Recurrence Relations

Jonathan Backer
backer@cs.ubc.ca

Department of Computer Science
University of British Columbia

May 28, 2007

Introduction

Reading:

- CLRS: "Recurrences" 4.1-4.2
- GT: "Divide-and-Conquer" 5.2.1 (not Master Theorem)

We analysed the running time of iterative algorithms. Some algorithms are inherently recursive and we need new techniques to deal with them.

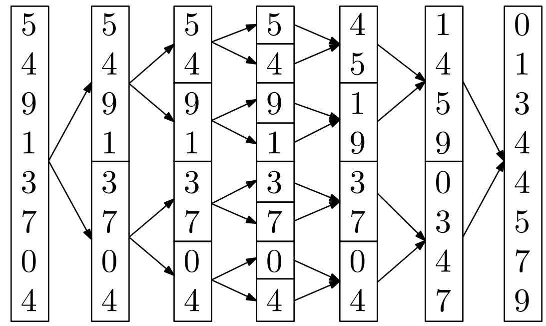
MergeSort

`MergeSort` uses an approach called "divide-and-conquer".

- Divide the problem into smaller subproblems.
- Solves the subproblems recursively.
- Combines the solutions of the subproblems to solve the initial problem.

Introduction (cont'd)

```
Algorithm MergeSort(A,p,r)
   if (p<r) then
      q ← ⌊(p+r)/2⌋
      MergeSort(A,p,q)
      MergeSort(A,q+1,r)
      merge(A,p,q,r)
```



Let $T(n)$ be the run-time on an input of size n. Then

$$T(n) = \begin{cases} b & \text{if } n = 1 \\ T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + dn & \text{if } n \geq 2 \end{cases}$$

For some $b, d \in \mathbb{Z}^+$.

Ignoring Discontinuities

- $\lfloor\rfloor$ and $\lceil\rceil$ often necessary to define recurrence.
- Tedious to analyse because of discontinuity.
- Recurrences with both are often worse because the discontinuities are out of sync.
- Ignore the problem:
  - Solve $T(n)$ when n is a power of 2.
  - Then $\lfloor n/2 \rfloor = \lceil n/2 \rceil = n/2$.
  - Not a true asymptotic bound, but result often generalizes.

$$T(n) = \begin{cases} b & \text{if } n = 1 \\ 2T(n/2) + dn & \text{if } n \geq 2 \end{cases}$$

Guess and Test: Inductive Step

Guess $T(n) \le cn\log n$, for $n = 2^k$.

Induction Step:

$$\begin{aligned} T(n) &= 2T(n/2) + dn \\ &\le 2[c(n/2)\log(n/2)] + dn \\ &\le cn\log(n/2) + dn \\ &\le cn[\log n - \log 2] + dn \\ &\le cn\log n - cn\log 2 + dn \\ &\le cn\log n \text{ if } cn\log 2 \ge dn \end{aligned}$$

Induction step goes through if $c \ge d/\log 2$.

Guess and Test: Base Case

Guess $T(n) \le cn\log n$, for $n = 2^k$.

Base case:

- $n = 2^0$: Doesn't work because $T(1) = b \ge 0 = cn\log n$.
- $n = 2^1$: Refers to $T(1)$. Must iterate recursion.

Carefully choose n_0.

$$\left.\begin{aligned} T(2) &= 2T(1) + 2d \\ &= 2b + 2d \\ &\le c \cdot 2\log 2 \end{aligned}\right\} \text{need } c \ge \frac{b+d}{\log 2}$$

Base case and inductive step restrict c. Take the maximum to satisfy both.

Hence, $T(n) \le \frac{b+d}{\log 2} \cdot n\log n$, for $n = 2^k$ and $k \ge 1$.

Handling Discontinuities

Squeeze $T(n)$ between two functions $m(n) \leq T(n) \leq M(n)$.

- $T(n) \in \Omega(m(n))$ and $T(n) \in O(M(n))$.
- Trade-off between
  - proving $m(n) \leq T(n) \leq M(n)$ and
  - solving for $m(n)$ and $M(n)$.

$$m(n) = \begin{cases} b & \text{if } n = 1 \\ 2m(\lfloor n/2 \rfloor) + dn & \text{if } n \geq 2 \end{cases}$$

$$M(n) = \begin{cases} b & \text{if } n = 1 \\ 2M(\lceil n/2 \rceil) + dn & \text{if } n \geq 2 \end{cases}$$

Prove Inequality: $T(n) \leq M(n)$.

