INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

Introduction to Cryptography

Dr. Ashok Kumar Das

Center for Security, Theory and Algorithmic Research
International Institute of Information Technology, Hyderabad

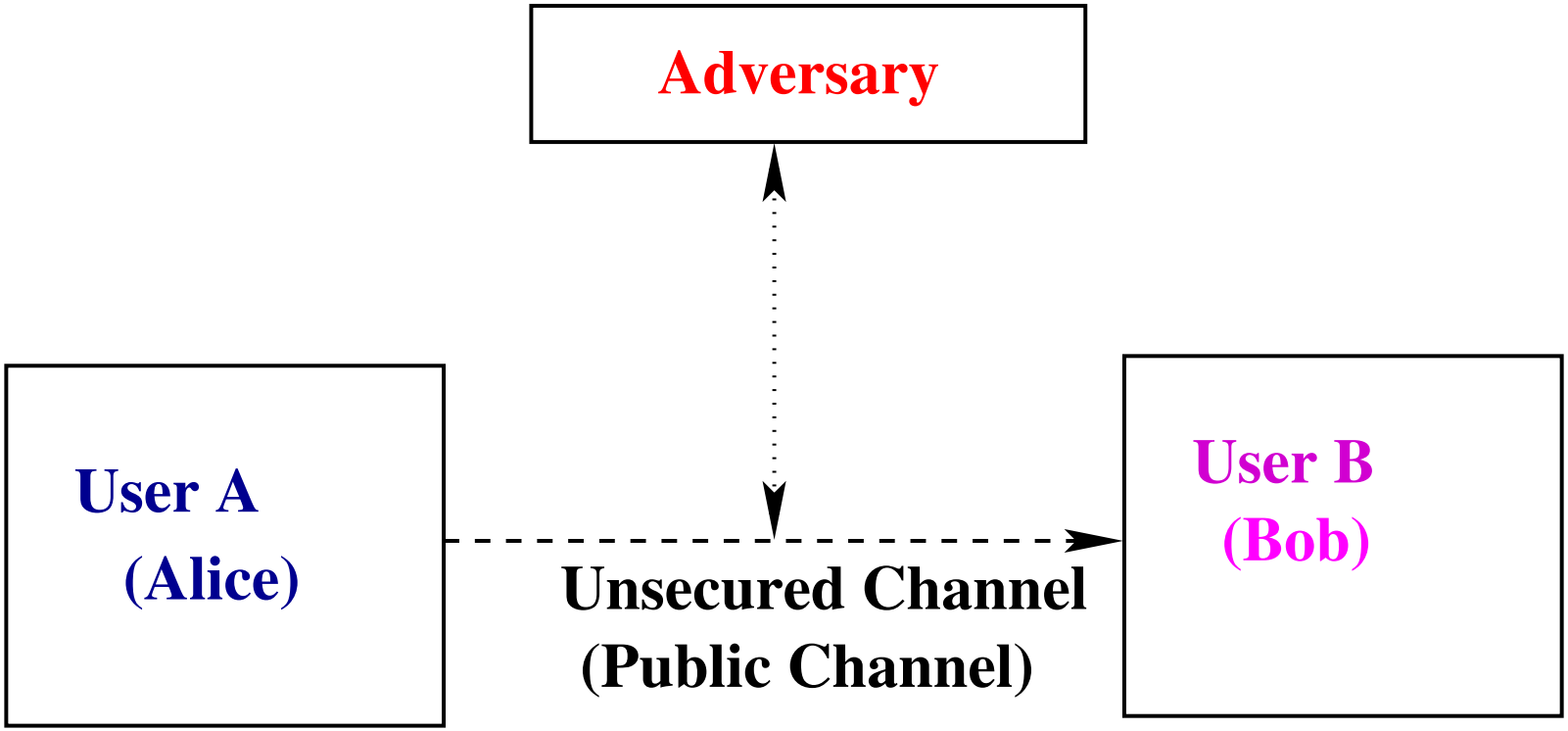
E-mail: *ashok.das@iiit.ac.in*
URL: http://www.iiit.ac.in/people/faculty/ashokkdas
https://sites.google.com/view/iitkgpakdas/

Overview of Cryptography

What is Cryptography?

- Cryptography is the study of **mathematical techniques** related to aspects of information security such as confidentiality, data integrity, entity authentication, message authentication (data origin authentication) and non-repudiation.
- Cryptography is not the only means of providing information security, but rather one set of techniques.
- Now-a-days, cryptography has moved from an art to a science. Thus, cryptography is the science of keeping secrets secret.

Consider the following simple two-party communication model:



- An "**adversary**" is an entity in a two-party communication which is neither the sender nor the receiver, and which tries to defeat the information security service being provided between the sender and the receiver.
- A "**channel**" is a means of conveying information from one entity to another entity.
- An "**unsecured channel**" is one from which parties other than the sender and the receiver can reorder, delete, insert, or read the data being transmitted.
- A "**secured channel**" is one from which an adversary does not have the ability to reorder, delete, insert, or read the data being transmitted.

Types of adversary

- A "**passive adversary**" is an adversary who is only capable of reading information from an unsecured channel.
- An "**active adversary**" is an adversary who is capable to transmit, alter, or delete information on an unsecured channel.

