Graph Theory & Algorithms

Aniket Pandey

23 December 2017

1   Introduction

Graph Theory is the study of Graphs, the mathematical objects modelling the pairwise relation of Vertices(also called nodes) and Edges where two nodes are connected by edges. A graph can be directed or undirected, cyclic or acyclic, linear or weighted etc.

Graph Theoretical concepts are widely used to study and model various applications, in different areas. For example, in Computer Science, problems like travelling salesman problem, the shortest spanning tree in a weighted graph and in Mathematics like hamiltonian graphs and Fermat's Little Theorem & Nielson-Schreier Theorem.

2   Notations

2.1   Big-O Notation

Big-O Notations are used in mathematics to characterize functions according to their growth rate. In Graph Theory, efficiency of an algorithm is measured in terms of the input length n as $n \to \infty$.

Formal definition would be
If $f : N \to N$ and $g : N \to N$ are two functions, then $f = O(g)$ if and only if $f(n) < c \cdot g(n)$ for a constant c as $n \to \infty$.

2.2   Other Notations

There are a few more notations which complement Big-O Notation. I will give a brief information about these.

For functions f & g from N to N

$$f = \Omega(g) \quad \text{if } g = O(f) \tag{1}$$
$$f = \Theta(g) \quad \text{if } f = O(g) \ \& \ g = O(f) \tag{2}$$
$$f = o(g) \quad \text{if there exists } \varepsilon \text{ such that } f(n) < \varepsilon \cdot g(n) \tag{3}$$
$$f = \omega(g) \quad \text{if } g = o(f) \tag{4}$$

3 Terminologies

Here are a few basic terminologies that are used to represent navigation through the Graph.

Walk A walk is any route through a graph from vertex to vertex along edges. A walk can end on the same vertex on which it began or on a different vertex. A walk can travel over any edge and any vertex any number of times.

Path A path is a walk that does not include any vertex twice, except that its first vertex might be the same as its last.

Trail A trail is a walk that does not pass over the same edge twice. A trail might visit the same vertex twice, but only if it comes and goes from a different edge each time.

Cycle A cycle is a path that begins and ends on the same vertex.

4 Data Structures used in Graphs

4.1 Computational Model

Computational model is a set of operations which are used to measure the resources(time and space) that are needed for a certain problem ot be solved. In layman's terms, using this model , we can determine if the given algorithm is efficient or not.

One such model is *Turing Machine* . It was invented by Alan Turing in 1936.

4.2 Turing Machine

Turing Machine is an abstract machine which is made up of infinite tape(s) divided into discrete boxes called *cells* and an imaginary head which can read the information in the boxes and can change the values of the cells. It also has a *state register* which stores the information about the state of the head.
The basic *Transition function* of a Turing Machine is defined as follows

$$\delta : Q \times \Gamma^k \longrightarrow Q \times \Gamma^{k-1} \times \{L, S, R\}^k \tag{5}$$

The above equation describes the functioning of a simple Turing Machine consisting of k tapes where $1st$ tape is the input tape, the rest $k-1$ tapes are called *work tapes.* Last one of them is designated as output tape.

Here, the input tape takes the input, the work tapes decide what to do with the input and in which direction to move the head. Then finally, changes(if any) are done in the output tape.

4.2.1 Properties of Turing Machine

1. If the time taken by a k tape Turing Machine to compute a function is $T(n)$, where $T : N \to N$ is a Time-Constructible function. Then the time required to compute the same function by a single tape Turing Machine is $5kT(n)^2$.

2. If a bi-directional Turing Machine computes a function in time $T(n)$, where $T : N \to N$ is a Time-Constructible function. Then the time required to compute the same function by a standard uni-directional Turing Machine is $4T(n)$.

4.2.2 Universal Turing Machine

A Universal Turing Machine is a machine which can run an arbitrary Turing Machine for a given input. It is the most general form of Turing Machine, with the general form of Time complexity of UTM being $O(TlogT)$ where T is for a normal TM . The reason I think it is because of the fact that a UTM has to deal with more bits of input than a normal one, which results in it being comparatively slower by a *log* factor.

Another important result associated with a Universal Turing Machine is *The Halting Problem.*The Halting Problem is a very interesting problem in context of *Uncomputability*, which proves that we cannot have a UTM which can tell with certainty whether a Turing Machine will compute a program for a given input in a finite time. In simpler words, no machine can solve the Halting Problem. This is a very important result in Computer Science because most of the problems that we generally encounter are Halting Problems, that is, a solution to them would solve The Halting Problem.

