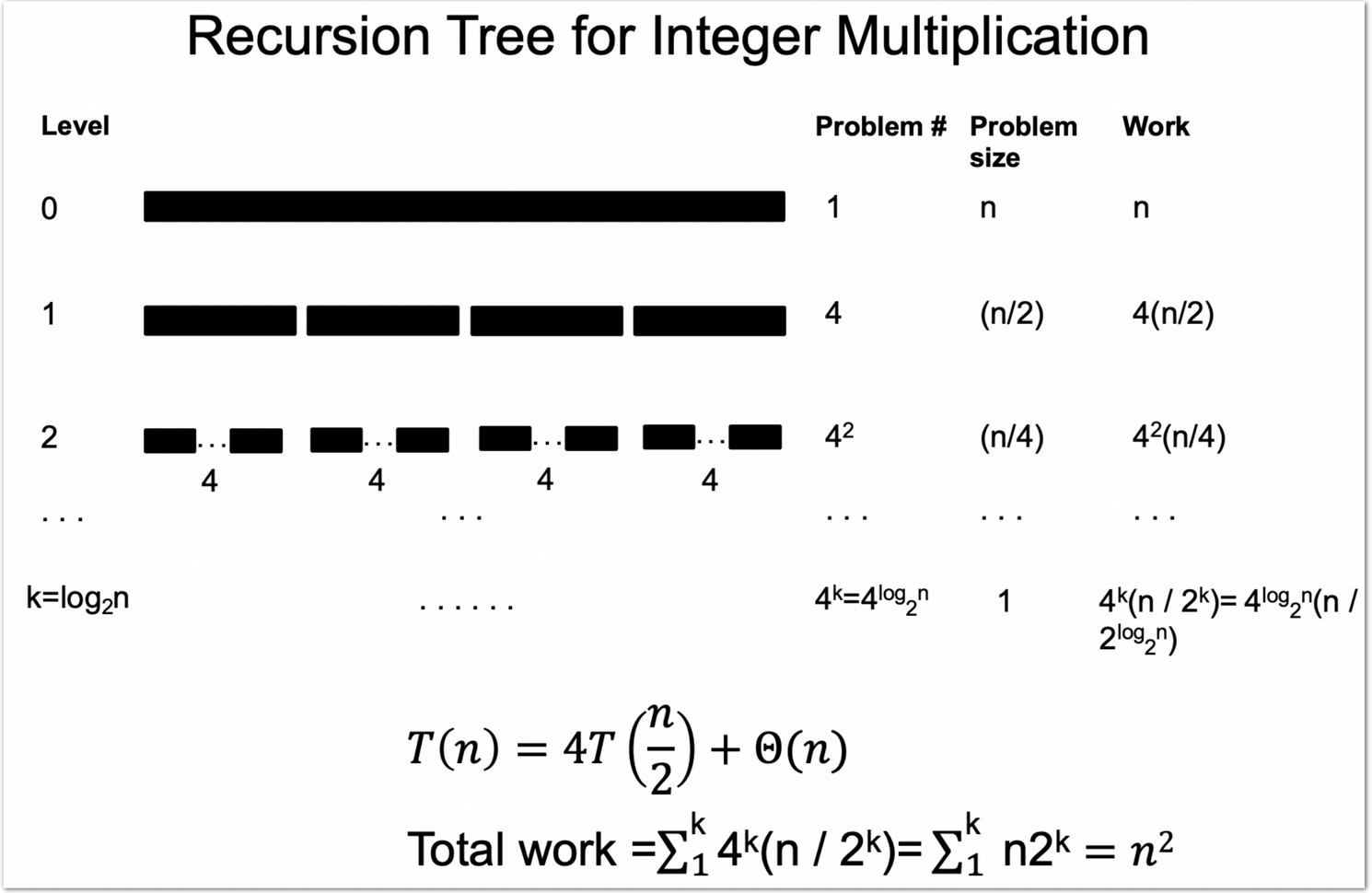
1. Topics that reviewed in this discussion may not be covered in the Exam
2. Topics that not reviewed in this discussion may be covered in the Exam