Cryptographic goals (objectives)

- **Confidentiality:** Privacy (confidentiality) is a service of keeping information secret from all but those who are authorized to see it.
- **Data integrity:** ensuring information has not been altered by unauthorized or unknown means.
- **Entity authentication or identification:** Corroboration of the identity of an entity (i.e., a person, a computer terminal, a credit card, etc.).
- **Message or data origin authentication:** Corroborating the source of information.
- **Non-repudiation:** Preventing the denial of the previous session (preventing the malicious nodes to hide their activities).

Cryptographic goals (objectives)

- **Authorization:** Conveyance to another entity such as a person or group of users. It ensures that the nodes (users) those who are authorized can be involved in providing information to network services.
- **Signature:** a means to bind information to an entity.
- **Access control:** restricting access to resources to privileged entity.
- **Certification:** endorsement of information by a trusted entity.

Cryptographic goals (objectives)

We need also to consider the forward and backward secrecy when new nodes join in the network and existing nodes depart from the network.

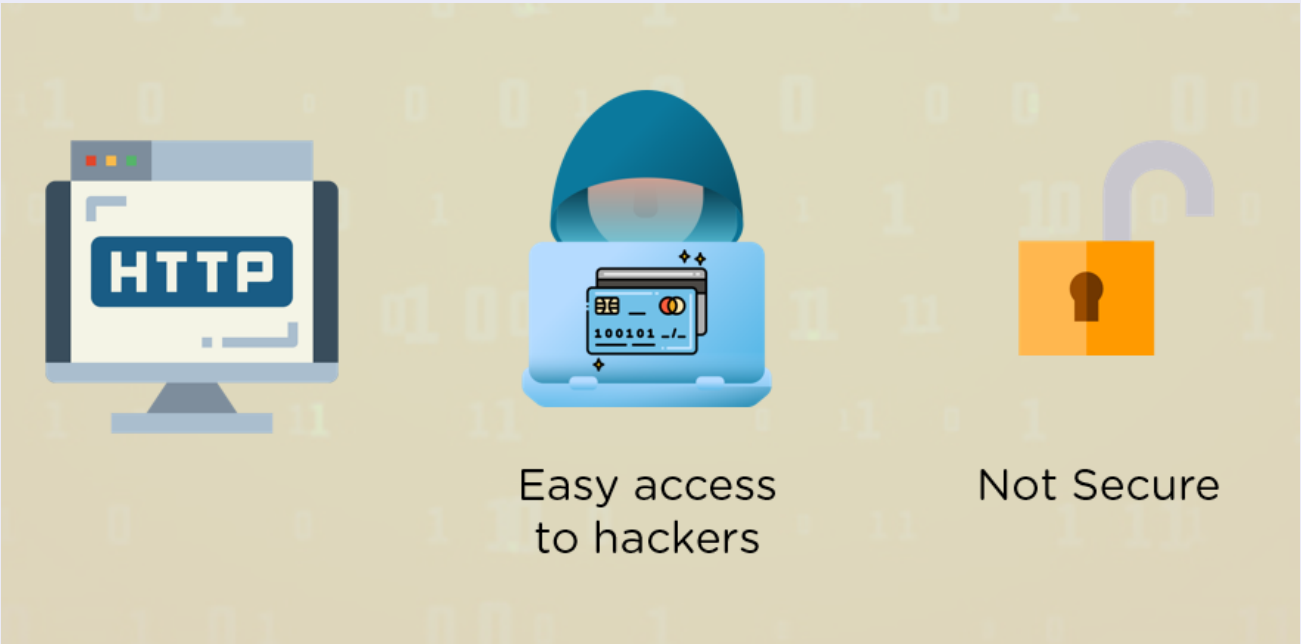
- ***Forward secrecy:*** When a node (user) leaves the network, it must not read any future messages after its departure.
- ***Backward secrecy:*** When a new node (user) joins in the network, it must not read any previously transmitted message.

What is the Need for Cryptography?

- Suppose Alice wants to look for a discount on the latest iPhone. After browsing the internet, she comes across a questionable website willing to offer a 50% discount on the first purchase.
- A few moments after she provides her payment details, the website withdraws a huge chunk of money from her account.
- Alice then wonders how she had failed in realizing that the website was a scam.

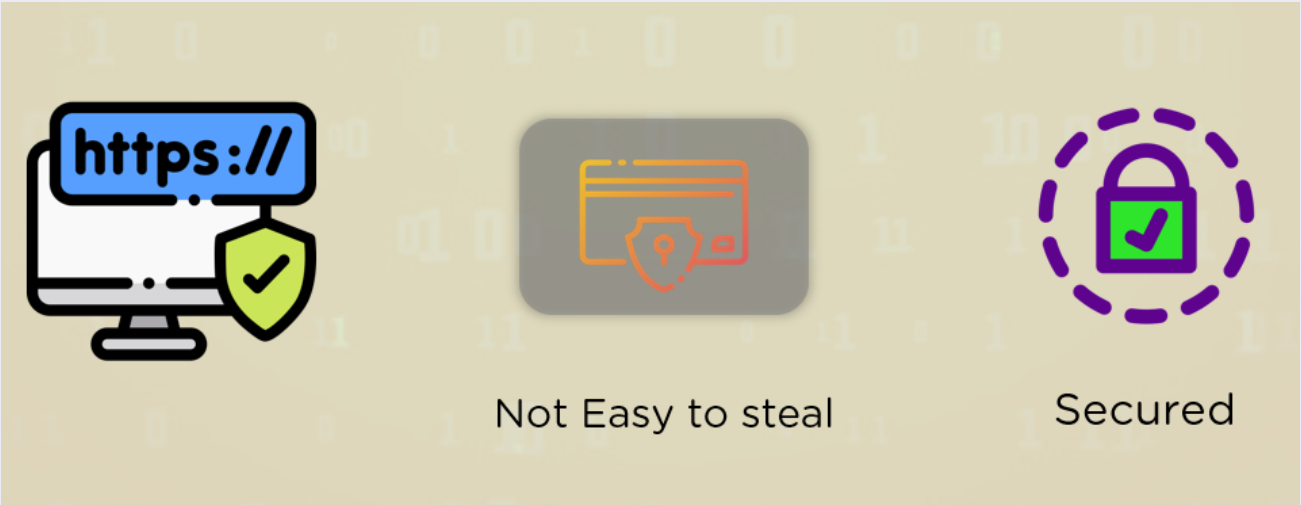
What is the Need for Cryptography?

