Master Method

Motivation

Design and Analysis of Algorithms I

Integer Multiplication Revisited

<u>Motivation</u> : potentially useful algorithmic ideas often need mathematical analysis to evaluate

<u>Recall</u> : grade-school multiplication algorithm uses $\theta(n^2)$ operation to multiply two n-digit numbers

A Recursive Algorithm

Recursive approach

Write $x = 10^{n/2}a + b$ $y = 10^{n/2}c + d$

[where a,b,c,d are n/2 – digit numbers]

So :

$$x \cdot y = 10^{n}ac + 10^{n/2}(ad + bc) + bd \qquad (*)$$

Algorithm#1 : recursively compute ac,ad,bc,bd, then compute (*) in the obvious way.

A Recursive Algorithm

T(n) = maximum number of operations this algorithm needs to multiply two n-digit numbers

Recurrence : express T(n) in terms of running time of recursive calls.

Base Case : T(1) <= a constant.

For all n > 1 : $$T(n) \leq 4T(n/2) + O(n)$$

Work done here

Work done by recursive calls

A Better Recursive Algorithm

<u>Algorithm #2 (Gauss)</u> : recursively compute $ac^{(1)}$, $bd^{(2)}$, $(a+b)(c+d)^{(3)}$ [recall ad+bc = (3) – (1) – (2)]

<u>New Recurrence :</u>

<u>Base Case :</u> T(1) <= a constant

Which recurrence best describes the running time of Gauss’s algorithm for integer multiplication?

- ○ $T(n) \leq 2T(n/2) + O(n^2)$
- ○ $3T(n/2) + O(n)$
- ○ $4T(n/2) + O(n)$
- ○ $4T(n/2) + O(n^2)$

A Better Recursive Algorithm

Algorithm #2 (Gauss) : recursively compute ac,(1) bd,(2)
(a+b)(c+d)(3) [recall ad+bc = (3) – (1) – (2)]

New Recurrence :

Base Case : T(1) <= a constant

For all n>1 : $T(n) \leq 3T(n/2) + O(n)$

Work done here

Work done by recursive calls