Week 6 - Monday

COMP 4500

Last time

- What did we talk about last time?
- Started compression

Questions?

Assignment 3

Logical warmup: Revenge of the hats

- 20 prisoners are scheduled to be executed tomorrow
- They will be lined up in a row facing the same direction, each randomly wearing a white hat or a black hat
- Thus, one prisoner can see 19 prisoners, the next can see 18 prisoners, etc.
- Tomorrow, each prisoner, in order, starting with the last (who can see the other 19) will be asked the color of his hat
- He can answer "white" or "black"
- If he answers correctly, he is spared, otherwise he is shot
- If anyone answers something other than "white" or "black," everyone is killed immediately
- Since the prisoners were told the scenario a day a head of time, what strategy can they develop to save the maximum number of prisoners?

Finish Exam 1 Post-Mortem

Data Compression

Algorithm description

- Take the two lowest frequency letters ***y*** and ***z***.
- Since they are neighbors in a full tree, we can stick them together and treat them like a meta-letter ***yz*** with the sum of their frequencies.
- Recursively repeat until everything is merged together.

Algorithm

- If S has two letters then
  - Encode one with 0 and the other with 1
- Else
  - Let y and z be the two lowest-frequency letters
  - Form a new alphabet S' by deleting y and z and replacing them with a new letter w of frequency $f_y + f_z$
  - Recursively construct a prefix code for S' with tree T'
  - Define a prefix code for S as follows:
    - Start with T'
    - Take the leaf labeled w and add two children below it labeled y and z

Practice: Make a prefix code tree

Letter	Frequency
a	0.06
b	0.05
c	0.16
d	0.26
e	0.19
f	0.28

Three-sentence Summary of Mergesort

Divide and Conquer

Divide and conquer

- **Divide and conquer** algorithms are ones in which we divide a problem into parts and recursively solve each part
- Then, we do some work to combine the solutions to each part into a final solution
- Divide and conquer algorithms are often simple
- However, their running time can be challenging to compute because recursion is involved

Mergesort algorithm

- If there are two elements in the array or fewer then
  - Make sure they're in order
- Else
  - Divide list into two halves
  - Recursively merge sort the two halves
  - Merge the two sorted halves together into the final list

Time for mergesort

- The algorithm is simple, but recursive
- We'll use ***T*(*n*)** to describe the total running time recursively
  - $T(n) \leq c, \qquad n \leq 2$
  - $T(n) \leq 2T\left(\frac{n}{2}\right) + cn, \qquad n > 2$
- Is it really the same constant ***c*** for both?
  - No, but it's an inequality, so we just take the bigger one

Recursive running time

- If we can, we want to turn the recursive version of ***T*(*n*)** into an explicit (non-recursive) Big Oh bound
- Before we do, note that we could similarly have written:

