Complexity Analysis of Recursive Algorithms: Substitution Method

Session No.: 8
Course Name: Design and analysis of algorithm
Course Code: R1UC407B
Instructor Name: Mili Dhar
Duration: 50 Min.
Date of Conduction of Class:

Review of the key concepts

1. Review of Master theorem

Q: In what Conditions master theorem does not work?

Learning Outcome

Solve recurrence relations using the **Substitution method**.

Session Outline

1 Complexity Analysis of Recursive Algorithms using Substitution method

2 Reflection learning activity

3 Conclusion and post-session activity

Recurrence Relation

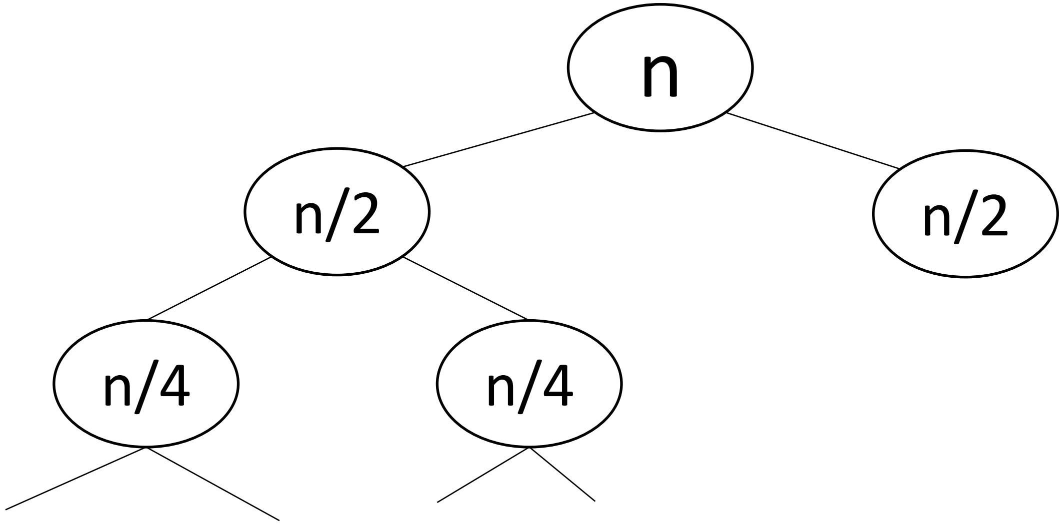
A **recurrence relation** is an equation that uses recursion to related terms in a sequence or elements in an array. It is a way to define a sequence or array in terms of itself.

It helps in finding the subsequent term (next term) dependent upon the preceding term (previous term). If we know the previous term in a given series, then we can easily determine the next term.

GALGOTIAS UNIVERSITY

Recurrence Relation: Binary search

```
BS (A, l, u, key)
{
        mid = (l+u)/2;
        if (A[mid] == key)
                return mid;
        else if (A[mid]<key)
                BS (A, mid+1, u, key)
        else
                BS (A, l, mid-1, key)
}
```



Total Time $T(n) = T(n/2) + c$

Method to solve the Recurrence Relation

1. **Substitution method**
2. Recursive tree method
3. Master Theorem method

Analysis using substitution method:

```
Fact(n)
Begin
        if (n>1) then
                return (n* Fact(n-1))
        endif
End
```

Recurrence relation:

$T(n) = 1$, when $n=1$

$=1+T(n-1)$ when $n>1$

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

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

$= 1+ T(n-2)$

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

$= 1+ T(n-3)$

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

$=2+T(n-2)$

$=2+1+T(n-3)$

$=3+T(n-3)$

$\ldots\ldots\ldots\ldots$

$=k+T(n-k)$

Here, $n-k =1$

$\Rightarrow k=n-1$

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

$=n-1+T(1)$

$=n-1+1$

$=n$

$T(n) = O(n)$

Recurrence relation:

$$T(n) = 1, \text{ when } n=1$$
$$= n + T(n-1) \text{ when } n>1$$

$$T(n-1)= n-1 + T((n-1)-1)$$
$$=n-1 +T(n-2)$$
$$T(n-2) = n-2+T(n-3)$$

$$T(n) = n+(n-1)+(n-2)+T(n-3)$$

After **k** levels of expansion

$$=n+(n-1)+(n-2)+(n-3)+\ldots\ldots.(n-(k-1)) +T(n-k)$$

Here, $n-k=1 \Rightarrow k=n-1$

$$T(n)= n+(n-1)+(n-2)+(n-3)+\ldots\ldots+T(1)$$
$$=n+(n-1)+(n-2)+(n-3)+\ldots\ldots+1$$
$$= [n(n+1)]/2$$

$T(n) = O(n^2)$

Solve the following recurrence relation:

T(n) =T(n-1) +3 for n>1

= 4 for n=1

→ Substituting the value of T(n-1) in the given equation, one can get the following equations:

T(n-1)=T(n-2)+3

T(n)=[T(n-2)+3]+3

= T(n-2)+2*3

=T(n-3)+3*3

……………………….

