3.24pt

Design and Analysis of Algorithms CSE2012

Dr. Ramesh Ragala

February 11, 2022

Introduction to Recurrence Relations

Recurrence Relation

Definition:

A Recurrence Relation for a sequence $\{ a_n \}$ is an equation that express a_n in terms of one or more of the previous terms in the sequence, a_1,a_2,a_3, ... a_{n-1} for all integers $n \geq n_0$ where n_0 is a non-negative integer.
A sequence is called a solution of a recurrence relation if its terms satisfy the recurrence relation.

- Motivating Examples:
    - Finding the Factorial of a given number
    - Fibonacci series
    - Towers of Hanoi
    - some of the problems solved by DAC approach

- Simple Example
  - Assume we have a set of integers as like {1,2,4,8,16,32,..}
  - What will be the next integer in above set

- Simple Example
  - Assume we have a set of integers as like $\{1,2,4,8,16,32,..\}$
  - What will be the next integer in above set
  - 64

- Simple Example
  - Assume we have a set of integers as like $\{1,2,4,8,16,32,..\}$
  - What will be the next integer in above set
  - 64
  - How did you find the answer

- Simple Example
  - Assume we have a set of integers as like {1,2,4,8,16,32,..}
  - What will be the next integer in above set
  - 64
  - How did you find the answer
  - Is there any procedure

- Simple Example
  - Assume we have a set of integers as like {1,2,4,8,16,32,..}
  - What will be the next integer in above set
  - 64
  - How did you find the answer
  - Is there any procedure yes

Introduction to Recurrence Relations

- Simple Example
  - Assume we have a set of integers as like {1,2,4,8,16,32,..}
  - What will be the next integer in above set
  - 64
  - How did you find the answer
  - Is there any procedure yes
  - After giving the first term, each term of the sequence can be defined from the previous term.
  - $a_1 = 1$

- Simple Example
  - Assume we have a set of integers as like {1,2,4,8,16,32,..}
  - What will be the next integer in above set
  - 64
  - How did you find the answer
  - Is there any procedure yes
  - After giving the first term, each term of the sequence can be defined from the previous term.
  - $a_1 = 1 \Rightarrow$

Introduction to Recurrence Relations

- Simple Example
  - Assume we have a set of integers as like $\{1,2,4,8,16,32,..\}$
  - What will be the next integer in above set
  - 64
  - How did you find the answer
  - Is there any procedure yes
  - After giving the first term, each term of the sequence can be defined from the previous term.
  - $a_1 = 1 \Rightarrow a_{n-1} = 2a_n$
- When a sequence is defined recursively, mathematical induction can be used to prove results about the sequence.

- A recurrence relation is like a recursively defined sequence, but without specifying any initial values (initial conditions).
- Therefore, the same recurrence relation can have (and usually has) multiple solutions.
- If both the initial conditions and the recurrence relation are specified, then the sequence is uniquely determined
- more generally, recurrences are have the form
- T(n) = α T(n-β) + f(n), T(δ) = c or
- T(n) = α T(n/β) + f(n), T(δ) = c is initial condition
- **Example on Recurrence Relation:**
  - Consider the Recurrence Relation: $a_n = 2a_{n-1} - a_{n-2}$ for n= 2,3,4..
  - Is the sequence { a_n } with a_n=3n a solution of this recurrence relation?

- A recurrence relation is like a recursively defined sequence, but without specifying any initial values (initial conditions).
- Therefore, the same recurrence relation can have (and usually has) multiple solutions.
- If both the initial conditions and the recurrence relation are specified, then the sequence is uniquely determined
- more generally, recurrences are have the form
- T(n) $= \alpha$ T(n-β) + f(n), T(δ) = c or
- T(n) $= \alpha$ T(n/β) + f(n), T(δ) = c is initial condition
- **Example on Recurrence Relation:**
  - Consider the Recurrence Relation: $a_n = 2a_{n-1} - a_{n-2}$ for n= 2,3,4..
  - Is the sequence { a_n } with a_n=3n a solution of this recurrence relation? yes