- She then notices that the website is an HTTP webpage instead of HTTPS.
- Hypertext Transfer Protocol (HTTP) is an application-layer protocol for transmitting hypermedia documents, such as HTML. It was designed for communication between web browsers and web servers, but it can also be used for other purposes.
- The payment information submitted was not encrypted and visible to anyone keeping an eye, including the website owner.



What is the Need for Cryptography?

- Now, if she had chosen to use a reputed website, which has encrypted transactions and employs cryptography, this iPhone enthusiast could have avoided this particular incident. This is why it's never recommended to visit unknown websites or share any personal information on them.
- This is where Cryptography comes to play, and is so essential.



Hypertext Transfer Protocol Secure (HTTPS) is an extension of the Hypertext Transfer Protocol (HTTP). It is used for secure communication over a computer network, and is widely used on the Internet.

Introduction to Cryptography

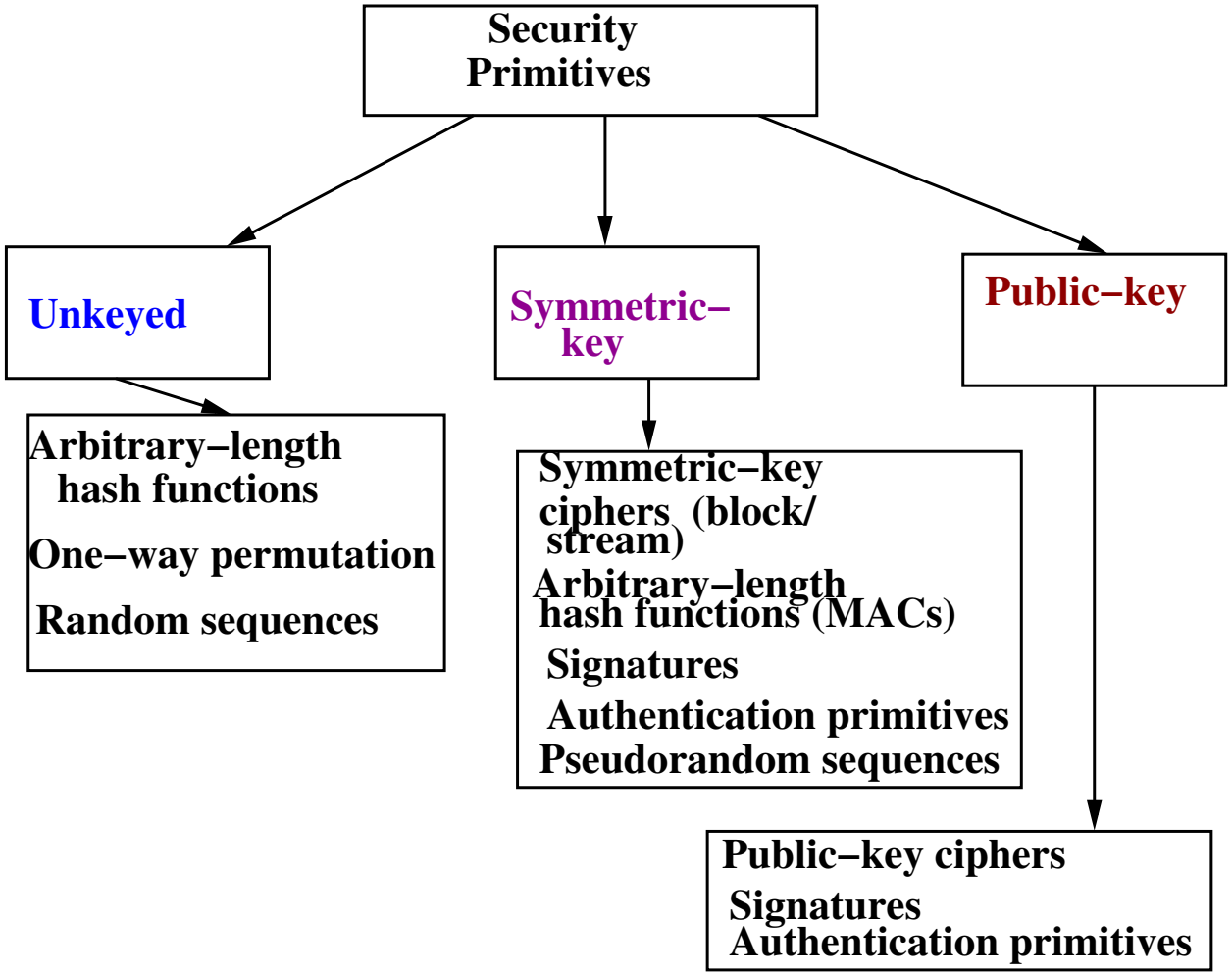


Figure: A taxonomy of cryptographic primitives

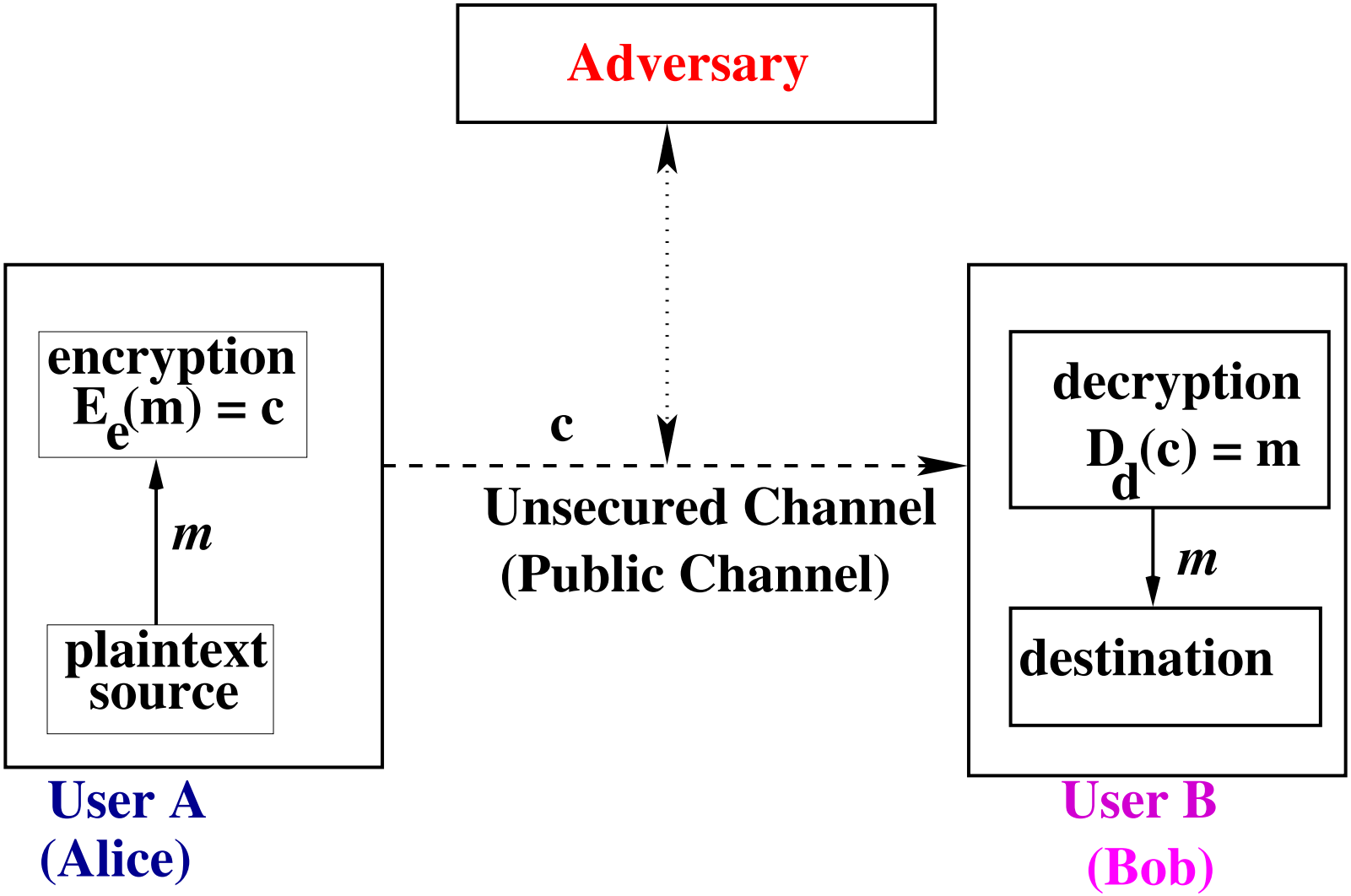
Evaluation criteria for the primitives

- ***Level of security:*** This is usually difficult to quantify.
- ***Functionality:*** Primitives will need to be combined to meet various information security objectives.
- ***Methods of operation:*** One primitive could provide very different functionality depending on its mode of operation or usage.
- ***Performance:*** This refers to the efficiency of a primitive in a particular mode of operation (For example, an encryption algorithm may be rated by the number of bits per second which it can encrypt).
- ***Easy of implementation:*** This might include the complexity of implementing the primitives in either a software or hardware environment.

Note that the relative importance of various criteria is very much dependent on the application and resources availability.

Introduction to Cryptography

Consider the following simple two-party communication model with encryption:



- **Encryption scheme 1:** Have only the encryption and decryption functions and these are kept secret to the sender and receiver only. No key is used in this method.
- **Encryption scheme 2:** Key is being used. However, the encryption and decryption functions are made public.

Quiz: Why keys are necessary? Why not just choose one encryption function and its corresponding decryption function?

- **Security of the scheme**
  - Depends entirely on the secrecy of the key
  - Does not depend on the secrecy of the algorithm (Needs to be public for criticism!)
- Hence, we make the **assumptions** as follows:
  - Algorithms for encryption/decryption are known to the public
  - Keys used are kept secret

P, NP, NP-Hard and NP-Completeness

P, NP, NP-Hard and NP-Completeness

Decision Problem

- Any problem for which the answer is either zero (0) or one (1) is called a decision problem.
- An algorithm for a decision problem is termed as a decision algorithm.