= T(n-k)+k*3

when k= n-1 , the resulting equation would be as follows:

$=T(n-(n-1))+(n-1)*3$

$=T(1)+(n-1)3$

$=4+3n-3$

$=3n+1$

$T(n)= O(n)$

$T(n) = T(n/2) + c$ if $n>1$
$= 1$ if $n = 1$

$T(n)=T(n/2)+C$

$T(n/2)=T(n/4)+C \rightarrow T(n/2^2)+C$

$T(n/4)=T(n/8)+C \rightarrow T(n/2^3)+C$

We assume,
$n= 2^k$
$k=\log_2 n$

$T(n) = T(n/2^3)+3C$

............After K level.................

$T(n) = T(n/2^3)+C$

$= T(n/2^k) + kc$

$= T(1) + (\log_2 n)c$ $=O(\log_2 n)$

$T(n) = n * T(n-1)$ if $n>1$

$= 1$ if $n = 1$

$T(n)=n\times T(n-1)$

$T(n-1)=(n-1)\times T(n-2)$

$T(n-2)=(n-2)\times T(n-3)$

The recurrence continues until n reduces to 1:

$T(n)=n\times(n-1)\times(n-2)\times\ldots\times 2\times 1$

$T(n)=n! = O(n^n)$

$T(n) = 2T(n/2) + n$ if $n>1$
$= 1$ if $n = 1$

$T(n)=2T(n/2)+n$
$T(n/2)=2T(n/4)+n/2$
$T(n/4)=2T(n/8)+n/4$

$T(n)=2^3\ T(n/2^3\ )+3n$
…………..After **k** levels of expansion…………

$T(n)=2^k\ T(n/2^k\ )+kn$
$T(n)=2^{\log 2n}\ T(1)+n\log_2 n = n + n\log_2 n = O(n\log_2 n)$

We assume,
$n = 2^k$
$k=\log_2 n$

$T(n) = T(n-1) + \log n$ if $n>1$

$= 1$ if $n = 1$

$T(n)=T(n-1)+\log n$

$T(n-1)=T(n-2)+\log(n-1)$

$T(n-2)=T(n-3)+\log(n-2)$

We assume,
n-1= k

$T(n)= T(n-3) + \log (n-2) + \log(n-1) + \log n$

……….After **k** levels of expansion…………….

$T(n) = T(n-k) + \log(n-(k-1)) + \log (n-(k-2)) +……+ \log n$

$= 1 + \log (2x3x4x5x………xn)$

$=1 + \log (n!)$

$=1 + \log (n^n) = 1+ n\log n = O(n\log n)$

Q. $T(n) = T(n-2)+n^2$ if $n>0$
$= 1$ if $n=0$

$T(n) = T(n-2) + n^2$
$= T(n-4) + (n-2)^2 + n^2$
$= T(n-6) + (n-4)^2 + (n-2)^2 + n^2$
....
$= T(n-2k) + (n-(2k-2))^2 + (n-(2k-4))^2 + \ldots\ldots\ldots + n^2$
$= 1 + 2^2 + 4^2 + \ldots\ldots\ldots\ldots + n^2$
$= 1 + (2^2.1^2 + 2^2.2^2 + 2^2.3^2 + \ldots\ldots + 2^2.k^2)$
$=1+ 2^2[1^2+2^2+3^2+\ldots.+k^2]$
$=1+2^2[k(k+1)(2k+1)]/6$
$=k^3 \quad = (n/2)^3 = O(n^3)$

We assume,
$n = 2k$
$k = n/2$

Q. $T(n) = 2T(n/2) + n\log n$ if $n>1$
$= 1$ if $n=1$

$T(n) = 2T(n/2)+ n\log n$

$= 2[2T(n/2^2)+ n/2 \log n/2] + n\log n$

$=2^2T(n/2^2)+ n \log n/2 +n\log n$

$=2^2[2T(n/2^3)+ n/2^2 \log n/2^2]+n \log n/2 +n\log n$

$=2^3T(n/2^3)+n\log n/2^2+n\log n/2+n\log n$

$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$

$= 2^k T(n/2^k)+n\log(n/2^{k-1})+\ldots\ldots\ldots\ldots+ n\log n/2^2 + n \log n/2 + n\log n$

$= 2^k T(n/2^k) + n [\log n/2^0 + \log n/2^1 + \log n/2^2 + \ldots\ldots\ldots\ldots + \log n/2^{k-1}$

$= n.1 + n [(\log n-0)+(\log n-1)+ \ldots\ldots+ (\log n- k-1)$

$= n+ n[k\log n –(1+2+3+\ldots..(k-1))$

$= n+n[(\log n)^2- \{k(k-1)/2\}]$

$= n+n [(\log n)^2 – (\log n)^2 - \log n/2 = O(n (\log n)^2)$

Post session activities

Q1. $T(n) = 5T(n/5) + (n/\log n)$ if $n>1$

$= 1$ if $n=1$

Q2. $T(n) = T(n-2) + 2\log n$ if $n>0$

$= 1$ if $n=0$

In the next session, Analysis of Recursive Algorithms using Recursive tree method will be discussed in detail.

Thank You