# Exercises Week 10
# Recurrence trees

Hans Georg Schaathun

12th November 2015

**Period** 21–27 October 2015

**Reading** Stein *et al* cover this material in Chapter 4.

**Programme** This document details the programme for the week, including exercises and pointers to other material. It is available in two versions:

1. as a PDF document.
2. as a web site. This depends on MathML and may require firefox/iceweasel to display correctly.

The web version includes inline video. The pdf version shows a still image from the video, providing a hyperlink directly to the video on the web site.

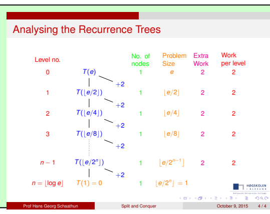
## 1 Wednesday 21 October

### 1.1 Recurrence trees

**Related reading:** Stein *et al.* p. 228-231+ or Rosen p. 359+,512

THEORY

We consider the number of multiplications needed to calculate $x^e \mod n$ using square and multiply. This is a short example of the use of recurrence trees.



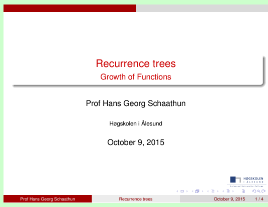

OGG

MP4

Slides

**THEORY**

This video from last year gives an example of a recurrence tree based on the Tower of Hanoi. It is similar to the previous video, but not as well made.




OGG

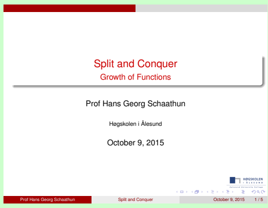
MP4

Slides

**THEORY**

This third video, also from last years, uses a recurrence tree to count the number of comparisons needed for Merge Sort.

1. Algorithm MergeSort$(A_1, A_2, \ldots, A_n)$
2. if $n == 1$, return $A$
3. $B = \text{MergeSort} A_{1,2,\ldots,\lfloor n/2 \rfloor}$
4. $C = \text{MergeSort} A_{\lfloor n/2 \rfloor+1, \lfloor n/2 \rfloor+2, \ldots, n}$
5. return Merge $B, C$




OGG

MP4

Slides

**Exercise 1.1** *Draw a recursion tree for*

$$T(n) = 3(T(n/3)) + n, \quad \textit{when } n \geq 2, \tag{1}$$
$$T(1) = 1. \tag{2}$$

*Assuming that $n$ is a power of three, use the recursion tree to find an exact solution for $T(n)$.*

**Exercise 1.2** *Draw a recursion tree for*

$$T(n) = 2(T(n/4)) + 3n, \quad \textit{when } n \geq 1, \tag{3}$$
$$T(1) = 1. \tag{4}$$

*Assuming that $n$ is a power of four, use the recursion tree to find an exact solution for $T(n)$.*

**Exercise 1.3** *Consider again the recursion trees from Exercises 1.1 and 1.2. Discuss what happens to the solution if you relax the assumption that $n$ be a power of three (resp. four).*

**Exercise 1.4** *Revisit the recursion trees from Exercises 1.1 and 1.2. Use the diagrams to find Big-O bounds on the two recurrence equations.*

**Exercise 1.5** *Consider an array $A$ of objects, where each object $o$ has a key $k(o)$. Consider an algorithm which takes a search key $k_0$ as input and outputs an element $o \in A$ so that $k_0 = k(o)$.*

1. *Write pseudo-code for a search algorithm. How many objects must be considered before the right element is found? Give answers for the worst case and the average case.*

2. *Suppose the array $A$ is sorted with keys $k(o)$ in increasing order. How does that affect searching? Write pseudo-code for a faster algorithm, taking advantage of the search order. How many objects must now be considered in the worst case before the right element is found?*

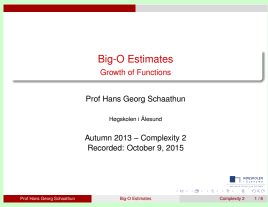
**Remark 1** *The two search algorithms in Exercise 1.5 are called* ***linear*** *and* ***binary search*** *respectively.*