Checklist - Divide & Conquer

- Divide a size n problem into a $(a \geq 1)$ sub-problems with size n/b.
- Template of Divide & Conquer
  - Step 1: Divide the problem into sub-problems (Divide)
  - Step 2: Solve each sub-problem (Conquer)
  - Step 3: Combine the results of sub-problems. (Combine)
- The examples you have learned:
  - Merge Sort, Quick Sort
  - Integer Multiplication, Matrix Multiplication

Checklist - Divide & Conquer

Solve Recursive Function: Recursion Tree



1. Calculate how many levels.
2. Calculate how many recursive calls each level has.
3. Calculate how many work does each recursive call has. (Then calculate the total work on each level)
4. Calculate how many total work you need to do.

Checklist - Divide & Conquer

Solve Recursive Function: Master theorem

If $T(n) = aT\left(\frac{n}{b}\right) + \Theta(n^d)$ for constants $a > 0, b > 1, d \geq 0$, then:

$$T(n) = \begin{cases} O(n^d) & if\ d > log_b a \\ O(n^d logn) & if\ d = log_b a \\ O(n^{log_b a}) & if\ d < log_b a \end{cases}$$

Checklist - Tree

Tree Structure

- Root, parent, children.
- Degree: the number of its children.
- Leaf: nodes with degree zero.
- Path: a sequence of nodes: $(a_0, a_1, \ldots, a_n)$
- Height: the maximum depth of any node. The height of a tree with one node is 0.
- Ancestor, descendant.

Checklist - Tree

Tree Traversal

- Without specification, traverse from left to right!
- BFS:
  - $\Theta(n)$ runtime and $O(n)$ memory, maximum nodes at a given depth.
  - Queue.
- DFS:
  - $\Theta(n)$ runtime and $\Theta(h)$ memory, h is the height of the tree.
  - Stack or recursion (still stack). (Inverse order)
- Pre-order / Post-order / In-order
  - Given in-order and pre-order, how to get post-order?
  - Recall pre-order: root, left, right; post-order: left, right, root; in-order: left, root, right

Remind

- If we ask you to write the time complexity, you must write the most . However, if the question says $\Theta(n) = O(n)$, that's true.

Checklist - Binary Tree

- Each node has at most 2 children.
- Full binary tree: all the nodes have 0 or 2 children.
- Complete binary tree: filled at each depth from left to right.
  - Recursive definition:
    - The left sub-tree is a complete tree of height $h-1$ and the right subtree is a perfect tree of height $h-2$, or
    - The left sub-tree is perfect tree with height $h-1$ and the right sub-tree is complete tree with height $h-1$
- No relationship between full and complete binary tree.
- Perfect binary tree: height h with $2^{h+1}-1$ nodes.
- Array storage may not be the best solution.

Checklist - Binary Heap

- Min-Heap: The key of the root is less than or equal to the keys of the sub-trees, and the sub-trees (if any) are also min-heaps. (Recursive definition)
  - There is not relationship between children!
- We will do better in complete binary tree. (Avoid unbalanced binary tree.)
- We may store a complete tree using an array. **E.g.** Starts at index = 1. Then, for index i, its parent is $i/2$, and its children are $2i$ and $2i + 1$. (If starts at index = 0?)
- Operations: **Top, Pop, Push**.
  - For push, insert at the back, and compare the value of its parent.
  - For pop, delete the index = 1, and then copy the last entry to the top. Then compare the
  - value of its both children.
  - Access: $\Theta(1)$, pop and push: $O(ln(n))$

Checklist - Binary Heap

Build Heap

- Build heap: **Floyd's Method**
  - No percolation for the leaf nodes (n/2 nodes)
  - At most n/4 nodes percolate down 1 level
  - At most n/8 nodes percolate down 2 levels
  - At most n/16 nodes percolate down 3 levels
  - …
  - $$1\frac{n}{4} + 2\frac{n}{8} + 3\frac{n}{16} + \cdots = \sum_{i=1}^{\log n} i\frac{n}{2^{i+1}} = \frac{n}{2}\sum_{i=1}^{\log n}\frac{i}{2^i} = n = \Theta(n)$$