- A recurrence relation is like a recursively defined sequence, but without specifying any initial values (initial conditions).
- Therefore, the same recurrence relation can have (and usually has) multiple solutions.
- If both the initial conditions and the recurrence relation are specified, then the sequence is uniquely determined
- more generally, recurrences are have the form
- T(n) = α T(n-β) + f(n), T(δ) = c or
- T(n) = α T(n/β) + f(n), T(δ) = c is initial condition
- **Example on Recurrence Relation:**
  - Consider the Recurrence Relation: $a_n = 2a_{n-1}$ - a_{n-2} for n= 2,3,4..
  - Is the sequence { a_n } with a_n=3n a solution of this recurrence relation? yes a_n=n+1.
  - So Initial Conditions + Recurrence Relation $\Rightarrow$ Uniquely determines the sequence.

- Fibonacci Series:
- $\{ f_n \} = \{ 0,1,1,2,3,5,8,13,21, \}$.
- Recursive Definition for Fibonacci Series is:
  - INITIALIZE: $f_0 = 0$, $f_1 = 1$
  - RECURSIVE: $f_n = f_{n-1} + f_{n-2}$ for $n > 1$
- The recurrence relation is the recursive part of the above Definition.
- Simply, The recurrence relation for a sequence consists of an equation that expresses each term in terms of lower terms.
- Is there another solution to the Fibonacci recurrence relation?

Introduction to Recurrence Relations

- Fibonacci Series:
- $\{ f_n \} = \{ 0,1,1,2,3,5,8,13,21, \}$.
- Recursive Definition for Fibonacci Series is:
  - INITIALIZE: $f_0 = 0$, $f_1 = 1$
  - RECURSIVE: $f_n = f_{n-1} + f_{n-2}$ for $n > 1$
- The recurrence relation is the recursive part of the above Definition.
- Simply, The recurrence relation for a sequence consists of an equation that expresses each term in terms of lower terms.
- Is there another solution to the Fibonacci recurrence relation?
- Yes. we can have different set of initial conditions, $f_0 = f_1 = 1$
- In this case, what will be the sequence????

Recurrence Relation for Counting Problem

- It is very difficult to get a closed formula for counting particular set.
- It is easy to get a recurrence relation for counting particular set.
- Examples:
  - Geometric example: counting the number of points of intersection of n lines.
  - A recurrence relation for the number of bit strings of length n which contain the string 00.
  - Partition Function
  - Financial Recurrence Relation

PROBLEM

RAMU deposited Rs.10,000/- in a savings account at a bank yielding 5% per year with interest compounded annually. How much money will be in the account after 30 years?

- Let P_n denote the amount in the account after n years.
- How can we determine P_n on the basis of P_{n-1}?
- We can derive the following recurrence relation:
- $P_n = P_{n-1} + 0.05P_{n-1} = 1.05P_{n-1}$.
- The initial condition is $P_0 = 10{,}000$.
- $P_1 = 1.05P_0$
- $P_2 = 1.05P_1 = (1.05)^2P_0$
- ...
- $P_n = 1.05P_{n-1} = (1.05)^nP_0 \Rightarrow$ no iteration. Just Formula $\Rightarrow$ Recurrence Relation

Recurrence Relation Examples on ... of String

PROBLEM

Let a_n denote the number of bit strings of length n that do not have two consecutive 0s (valid strings). Find a recurrence relation and give initial conditions for the sequence $\{ a_n \}$.

- The number of valid strings equals the number of valid strings ending with a 0 plus the number of valid strings ending with a 1.
- Let us assume that $n \geq 3$, so that the string contains at least 3 bits.
- Let us further assume that we know the number a_{n-1} of valid strings of length (n -1).
- Then how many valid strings of length n are there, if the string ends with a 1?

Recurrence Relation Examples on of String

- There are a $_{n-1}$ such strings, namely the set of valid strings of length (n-1) with a 1 appended to them.
- Now we need to know: How many valid strings of length n are there, if the string ends with a 0?
- Valid strings of length n ending with a 0 must have a 1 as their (n-1)st bit (otherwise they would end with 00 and would not be valid).
- And what is the number of valid strings of length (n - 1) that end with a 1?
- We already know that there are a n-1 strings of length n that end with a 1.
- Therefore, there are a$_{n-2}$ strings of length (n - 1) that end with a 1.

