UNIT-1

Learning Objective

- Notion of Algorithm
- Growth of Functions
- Recurrences
- Asymptotic Notations (Big O , ?, ?)
- Basic Efficiency Classes
- Mathematical Analysis of Algorithms
- Proof of Correctness.

Notion of Algorithms

- An Algorithm is a finite set of instructions that, it followed, accomplishes a particular task.

In addition, all Algorithms must satisfy the following criteria.

1 Input

2 Output

3 Definiteness

4 Finiteness

5 Effectiveness.

Growth of Functions

- **Growth of Function**

To Study how the running time of an algorithm increases with the size of inputs

- **Asymptotic Notation**

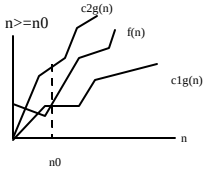
The Notations used to describe the running time of an algorithm.

Growth of Functions Cont...

1)OH ((-)Notation):
These Notation is used for finding from upper and lower bound of an algorithm.

0<=c1g(n)<=f(n)<=c2g(n)
where c1,c2 is positive.

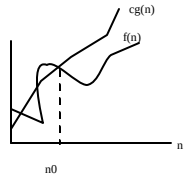
f(n)=(-)(g(n))



Growth of Functions Cont...

2)O Notation:
OH Notation bounds a function from upper and lower bound but if a function is only having its upper bound then it is denoted by O notation.

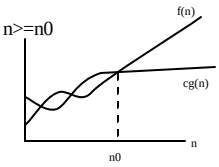
f(n)<=cg(n)
n>=n0



f(n)=0(g(n))

Growth of Functions Cont...

3) W **Notation:**
As 0 Notation provides upper bound omega notation provides the lower bound of asymptote.
cg(n)<=f(n)



f(n)= Ω (g(n))

Recurrences

- When an Algorithm contains a recursive call to itself, its running time can often be describe by a recurrence.
  T(n)=aT(n/b)+f(n)

Recurrences Cont...

1)Substitution method

This is the method of solving Recurrences which is totally based on the guess work

Recurrences Cont...

2)Iteration method

This method converts the recurrence into summation as then relies on techniques for bounding summations to solve the recurrence

Recurrences Cont...

3)Master Method:
The master method provides bounds for recurrences of the form

$T(n)=aT(n/b)+f(n)$

Recurrences Cont...

- Where a>=1,b>1, and f(n) is a given function; it requires memorization of three cases, but once you do that, determining asymptotic bounds for many recurrences is easy.

Basic Efficiency Classes

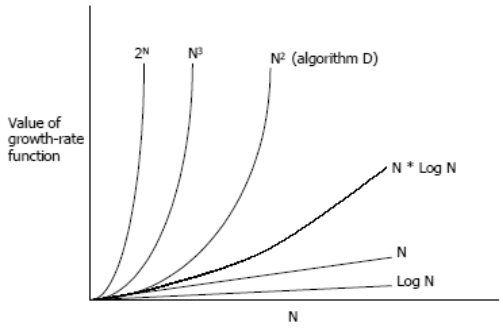
Basic Efficiency Classes

fast

1	constant
log *n*	logarithmic
n	linear
n log *n*	*n* log *n*
n^2	quadratic
n^3	cubic
2^n	exponential
n!	factorial

slow

Basic Efficiency Classes Cont…



Proofs of Correctness

Proofs of Correctness of Algorithms

- A proof of correctness of an algorithm is a proof of the following: Whenever the algorithm is run on a set of inputs that satisfy a problem's precondition, the algorithm halts, and its outputs (and inputs) satisfy the problem's post condition.
- A proof that a program is correct often has two pieces (that can be developed separately):

Proofs of Correctness Cont…

- **Proof of partial correctness:** This is a proof that, whenever an algorithm is run on a set of inputs satisfying the problem's precondition, either
  - the algorithm halts, and the outputs (and inputs) satisfy the problem's post condition, or
  - the algorithm does not halt at all!

So, an algorithm might be "partially correct" because it never, ever halts.

- **Proof of termination:** This is a proof that the algorithm always halts, whenever it is run on a set of inputs that satisfy the precondition.

