

Fundamentals of algorithm & problem solving!

The essence of this architecture is captured by RAM.

→ In this instruction executed by one after another, one operation at time.

Understand the problem →

↓
Decide on computational

means Exact Vs approximate Solving data structures.

Algorithm design techniques.

↓
Design an algorithm.

↓
correctness

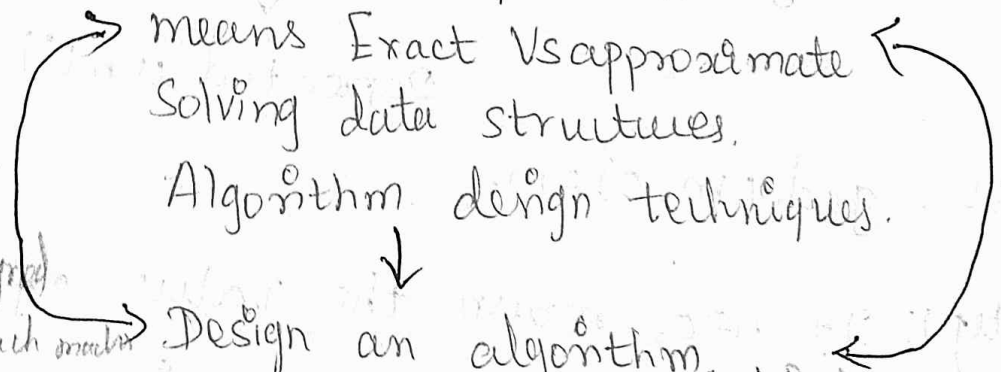
↓
Analyze the alg

↓
code the alg

→ Begin
• I/P A
• I/P B
• compute $50M + B$
End.

Seq
↓
→ Seq Algorithm designed to be executed on such machine
→ Alg that takes adv of this machine capability are called parallel.

parallel.



1. Understanding the problem:

- To design an alg
- Understand the problem completely
- Ask questions to clarify the doubts.
- Divide the problem into smaller problems until they become manageable size.

2. Computational devices:

- Machine instruction executed at one operation at a time.
- Executed only sequential alg.
- The operation executed is parallel.

3. Choosing b/w exact & appropriate problem solving:

- The Decision is to choose b/w solving the problem exactly & approximately.

Exact alg:

4. Solving Data structure:

- OOPS, DS is important for both DAA.

5. Alg Design Technique:

Def: It's a general approach to solving problems algorithmically that is applicable to a variety of problems from different areas of computing.

6. Methods of specifying an alg:

1. Euclid alg { simple eng statement.
step by step format.
2. Pseudo code.
3. Flow chart.

7. proving an alg correctness:

→ Alg has been specified, then its must be correct.

→ If the alg is incorrect, we need to either redesign it.

→ The correctness for an approximation alg, is if we are able to show that the error produced.

Analyze an alg:

→ An alg implemented, uses the computer primary memory & CPU.

→ We analyse the amount of resources (time & space).

→ Analyse is done at 2 stage $\left\{ \begin{array}{l} \text{Prior} \\ \text{Posterior} \end{array} \right.$

→ Qualities of alg (i) correctness, Efficiency, simplicity (iv) Generality.

→ Alg efficiency $\left\{ \begin{array}{l} \text{Time: how fast the alg runs} \\ \text{space: how much extra memory the alg needs.} \end{array} \right.$

→ Alg should be simple, because its easy to understand & easy to program.

Coding an alg:

→ The correctness of the pgm is done by proving it mathematically.

→ The Validity of the program is done by Testing.

→ Using code optimization technique.

Important problem types:

- Sorting
- Searching
- String processing
- Graph problem
- Combinatorial problem
- Geometric problems
- Numerical problems.

Fundamentals of analysis of Alg efficiency:

→ when alg implemented, uses the computer primary memory & CPU.

→ Analysing the amount of resource (time/space) for solving problems.

→ Alg efficiency depends on resource { Running Time
memory space.

Analyze Framework:

- ① Alg efficiency { Time
space.
- ② Measuring an Input size
- ③ Units for measuring Running Time.
- ④ Orders of Growth
- ⑤ Approaches in Analysis of alg. { Theoretical
Empirical.
- ⑥ Worst, Best, Average case Efficiencies.

Measuring Space complexity:-

→ An amount of storage space or memory required to execute.

$$S(P) = C + SP$$

Asymptotic Notations

where

$S(p)$ \rightarrow space complexity of a pgm 'p'

c \rightarrow constants for I/P & O/P.

S_p \rightarrow Instance characteristics of a pgm.

② Meaning Time complexity:

\rightarrow An amount of computer time required to completion.

\rightarrow Multitask sys execution time depends on

\hookrightarrow sys load.

\hookrightarrow Instruction set used

\hookrightarrow No. of other pgm running.

\hookrightarrow speed of underlying h/w.

\rightarrow Time complexity is given in terms of frequency count.

\rightarrow Denoted by Big Oh notation (O).

③ Meaning an I/P size:

\rightarrow The efficiency of an alg can be computed as a function

\rightarrow I/P size is passed as a parameter.

\rightarrow It can be \hookrightarrow exact value

\hookrightarrow Approximate value.

Ex:

Spell-check Alg.

\rightarrow NO of char

\rightarrow NO of words

} I/P size

④ Measuring Running Time:

- Identify basic operation.
- Understand the concept of basic operation.
- Compute Total no. of time taken by basic operation.

$$T(n) = C_{op} \cdot C(n)$$

$T(n)$ → Running time of basic operation.

C_{op} → Time take basic "

$C(n)$ → No. of times the operation needs to be executed.

⑤ Computing Order of Growth:

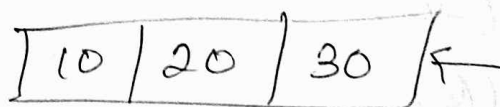
→ Measuring the performance of an alg in relation with I/P size.

Eg:

$\log n, n \log n, n^2, 2^n \dots$

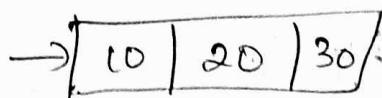
⑥ Worst, Best, Average case efficiency

worst case



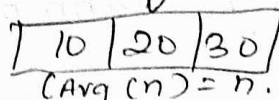
$$C_w(n) = n$$

Best



$$C_B(n) = n$$

Average



$$C_{Avg}(n) = n$$