Lecture 8: Recurrences

Agenda:

- Iterated substitution —– leading to
- Master theorem (simple version)

Reading:

- Textbook pages 63 – 75

Solving recurrence relations:

- Iterated substitution (done)
- Recursion tree (done)
- Master theorem (now)
- Divide-and-conquer: what form of recurrence relation does it have?
- Typical procedure:

```
Proc dnq(n)

    ......
    dnq(n/b) ... dnq(n/b)
    ......
    return
end dnq
```

- For the call dnq(n) assume:
  - running time is n^c — excluding recursive calls
  - there are a total of a calls to dnq($\frac{n}{b}$)
- Recurrence relation for total time $T(n)$

$$T(n) = \begin{cases} \text{bounded}, & \text{if } n < b \\ a \times T(\frac{n}{b}) + n^c, & \text{if } n \geq b \end{cases}$$

- Closed form solution?
  - Iterated substitution !!!
  - Simplifying assumption to $n = b^k$

Iterated substitution for $T(n)$:

$$
\begin{array}{lll}
 & T(b^k) & \\
= & a \times T(\frac{b^k}{b}) + b^{kc} & \text{definition} \\
= & a \times T(b^{k-1}) + b^{kc} & \text{arithmetic} \\
 & & \\
= & a \times \left(a \times T(\frac{b^{k-1}}{b}) + b^{(k-1)c}\right) + b^{kc} & \text{definition} \\
= & a^2 \times T(b^{k-2}) + a \times b^{(k-1)c} + b^{kc} & \text{arithmetic} \\
 & & \\
= & a^2 \times \left(T(\frac{b^{k-2}}{b}) + b^{(k-2)c}\right) + a \times b^{(k-1)c} + b^{kc} & \text{definition} \\
= & a^3 \times T(b^{k-3}) + a^2 \times b^{(k-2)c} + a \times b^{(k-1)c} + b^{kc} & \text{arithmetic} \\
 & & \\
= & a^3 \times \left(T(\frac{b^{k-3}}{b}) + b^{(k-3)c}\right) + a^2 \times b^{(k-2)c} + a \times b^{(k-1)c} + b^{kc} & \text{definition} \\
= & a^4 \times T(b^{k-4}) + a^3 \times b^{(k-3)c} + a^2 \times b^{(k-2)c} + a \times b^{(k-1)c} + b^{kc} & \text{arithmetic} \\
 & & \\
= & a^5 \times T(b^{k-5}) + a^4 \times b^{(k-4)c} + a^3 \times b^{(k-3)c} & \\
 & \quad + a^2 \times b^{(k-2)c} + a \times b^{(k-1)c} + b^{kc} & \text{guessing} \\
 & & \\
= & \ldots\ldots & \\
 & & \\
= & a^k \times T(b^{k-k}) + \sum_{i=0}^{k-1} \left(a^i \times b^{(k-i)c}\right) & \text{guessing} \\
 & & \\
= & a^k \times T(1) + \left(\sum_{i=0}^{k-1} \left(\frac{a}{b^c}\right)^i\right) b^{kc} & \text{arithmetic} \\
= & a^k \times T(1) + \begin{cases} b^{kc}\dfrac{1-(\frac{a}{b^c})^k}{1-\frac{a}{b^c}} & \text{if } a \neq b^c \\ b^{kc} \times k & \text{if } a = b^c \end{cases} & \text{arithmetic}
\end{array}
$$

Replace k by $\log_b n$, we have

$$
T(n) = a^{\log_b n} \times T(1) + \begin{cases} n^c \dfrac{1-(\frac{a}{b^c})^{\log_b n}}{1-\frac{a}{b^c}} & \text{if } a \neq b^c \\ n^c \times \log_b n & \text{if } a = b^c \end{cases}
$$

Iterated substitution for $T(n)$:

$$
\begin{array}{lll}
 & T(b^k) & \\
= & a \times T(\frac{b^k}{b}) + b^{kc} & \text{definition} \\
= & a \times T(b^{k-1}) + b^{kc} & \text{arithmetic} \\
= & a \times \left(a \times T(\frac{b^{k-1}}{b}) + b^{(k-1)c}\right) + b^{kc} & \text{definition} \\
= & a^2 \times T(b^{k-2}) + a \times b^{(k-1)c} + b^{kc} & \text{arithmetic} \\
= & a^2 \times \left(T(\frac{b^{k-2}}{b}) + b^{(k-2)c}\right) + a \times b^{(k-1)c} + b^{kc} & \text{definition} \\
= & a^3 \times T(b^{k-3}) + a^2 \times b^{(k-2)c} + a \times b^{(k-1)c} + b^{kc} & \text{arithmetic} \\
= & a^3 \times \left(T(\frac{b^{k-3}}{b}) + b^{(k-3)c}\right) + a^2 \times b^{(k-2)c} + a \times b^{(k-1)c} + b^{kc} & \text{definition} \\
= & a^4 \times T(b^{k-4}) + a^3 \times b^{(k-3)c} + a^2 \times b^{(k-2)c} + a \times b^{(k-1)c} + b^{kc} & \text{arithmetic} \\
= & a^5 \times T(b^{k-5}) + a^4 \times b^{(k-4)c} + a^3 \times b^{(k-3)c} & \\
 & \quad + a^2 \times b^{(k-2)c} + a \times b^{(k-1)c} + b^{kc} & \text{guessing} \\
= & \ldots\ldots & \\
= & a^k \times T(b^{k-k}) + \sum_{i=0}^{k-1} \left(a^i \times b^{(k-i)c}\right) & \text{guessing} \\
= & a^k \times T(1) + \left(\sum_{i=0}^{k-1} \left(\frac{a}{b^c}\right)^i\right) b^{kc} & \text{arithmetic} \\
= & a^k \times T(1) + \begin{cases} b^{kc}\dfrac{1-(\frac{a}{b^c})^k}{1-\frac{a}{b^c}} & \text{if } a \neq b^c \\ b^{kc} \times k & \text{if } a = b^c \end{cases} & \text{arithmetic}
\end{array}
$$