Recurrence Relation Examples on ... of String

- So there are a_{n-2} valid strings of length n that end with a 0 (all valid strings of length (n -2) with 10 appended to them).
- As we said before, the number of valid strings is the number of valid strings ending with a 0 plus the number of valid strings ending with a 1.
- That gives us the following recurrence relation:
  $\mathbf{a_n = a_{n-1} + a_{n-2}}$
- The Initial Conditions are:
- a_1=2 (0 and 1), $a_2 = 3$ (01, 10 and 11), $a_3 = a_2 + a_1 = 3 + 2 = 5$, $a_4 = a_3 + a_2 = 5 + 3 = 8$
- This sequence satisfies the same recurrence relation as the Fibonacci sequence. Since $a_1 = f_3$ and $a_2 = f_4$, we have $a_n = f_{n+2}$.

- There are many methods to solve the recurrence relations
  - Characteristic Equations
  - Forward and Backward Substitution
  - Master Theorem
  - Recurrence Trees

- "Making a good guess" method.
- Guess the form of the answer, then use induction to find the constants and show that solution works.
- The substitution method can be used to establish either upper or lower bounds on a recurrence.
- This method can be applied only $\Rightarrow$ when it is easy to guess the form of the answer.
- Example: $\mathrm{T(n) = 2T(\lfloor n/2 \rfloor) + n}$
- The guess solution is $\mathrm{T(n) = O(n \log n)}$.

$$
\begin{aligned}
T(n) \leq 2(c\lfloor n/2 \rfloor) log(\lfloor n/2 \rfloor)) + n \\
&\leq cnlog(n/2) + n \\
&= cnl0gn - cnlog2 + n \\
&= cnlogn - cn + n \\
&\leq cnlogn, \quad c \geq 1.
\end{aligned}
$$

Substitution Method

- Mathematical induction now requires us to show that our solution holds for the boundary conditions.
- we do so by showing that the boundary conditions are suitable as base cases for the inductive proof.
- Suppose , T(1)=1 is the sole boundary condition of the recurrence.
- Then the bound T(n) $\leq$ c n lg n yields T(1) $\leq$c log 1=0 Which is odds with T(1) = 1.
- The base case of our inductive proof fails to hold.
- To overcome this difficulty, we can take advantage of the asymptotic notation.
- we need to prove T(n)$\leq$c n lg n for n$\geq$ m.
- The idea is to remove the difficult boundary condition T(1)= 1 from consideration.

- Thus , we can replace T(1) by T(2) as the base cases in the inductive proof , letting m=2.
- From the recurrence , with $T(1) = 1$, we get $T(2)=4$.
- We require $T(2) \leq c\ 2 \lg 2$.
- It is clear that , any choice of $c \geq 2$ suffices for the base cases.
- But it will not work for n=3.
- choice of $c \geq 2$ is sufficient for this to hold.
- Finally $T(n) \leq c\ n \lg n$ for any $c \geq 2$ and $n \geq 2$.

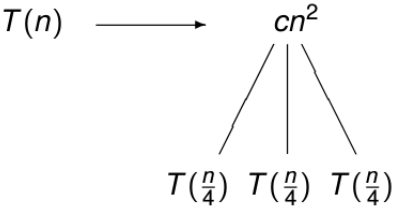
Disadvantages of Substitution Met

- There is no general way to guess the correct solution to the recursion.
- Guessing a solution takes experience and, occasionally, creativity.
- Some heuristics that can help you become a good guesser
- We can use recursion trees to generate good guess to recursion.
- Another way to make a good guess is to prove loose upper and lower bounds.
- Examples in class
- For some application, even we can (nearly)correctly guess at asymptotic bound $\Rightarrow$ Induction does not work properly.
- Proving exactness is missing $\Rightarrow$ Asymptotic Notations