Base Case: $T(1) = b = M(n)$. *Inductive Step*:

$$\begin{aligned} T(n) &= T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + dn \\ &\leq M(\lceil n/2 \rceil) + M(\lfloor n/2 \rfloor) + dn \text{ induction hypothesis} \\ &\leq M(\lceil n/2 \rceil) + M(\lceil n/2 \rceil) + dn \text{ if } M \text{ non-decreasing} \\ &= M(n) \end{aligned}$$

$M(n)$ is non-decreasing if $M(n) \leq M(n+1)$.

Base Case: $M(1) = b \leq 2b + dn = M(2)$. *Inductive Step*: If n is even,

$$\begin{aligned} M(n) &= 2M(\lceil n/2 \rceil) + dn \\ &\leq 2M(\lceil (n+1)/2 \rceil) + dn \text{ induction hypothesis} \\ &= M(n+1) \end{aligned}$$

If n is odd, $\lceil n/2 \rceil = \lceil (n+1)/2 \rceil$. So $M(n) = M(n+1)$.

Solve Recurrence: Inductive Step

Guess $M(n) \leq cn \log n$

Inductive Step:

$$\begin{aligned}
M(n) &= 2M(\lceil n/2 \rceil) + dn \\
&\leq 2c\left(\lceil n/2 \rceil \log \lceil n/2 \rceil\right) + dn \text{ induction hypothesis} \\
&\leq 2c([n+1]/2) \cdot \log([n+1]/2) + dn \text{ rounding up} \\
&= c[n+1] \log([n+1]/2) + dn \\
&= [cn + c] \log([n+1]/2) + dn \\
&= cn \log([n+1]/2) + c \log([n+1]/2) + dn \\
&\leq cn \log([n+1]/2) + c \log([n+n]/2) + dn \\
&\leq cn \log([n+1]/2) + c \log n + dn
\end{aligned}$$

Solve Recurrence: Inductive Step (cont'd)

Guess $M(n) \leq cn \log n$ (cont'd)

$$\begin{aligned}
M(n) &\leq cn \log([n+1]/2) + c \log n + dn \\
&= cn[\log(n+1) - \log 2] + c \log n + dn \\
&= cn \log(n+1) - cn \log 2 + c \log n + dn \\
&\leq cn(\log n + 1/n) - cn \log 2 + c \log n + dn \text{ sublinear} \\
&\leq cn \log n + c - cn \log 2 + c \log n + dn \\
&\leq cn \log n + dn - cf(n), \; f(n) = n \log 2 - \log n - 1 \\
&\leq cn \log n, \text{ if } cf(n) \geq dn
\end{aligned}$$

Can we choose c large enough? Depends on $f(n)$.

By inspection, $f(n) \in \Theta(n)$. Consider n_0 and e such that $f(n) \geq en$, for $n \geq n_0$. Then $cf(n) \geq cen \geq dn$, if $c \geq d/e$ and $n \geq n_0$.

Solve Recurrence: Base Case

Guess $M(n) \le cn \log n$

- Inductive step only worked for $n \ge n_0$.
- Must consider every $n < n_0$ a base case.
- Choose

$$c > \frac{M(n)}{n \log n}, \quad \text{for } 1 < n < n_0$$

May be many base cases.

- Make sure $c > d/e$ as well
- Then base cases and inductive step go through.
- Proves $M(n) \in O(n \log n)$.

Guess and Test (cont'd)

Recurrence: $T(n) = 2T(\lfloor n/2 \rfloor) + n$ with $T(1) = 1$

Guess $T(n) \le cn$

Inductive Step:

$$\begin{aligned} T(n) &= 2T(\lfloor n/2 \rfloor) + n \\ &\le 2c\lfloor n/2 \rfloor + n && \text{induction hypothesis} \\ &\le \tfrac{2cn}{2} + n = (c+1)n \end{aligned}$$

So $T(n) \le dn$, but $d \ne c$.

Only allowed to choose c once!

Guess and Test (cont'd)

Recurrence: $T(n) = T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + 1$ with $T(1) = 1$

Guess $T(n) \leq cn$

Inductive Step

$$\begin{aligned} T(n) &= T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + 1 \\ &\leq c\lfloor n/2 \rfloor + c\lceil n/2 \rceil + 1 && \text{induction hypothesis} \\ &= cn + 1 \end{aligned}$$

Sometimes a stronger guess makes the math work.

