lecture 17 - Oct 14 - midterm review

ProctorTrack onboarding:
- did you get an email confirming success?
- did you see how to enter password in URCourses and see the exam question?

- Midterm exam password will be shared prior to exam, but will also be shown in ProctorTrack when starting the exam. Be sure to enter it in UR Courses!
- Have blank paper and pencil ready on the side, BUT: All answers entered in URCourses via keyboard (no image upload)
- be alone in your room!
- use URCourses chat ONLY in order to ask me questions to clarify the meaning of an exam problem.
- write Big-O, Omega, Theta, N^2, 2^k, etc., to ease typing of mathematical notation.

<= >=

Chapter 0. Introduction

How did our two algorithms for Fibonacci numbers differ? To what effect?

$\Rightarrow$ algorithm design can greatly affect runtime

note: space - time trade-off

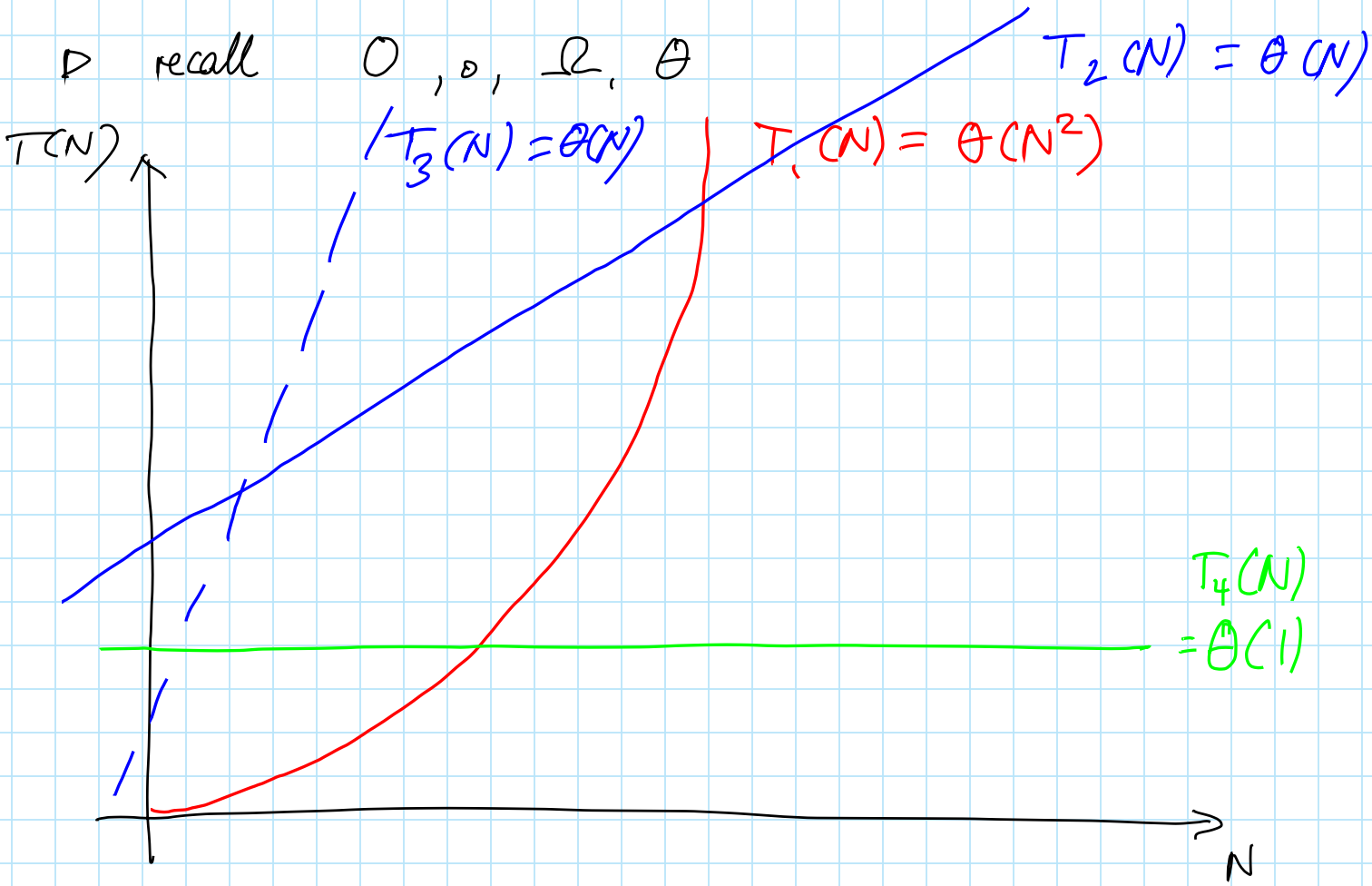
Chapter 1. Algorithm Analysis

▷ simplified, abstract model (why?)

$T_{avg}(N)$, $T_{worst}(N)$, $T_{best}(N)$

$T_{avg}(N)$ is <u>not</u> related to <u>amortized</u> cost!

▷ recall O, o, Ω, Θ



$T_2(N) = \Theta(T_3(N))$; $T_2(N) = o(T_1(N))$; $T_2(N) = \Omega(T_4(N))$

Theorems 1 and 2 about O, Ω, Θ are important to understand.

▷ Running time calculations

- consecutive fragments
- if / else
- loops, nested loop
- recursion

loops often yield standard sums:

$$\sum_{k=1}^{N} k = 1+2+3+\ldots+N = \frac{N(N+1)}{2} = \Theta(N^2)$$

$$\sum_{k=1}^{N} \frac{1}{2^k} = \frac{1}{2}+\frac{1}{4}+\frac{1}{8}+\ldots+\frac{1}{2^N} < 1 = \Theta(1)$$

recursion: formulate reccurence relation, try it out for a few steps (small values of N), until you see a pattern, guess a general formula, and verify by induction

▷ recall important basics on trees / BSTs

▷ splay tree operations and the notion of amortized cost

- insertion into splay tree of size N has worst-case

- however, amortized cost for insertion is $O(\log(N))$

Chapter 2. Priority Queues

▷ binary heap (min-heap)

array representation

percolating up/down

all operations, e.g.:

$O(\log(N))$ insertion, deletemin

$\Theta(N)$ buildHeap

▷ d-heaps

▷ leftist heaps

▷ skew heaps

▷ binomial queues

Chapter 3. Sorting

always in increasing order

▷ Insertion Sort $T_{worst}(N) = T_{avg}(N) = \Theta(N^2)$

to do better, one must sometimes exchange non-adjacent elements

▷ Shellsort: different gap sequences

$\Rightarrow$ effect on runtime?

When a general formula, loop analysis, ..., depending on N, is confusing, first try out $N = 1, 2, 3, 4, \ldots$