P, NP, NP-Hard and NP-Completeness

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

Time complexity classes

- Let $t : \mathcal{N} \rightarrow \mathcal{N}$ be a function, where $\mathcal{N}$ is the set of natural numbers.
- Define the time complexity class, $TIME(t(n))$, to be $TIME(t(n)) = \{L | L$ is a language decided by an $O(t(n))$-time deterministic Turing machine (DTM) $\}$.
- Define the time complexity class, $NTIME(t(n))$, to be $NTIME(t(n)) = \{L | L$ is a language decided by an $O(t(n))$-time non-deterministic Turing machine (NTM) $\}$.

P, NP, NP-Hard and NP-Completeness

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

The Class P

- $P = \{L | L$ is a language or problem decided by a deterministic Turing machine (DTM) in polynomial time $\}$. In other words, $P = \cup_{k \in \mathcal{N}} TIME(n^k)$, where $\mathcal{N}$ is the set of natural numbers.
- P is the class of languages that can be decided quickly on a DTM.
- P is the set of all decision problems solvable by deterministic algorithms in poly-time.
- P is the class of all "practically" solvable problems.

P, NP, NP-Hard and NP-Completeness

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

Path problem in a graph

- Let $G = (V, E)$ be a (directed/undirected) graph, where V be the set of vertices (nodes) and E the set of edges of G. Then the adjacency matrix or link list representation $\langle G \rangle$ is an encoding (reasonable encoding) of the graph G.
- Define a formal problem as follows:
  $PATH := \{\langle G, s, t \rangle \rangle | G$ is a directed graph with a path from node s to node t in $G\}$.

P, NP, NP-Hard and NP-Completeness

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY HYDERABAD

Example of the class P:

Theorem

PATH is in P.

- Input: $\langle G, s, t \rangle$, for all valid encoding of graph G.
- Output: Accept, if there is a path from s to t in G; reject, otherwise.

Stages (Steps or Iterations):

1. Place a mark on the node s,
2. Repeat the following until one fails to mark an additional node:
3. Scan all the edges of G. It there is an edge $(u, v) \in E$ from marked node u to an unmarked node v, then mark v.
4. If t is marked, then “accept”; else “reject”.

Example of the class P (Continued...)

- Correctness: Clear.
- Complexity (Running time):
  – Stage 1 requires only one step, that is, Stage 1 is executed only once.
  – Stage 4 is executed only once.
  – Stages 2 and 3 are executed at most $n(n-1) = O(n^2)$ time, where $n = |V|$ times.
- Therefore, the total number of stages is polynomial in the input size (n).
- Again, each stage can be implemented on a DTM using polynomially many steps (transitions) on the DTM. This means that the entire algorithm takes polynomially many steps in the input size.
- Thus, we have a deterministic algorithm for PATH decidable in polynomial time.
- Consequently, $PATH$ is in P, that is, $PATH \in P$.

P, NP, NP-Hard and NP-Completeness

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

The Class NP

- $NP = \{L | L$ is a language or problem decided by a non-deterministic Turing machine (NTM) in polynomial time $\}$. In other words, $NP = \cup_{k \in \mathcal{N}} NTIME(n^k)$, where $\mathcal{N}$ is the set of natural numbers.
- NP is the class of languages that can be verified quickly on a DTM.
- P is the set of all decision problems solvable by non-deterministic algorithms in poly-time.
- Is it practical? Not known till date.

The Class NP (Continued...)

Verifier of a language L:

Definition

Let L be a language (problem). A verifier for L is a DTM V such that $L = \{\alpha | \langle \alpha, \beta \rangle$ is accepted by V for some string $\beta\}$.

In other words, for every $\alpha \in L$ there exists a certificate β such that V accepts $\langle \alpha, \beta \rangle$. That is, every $\alpha \in L$ is a certificate (prescription) for membership.
If V runs in poly-time (in $|\alpha|, |\alpha|$ is the length of α), then V is called poly-time verifier for L.

The Class NP (Continued...)

Theorem

$L \in NP$ if and only if L has a poly-time verifier.

To prove a problem L is in NP, we should have any one of the following methods:

1. A poly-time non-deterministic algorithm (NTM) for L.
2. A certificate (i.e., verifier) for L, that is, a poly-time verifier for L.

P, NP, NP-Hard and NP-Completeness

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

CLIQUE in a graph

- A k-clique G' in an undirected graph $G = (V, E)$ is a subgraph G' of G isomorphic to the complete graph K_k.
- A maximal subgraph of an undirected graph $G = (V, E)$ is a clique.
- If the size of the clique is k (the number of vertices in $G' \subseteq G$), then it is known as k-clique.

CLIQUE in a graph (Continued...)

