Recurrence

Dr. Son P. Nguyen

UEL
VNU-HCMC

October 4, 2015

Example

- The Tower of Hanoi. How do you solve this using recursion ?
- Equation gotten:

$$M(n) = 2M(n-1) + 1$$

- This is an example of a *recurrence equation*
- Solving the equation, we'll know how much time needed to finish the game.

Example

- The Tower of Hanoi. How do you solve this using recursion ?
- Equation gotten:

$$M(n) = 2M(n-1) + 1$$

- This is an example of a *recurrence equation*
- Solving the equation, we'll know how much time needed to finish the game.
- The easiest way is to list the first few $M(n)$ and make a guess about a general formula.
- Then try to prove the formula by *induction*

Example

- The Tower of Hanoi. How do you solve this using recursion ?
- Equation gotten:

$$M(n) = 2M(n-1) + 1$$

- This is an example of a *recurrence equation*
- Solving the equation, we'll know how much time needed to finish the game.
- The easiest way is to list the first few $M(n)$ and make a guess about a general formula.
- Then try to prove the formula by *induction*

Examples

Example 1

The empty set $\emptyset$ is a set with no elements. How many subsets does it have ? How many subsets does the one-element set $\{1\}$ have? How many subsets does the two-element set $\{1, 2\}$ have? How many of these subsets contain 2 ? How many subsets does $\{1, 2, 3\}$ have ? How many contain 3 ? Give a recurrence for the number $S(n)$ of subsets of an n-element set, and prove that your recurrence is correct.

Example 2

When paying off a loan with initial amount A and monthly payment M at an interest rate of p percent, the total amount $T(n)$ of the loan after n months is computed by adding $p/12$ percent to the amount due after $n-1$ months and then subtracting the monthly payment M. Convert this description into a recurrence for the amount owed after n months.

Examples

Example 1

The empty set $\emptyset$ is a set with no elements. How many subsets does it have ? How many subsets does the one-element set $\{1\}$ have? How many subsets does the two-element set $\{1, 2\}$ have? How many of these subsets contain 2 ? How many subsets does $\{1, 2, 3\}$ have ? How many contain 3 ? Give a recurrence for the number $S(n)$ of subsets of an n-element set, and prove that your recurrence is correct.

Example 2

When paying off a loan with initial amount A and monthly payment M at an interest rate of p percent, the total amount $T(n)$ of the loan after n months is computed by adding $p/12$ percent to the amount due after $n - 1$ months and then subtracting the monthly payment M. Convert this description into a recurrence for the amount owed after n months.

Example (cont.)

Example 3

Given the recurrence

$$T(n) = rT(n-1) + a$$

where r and a are constants, find a recurrence that expresses $T(n)$ in terms of $T(n-2)$ instead of $T(n-1)$. Now find a recurrence that expresses $T(n)$ in terms of $T(n-3)$ instead of $T(n-2)$ or $T(n-1)$. Now find a recurrence that expresses $T(n)$ in terms of $T(n-4)$ rather than $T(n-1)$, $T(n-2)$, or $T(n-3)$. Based on your work so far, find a general formula for the solution to the recurrence

$$T(n) = rT(n-1) + a$$

with $T(0) = b$ and where r and a are constants.

First theorem on recurrence

Theorem

If $T(n) = rT(n-1) + a$, $T(0) = b$, *and* $r \neq 1$, *then*

$$T(n) = r^n b + a \cdot \frac{1 - r^n}{1 - r}$$

for all nonnegative integers n

Second theorem on recurrence

A generalization of the first theorem when a is no longer a constant.

Theorem

For any positive constants b and r and any function g defined on the nonnegative integers, the solution to the first-order linear recurrence

$$T(n) = \begin{cases} rT(n-1) + g(n) & \text{if} \quad n > 0 \\ b & \text{if} \quad n = 0 \end{cases}$$

is

$$T(n) = r^n b + \sum_{i=1}^{n} r^{n-i} g(i)$$

Some remarks

- To bound $T(n)$, it boils down to whether $r^n b$ dominates or $\sum_{i=1}^{n} r^{n-i} g(i)$ dominates
- The second sums in a few examples are easy enough for a kind-of geometric series analysis

Example

Solve the recurrence $T(n) = 4T(n-1) + 2n$, with $T(0) = 6$.

