MCAC 301: Design and Analysis of Algorithms

Neelima Gupta

ngupta@cs.du.ac.in

August 23, 2020

Input Size : The Units

Input size is the number of memory words required to store the input. We can also use bits, bytes OR 2 or 4 memory words as a unit to define memory requirement. Constants do not matter. Some units that will be useful:

Input Size : The Units

Input size is the number of memory words required to store the input. We can also use bits, bytes OR 2 or 4 memory words as a unit to define memory requirement. Constants do not matter. Some units that will be useful:

- IW (Integer Word): Number of memory words required to store an integer.

Input Size : The Units

Input size is the number of memory words required to store the input. We can also use bits, bytes OR 2 or 4 memory words as a unit to define memory requirement. Constants do not matter. Some units that will be useful:

- IW (Integer Word): Number of memory words required to store an integer.
- IW CW (Character Word): Number of memory words required to store a character.

Input Size : The Units

Input size is the number of memory words required to store the input. We can also use bits, bytes OR 2 or 4 memory words as a unit to define memory requirement. Constants do not matter.
Some units that will be useful:

- IW (Integer Word): Number of memory words required to store an integer.
- IW CW (Character Word): Number of memory words required to store a character.
- IW FPW (Floating Point Word): Number of memory words required to store a floating point number.

Input Size : The Units

Input size is the number of memory words required to store the input. We can also use bits, bytes OR 2 or 4 memory words as a unit to define memory requirement. Constants do not matter.
Some units that will be useful:

- IW (Integer Word): Number of memory words required to store an integer.
- IW CW (Character Word): Number of memory words required to store a character.
- IW FPW (Floating Point Word): Number of memory words required to store a floating point number.

One can define more units like $2IWs$, $4FPWs$, $100CWs$ etc so long as they are constant number of memory words.

Input Size: Dealing with Sets

Set of numbers $\{x_1, x_2 \dots x_n\}$, set of jobs $\{j_1, j_2 \dots j_n\}$, set of points $\{p_1, p_2 \dots p_n\}$, set of objects $\{x_1, x_2 \dots x_n\}$, set of students $\{S_1, S_2 \dots S_n\}$. Input Size is the size of the Set.

Input Size: Dealing with Sets

Set of numbers $\{x_1, x_2 \dots x_n\}$, set of jobs $\{j_1, j_2 \dots j_n\}$, set of points $\{p_1, p_2 \dots p_n\}$, set of objects $\{x_1, x_2 \dots x_n\}$, set of students $\{S_1, S_2 \dots S_n\}$. Input Size is the size of the Set.

- Sort n real numbers: Each number requires 1 FPW. Input Size: n FPWs.

Input Size: Dealing with Sets

Set of numbers $\{x_1, x_2 \dots x_n\}$, set of jobs $\{j_1, j_2 \dots j_n\}$, set of points $\{p_1, p_2 \dots p_n\}$, set of objects $\{x_1, x_2 \dots x_n\}$, set of students $\{S_1, S_2 \dots S_n\}$. Input Size is the size of the Set.

- Sort n real numbers: Each number requires 1 FPW. Input Size: n FPWs.
- Set of points in a $2D$ plane: Each point requires 2 FPWs. Input Size: n (2 FPWs).

Input Size: Dealing with Sets

Set of numbers $\{x_1, x_2 \dots x_n\}$, set of jobs $\{j_1, j_2 \dots j_n\}$, set of points $\{p_1, p_2 \dots p_n\}$, set of objects $\{x_1, x_2 \dots x_n\}$, set of students $\{S_1, S_2 \dots S_n\}$. Input Size is the size of the Set.

- Sort n real numbers: Each number requires 1 FPW. Input Size: n FPWs.
- Set of points in a $2D$ plane: Each point requires 2 FPWs. Input Size: n (2 FPWs).
- Set of students (say name): Each name requires some maximum number say 100 of CWs. Input Size: n (100 CWs).

Input Size: Dealing with Sets

Set of numbers $\{x_1, x_2 \dots x_n\}$, set of jobs $\{j_1, j_2 \dots j_n\}$, set of points $\{p_1, p_2 \dots p_n\}$, set of objects $\{x_1, x_2 \dots x_n\}$, set of students $\{S_1, S_2 \dots S_n\}$. Input Size is the size of the Set.