 UL 16

Proofs of Correctness Cont…

- Moreover, Various strategies have been found to prove the correctness of different kinds of algorithms — including single statements, sequences of simpler programs, tests, and loops.

 UL 17

Write Algorithms

- Swap two numbers with out any temp. variable.
- Fibonacci Number upto 'N' Terms
- Factorial of a number
- Calculate the Percentage of a student and diplay Percentage and Division (Take assumption)
- String reversal

 UL 18

Four Equations

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

Mathematical Analysis

Four Equations

- T(n)=2T(n/2)+n // best and avg. CaseQS.
- T(n)=T(n-1)+n //worst case QS.
- T(n)=2T(n/2)+1//best Case MS.
- T(n)= 1 if n=1
  2T(n/2)+n if n>1 //worst and avg. Case MS.

Conclusion

- The word "Algorithm" has come to refer to a method that can be used by a computer for the solution of a problem.
- When an Algorithm contains a recursive call to itself, its running time can often be describe by a recurrence.

Review Questions

1. Out of the these two function n!, n^n _____ has the largest growth rate.
2. Solution of T(n)=2T(n/2)+n is_____
3. Sorting algorithm cannot be faster than_____
4. The Notations used to describe the running time of an algorithm is know as _____
5. Two main measures for the efficiency of an algorithm are
   a. Processor and memory
   b. Complexity and capacity
   c. Time and space
   d. Data and space

Review Questions Cont...

6. The time factor when determining the efficiency of algorithm is measured by
   a. Counting microseconds
   b. Counting the number of key operations
   c. Counting the number of statements
   d. Counting the kilobytes of algorithm

Review Questions Cont...

7. The space factor when determining the efficiency of algorithm is measured by
   a. Counting the maximum memory needed by the algorithm
   b. Counting the minimum memory needed by the algorithm
   c. Counting the average memory needed by the algorithm
   d. Counting the maximum disk space needed by the algorithm

Review Questions Cont...

8. Which of the following case does not exist in complexity theory
   a. Best case
   b. Worst case
   c. Average case
   d. Null case

Review Questions Cont...

9. The Worst case occur in linear search algorithm when
   a. Item is somewhere in the middle of the array
   b.Item is not in the array at all
   c. Item is the last element in the array
   d. Item is the last element in the array or is not there at all

Review Questions Cont...

10.The Average case occur in linear search algorithm
   a. When Item is somewhere in the middle of the array
   b. When Item is not in the array at all
   c. When Item is the last element in the array
   d. When Item is the last element in the array or is not there at all

Review Questions Cont...

Short answer type Questions

1. Discuss the entire cases of master method for solving recurrences with example.
2. Solve the recurrence relation T(n)=T(n-1)+n^4
3. Write any one algorithm (with example) for Sorting in Linear time.
4. Find the complexity of ?i^2.
5. Explain the master method with the help of an example of each case.(3)

Review Questions Cont...

6. Prove that sorting algorithm cannot be faster than nlogn.
7. What is the various Basic Efficiency Classes?
8. Explain ? Notation.
9. Solve T(n)=T(n/3)+1
10. Solve T(n)=2T(n/3)+ n (using iteration method)

Review Questions cont..

Long answer type Questions

1. What is stable sort algorithm? Explain with example.
2. For the given recurrence T(n)=aT(n/b)+f(n)
3. Solve the above equation for the case when a=b and f(n)=cn.
4. What do you mean by recurrences? Explain all the three methods for solving recurrences with the help of an example.
5. Explain Four golden equations ?
6. Find the complexity of merge Sort?
7. Explain is the various Basic Efficiency Classes?

Review Questions cont..

8. Explain the master method with the help of an example of each case?
9. Where do we use iteration method explain?
10. Explain substitution method? Give example as well.

Suggested Reading/References

1. T. H. Cormen, C. E. Leiserson, R. L. Rivest, Clifford Stein, "Introduction to Algorithms", 2nd Ed., PHI, 2004.
2. A. V. Aho, J. E. Hopcroft, J. D. Ullman, "The Design and Analysis of Computer Algorithms", Addition Wesley, 1998.
3. Ellis Horowitz and Sartaz Sahani, "Computer Algorithms", Galgotia Publications,1999.
4. D. E. Knuth, "The Art of Computer Programming", 2nd Ed., Addison Wesley, 1998

