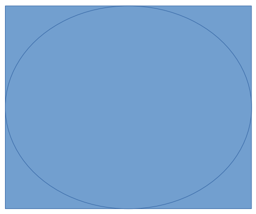
Deterministic and Non-Deterministic Algorithms

1. Random Coin Toss Example
   >> used to estimate the turn to pick vegetables from the market.
   >> Some apps that uses as a random process.

2. Example: PI Estimation
   Monte-Carlo Simulation.



Are, non-deterministic algorithms.

Choice ?? Each step, there exists.

Deterministic:

The computation with a given particular input, will result always the same output.

- The solution to a problem is bounded in polynomial time.
- The consecutive steps of the algorithm in a particular point of time is defined.

Non-Deterministic Algorithms

For the same input, the computation will result different outputs on different execution.

- The solution to a problem cannot be bounded in polynomial time.
- The consecutive steps of the algorithm in a particular point of time are not defined, but can be guessed.

Divide and Conquer Algorithms

This is a strategy to solve a large problem by:

a. partitioning into sub-problems.

b. solve sub-problems

c. accumulate the solution of the sub-problems to get the final output of the large problem.

Example: A merge sort

1. Divide : Split unsorted list.
2. Conquer : sort the smaller list.
3. Combine: merge the sorted list.

<u>Sequential and Parallel Algorithms</u>

```
   Example:
        Sequential aggregators.
        Parallel aggregators.

Sum of numbers:

   [1, 2, 5, 7, 3]:

Sequence: of computation.
      [1+2, 5, 7, 3]
      [1+2+5, 7, 3]
      [1+2+5+7, 3]
      [1+2+5+7+3], final output.

Parallel:
[1, 2, 5, 7, 3] => split (randomize).
[1, 2, 5] [7, 3]=> compute(map).
                => aggregate(reduce).
```

The Big O notations

>> It gives you the efficiency of an algorithm.
>> It tells you the growth of an algorithm.

Accurate definition:

$$f = O(g)$$

if there is a constant $c > 0$ such that $f(n) \leq c.g(n)$.

Example

Time consumed for a computation using some sort of algorithm. The time grows based on input size.

(Time could be machine dependent, but we measure based in input size regardless of machine time.)

Log Time: Binary Search O(log n)

Exponential Time:

    Bin Packing Problem. $O(2^n)$

Example: Solving sum of number problem.

Display all the positive integers x, y and z that makes sum = 5.

???

Given: x, y and z three numbers (positive integer). Display all numbers that makes x + y + z = 5.

(x = 0, y = 0, z = 5)
(x = 1, y = 0, z = 4)…
Lets say, max = n.

Algo. 1: ??

```
x goes from 0 to n,
    y goes from 0 to n,
        z goes from 0 to n,
            if(x+y+z) = 5 then display.
```

Total runs:

$(n+1) * (n+1) * (n+1) \Rightarrow (n+1)^3$

$\Rightarrow O(n^3)$

How to reduce this problem in square time ?

Algo. 2: ??

```
x goes from 0 to n,
   y goes from 0 to n,
      z = n - (x + y)
      if(z >= 0) then display.
```

x = 0

y = 0,

O(n^2) ??

Sorting: List sorting

bubble sort, selection sort => n^2

Merge sort, quick sort => nlogn

Classes of Running Algorithm Time

>> Constant – $O(1)$
Hashing,

>> Logarithmic – $O(\log N)$
Binary Search

>> Linear – $O(N)$
Linear Search

>> Log Linear – $O(N\log N)$
Quick Sort, Merge Sort …

>> Polynomial –
$O(N^2)$, $O(N^3)$, $O(N^4)$, etc.
Bubble Sort

>> Exponential –
$O(2^N)$, $O(3^N)$, $O(4^N)$, etc.
Bin Packing problem, TSP...

Heuristic and Approximate Algorithm

Heuristics:

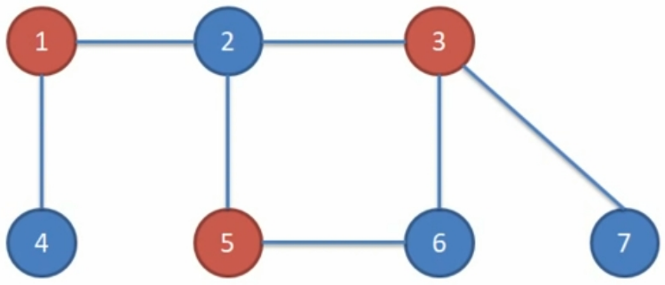
>> strategy to find a solution for a problem among all possible ones without guaranteeing the best.

>> this strategy are designed hoping to find the best solution and can be close to the best one.

>> usually are fast but might not be complete.

Example: Greedy Algorithm

Vertex Cover Problem in a Graph: What is the minimum number of vertices that covers all the edges ?



Approximate Algorithm:

>> when heuristics are applied for a solution to a problem, then it becomes an approximate solution, it might not be the exact one.

>> the algorithm becomes approximate algorithm.

>> generally applied on non-deterministic polynomial(complete) class of problems. (NP-Complete)

>> are use to get near optimal solution to a problem.

Example: Travelling salesman problem and nearest neighbors.