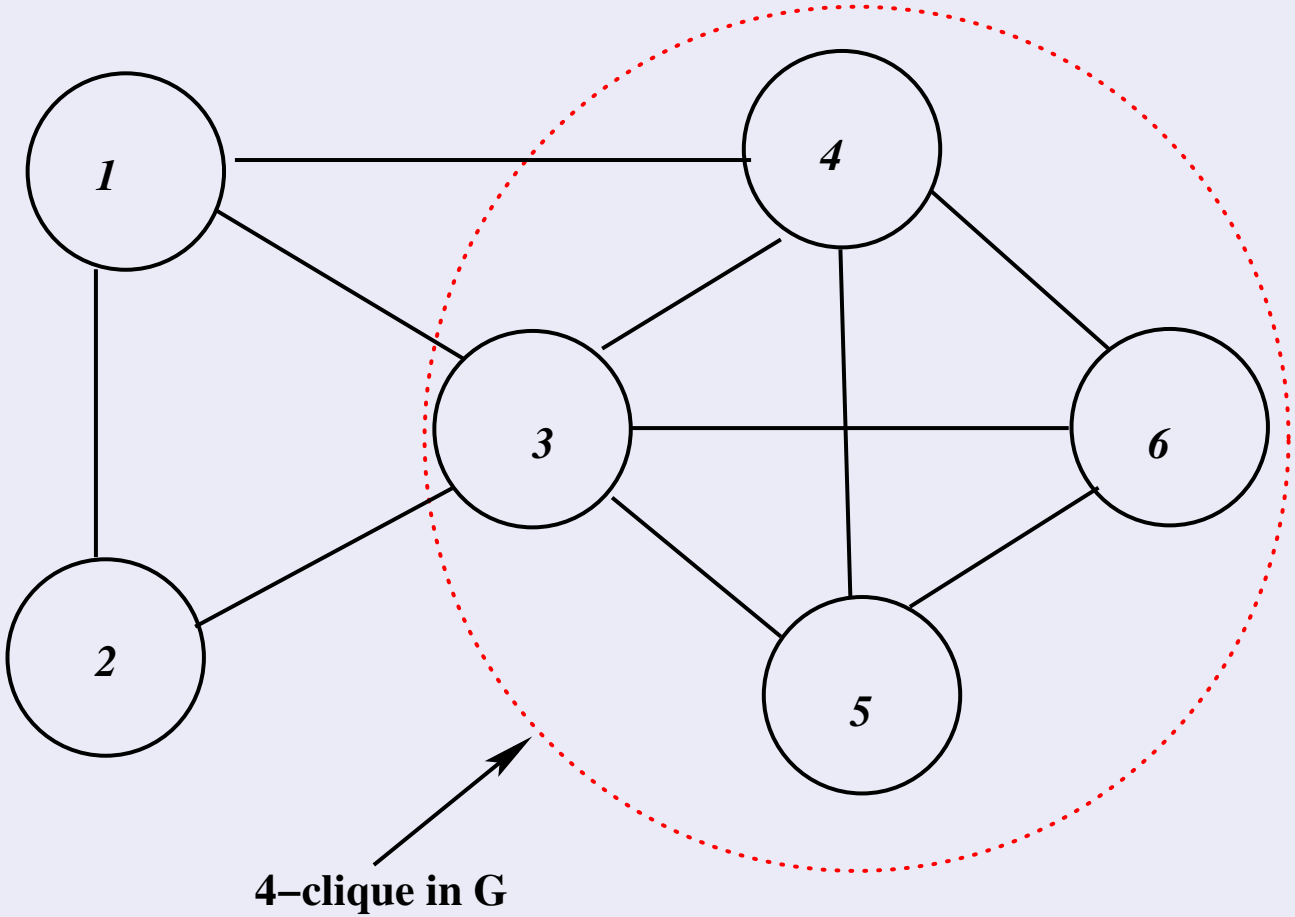


Figure: A 4-clique in a graph G.

CLIQUE in a graph (Continued...)

Consider the following problem:
$CLIQUE := \{\langle G, k\rangle | G$ is an undirected graph with a k-clique, where $k > 0$ is an integer constant $\}$.

Theorem

CLIQUE is in NP.

Proof: **Poly-time verifier for CLIQUE**
Input: $\langle\langle G, k\rangle, T\rangle$, where T is the certificate for $\langle G, k\rangle$, that is, T is the subset of $G = (V, E)$.
Output: Accept, if T is k-clique in G; reject, otherwise.
Stages:

1. If T does not contain k nodes, then "reject".
2. IF G does not contain an edge (u, v) with $u, v \in T$, then "reject".
3. Otherwise, "accept".

CLIQUE in a graph (Continued...)

Time complexity analysis:

- Stage 1 runs in $O(k)$ time.
- Stage 2 requires at most $k(k-1)/2 = O(k^2)$ time to check whether T is a complete graph or not.
- Thus, T is the verifier for CLIQUE runs in poly-time ($O(k^2)$).
- Hence, $CLIQUE \in NP$.

P, NP, NP-Hard and NP-Completeness

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

Polynomial-time reduction

- Let L_1 and L_2 be two problems (languages).
- Problem L_1 reduces to another problem L_2 in polynomial time, denoted by $L_1 \leq_p L_2$, if and only if there is a way to solve L_1 by a deterministic poly-time algorithm and using that deterministic algorithm we can solve L_2 also in poly-time.
- This definition implies that if we have a poly-time deterministic algorithm for L_1, then we can solve L_2 in poly-time.
- Moreover, if $L_1 \leq_p L_2$ and $L_2 \leq_p L_3$, then $L_1 \leq_p L_3$. In other words, the relation (reduction function) $\leq_p$ is transitive.

NP-Hard and NP-Completeness

Definition

A problem B is called **NP-hard** if every problem $A \in NP$ reduces to B in poly-time, that is, $A \leq_p B$, for all $A \in NP$.

Definition

A problem B is called **NP-complete** if
(i) $B \in NP$, and
(ii) B is NP-hard.

Note: All NP-complete problems are NP-hard. However, all NP-hard problems may or may not be NP-complete! (why?)