Guess and Test (cont'd)

Revised Guess: $T(n) \leq cn - 1$

Inductive Step

$$\begin{aligned} T(n) &= T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + 1 \\ &\leq (c\lfloor n/2 \rfloor - 1) + (c\lceil n/2 \rceil - 1) + 1 && \text{induction hypothesis} \\ &= c(\lfloor n/2 \rfloor + \lceil n/2 \rceil) - 1 \\ &= cn - 1 \end{aligned}$$

So the inductive step holds for all c.

Base case $(n = 1)$:

- Want $T(1) = 1 \leq c - 1$
- Satisfied for $c \geq 2$.

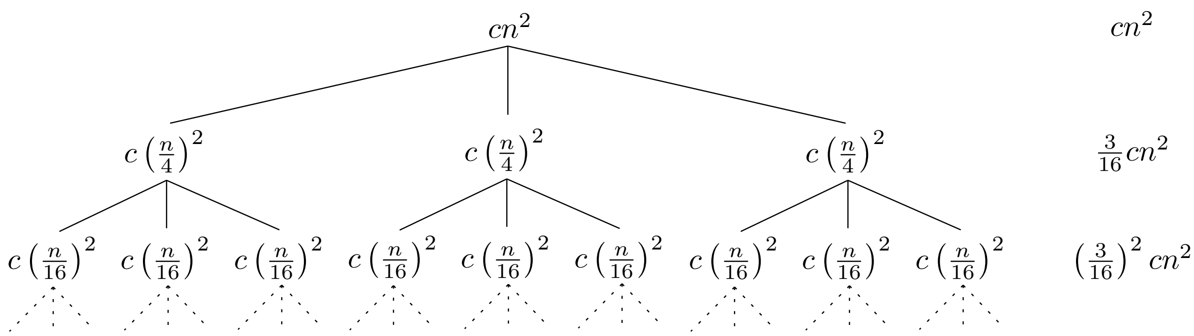
So the proof works for $c \geq 2$.

Iteration

- A tree represents the recursion.
- Add the costs for each level of the tree.
- The sums and the tree height bound the running time
- Used to generate guess, if accounting is sloppy.
- Assume n is appropriate power.

Internal Nodes Dominate

$$T(n) = \begin{cases} 3T(\lfloor n/4 \rfloor) + cn^2 & n \geq 4 \\ b & n \leq 4 \end{cases}$$



Internal Nodes Dominate (cont'd)

So we have

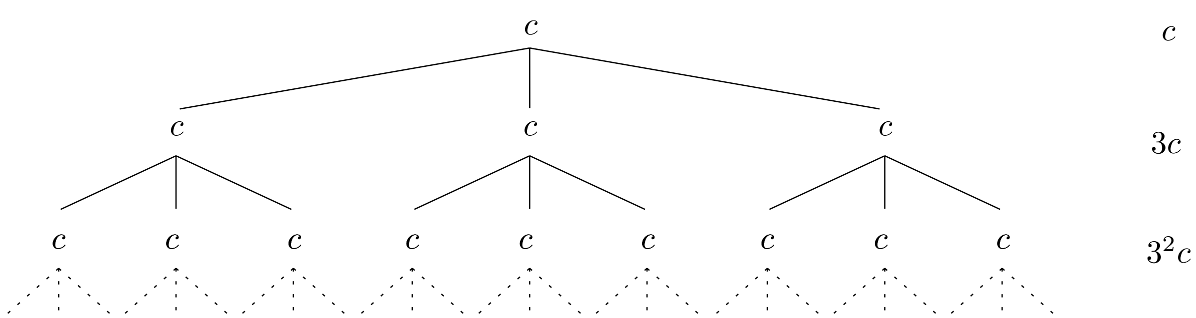
$$\begin{array}{lll} & \textit{internal nodes} & \textit{leaves} \\ T(n) & \leq cn^2\left[1+\frac{3}{16}+\left(\frac{3}{16}\right)^2+\ldots+\left(\frac{3}{16}\right)^{\log_4 n}\right] & +b\cdot 3^{\log_4 n} \\ & \leq cn^2\left[1+\frac{3}{16}+\left(\frac{3}{16}\right)^2+\ldots\right] & +b\cdot n^{\log_4 3} \end{array}$$