Recursion Tree Method

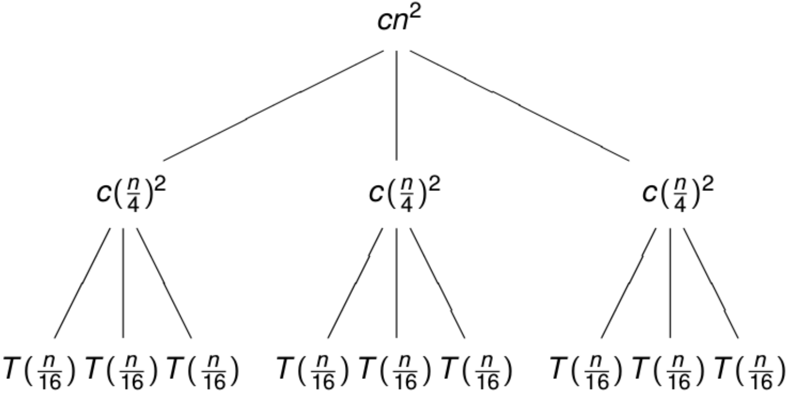
- Expanding the recurrence into a tree
- Summing the cost at each level
- Applying the substitution method

- Consider the Recurrence Relation: T(n) = 3T(n/2)+cn^2 for some constant c.
- Assumption "n" is an exact power of 4.
- In the recursion-tree method we expand T (n) into a tree:



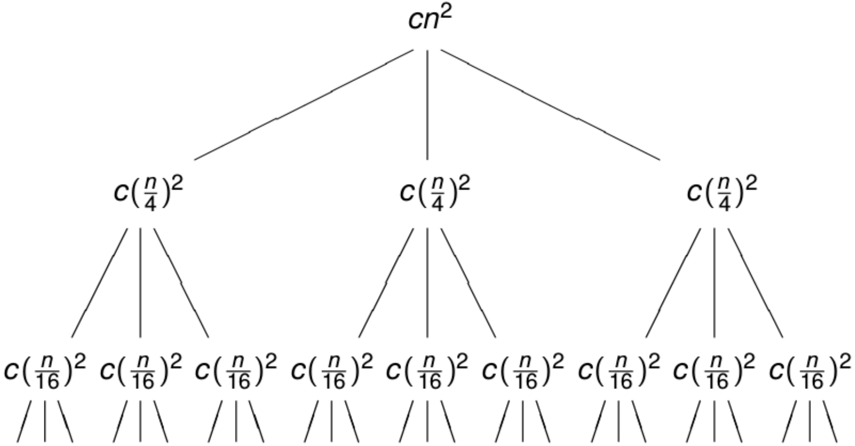
Recursion Tree Method

- Applying $T(n) = 3T(n/4) + cn^2$ to $T(n/4)$ leads to $T(n/4) = 3T(n/16) + c(n/4)^2$, expanding the leaves:



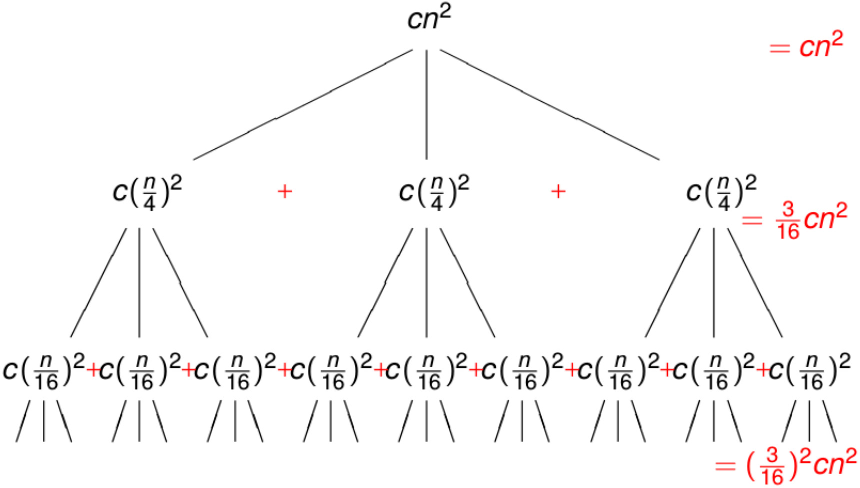
Recursion Tree Method

- Applying T(n)=3T(n/4)+cn^2 to T(n/16) leads to T(n/16)=3T(n/64)+c(n/16)^2, expanding the leaves:



Recursion Tree Method

- Summing the cost at each level.