Replace k by $\log_b n$, we have

$$
T(n) = n^{\log_b a} \times T(1) + \begin{cases} n^c \dfrac{1-(\frac{a}{b^c})^{\log_b n}}{1-\frac{a}{b^c}} & \text{if } a \neq b^c \\ n^c \times \log_b n & \text{if } a = b^c \end{cases}
$$

Growth of function $T(n)$:

$$T(n) = n^{\log_b a} \times T(1) + \begin{cases} n^c \dfrac{1 - (\frac{a}{b^c})^{\log_b n}}{1 - \frac{a}{b^c}} & \text{if } a \neq b^c \\ n^c \times \log_b n & \text{if } a = b^c \end{cases}$$

- If $a = b^c$ then $\log_b a = c$ and
$$T(n) = T(1) n^{\log_b a} + n^c \log_b n \in \Theta(n^c \log n).$$

- If $a < b^c$ then $\log_b a < c$ and
$$T(n) = T(1) n^{\log_b a} + n^c \frac{1 - (\frac{a}{b^c})^{\log_b n}}{1 - \frac{a}{b^c}} = T(1) n^{\log_b a} + \frac{n^c - n^{\log_b a}}{1 - \frac{a}{b^c}} \in \Theta(n^c).$$

- If $a > b^c$ then $\log_b a > c$ and
$$T(n) = T(1) n^{\log_b a} + n^c \frac{(\frac{a}{b^c})^{\log_b n} - 1}{\frac{a}{b^c} - 1} = T(1) n^{\log_b a} + \frac{n^{\log_b a} - n^c}{\frac{a}{b^c} - 1} \in \Theta(n^{\log_b a}).$$

Master Theorem (simple version):

Let $a \geq 1$ and $b > 1$ be constants, let $f(n) \in \Theta(n^c)$ for some positive c, and let $T(n)$ be defined by the recurrence
$$T(n) = aT(\frac{n}{b}) + f(n).$$
Then $T(n)$ can be bounded asymptotically as follows:

1. if $c = \log_b a$, then $T(n) \in \Theta(n^c \log n)$,
2. if $c > \log_b a$, then $T(n) \in \Theta(n^c)$,
3. if $c < \log_b a$, then $T(n) \in \Theta(n^{\log_b a})$,

Algorithm design consequences:

- $c \stackrel{?}{=} \log_b a$ — $T(n) \uparrow$ if $\log_b a \uparrow$ or $c \uparrow$
  Redesign: no

- $c \stackrel{?}{>} \log_b a$ — $T(n) \downarrow$ if $\log_b a \uparrow$ or $c \downarrow$
  Redesign: more/bigger calls, less body

- $c \stackrel{?}{<} \log_b a$ — $T(n) \downarrow$ if $\log_b a \downarrow$ or $c \uparrow$
  Redesign: fewer/less calls, more body

Some examples:

1. $T(n) = \begin{cases} 2T(\frac{n}{2}) + n & \text{if } n \geq 2 \\ T(1) & \text{if } n = 1 \end{cases}$

2. $T(n) = \begin{cases} 4T(\frac{n}{2}) + n^3 & \text{if } n \geq 2 \\ T(1) & \text{if } n = 1 \end{cases}$

3. $T(n) = \begin{cases} 7T(\frac{n}{2}) + n^2 & \text{if } n \geq 2 \\ T(1) & \text{if } n = 1 \end{cases}$

4. $T(n) = \begin{cases} 7T(\frac{n}{2}) + \dfrac{n^2}{\lg n} & \text{if } n \geq 2 \\ T(1) & \text{if } n = 1 \end{cases}$

Some examples:

1. $T(n) = \begin{cases} 2T(\frac{n}{2}) + n & \text{if } n \geq 2 \\ T(1) & \text{if } n = 1 \end{cases}$

   $a = 2, b = 2, c = 1 \Longrightarrow c = \log_b a$, so $T(n) \in \Theta(n \log n)$

2. $T(n) = \begin{cases} 4T(\frac{n}{2}) + n^3 & \text{if } n \geq 2 \\ T(1) & \text{if } n = 1 \end{cases}$

   $a = 4, b = 2, c = 3 \Longrightarrow c > \log_b a = 2$, so $T(n) \in \Theta(n^3)$

3. $T(n) = \begin{cases} 7T(\frac{n}{2}) + n^2 & \text{if } n \geq 2 \\ T(1) & \text{if } n = 1 \end{cases}$

   $a = 7, b = 2, c = 2 \Longrightarrow c < \log_b a$, so $T(n) \in \Theta(n^{\log_2 7})$

4. $T(n) = \begin{cases} 7T(\frac{n}{2}) + \dfrac{n^2}{\lg n} & \text{if } n \geq 2 \\ T(1) & \text{if } n = 1 \end{cases}$

   $a = 7, b = 2, c = ??? \Longrightarrow$, so simple Master Theorem does NOT apply

Exercise:

In the last example, show that $T(n) \in O(n^{\log_2 7})$.

Have you understood the lecture contents?

well	ok	not-at-all	topic
☐	☐	☐	iterated substitution method
☐	☐	☐	closed form guessing
☐	☐	☐	prove by math induction
☐	☐	☐	logarithm (base substitution)
☐	☐	☐	Master Theorem: simple version
☐	☐	☐	applying Master Theorem