The infinite series $\sum_i (3/16)^i$ converges to a constant d, so

$$T(n) \leq cdn^2 + bn^{\log_4 3} \in O(n^2)$$

Leaves Dominate

$$T(n) = 3T(\lfloor n/2 \rfloor) + c \text{ with } T(1) = d$$



So we have

$$\begin{array}{lll} & \textit{internal nodes} & \textit{leaves} \\ T(n) & = c + 3c + \ldots + 3^{\log_2 n}c & +d3^{\log_2 n} \\ & = c\sum_{i=0}^{\log_2 n} 3^i & +dn^{\log_2 3} \end{array}$$

Leaves Dominate (cont'd)

Recall that $\sum_{i=0}^{k} a^i = \frac{a^{k+1}-1}{a-1}$. So

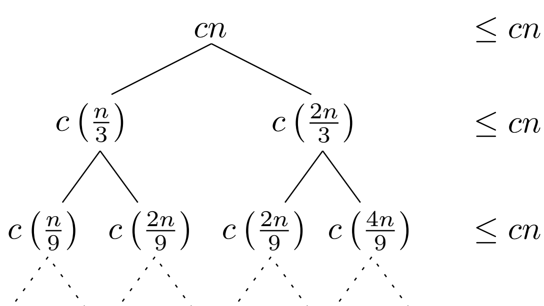
$$\begin{aligned}
\sum_{i=0}^{\log_2 n} 3^i &= \frac{3^{(\log_2 n)+1} - 1}{3-1} \\
&\leq \frac{3^{(\log_2 n)+1}}{3-1} \\
&\leq 3^{(\log_2 n)+1} \\
&\leq 3 \cdot 3^{\log_2 n} \\
&\leq 3 \cdot n^{\log_2 3}
\end{aligned}$$

Therefore,

$$T(n) \leq 3cn^{\log_2 3} + d^{\log_2 3} \in O(n^{\log_2 3})$$

Unbalanced Trees

$$T(n) = T(\lfloor n/3 \rfloor) + T(\lfloor 2n/3 \rfloor) + cn \text{ with } T(1) = T(2) = 1.$$



The leftmost branch is shorter than the rightmost branch.

So

$$\begin{aligned}
& \quad \textit{internal nodes} & & \textit{leaves} \\
T(n) & \leq cn \log_{\frac{3}{2}} n & & + 2^{\log_{\frac{3}{2}} n} \\
& = cn \log_{\frac{3}{2}} n & & + n^{\log_{\frac{3}{2}} 2}
\end{aligned}$$

Unbalanced Trees (cont'd)

- Leaf estimate assumes that the last level is full.
- Try node bound as a guess: $T(n) \le dn \log n$.

Inductive Hypothesis:

$$\begin{aligned}
T(n) &= T\left(\left\lfloor \frac{n}{3} \right\rfloor\right) + T\left(\left\lfloor \frac{2n}{3} \right\rfloor\right) + cn \\
&\le d\left\lfloor \frac{n}{3} \right\rfloor \log \left\lfloor \frac{n}{3} \right\rfloor + d\left\lfloor \frac{2n}{3} \right\rfloor \log \left\lfloor \frac{2n}{3} \right\rfloor + cn \\
&\le d\left(\frac{n}{3}\right) \log \left(\frac{n}{3}\right) + d\left(\frac{2n}{3}\right) \log \left(\frac{2n}{3}\right) + cn \\
&\le d\left(\frac{n}{3}\right) [\log n - \log 3] + d\left(\frac{2n}{3}\right) [\log n + \log 2 - \log 3] + cn \\
&= d\left(\frac{n}{3} + \frac{2n}{3}\right) \log n - d\left(\frac{n}{3} + \frac{2n}{3}\right) \log 3 + d\left(\frac{2n}{3}\right) \log 2 + cn \\
&= dn \log n - dn \log 3 + d\left(\frac{2n}{3}\right) \log 2 + cn \\
&= dn \log n + n\left[-d \log 3 + d\left(\frac{2}{3}\right) \log 2 + c\right]
\end{aligned}$$

Unbalanced Trees (cont'd)

Inductive Step:

$$T(n) \le dn \log n + n\,[-d \log 3 + 2d/3 \cdot \log 2 + c]$$

- Fortunately, $\log 3 > 2/3 \cdot \log 2$.
- Can make d large enough to cancel c.

Base Cases:

- Does not hold for $n = 1$ because $\log 1 = 0$.
  - So $n_0 = 2$.
  - Every $T(i)$ referring to $T(1)$ must be a base case.