Now since Halting Problem is unsolvable , it gives us another important result, that *most problems are uncomputable*. For example, consider the *Decision Problems* which can be considered to have input as an infinite set of either 0 or 1 ,i.e $\{0, 1\}^*$. Any combination of 0's and 1's is possible, so its impossible to compute it in finite time. Similarly, many such problems exist.

5 Cryptography

Crytpography, or encryption in modern terms is referred to the process of converting a normal message (plaintext) into an unintelligible text (ciphertext) using a key called encryption key. Reverse of the process, decryption uses decryption key to recover back the original message.

$$D_k(E_k(x)) = x \tag{6}$$

Here,

$$E_k = \text{Encryption algorithm} \tag{7}$$
$$D_k = \text{Decryption algorithm} \tag{8}$$
$$x = \text{Plain text message} \tag{9}$$

A Cipher is the pair of algorithms which govern the encryption and decryption of data. The secrecy of data relies on the fact that only the concerned party has the knowledge of cipher key. As shown in the image below, only Alice and Bob know the actual content. A particular eavesdropper, i.e Eve recovers the data being transmitted by Alice. But to her dismay, she will only have some gibberish in the data.
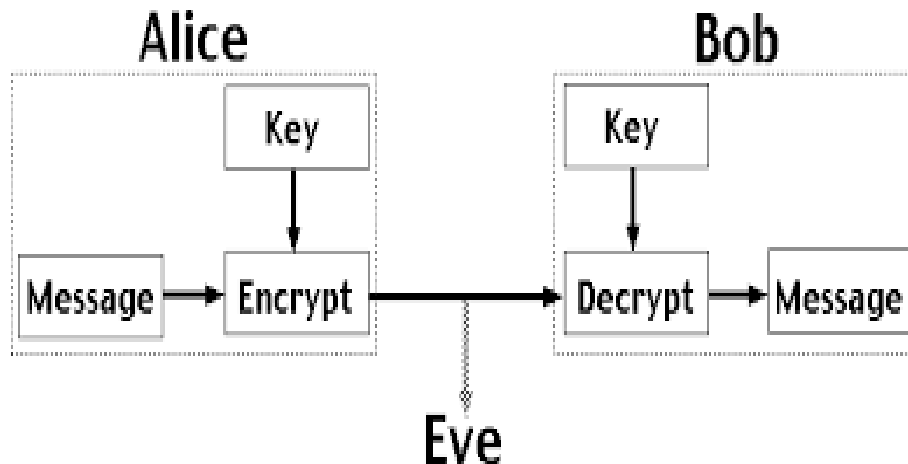
5.1 Public Key Cryptography

The two keys, i.e Encryption and Decryption key, can either be public or private. But the frequently observed case is that the encryption cipher is public and decryption cipher is private.

This is also called **Asymmetric Cryptography**.

Public key Cryptography ensures confidentiality. That is, anyone will be able to encrypt their data but only the party with the private key will be able to decrypt it.

5.2 Perfect Secrecy

An Encryption algorithm, however complex, can be easily broken unless its perfectly secret. Perfect Secrecy relies on the condition that the ciphertext cannot be decrypted even if the decryption machine has unlimited computational power.

Perfect secrecy is like completely randomised ciphertext. There is no fixed rule that governs the encryption. The Encryption key is chosen by pseudorandom generators which are 100% random.

5.2.1 One Time Pad

Just like Caeser shift, One time pad works on the principle of translating the letters by a certain length. However, the shift is not constant. For each letter in

the original message, there is a unique encryption key. And that key is completely random, so the eavesdropper has no idea what the message is even after brute-forcing the cipher-text with unlimited computational power.

5.2.2 Demonstration

I will show a demonstration of how perfectly secret texts cannot be deciphered.

Suppose you want to encrypt a message BAD DOG with a One time pad XIF CLM (each letter chosen randomly). So after running the encryption algorithm..

$$(XIF\ CLM):(BAD\ DOG) \longrightarrow ZJJ\ GAT$$

Now, the eavesdropper will try to identify the pattern in the ciphertext. Since the encryption key was chosen completely randomly and is of the same length of the message, so there will not be any observable pattern in the text. Using Brute force as the last resort, he'll try to get every possible feasible outcome. On trying out a few keys

$$(UIP\ DZZ):(ZJJ\ GAT) \longleftarrow EAT\ CAT$$
$$(XIF\ CLM):(ZJJ\ GAT) \longleftarrow BAD\ DOG$$
$$(TIP\ FMZ):(ZJJ\ GAT) \longleftarrow FAT\ ANT$$

Since, each of the decrypted messages EAT CAT, BAD DOG, FAT ANT have equal probability of being the original message, there is now way the eavesdropper will be certain of the actual message.