CS3383 Lecture 1.2: Substitution method and randomized d&c

David Bremner David Bremner

February 15, 2024

Even More Divide and Conquer

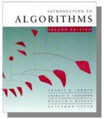
- Substitution Method for recurrences
- Quicksort
- Randomized Quicksort

/Subtype /Text/F 1/T (Video)/Contents (video/12.0-intro-a[redacted]

Even More Divide and Conquer

- Substitution Method for recurrences
- Quicksort
- Randomized Quicksort

/Subtype /Text/F 1/T (Video)/Contents (video/12.0-intro-[redacted]

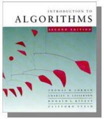


Substitution method

The most general method:

1. ***Guess*** the form of the solution.
2. ***Verify*** by induction.
3. ***Solve*** for constants.

/Subtype /Text/F 1/T (Video)/Contents (video/12.1-subst.mkv)



Substitution method

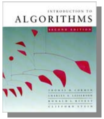
The most general method:

1. ***Guess*** the form of the solution.
2. ***Verify*** by induction.
3. ***Solve*** for constants.

/Subtype /Text/F 1/T (Video)/Contents (video/12.2-example.mkv)

EXAMPLE: $T(n) = 4T(n/2) + n$

- [Assume that $T(1) = \Theta(1)$.]
- Guess $O(n^3)$. (Prove O and Ω separately.)
- Assume that $T(k) \le ck^3$ for $k < n$.
- Prove $T(n) \le cn^3$ by induction.



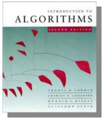
Example of substitution

$$
\begin{aligned}
T(n) &= 4T(n/2) + n \\
&\le 4c(n/2)^3 + n \\
&= (c/2)n^3 + n \\
&= cn^3 - ((c/2)n^3 - n) \longleftarrow \textit{desired} - \textit{residual} \\
&\le cn^3 \longleftarrow \textit{desired}
\end{aligned}
$$

/Subtype /Text/F 1/T (Video)/Contents (video/12.3-working-a.mkv)

whenever $(c/2)n^3 - n \ge 0$, for example, if $c \ge 2$ and $n \ge 1$.

↖ *residual*



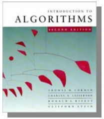
Example of substitution

$$
\begin{aligned}
T(n) &= 4T(n/2) + n \\
&\le 4c(n/2)^3 + n \\
&= (c/2)n^3 + n \\
&= cn^3 - ((c/2)n^3 - n) \leftarrow \textit{desired} - \textit{residual} \\
&\le cn^3 \leftarrow \textit{desired}
\end{aligned}
$$

/Subtype /Text/F 1/T (Video)/Contents
(video/12.3-working-b.mkv)

whenever $(c/2)n^3 - n \ge 0$, for example, if $c \ge 2$ and $n \ge 1$.

residual



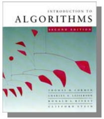
A tighter upper bound?

We shall prove that $T(n) = O(n^2)$.

Assume that $T(k) \leq ck^2$ for $k < n$:

/Subtype /Text/F 1/T (Video)/Contents (video/12.4-tighter.mkv)

$$\begin{aligned} T(n) &= 4T(n/2) + n \\ &\leq 4c(n/2)^2 + n \\ &= cn^2 + n \\ &= O(n^2) \end{aligned}$$



A tighter upper bound?

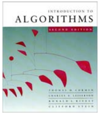
We shall prove that $T(n) = O(n^2)$.

Assume that $T(k) \le ck^2$ for $k < n$:

/Subtype /Text/F 1/T (Video)/Contents (video/12.5-fail.mkv)

$$
\begin{aligned}
T(n) &= 4T(n/2) + n \\
&\le 4c(n/2)^2 + n \\
&= cn^2 + n \\
&= O(n^2)
\end{aligned}
$$

Wrong! We must prove the I.H.



Divide and conquer

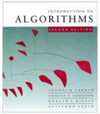
Quicksort an n-element array:

1. ***Divide:*** Partition the array into two subarrays around a ***pivot*** x such that elements in lower subarray $\leq x \leq$ elements in upper subarray.

/Subtype /Text/F 1/T (Video)/Contents (video/12.6-quicksort.mkv)

2. ***Conquer:*** Recursively sort the two subarrays.
3. ***Combine:*** Trivial.

Key: *Linear-time partitioning subroutine.*



Divide and conquer

Quicksort an n-element array:

1. ***Divide:*** Partition the array into two subarrays around a ***pivot*** x such that elements in lower subarray $\leq x \leq$ elements in upper subarray.

/Subtype /Text/F 1/T (Video)/Contents (video/12.6-quicksort-b.mkv)

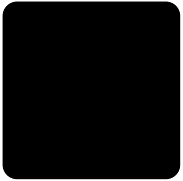
2. ***Conquer:*** Recursively sort the two subarrays.
3. ***Combine:*** Trivial.

Key: *Linear-time partitioning subroutine.*

Analysis of quicksort

- Quicksort is $\Theta(n^2)$ in the worst case. What kind of input is bad?
- Quicksort is supposed to be fast “in practice”.
- We can choose a better pivot in $O(n)$ time, but we’ll see it’s a bit complicated.
- What if we choose a random element as pivot?

/Subtype /Text/F 1/T (Video)/Contents (video/12.7-qs-time-a.mkv)



Analysis of quicksort

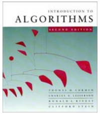
- Quicksort is $\Theta(n^2)$ in the worst case. What kind of input is bad?
- Quicksort is supposed to be fast "in practice".
- We can choose a better pivot in $O(n)$ time, but we'll see it's a bit complicated.
- What if we choose a random element as pivot?

/Subtype /Text/F 1/T (Video)/Contents
(video/12.7-qs-time-b.mkv)

Random pivot

- `pivot ←A[random(1..n)]`
- indicator $X_k = 1$ if we generate a $k : n - k - 1$ split, 0 otherwise
- $E[X_k] = Pr[X_k = 1] = 1/n$, assuming distinct elements.

/Subtype /Text/F 1/T (Video)/Contents (video/12.8-pivot.mkv)



Analysis (continued)

$$T(n) = \begin{cases} T(0) + T(n-1) + \Theta(n) & \text{if } 0 : n-1 \text{ split,} \\ T(1) + T(n-2) + \Theta(n) & \text{if } 1 : n-2 \text{ split,} \\ \vdots & \\ T(n-1) + T(0) + \Theta(n) & \text{if } n-1 : 0 \text{ split,} \end{cases}$$

$$= \sum_{k=0}^{n-1} X_k \big(T(k) + T(n-k-1) + \Theta(n) \big)$$

/Subtype /Text /F 1 /T (Video) /Contents (video/12.9-sum-of-indicators.mkv)