P, NP, NP-Hard and NP-Completeness

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

NP-Hard and NP-Completeness

Theorem

Let A and B be two problems. If $A \leq_p B$ and $B \in P$, then $A \in P$.

Theorem

Let A and B be two problems. If $A \leq_p B$ and $B \in NP$, then $A \in NP$.

Theorem

If a problem B is NP-complete and $B \leq_p C$ for another problem C in NP, then C is NP-complete.

Example of NP-Hard and NP-Complete problem

Hamiltonian Cycle Problem

- A directed Hamiltonian cycle in a directed graph $G = (V, E)$ is a directed cycle of length $n = |V|$. So, the cycle goes through every vertex (node) exactly once and then returns back to the starting vertex.

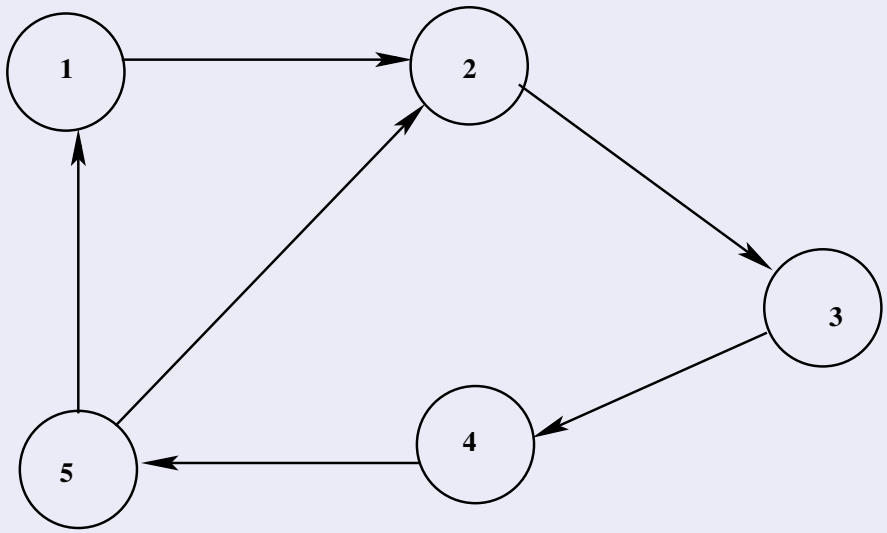


Figure: A directed graph G with a Hamiltonian cycle 1, 2, 3, 4, 5, 1.

P, NP, NP-Hard and NP-Completeness

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

Hamiltonian Cycle Problem

- Consider the following problem formally as $HAMCYCLE := \{\langle G \rangle |$ there is a Hamiltonian cycle in the directed graph $G\}$.

Theorem

HAMPATH is NP-complete.

P, NP, NP-Hard and NP-Completeness

Hamiltonian Cycle Problem

- Consider the following problem formally as $HAMCYCLE := \{\langle G \rangle |$ there is a Hamiltonian cycle in the directed graph $G\}$.

Theorem

Using the fact that if HAMPATH is NP-complete, HAMCYCLE is also NP-complete.

Proof: To show HAMCYCLE is NP-complete, we must demonstrate two things:
(1) that HAMCYCLE is in NP; and
(2) that every language $A \in NP$ is poly-time reducible to HAMCYCLE.
That is, to prove HAMCYCLE is NP-hard, we take a poly-time reduction HAMPATH $\leq_p$ HAMCYCLE.

Hamiltonian Cycle Problem

Part 1. $HAMCYCLE \in NP$

- We need to construct a poly-time NTM for HAMCYCLE.
- **NTM for HAMCYCLE:**
  Input: $\langle G \rangle$.
  Output: Accept, if there is a Hamiltonian cycle; reject, otherwise.

Stages:

1. Non-deterministically generate a sequence of $n+1$ nodes, where $n = |V|$ is the number of nodes in G, say $p_1, p_2, p_3, \ldots, p_n, p_{n+1}$ from the set $\{1, 2, 3, \ldots, n\}$.
2. If $p_1 \neq p_{n+1}$, then "reject".
3. If there is a repetition in $p_1, p_2, \ldots, p_n, p_{n+1}$, then "reject".

Hamiltonian Cycle Problem

Part 1. *HAMCYCLE* $\in$ *NP* (Continued...)

4. If for some $i = 1, 2, \ldots, n-1$, the edge (p_i, p_{i+1}) is not an edge of G, then "reject".
5. If (p_n, p_{n+1}) is not an edge of G, then "reject".
6. "Accept".

Obviously, HAMCYCLE runs in poly-time by the NTM. Hence, *HAMCYCLE* $\in$ *NP*.