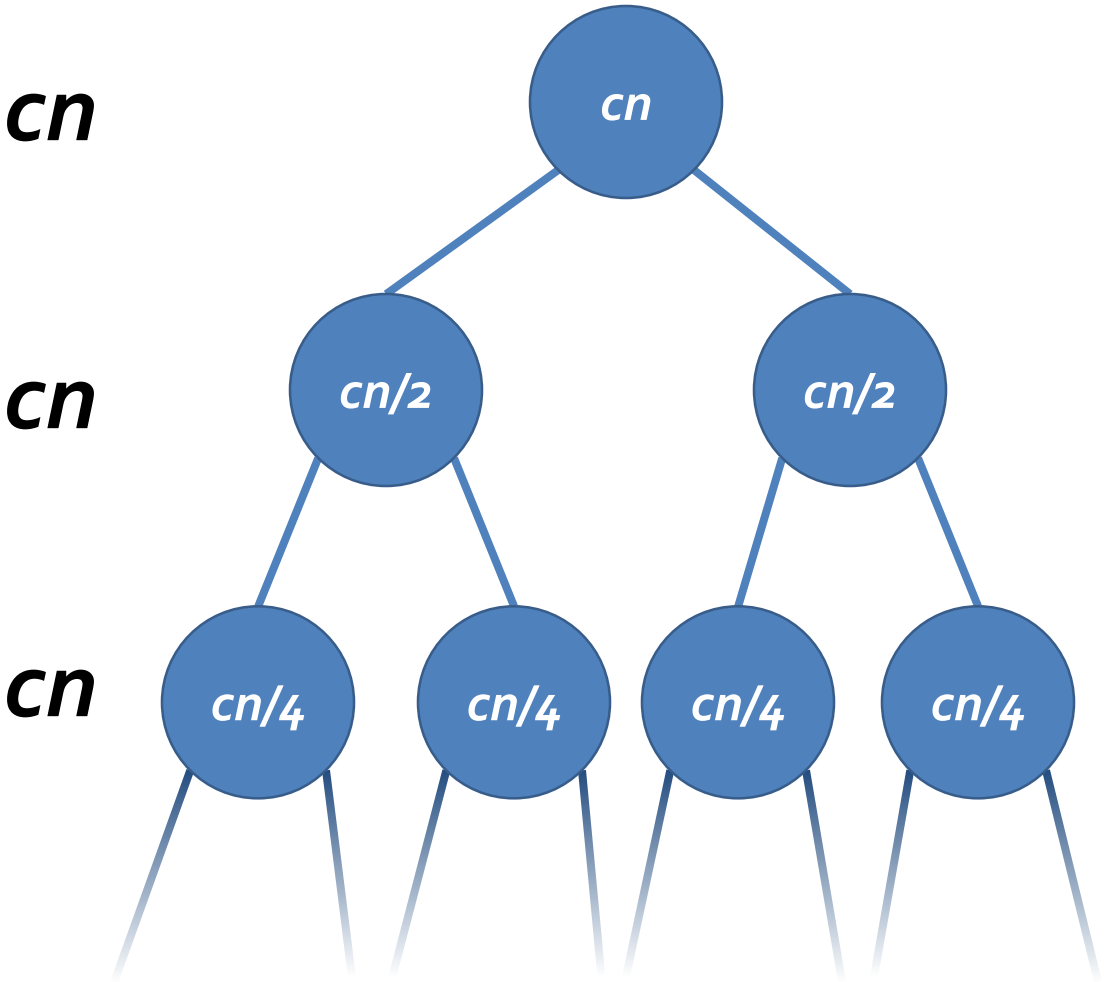
$$T(n) \leq 2T\left(\frac{n}{2}\right) + O(n)$$

- Also, we can't guarantee that ***n*** is even
- A more accurate statement would be

$$T(n) \leq T\left(\left\lceil\frac{n}{2}\right\rceil\right) + T\left(\left\lfloor\frac{n}{2}\right\rfloor\right) + cn$$

- Usually, we ignore that issue and assume that ***n*** is a power of 2, evenly divisible forever

Intuition about mergesort recursion



- Each time, the recursion cuts the work in half while doubling the number of problems
  - The total work at each level is thus always ***cn***
- To go from ***n*** to 2, we have to cut the size in half $(\log_2 \boldsymbol{n}) - 1$ times

Checking a solution

- We know that there's ***cn*** work at each level and approximately $\log_2$ ***n*** levels
- If we think that the running time O(***n*** log ***n***), we can guess that ***T***(***n***) ≤ ***cn*** $\log_2$ ***n*** and substitute that in for ***T***(***n***/2)

$$T(n) \leq 2T\left(\frac{n}{2}\right) + cn$$
$$\leq 2c\left(\frac{n}{2}\right)\log_2\left(\frac{n}{2}\right) + cn$$
$$= cn\,(\log_2 n - 1) + cn$$
$$= cn\,\log_2 n - cn + cn$$
$$= cn\log_2 n$$

Upcoming

Next time…

- More recurrence relations

Reminders

- **Assignment 3 is due on Friday**
- Read section 5.2
- Extra credit opportunities (0.5% each):
  - **Phadke research talk: 2/12 3-4 p.m. in Point 139**
  - Phadke teaching demo: 2/13 10-10:55 a.m. in Towers 112
  - Hristov teaching demo: 2/19 11:30-12:25 a.m. in Point 113
  - Hristov research talk: 2/19 4:30-5:30 p.m. in Point 139