Some remarks

- To bound $T(n)$, it boils down to whether $r^n b$ dominates or $\sum_{i=1}^{n} r^{n-i} g(i)$ dominates
- The second sums in a few examples are easy enough for a kind-of geometric series analysis

Example

Solve the recurrence $T(n) = 4T(n-1) + 2n$, with $T(0) = 6$.

Example

Solve the recurrence $T(n) = 3T(n-1) + n$, with $T(0) = 10$.

Some remarks

- To bound $T(n)$, it boils down to whether $r^n b$ dominates or $\sum_{i=1}^{n} r^{n-i} g(i)$ dominates
- The second sums in a few examples are easy enough for a kind-of geometric series analysis

Example

Solve the recurrence $T(n) = 4T(n-1) + 2n$, with $T(0) = 6$.

Example

Solve the recurrence $T(n) = 3T(n-1) + n$, with $T(0) = 10$.

Exercises

Do exercises 7,9,11 p. 197

Asymptotic notations

Let $f(n)$ be the running time of an algorithm where n is the input size. We always have the need to bound $f(n)$.

Example

Say $f(n) = 7n^{3/5} + x^{1/5}\ln^3 n$. We can "bound" f by a simpler function $g(n) = n^{3/5}$

Standard form

A function $g(n)$ is said to be in *standard form* if it is the product of terms of the following types:

1. Constants such as $\sqrt{2\pi}$, 6, e^{-2}.
2. Constant powers of n such as n, $\sqrt{n}$, $n^{5/2}$, n^{-3}.
3. Constant powers of $\ln n$ such as $\ln n$, $\sqrt{\ln n}$, $\frac{1}{\ln n}$.
4. Exponentials such as 2^n, e^n, $2^{n/2}$.
5. n^{cn} for constant c, such as n^n.

Asymptotic notations

Let $f(n)$ be the running time of an algorithm where n is the input size. We always have the need to bound $f(n)$.

Example

Say $f(n) = 7n^{3/5} + x^{1/5} \ln^3 n$. We can "bound" f by a simpler function $g(n) = n^{3/5}$

Standard form

A function $g(n)$ is said to be in *standard form* if it is the product of terms of the following types:

1. Constants such as $\sqrt{2\pi}$, 6, e^{-2}.
2. Constant powers of n such as n, $\sqrt{n}$, $n^{5/2}$, n^{-3}.
3. Constant powers of $\ln n$ such as $\ln n$, $\sqrt{\ln n}$, $\frac{1}{\ln n}$.
4. Exponentials such as 2^n, e^n, $2^{n/2}$.
5. n^{cn} for constant c, such as n^n.

Asymptotic notations (cont.)

Example of standard form

Stirling's formula

$$n! \sim n^n e^{-n}\sqrt{2\pi n}$$

- In our applications we imagine $f(n)$ as a complicated function and $g(n)$ in standard form.

Definition

We write $f(n) \sim g(n)$ and say $f(n)$ *is asymptotic to* $g(n)$ when

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 1$$

Asymptotic notations (cont.)

Example of standard form

Stirling's formula

$$n! \sim n^n e^{-n} \sqrt{2\pi n}$$

- In our applications we imagine $f(n)$ as a complicated function and $g(n)$ in standard form.

Definition

We write $f(n) \sim g(n)$ and say $f(n)$ *is asymptotic to* $g(n)$ when

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 1$$

Asymptotic notations (cont.)

Big O

We write $f(n) = O(g(n))$ and say $f(n)$ is *big oh* of $g(n)$ when there is a positive constant C such that for all sufficiently large n

$$f(n) \leq Cg(n)$$

i.e. the graph of g will eventually climb above the graph of f

Big Omega

We write $f(n) = \Omega(g(n))$ and say $f(n)$ is *omega* of $g(n)$ when there is a positive constant ϵ such that for all sufficiently large n,

$$f(n) \geq \epsilon g(n)$$

Asymptotic notations (cont.)

Big O

We write $f(n) = O(g(n))$ and say $f(n)$ is *big oh* of $g(n)$ when there is a positive constant C such that for all sufficiently large n

$$f(n) \leq Cg(n)$$

i.e. the graph of g will eventually climb above the graph of f

Big Omega

We write $f(n) = \Omega(g(n))$ and say $f(n)$ is *omega* of $g(n)$ when there is a positive constant ϵ such that for all sufficiently large n,

$$f(n) \geq \epsilon g(n)$$

Asymptotic notations (cont.)