- Sort n real numbers: Each number requires 1 FPW. Input Size: n FPWs.
- Set of points in a $2D$ plane: Each point requires 2 FPWs. Input Size: n (2 FPWs).
- Set of students (say name): Each name requires some maximum number say 100 of CWs. Input Size: n (100 CWs).

Thus, Input Size is $n = |S|$, where $|S|$ is the input set.
Pls do not forget, we are assuming that the elements of the set can be stored in constant number of memory words. This will be the case most of the times for the purpose of this course.

Input Size : Dealing with Graphs

Given a directed graph $G = (V, E)$ and a pair of vertices s and t, find a shortest path from s to t. Input size is the number of words required to store the input graph.

- Graph is nothing but a pair of sets V and E.
- Let $|V| = n$ and $|E| = m$
- Thus, Input Size: $(n + m)$.

Input Size : Dealing with Graphs

Given a directed graph $G = (V, E)$ and a pair of vertices s and t, find a shortest path from s to t. Input size is the number of words required to store the input graph.

- Graph is nothing but a pair of sets V and E.
- Let $|V| = n$ and $|E| = m$
- Thus, Input Size: $(n + m)$.
- However, in adjacency matrix representation, it is n^2.

The space required by s and t, in this case, is constant as each element of the vertex set can be stored in constant number of MWs. Hence can be ignored.

Input Size: Dealing with Values

Primality Testing - Test whether a given integer n is prime.

- What do you mean by, "as the input grows" in case of sets?

Input Size: Dealing with Values

Primality Testing - Test whether a given integer n is prime.

- What do you mean by, "as the input grows" in case of sets?
  Size of the set grows, right?

Input Size: Dealing with Values

Primality Testing - Test whether a given integer n is prime.

- What do you mean by, "as the input grows" in case of sets?
  Size of the set grows, right?
- What do you mean by, "as the input grows" in case of graphs?

Input Size: Dealing with Values

Primality Testing - Test whether a given integer n is prime.

- What do you mean by, "as the input grows" in case of sets?
  Size of the set grows, right?
- What do you mean by, "as the input grows" in case of graphs?
  Size of the graph grows, right?

Input Size: Dealing with Values

Primality Testing - Test whether a given integer n is prime.

- What do you mean by, "as the input grows" in case of sets?
  Size of the set grows, right?
- What do you mean by, "as the input grows" in case of graphs?
  Size of the graph grows, right?
- What do you mean by, "as the input grows" in this case?

Input Size: Dealing with Values

Primality Testing - Test whether a given integer n is prime.

- What do you mean by, "as the input grows" in case of sets?
  Size of the set grows, right?
- What do you mean by, "as the input grows" in case of graphs?
  Size of the graph grows, right?
- What do you mean by, "as the input grows" in this case?

In this case, input grows as the value of n grows. The number of memory words required to store n increases as the value of n increases - it is not a constant.

Input size: $\log_2 n$ bits. Note the unit and the base of log.

Input Size: Dealing with Values

Primality Testing - Test whether a given integer n is prime.

- What do you mean by, "as the input grows" in case of sets?
  Size of the set grows, right?
- What do you mean by, "as the input grows" in case of graphs?
  Size of the graph grows, right?
- What do you mean by, "as the input grows" in this case?

In this case, input grows as the value of n grows. The number of memory words required to store n increases as the value of n increases - it is not a constant.

Input size: $\log_2 n$ bits. Note the unit and the base of log.
Thus an algorithm that runs in $O(n)$ time is actually exponential in the input size.

Dealing with Values: More Examples

1. gcd(n, m). Input Size: ?

Dealing with Values: More Examples

1. gcd(n, m). Input Size: ? $\log_2 n + \log_2 m$ bits.

Dealing with Values: More Examples

1. gcd(n, m). Input Size: ? $\log_2 n + \log_2 m$ bits.
2. Compute x^n *modulo m*. Input Size: ?

Dealing with Values: More Examples

1. gcd(n, m). Input Size: ? $\log_2 n + \log_2 m$ bits.
2. Compute x^n *modulo* m. Input Size: ?
   $\log_2 x + \log_2 n + \log_2 m$ bits.

Dealing with Values: More Examples

1. gcd(n, m). Input Size: ? $\log_2 n + \log_2 m$ bits.
2. Compute x^n *modulo* m. Input Size: ?
   $\log_2 x + \log_2 n + \log_2 m$ bits.