Checklist - Heap Sort

- Place the objects into a heap (Floyd's Method)
  - $O(n)$
- Repeatedly popping the top object until the heap is empty
  - $O(nlogn)$
- Time Complexity:
  - $O(nlogn)$

Common Mistakes in Homework

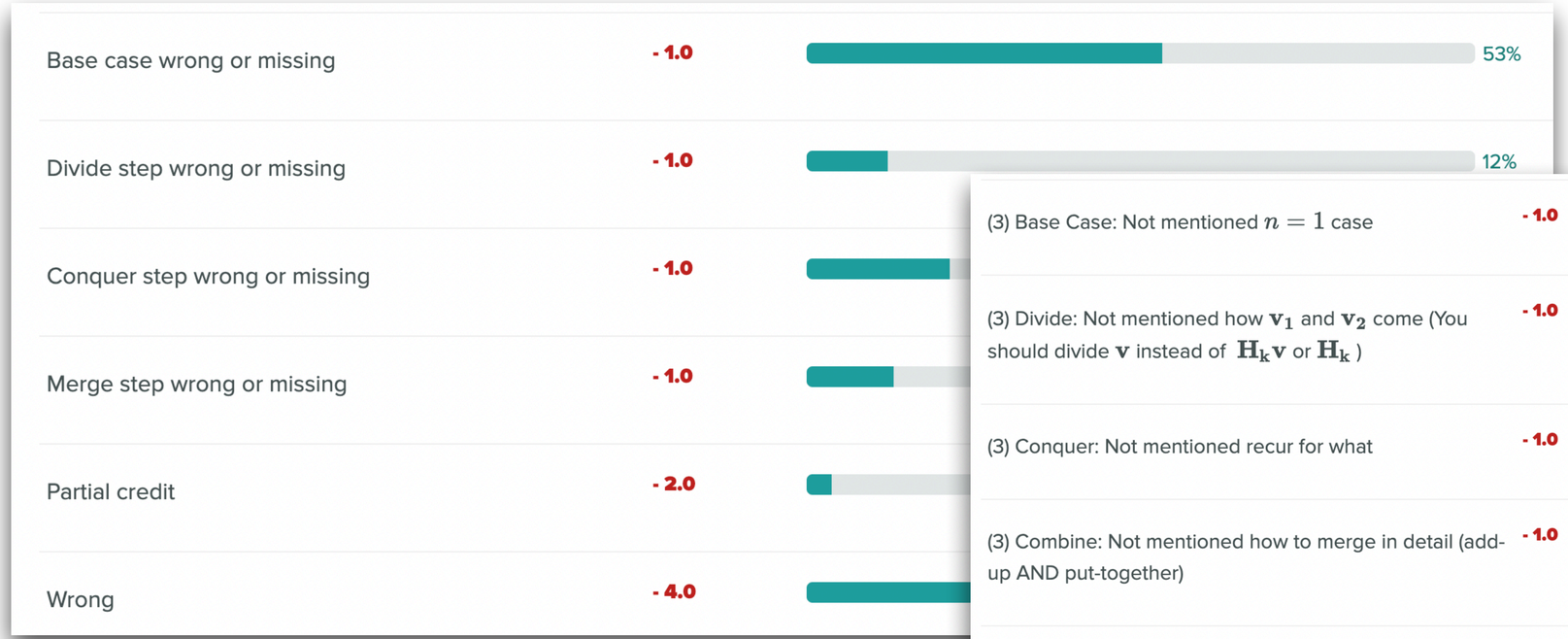
HW4

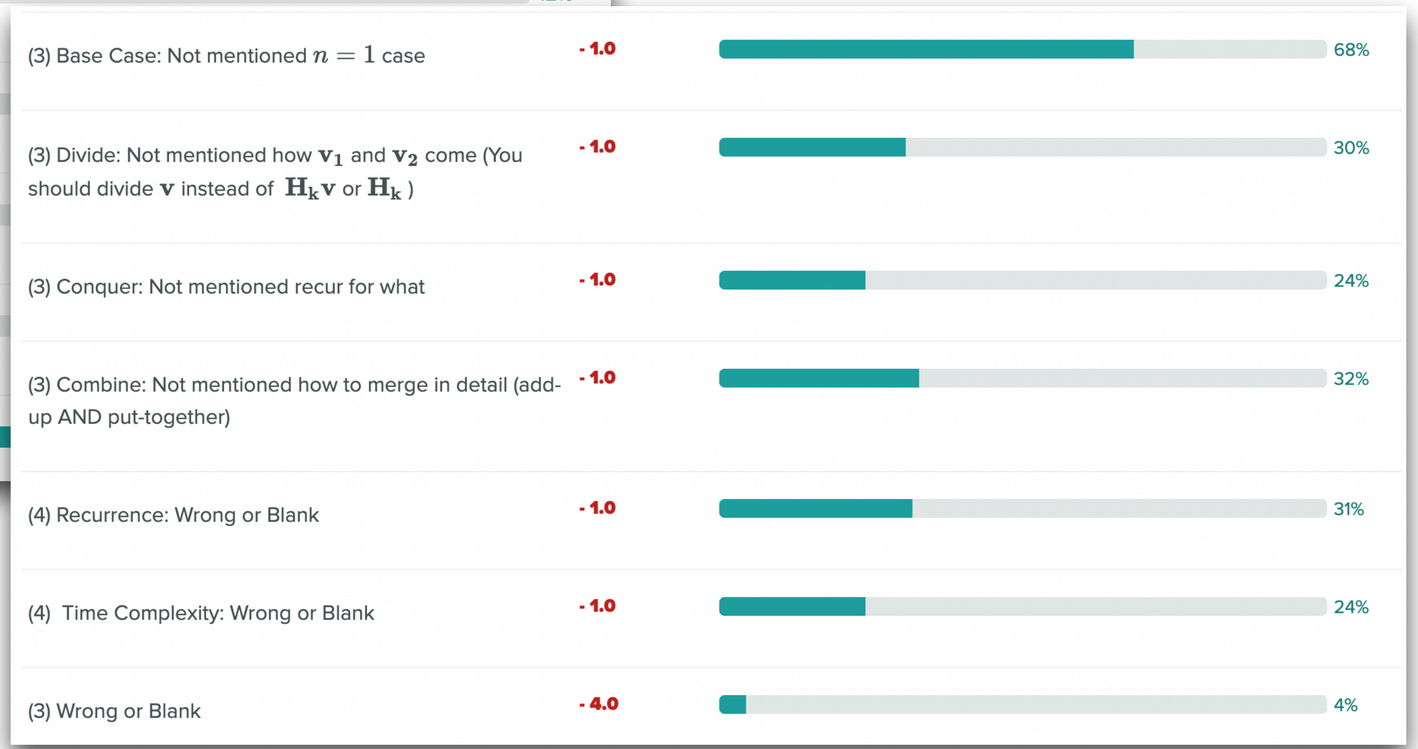
- Divide and Conquer Algorithm

不要直接想问题怎么解决，先分成子问题，一般是分2份或者4份；
然后假设子问题已经解决了，只需要想如何combine

Common Mistakes in Quiz

- Quiz 4
  - Divide and Conquer Algorithm





Pay more attention to details!

(3) Use Strassen's algorithm from (2) to come up with a divide-and-conquer algorithm to calculate the matrix multiplication $\mathbf{A} \times \mathbf{B}$ in more efficient than $\Theta(n^3)$ time. Write down your main idea briefly. (4pts)

1. If the problem is reduced into $n = 1$ i.e. $k = 0$, return the scalar product ab.
2. Else we partition $\mathbf{A}$ and $\mathbf{B}$, and calculate all multipliers and multiplicands in each $\mathbf{P_i}$. **(Divide)**
3. Recur for each $\mathbf{P_i}$, all seven of which are subproblems of size $n/2$. **(Conquer)**
4. Compute $(\mathbf{A} \times \mathbf{B})_{\mathbf{ij}}$ accordingly using linear combinations of $\mathbf{P_i}$, put them together to form the solution. **(Merge)**

Good Luck