- Adding up the cost

$$T(n) = cn^2 + (3/16)cn^2 + (3/16)^2 cn^2 + \ldots$$
$$= cn^2(1 + 3/16 + (3/16)^2 + \ldots)$$

- The above equation will be disappear when n = 16
- The tree has depth at least 2 if n $\geq 16 = 4^2$.
- For n $= 4^k$, k $= \log_4$(n), we have:

$$T(n) = cn^2 \sum_{i=0}^{\log_4(n)} \left(\frac{3}{16}\right)^i.$$

- Apply the Geometric Sum

Applying

$$S_n = \sum_{i=0}^{n} r^i = \frac{r^{n+1} - 1}{r - 1}$$

to

$$T(n) = cn^2 \sum_{i=0}^{\log_4(n)} \left(\frac{3}{16}\right)^i$$

with $r = \frac{3}{16}$ leads to

$$T(n) = cn^2 \frac{\left(\frac{3}{16}\right)^{\log_4(n)+1} - 1}{\frac{3}{16} - 1}.$$

Instead of $T(n) \leq dn^2$ for some constant d, we have

$$T(n) = cn^2 \frac{\left(\frac{3}{16}\right)^{\log_4(n)+1} - 1}{\frac{3}{16} - 1}.$$

Recall

$$T(n) = cn^2 \sum_{i=0}^{\log_4(n)} \left(\frac{3}{16}\right)^i.$$

To remove the $\log_4(n)$ factor, we consider

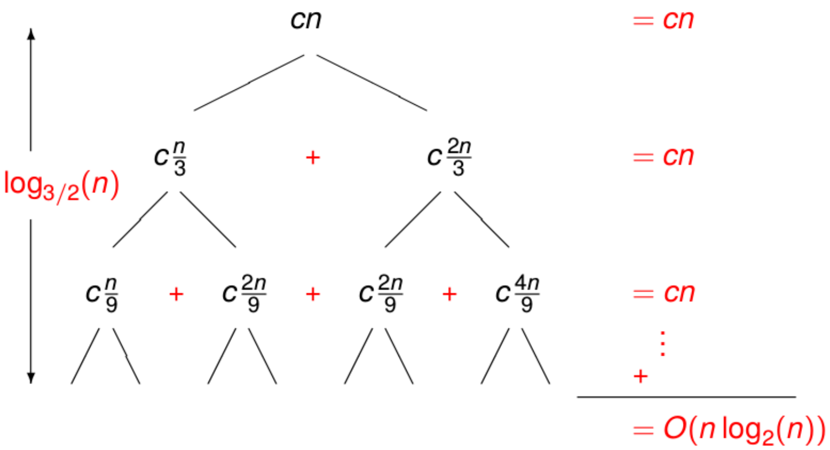
$$\begin{aligned} T(n) &\leq cn^2 \sum_{i=0}^{\infty} \left(\frac{3}{16}\right)^i \\ &= cn^2 \frac{-1}{\frac{3}{16} - 1} \leq dn^2, \text{ for some constant } d. \end{aligned}$$

Let us see if $T(n) \leq dn^2$ is good for $T(n) = 3T(n/4) + cn^2$.

Applying the substitution method:

$$
\begin{aligned}
T(n) &= 3T(n/4) + cn^2 \\
&\leq 3d\left(\frac{n}{4}\right)^2 + cn^2 \\
&= \left(\frac{3}{16}d + c\right)n^2 \\
&= \frac{3}{16}\left(d + \frac{16}{3}c\right)n^2 \\
&\leq \frac{3}{16}(2d)n^2, \quad \text{if } d \geq \frac{16}{3}c \\
&\leq dn^2
\end{aligned}
$$

Consider $T(n) = T(n/3) + T(2n/3) + cn$.