   Often we assume that x is a small number which can be stored in a constant number of MWs. In that case, we can ignore $\log_2 x$ and drop it from the IS.

The Output Size

Output Size is the number of memory words required to write the output. Though we can talk about input independent of the problem, we cannot talk about the output independent of the problem.

- Sort n real numbers:
  Output: ?

The Output Size

Output Size is the number of memory words required to write the output. Though we can talk about input independent of the problem, we cannot talk about the output independent of the problem.

- Sort n real numbers:
  Output: ? Sorted sequence of numbers

The Output Size

Output Size is the number of memory words required to write the output. Though we can talk about input independent of the problem, we cannot talk about the output independent of the problem.

- Sort n real numbers:
  Output: ? Sorted sequence of numbers Output Size:?

The Output Size

Output Size is the number of memory words required to write the output. Though we can talk about input independent of the problem, we cannot talk about the output independent of the problem.

- Sort n real numbers:
  Output: ? Sorted sequence of numbers Output Size:?
  Same as the input size - n (FPWs)

The Output Size

Output Size is the number of memory words required to write the output. Though we can talk about input independent of the problem, we cannot talk about the output independent of the problem.

- Sort n real numbers:
  Output: ? Sorted sequence of numbers Output Size:?
  Same as the input size - n (FPWs)
- Given a set of points in a $2D$ plane, find a closest of pair of points:
  Output :?

The Output Size

Output Size is the number of memory words required to write the output. Though we can talk about input independent of the problem, we cannot talk about the output independent of the problem.

- Sort n real numbers:
  Output: ? Sorted sequence of numbers Output Size:?
  Same as the input size - n (FPWs)
- Given a set of points in a $2D$ plane, find a closest of pair of points:
  Output :? Just 2 points.

The Output Size

Output Size is the number of memory words required to write the output. Though we can talk about input independent of the problem, we cannot talk about the output independent of the problem.

- Sort n real numbers:
  Output: ? Sorted sequence of numbers Output Size:?
  Same as the input size - n (FPWs)
- Given a set of points in a $2D$ plane, find a closest of pair of points:
  Output :? Just 2 points.
  Output Size:?

The Output Size

Output Size is the number of memory words required to write the output. Though we can talk about input independent of the problem, we cannot talk about the output independent of the problem.

- Sort n real numbers:
  Output: ? Sorted sequence of numbers Output Size:?
  Same as the input size - n (FPWs)
- Given a set of points in a $2D$ plane, find a closest of pair of points:
  Output :? Just 2 points.
  Output Size:? 2 FPWs - constant.
- Given a set of students (say name): Determine the student whose name appears first in the lexicographic order.
  Output :?

The Output Size

Output Size is the number of memory words required to write the output. Though we can talk about input independent of the problem, we cannot talk about the output independent of the problem.

- Sort n real numbers:
  Output: ? Sorted sequence of numbers Output Size:?
  Same as the input size - n (FPWs)
- Given a set of points in a $2D$ plane, find a closest of pair of points:
  Output :? Just 2 points.
  Output Size:? 2 FPWs - constant.
- Given a set of students (say name): Determine the student whose name appears first in the lexicographic order.
  Output :? Just 1 name

The Output Size

Output Size is the number of memory words required to write the output. Though we can talk about input independent of the problem, we cannot talk about the output independent of the problem.

- Sort n real numbers:
  Output: ? Sorted sequence of numbers Output Size:?
  Same as the input size - n (FPWs)
- Given a set of points in a $2D$ plane, find a closest of pair of points:
  Output :? Just 2 points.
  Output Size:? 2 FPWs - constant.
- Given a set of students (say name): Determine the student whose name appears first in the lexicographic order.
  Output :? Just 1 name
  Output Size :?

The Output Size

Output Size is the number of memory words required to write the output. Though we can talk about input independent of the problem, we cannot talk about the output independent of the problem.

- Sort n real numbers:
  Output: ? Sorted sequence of numbers Output Size:?
  Same as the input size - n (FPWs)
- Given a set of points in a $2D$ plane, find a closest of pair of points:
  Output :? Just 2 points.
  Output Size:? 2 FPWs - constant.
- Given a set of students (say name): Determine the student whose name appears first in the lexicographic order.
  Output :? Just 1 name
  Output Size :? maximum 100 (CWs) - constant.

Output Size contd . . .

- Given a directed graph $G = (V, E)$ and a pair of vertices s and t, find a shortest path from s to t.
  Output :?