Big Theta

We write $f(n) = \Theta(g(n))$ and say $f(n)$ is *theta* of $g(n)$ when there exist positive constants C, ϵ so that for n sufficiently large

$$\epsilon g(n) \leq f(n) \leq Cg(n)$$

or equivalently, $f = O(g)$ *and* $g = O(f)$.

Polylog

A function $f(n)$ is said to be *polylog* if $f(n) = \Theta(\ln(cn))$ for some positive constant c.

Asymptotic notations (cont.)

Big Theta

We write $f(n) = \Theta(g(n))$ and say $f(n)$ is *theta* of $g(n)$ when there exist positive constants C, ϵ so that for n sufficiently large

$$\epsilon g(n) \leq f(n) \leq Cg(n)$$

or equivalently, $f = O(g)$ *and* $g = O(f)$.

Polylog

A function $f(n)$ is said to be *polylog* if $f(n) = \Theta(\ln(cn))$ for some positive constant c.

Asymptotic notation (cont.)

Theorem

Let $f(n) = f_1(n) + f_2(n)$. Suppose $f_1(n) = O(g(n))$ and $f_2(n) = O(g(n))$. Then $f(n) = O(g(n))$.

Little oh one

When $f(n) = o(1)$, it means that $f(n) \to 0$ as $n \to \infty$. Of particular use is the factor $1 + o(1)$. This is a term that approaches one. (Warning: The o(1) term may be negative here. If $h(n) = 1 + o(1)$, then with arbitrarily small positive ϵ, we must have $1 - \epsilon \leq h(n) \leq 1 + \epsilon$ for n sufficiently large.) Thus

$$f(n) = g(n)(1 + o(1)) \quad \text{if and only if} \quad f(n) \sim g(n)$$

Asymptotic notation (cont.)

Theorem

Let $f(n) = f_1(n) + f_2(n)$. Suppose $f_1(n) = O(g(n))$ and $f_2(n) = O(g(n))$. Then $f(n) = O(g(n))$.

Little oh one

When $f(n) = o(1)$, it means that $f(n) \to 0$ as $n \to \infty$. Of particular use is the factor $1 + o(1)$. This is a term that approaches one. (Warning: The o(1) term may be negative here. If $h(n) = 1 + o(1)$, then with arbitrarily small positive ϵ, we must have $1 - \epsilon \leq h(n) \leq 1 + \epsilon$ for n sufficiently large.) Thus

$$f(n) = g(n)(1 + o(1)) \quad \text{if and only if} \quad f(n) \sim g(n)$$

Asymptotic notation (cont.)

There is a natural *ordering*' of the basic types of functions:

1. constants,
2. constant positive powers of $\ln n$,
3. constant positive powers of n,
4. exponentials $c^n,\ c > 1$,
5. n^{cn} for constant positive c, such as n^n.

Each type below grows slower than the following ones.

Example

Let $T(n)$ be the number of questions in a binary search on the range of numbers between 1 and n. Assuming that n is a power of 2, give a recurrence for $T(n)$.

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

Here, we would like to analyze a popular method for analyzing running time of an algorithm, namely the divide and conquer method.

Example

Let $T(n)$ be the number of questions in a binary search on the range of numbers between 1 and n. Assuming that n is a power of 2, give a recurrence for $T(n)$.

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

Here, we would like to analyze a popular method for analyzing running time of an algorithm, namely the divide and conquer method.

Let's analyze *Merge Sort*

```
MergeSort(A,low,high)

// This algorithm sorts the portion of list A from
// location low to location high.
if (low == high)
     return
else
     mid = ⌊(low + high)/2⌋
     MergeSort(A,low,mid)
     MergeSort(A,mid+1,high)
     Merge the sorted lists from the previous two steps
     return
```

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

Let's analyze *Merge Sort*

```
MergeSort(A,low,high)

// This algorithm sorts the portion of list A from
// location low to location high.
if (low == high)
     return
else
     mid = ⌊(low + high)/2⌋
     MergeSort(A,low,mid)
     MergeSort(A,mid+1,high)
     Merge the sorted lists from the previous two steps
     return
```

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

Recursion trees

Number of problems	Problem size	Work per problem	Work per level
1	n	n	n
2	$n/2$	$n/2$	$n/2 + n/2 = n$
$2^2 = 4$	$n/4$	$n/4$	$n/4 + n/4 + n/4 + n/4 = n$
$2^3 = 8$	$n/8$	$n/8$	$8(n/8) = n$