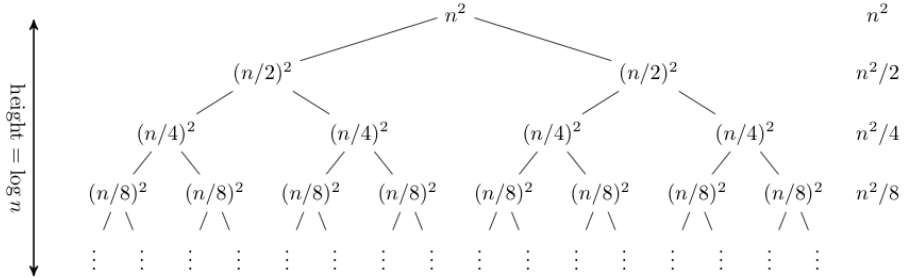


Master Method

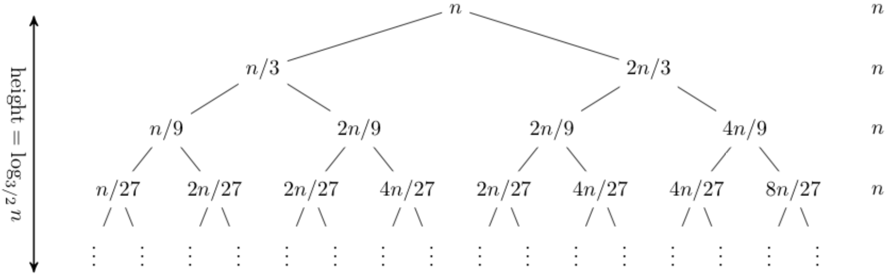
- It is mainly used to solve the recurrence relations of the form $T(n) = a\,T(n/b) + f(n)$ where $a \geq 1$, $b > 1$ and $f(n)$ is an asymptotically positive function.
- The above recurrence relation describes the running time of an algorithm that divides a problem of size n into a subproblems, each of size n/b, where a and b are positive constants.
- The a subproblems are solved recursively, each in time $T(n/b)$.
- The cost of dividing the problem and combining the results of the subproblems is described by the function $f(n)$.

Example on Recursion Tree Meth

- The Recurrence Relation is: $T(n) = 2T(n/2)+n^2$



- The Recurrence Relation(not balanced) is:
  T(n) = 2T(n/3)+T(2n/3)+n



Master Theorem Method

- The master method uses the following theorem
- Let a$\geq$1 and b$>$1 be constants, let f(n) be a function, and let T(n) be defined on the non-negative integers by the recurrence T(n) $=$ aT(n/b)+f(n),
  where we interpret n/b to mean either n/b or n/b .
- Then T(n) can be bounded asymptotically as follows.
- CASE-1: If f(n) $=$ O(n$^{log_b a-\epsilon}$) for some constant $\epsilon > 0$, then T(n) $=$ Θ(n$^{log_b a}$)
- CASE-2: If f(n) $=$ Θ(n$^{log_b a}$), then T(n) $=$ Θ(n$^{log_b a}$log n)
- CASE-3: If f(n) $=$ Ω(n$^{log_b a+\epsilon}$) for some constant ϵ $>$0, and if a f(n/b) $\leq$ cf(n) for some constant c$<$1 and all sufficiently large n, then T(n) $=$ $\Theta(f(n))$

- There is a gap between cases 1 and 2 when f(n) is smaller than n $^{log_b a}$ but not polynomially smaller.
- There is a gap between cases 2 and 3 when f(n) is larger than $n^{log_b a}$ but not polynomially larger.
- If the function f(n) falls into one of these gaps, or if the regularity condition in case 3 fails to hold, the master method cannot be used to solve the recurrence.
- **Example**
  - $T(n) = 9T(n/3)+n$.
  - Here a = 9, b=3, f(n) =n and $n^{log_b a} = n^{log_3 9} = \Theta(n^2)$.
  - Since $f(n) = O(n^{log_3 9 - \epsilon})$, where ϵ=1.
  - we can apply the CASE-1 of the master theorem.
  - So Solution is $T(n) = \Theta(n^2)$