## 1.2 More on complexity

THEORY

**Definition 1** *Let $f : \mathbb{N} \to \mathbb{R}$ and $g : \mathbb{N} \to \mathbb{R}$. We say that $f(n)$ is $O(g(n))$ if there are constants $c$ and $k$ such that*

$$\forall n \geq k, |f(n)| \leq c \cdot |g(n)|$$




OGG

MP4

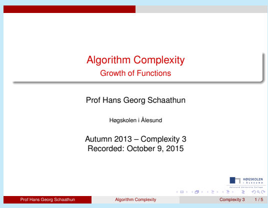
Slides

**Related reading:** Rosen p. 210+

EXERCISE EXAMPLE

**Exercise 1.6** *Consider the solution to the Tower of Hanoi. What is the complexity, in terms of the number of disks $n$?*

```
procedure HanoiMove(n) from s to t using a
if n = 1, then MoveDisc from s to t
else
    HanoiMove(n − 1) from s to a using t
    MoveDisc from s to t
    HanoiMove(n − 1) from a to t using s
```




OGG MP4 Slides

**Related reading:** Stein *et al.* p. 228-231 or Rosen p. 219+

**Exercise 1.7** *Consider*

$$f(x) = \sum_{i=0}^{n} a_i x^i$$

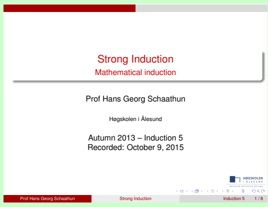
*Prove that $f(x) \in O(x^n)$.*

*In other words, find $c$ and $k$ so that $|f(x)| \leq c\,|g(x)|$ for $x \geq k$.*

# 2 Thursday 22 October

## 2.1 Other types of mathematical induction

THEORY



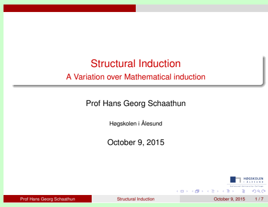

**Related reading:** Stein *et al.* p. 199-201 or Rosen p. 328-331

OGG

MP4

Slides

THEORY




**Related reading:** Stein *et al.* p. 206-208 or Rosen p. 347-350

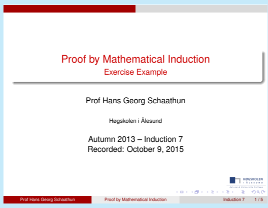
OGG

MP4

Slides

EXERCISE EXAMPLE

**Problem 2.1** *Prove that every natural number greater than 11 is the sum of a non-negative integer multiple of 3 and a non-negative integer multiple of 7.*




OGG

MP4

Slides

**Exercise 2.1** *Prove that every natural number greater than 7 is the sum of a non-negative integer multiple of 3 and a non-negative integer multiple of 5.*

**Exercise 2.2** *Prove that the Strong Principle of Mathematical induction is valid. (You can use contradiction, following the pattern used in the Video introducing induction.)*

**Exercise 2.3** *A set $S$ can be defined as follows.*

*Either $S$ is the empty set $\emptyset$, or $S$ is the union $S = S' \cup \{x\}$ of a set $S'$ and a singleton set containing some element $x$.*

*Use structural induction to show that the number of possible subsets of $S$ is $2^n$ where $|S| = n$ is the number of elements of $S$.*

## 2.2 More on complexity

**THEORY**

We have seen how Big-O is used to give asymptotic *upper* bounds. Similarly, we have Big-$\Omega$ as an asymptotic *lower* bound. And if we have the same Big-$O$ and -$\Omega$ bound, we write Big-$\Theta$ for it.