Hamiltonian Cycle Problem

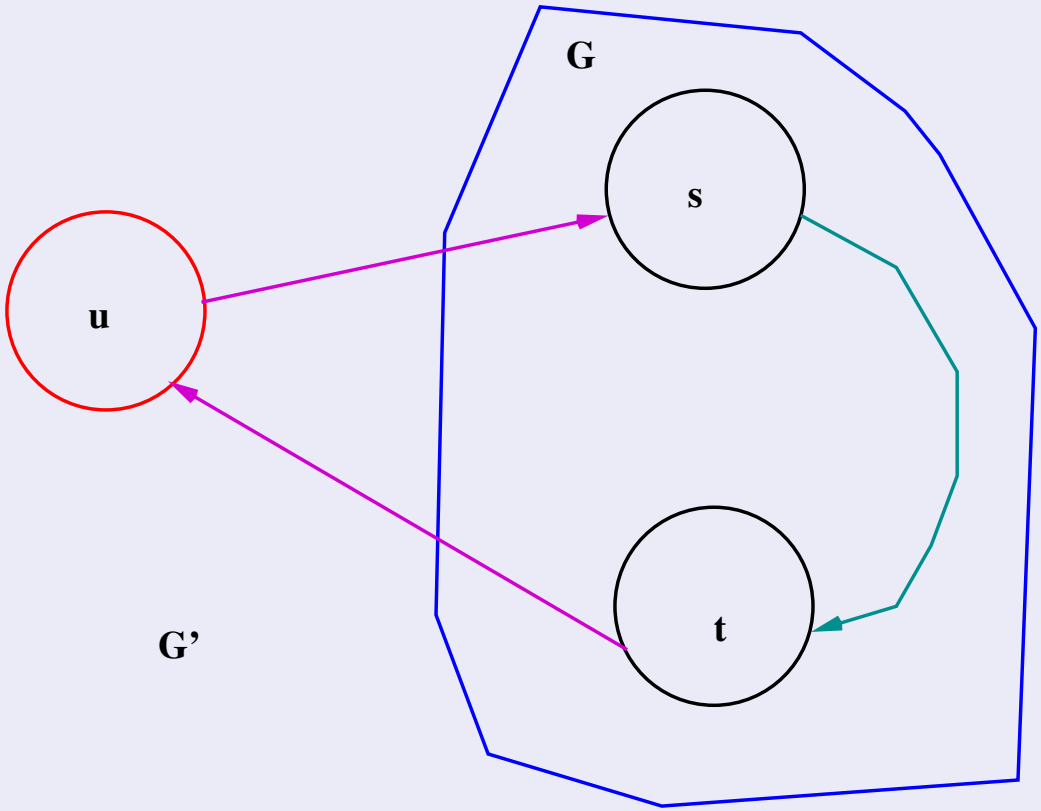
Part 2. $HAMPATH \leq_p HAMCYCLE$

- We have, $HAMPATH := \{\langle G, s, t\rangle |$ there is a (directed) Hamiltonian path from node s to node t in the directed graph $G\}$.
- Let $G = (V, E)$ be a directed graph with two vertices s and t.
- We plan to convert $\langle G, s, t\rangle$ to another directed graph $G' = (V', E')$ such that G has an (s, t)-Hamiltonian path if and only if G' has a Hamiltonian cycle.
- **Construction of** $G' = (V', E')$:
  1. $V' := V \cup \{u\}$.
  2. $E' := E \cup \{(u, s), (t, u)\}$.

Hamiltonian Cycle Problem

Part 2. *HAMPATH* $\leq_p$ *HAMCYCLE*

Construction of $G' = (V', E')$ (Continued...):



Hamiltonian Cycle Problem

Part 2. $HAMPATH \leq_p HAMCYCLE$
Construction of $G' = (V', E')$ (Continued...):

- Suppose that G has an s, t-Hamiltonian path $s, u_1, u_2, \ldots, u_m, t$. Then, $u, s, u_1, u_2, \ldots, u_m, t, u$ becomes a Hamiltonian cycle in G'.
- Conversely, let G' have a Hamiltonian cycle. If we traverse around the cycle starting from u, we must first reach s after leaving u. In order to complete the cycle we must take the edge (t, u). Between s and t the cycle visits every other node of G exactly once, that is, this cycle must be of the form $u, s, v_1, v_2, \ldots, v_m, t, u$. But then $s, v_1, v_2, \ldots, v_m, t$ is an s, t-Hamiltonian path in G.
- Clearly, this reduction runs in poly-time.

Cryptology = Cryptography + Cryptanalysis

Definition

An encryption scheme (cipher or cryptosystem) is said to be ***breakable*** if a third party, without prior knowledge of the key pair (e, d) where e is the encryption key and d is the corresponding decryption key, can systematically recover plaintext from corresponding ciphertext within some appropriate time frame.

Goal: We want this problem for an adversary (attacker) to be NP-hard (computationally infeasible).

Definition (Brute-force attack)

An encryption scheme can be broken by trying all possible keys to see which one the communicating parties are using (assuming that the class of encryption functions is public knowledge).
This is called an exhaustive search of the key space.

What is meant by "Security lies in the keys" (using brute-force attack)

Key size (bits)	**Number of alternative keys**	**Time required at 10^6 decryptions per microsecond**
32	$2^{32} = 4.3 \times 10^9$	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	10 hours
128	$2^{128} = 3.4 \times 10^{38}$	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	5.9×10^{30} years

Definition (Unconditionally secure scheme)

An encryption scheme is “unconditionally secure” if the ciphertext generated by the scheme does not contain enough information to determine uniquely the corresponding plaintext, no matter how many ciphertexts are available. That is, no matter how much time an opponent has, it is impossible for him/her to decrypt the ciphertext, simply because the required information is not there.

Definition (Computationally secure scheme)

An encryption scheme is said to be “computationally secure” if the following two criteria are met:

- The cost of breaking the cipher exceeds the value of the encrypted information.
- The time required to break the cipher exceeds the useful lifetime of the information.

Complexity Theoretic Connections

- We want **encryption and decryption** must be done in polynomial time (Class: P problems) to encrypt and decrypt the messages, respectively
- We want **cryptanalysis part** by an adversary must be computationally infeasible (Class: NP-hard problem) to break the cryptosystem