S. KIMMEL

How to Solve Recurrances

* 1. Master Method *

2. Guess & Check

Assumptions:

- all subproblems have same size
- $T(n) \leq$ constant for $n \leq n_0$ (where n_0 is some small integer)
- $n > n_0$:

$$T(n) = aT\left(\frac{n}{b}\right) + O(n^d)$$ (a, b, d constants independent of n)

Q: What are a, b, d in words?

a: # of recursive calls

b: factor by which problem shrinks in recursive call

d: characterizes extra work outside recursive call (in combine step)

$$T(n) = \begin{cases} O(n^d \log n) & a = b^d \\ O(n^d) & a < b^d \\ O(n^{\log_b a}) & a > b^d \end{cases}$$

Three Cases

S. KIMMEL

Examples (divide into groups & have each group present)

1. Binary Search

$a = 1$ (only 1 recursive call)

$b = 2$ (new problem is ½ size of old)

$d = 0$ (constant work outside recursive call)

$T(1) = \text{constant}$

$b^d = 2^0 = 1 = a \Rightarrow$ Case 1

$T(n) = n^d \log n = n^0 \log n = \log n$ ✓ Sanity Check!

2. Merge Sort

$a = 2$ (2 recursive calls)

$b = 2$ (new problems ½ size of old)

$d = 1$ ($O(n)$ work outside of recursive call)

$T(1) = \text{constant}$

$b^d = 2 = a \Rightarrow$ case 1 ✓ Sanity Check!

$T(n) = n^d \log n = n \log n$