Output Size contd . . .

- Given a directed graph $G = (V, E)$ and a pair of vertices s and t, find a shortest path from s to t.
  Output :? Path - the sequence of vertices from s to t.

Output Size contd ...

- Given a directed graph $G = (V, E)$ and a pair of vertices s and t, find a shortest path from s to t.
  Output :? Path - the sequence of vertices from s to t.
  Output Size:?

Output Size contd . . .

- Given a directed graph $G = (V, E)$ and a pair of vertices s and t, find a shortest path from s to t.
  Output :? Path - the sequence of vertices from s to t.
  Output Size:? - length of the path which is maximum n.

Output Size contd ...

1. Primality Testing - Find out whether n is a prime number.
   Output:?

Output Size contd . . .

1. Primality Testing - Find out whether n is a prime number.
   Output:? Yes/No (1/0)

Output Size contd . . .

1. Primality Testing - Find out whether n is a prime number.
   Output:? Yes/No (1/0)
   Output Size:?

Output Size contd . . .

1. Primality Testing - Find out whether n is a prime number.
   Output:? Yes/No (1/0)
   Output Size:? Just 1 bit - constant

Output Size contd . . .

1. Primality Testing - Find out whether n is a prime number.
   Output:? Yes/No (1/0)
   Output Size:? Just 1 bit - constant
2. gcd(n, m).
   Output:?

Output Size contd ...

1. Primality Testing - Find out whether n is a prime number.
   Output:? Yes/No (1/0)
   Output Size:? Just 1 bit - constant
2. gcd(n, m).
   Output:? some integer value $\leq \min\{n, m\}$

Output Size contd ...

1. Primality Testing - Find out whether n is a prime number.
   Output:? Yes/No (1/0)
   Output Size:? Just 1 bit - constant
2. gcd(n, m).
   Output:? some integer value $\leq \min\{n, m\}$
   Output Size:?

Output Size contd . . .

1. Primality Testing - Find out whether n is a prime number.
   Output:? Yes/No (1/0)
   Output Size:? Just 1 bit - constant
2. gcd(n, m).
   Output:? some integer value $\leq \min\{n, m\}$
   Output Size:? in the worst case $\min\{\log_2 n, \log_2 m\}$.
3. Compute x^n *modulo* m.
   Output:?

Output Size contd . . .

1. Primality Testing - Find out whether n is a prime number.
   Output:? Yes/No (1/0)
   Output Size:? Just 1 bit - constant
2. gcd(n, m).
   Output:? some integer value $\leq \min\{n, m\}$
   Output Size:? in the worst case $\min\{\log_2 n, \log_2 m\}$.
3. Compute x^n *modulo* m.
   Output:? some integer value $\leq$?

Output Size contd ...

1. Primality Testing - Find out whether n is a prime number.
   Output:? Yes/No (1/0)
   Output Size:? Just 1 bit - constant
2. gcd(n, m).
   Output:? some integer value $\leq \min\{n, m\}$
   Output Size:? in the worst case $\min\{\log_2 n, \log_2 m\}$.
3. Compute x^n *modulo* m.
   Output:? some integer value $\leq ?$
   Output Size:?

Output Size contd ...

1. Primality Testing - Find out whether n is a prime number.
   Output:? Yes/No (1/0)
   Output Size:? Just 1 bit - constant
2. gcd(n, m).
   Output:? some integer value $\leq \min\{n, m\}$
   Output Size:? in the worst case $\min\{\log_2 n, \log_2 m\}$.
3. Compute x^n *modulo* m.
   Output:? some integer value $\leq$?
   Output Size:? in the worst case $\log_2 m$.

   Output is independent of x, Why?

Output Size contd . . .

1. Primality Testing - Find out whether n is a prime number.
   Output:? Yes/No (1/0)
   Output Size:? Just 1 bit - constant
2. gcd(n, m).
   Output:? some integer value $\leq \min\{n, m\}$
   Output Size:? in the worst case $\min\{\log_2 n, \log_2 m\}$.
3. Compute x^n *modulo* m.
   Output:? some integer value $\leq ?$
   Output Size:? in the worst case $\log_2 m$.

   Output is independent of x, Why?
   Ans: because of modulo m.

Frame Title

What is the minimum time any meaningful algorithm will take?

Frame Title

What is the minimum time any meaningful algorithm will take?
Ans :$\max\{InputSize, OutputSize\}$