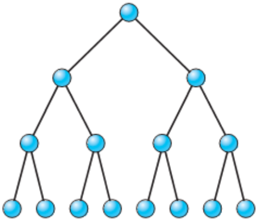

Figure 4.5: Four levels of a recursion tree diagram

We need to determine four things

- the number of subproblems
- the size of each subproblem
- the amount of work done per subproblem
- the total work done at that level

Example

Use a recursion tree to find a big Θ bound for the solution to the recurrence

$$T(n) = \begin{cases} 3T(n/3) + n & \text{if} \quad n \geq 3 \\ 1 & \text{if} \quad n < 3 \end{cases}$$

assuming n is a power of 3

Example

Use a recursion tree to solve the recurrence

$$T(n) = \begin{cases} 4T(n/2) + n & \text{if} \quad n \geq 2 \\ 1 & \text{if} \quad n = 1 \end{cases}$$

Assume that n is a power of 2. Convert your solution to a big Θ statement about the behavior of the solution

Recursion trees

Example

Use a recursion tree to find a big Θ bound for the solution to the recurrence

$$T(n) = \begin{cases} 3T(n/3) + n & \text{if} \quad n \geq 3 \\ 1 & \text{if} \quad n < 3 \end{cases}$$

assuming n is a power of 3

Example

Use a recursion tree to solve the recurrence

$$T(n) = \begin{cases} 4T(n/2) + n & \text{if} \quad n \geq 2 \\ 1 & \text{if} \quad n = 1 \end{cases}$$

Assume that n is a power of 2. Convert your solution to a big Θ statement about the behavior of the solution

Recursion trees

Lemma

Suppose that we have a recurrence of the form

$$aT\left(\frac{n}{2}\right) + n$$

where a is a positive integer and $T(1)$ is nonnegative. Then we have the following big Θ bounds on the solution:

1. If $a < 2$, then $T(n) = \Theta(n)$.
2. If $a = 2$, then $T(n) = \Theta(n \log n)$.
3. If $a > 2$, then $T(n) = \Theta(n^{\log_2 a})$.

Exercises

Do 1,4,8,9 on p. 212

The master theorem

Theorem

(Master Theorem) *Let a and b be positive real numbers, with $a \geq 1$ and $b > 1$. Let $T(n)$ be defined for integers n that are powers of b by*

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

Then we have the following:

1. *If* $f(n) = \Theta(n^c)$ *where* $\log_b a < c$ *then*

$$T(n) = \Theta(n^c) = \Theta(f(n))$$

2. *If* $f(n) = \Theta(n^c)$ *where* $\log_b a = c$ *then*

$$T(n) = \Theta(n^c \log n) = \Theta(f(n) \log n)$$

The master theorem (cont.)

3. If $f(n) = \Theta(n^c)$ where $\log_b a > c$ then

$$T(n) = \Theta(n^{\log_b a})$$

Example

What can we say about the big Θ behavior of the solution to

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

where n can be any nonnegative power of 3 ?

3. If $f(n) = \Theta(n^c)$ where $\log_b a > c$ then

$$T(n) = \Theta(n^{\log_b a})$$

Example

What can we say about the big Θ behavior of the solution to

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

where n can be any nonnegative power of 3 ?

The master theorem (cont.)

Example

If $f(n) = n\sqrt{n+1}$, what can we say about the big Θ behavior of solutions to

$$S(n) = \begin{cases} 2S(n/3) + f(n) & \text{if} \quad n \geq 2 \\ d & \text{if} \quad n = 1 \end{cases}$$

where n can be any nonnegative power of 3 ?

Example

What does the master theorem tell us about the solutions to the recurrence

$$S(n) = \begin{cases} 3S(n/2) + n\sqrt{n+1} & \text{if} \quad n \geq 2 \\ 1 & \text{if} \quad n = 1 \end{cases}$$

The master theorem (cont.)

Example

If $f(n) = n\sqrt{n+1}$, what can we say about the big Θ behavior of solutions to

$$S(n) = \begin{cases} 2S(n/3) + f(n) & \text{if} \quad n \geq 2 \\ d & \text{if} \quad n = 1 \end{cases}$$

where n can be any nonnegative power of 3 ?

Example

What does the master theorem tell us about the solutions to the recurrence

$$S(n) = \begin{cases} 3S(n/2) + n\sqrt{n+1} & \text{if} \quad n \geq 2 \\ 1 & \text{if} \quad n = 1 \end{cases}$$

Exercises

Do problem 1,2,4 on page 222