csce350 — Data Structures and Algorithms
Fall 2019 — Lecture Notes: Counting Operations in Recursive Algorithms

This document contains slides from the lecture, formatted to be suitable for printing or individual reading, and with occasional supplemental explanations added. It is intended as a supplement to, rather than a replacement for, the lectures themselves — you should not expect the notes to be self-contained or complete on their own.

1 Introduction

2.4

If we want to count the basic operations executed by a recursive algorithm, then we can usually:

WRITE AND SOLVE A RECURRENCE.

2 What is a recurrence?

Informally, **recurrence** is an equation that defines a function of n in terms of that function's value for smaller values of n.

Example:

$$D(n) = \begin{cases} D\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + 1 & \text{if } n > 1 \\ 0 & \text{otherwise} \end{cases}$$

3 Example: Factorial

```
FACTORIAL(n)
  if n = 0 then
    return 1
  else
    return FACTORIAL(n − 1) · n
  end if
```

$$M(n) = \begin{cases} M(n-1) + 1 & \text{if } n > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{aligned} M(n) &= M(n-1) + 1 \\ M(0) &= 0 \end{aligned}$$

4 Solving via backward substitution

We can solve many recurrences like this using **backward substitution**:

$$
\begin{aligned}
M(n) &= M(n-1)+1 \\
&= (M(n-2)+1)+1 \\
&= M(n-2)+2 \\
&= (M(n-3)+1)+2 \\
&= M(n-3)+3 \\
&= \cdots \\
&= M(n-i)+i \\
&= M(n-n)+n \\
&= M(0)+n \\
&= n \\
&\in \Theta(n)
\end{aligned}
$$

5 Backward substitution: Steps

1. Write a recurrence, including its base case.
2. Establish a pattern in terms of i.
3. Choose i to get the base case to happen.

6 Example: Tower of Hanoi

7 Example: Tower of Hanoi

```
SOLVETOWER(n, a, c, b)
  if n > 1 then
    SOLVETOWER(n − 1, a, b, c)
  end if
  Move a disk from a to c.
  if n > 1 then
    SOLVETOWER(n − 1, b, c, a)
  end if
```

8 Tower of Hanoi recurrence

$$
\begin{aligned}
M(n) &= M(n-1) + 1 + M(n-1) \\
&= 2M(n-1) + 1 \\
&= 2(2M(n-2) + 1) + 1 \\
&= 4M(n-2) + 2 + 1 \\
&= 4(2M(n-3) + 1) + 2 + 1 \\
&= 8M(n-3) + 4 + 2 + 1 \\
&= \cdots \\
&= 2^i M(n-i) + \sum_{j=0}^{i-1} 2^j \\
&= 2^i M(n-i) + 2^i - 1 \\
&= 2^{n-1} M(n-(n-1)) + 2^{n-1} - 1 \\
&= 2^{n-1} M(1) + 2^{n-1} - 1 \\
&= 2^{n-1} + 2^{n-1} - 1 \\
&= 2^n - 1 \in \Theta(2^n)
\end{aligned}
$$

9 Example: Binary Digits (recursive version)

```
COUNTBINARYDIGITSRECURSIVELY(n)
  if n ≤ 1 then
    return 1
  else
    return COUNTBINARYDIGITSRECURSIVELY(⌊n/2⌋) + 1
  end if
```

10 Binary digits recurrence

$$
\begin{aligned}
A(n) &= A\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + 1 \\
&= A\left(\left\lfloor \frac{2^k}{2} \right\rfloor\right) + 1 \\
&= A\left(2^{k-1}\right) + 1 \\
&= A\left(2^{k-2}\right) + 2 \\
&= \cdots \\
&= A\left(2^{k-i}\right) + i \\
&= A\left(2^{k-k}\right) + k \\
&= A\left(1\right) + k \\
&= k \\
&= \log_2 n \in \Theta(\log n)
\end{aligned}
$$