Design and Analysis of Algorithms I

Introduction

Guiding Principles

Guiding Principle #1

“worst – case analysis” : our running time bound holds for every input of length n.

-Particularly appropriate for “general-purpose” routines

As Opposed to

--”average-case” analysis

--benchmarks

REQUIRES DOMAIN KNOWLEDGE

BONUS : worst case usually easier to analyze.

Tim Roughgarden

Guiding Principle #2

Won't pay much attention to constant factors, lower-order terms

Justifications

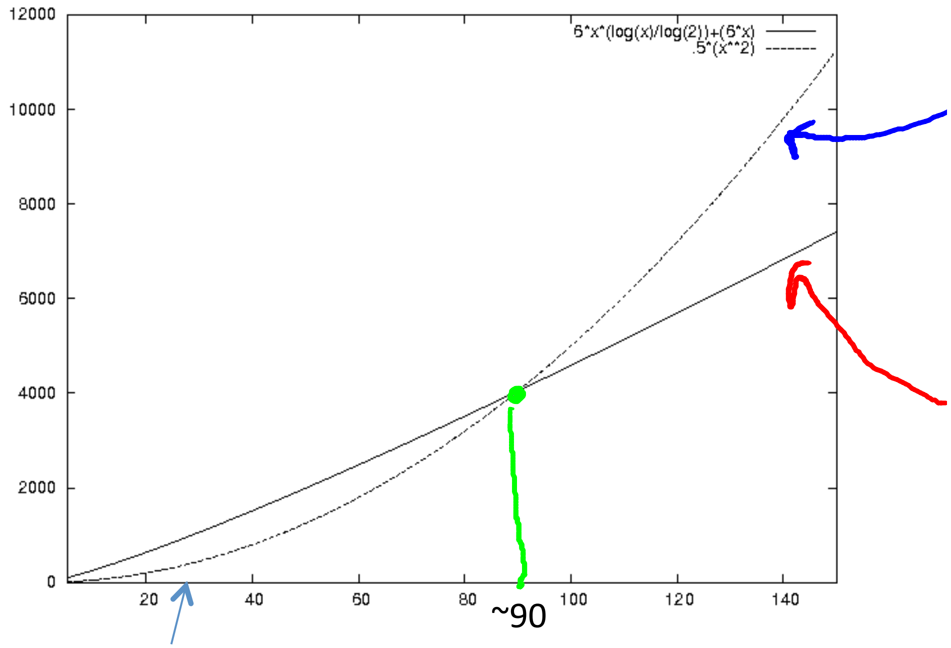
1. Way easier
2. Constants depend on architecture / compiler / programmer anyways
3. Lose very little predictive power
   (as we'll see)

Guiding Principle #3

<u>Asymptotic Analysis</u> : focus on running time for large input sizes n

Eg : $\underbrace{6n\log_2 +6n}_{\text{MERGE SORT}}$ “better than” $\underbrace{\frac{1}{2}n^2}_{\text{INSERTION SORT}}$

<u>Justification:</u> Only big problems are interesting!



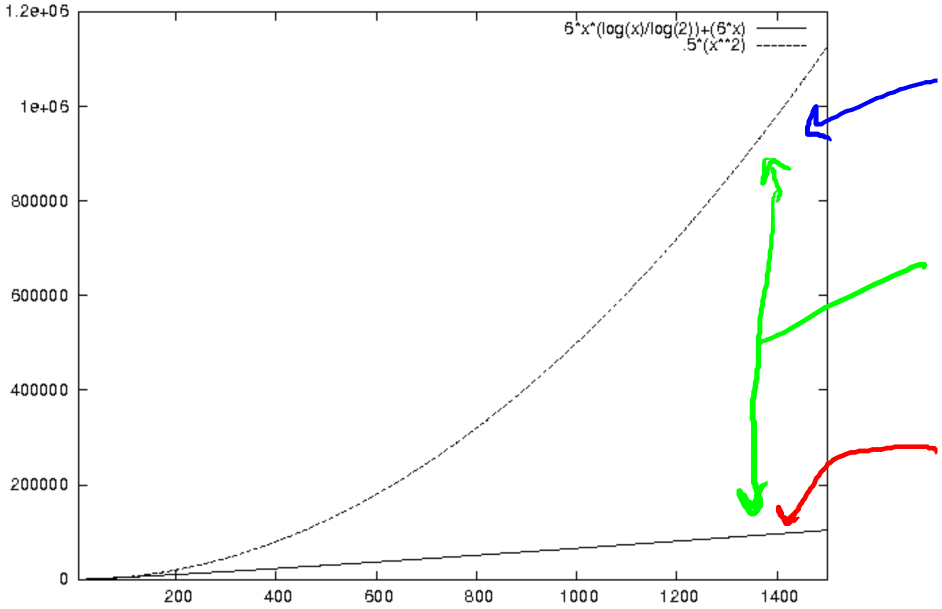
$\frac{1}{2}n^2$

$6n \log_2 n + 6n$

~90

Small n

Tim Roughgarden



$\frac{1}{2}n^2$

$10x$

$6n\log_2 n + 6n$

What Is a "Fast" Algorithm?

This Course : adopt these three biases as guiding principles

fast algorithm $\approx$ worst-case running time grows slowly with input size

Usually : want as close to linear (O(n)) as possible

Tim Roughgarden