OGG

MP4

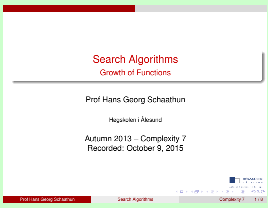
Slides

**Related reading:** Rosen p. 215-217

**THEORY**

**Problem 2.2** *Consider an arbitrary (unsorted) array $A$ of objects, and a search key $k_0$. Every element $x \in A$ has a key $k(x)$. Give an algorithm to return $x$ such that $k(x) = k_0$.*

**Problem 2.3** *Consider an sorted array $A$ of objects, and a search key $k_0$. Every element $x \in A$ has a key $k(x)$. Give an algorithm to return $x$ such that $k(x) = k_0$.*




OGG MP4 Slides

**Related reading:** Rosen p. 196-198

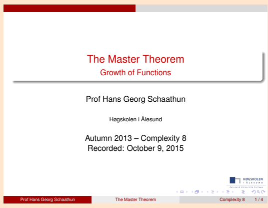
**SUPPORTING THEORY**

**Theorem 1** *Consider a natural number $a$ and real numbers $b \geq 1$, $c > 0$, and $d \geq 0$. Given a recurrence of the form*

$$T(n) = \begin{cases} aT(n/b) + n^c & \textit{for } n > 1, \\ d & \textit{for } n = 1, \end{cases}$$

*restricting $n$ to powers of $b$, we get that*

$$T(n) \in \begin{cases} \Theta(n^c), & \textit{if } log_b a < c, \\ \Theta(n^c \log n), & \textit{if } log_b a = c, \\ \Theta(n^{\log_b a}), & \textit{if } log_b a > c. \end{cases} \tag{5}$$




OGG MP4 Slides

**Related reading:** Stein *et al.* p. 244-247 or Rosen p. 516

**Exercise 2.4** *Consider the binary search:*

1. *How many comparisons are needed in the worst case? Give a recurrence equation.*
2. *Draw er recurrence tree.*
3. *Give a closed form expression solving the recurrence.*

**Exercise 2.5** *Give Big-O bounds for*

1. *linear search.*
2. *binary search.*

*Can the bounds be improved? Give Big-*$\Theta$ *bounds.*

**Exercise 2.6** *How many multiplications are needed to calculate* $a^x \mod n$ *in the worst case, using square-and-multiply?*

1. *Give a recurrence equation in terms of* $x$.
2. *Solve the recurrence.*
3. *Give a Big-O bound on the complexity.*

## 3 Compulsory Exercises (Tuesday 27 October 2015)

**Exercise 3.1** *Show that* $\log_{10} n \in \Theta(\log_2 n)$. *(You need to find the constants* $k$ *and* $c$ *for both the Big-O and Big-*$\Omega$ *bound.)*

**Exercise 3.2** *Find exact solutions to the following recurrence equations*

1. $T(n) = 2(T(n/2)) + 1$, $T(1) = 1$
2. $T(n) = 2(T(n/3)) + n$, $T(1) = 1$

**Exercise 3.3** *Find exact solutions to the following recurrence equations*

1. $T(n) = 3(T(n/2)) + n$, $T(1) = 1$
2. $T(n) = 2(T(n/2)) + n^2$, $T(1) = 1$

**Exercise 3.4** *Consider the claim* ***there is no largest prime number****. (In other words, the set of prime numbers is infinite.) Complete the following proof for the claim.*

*We make the proof by contradiction, so we assume that there is a largest prime number which we call* $p$.

*Let* $N$ *be one more than the product of all the primes from 2 to* $p$ *inclusive. In other words*

$$N = 1 + \prod_{i=1}^{n} p_i,$$

*where* $p_1, p_2, \ldots, p_n$ *is the list of all primes from 2 to* $p$.

*We can show that* $N$ *is not divisible by any prime* $p_i \leq p$ *because* $N \mod p' = 1$ *by the definition of* $N$.

...

**Exercise 3.5** *Consider the Fibonacci sequence* $1, 1, 2, 3, 5, 8, 13, \ldots$. *Let* $f_n$ *denote the* $n$*th number in this sequence. Give a recursive definition of* $f_n$.

**Exercise 3.6** *Use mathematical induction to show that for* $n \geq 3$ *we have*

$$f_n > \left(\frac{1+\sqrt{5}}{2}\right)^{n-2}$$

*where* $f_n$ *is the* $n$*th number of the Fibonacci sequence.*