SHRIMATI INDIRA GANDHI COLLEGE

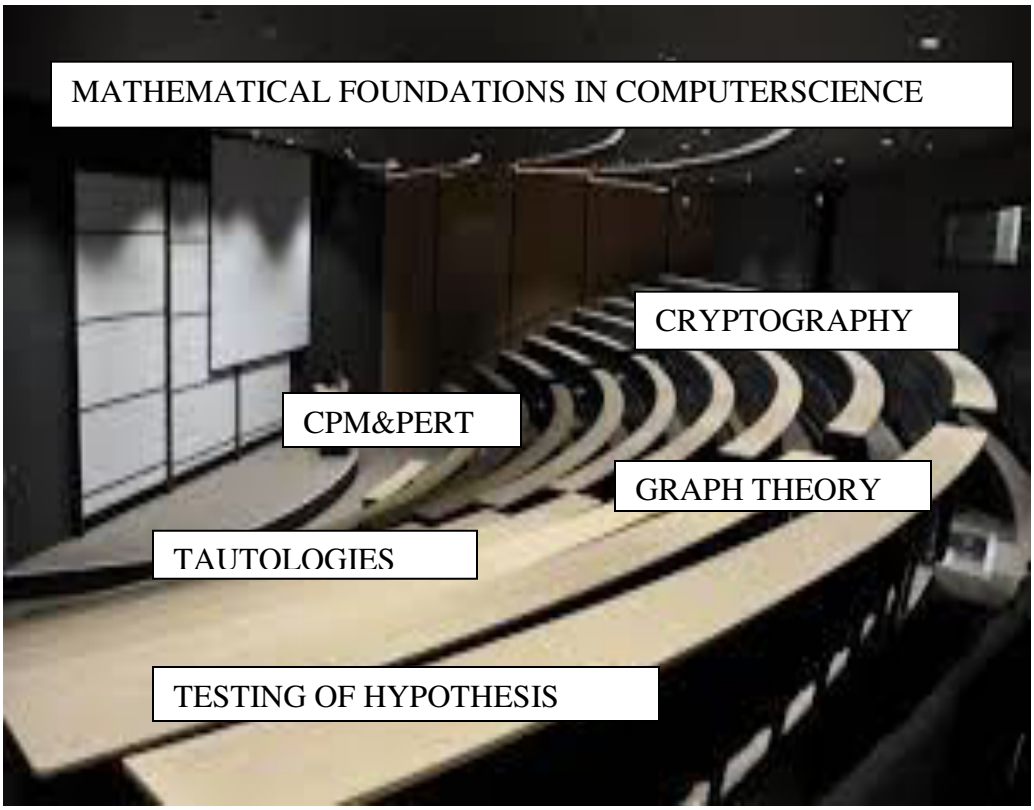
(NATIONALLY ACCREDITED AT "A" GRADE (3RDCYCLE) BY NAAC)

TIRUCHIRAPPALLI-2

INSTRUCTION MATERIAL

MATHEMATICAL FOUNDATIONS IN COMPUTER SCIENCE

DEPARTMENT OF COMPUTER SCIENCE


MATHEMATICAL FOUNDATIONS IN COMPUTERSCIENCE
CRYPTOGRAPHY
CPM&PERT
GRAPH THEORY
TAUTOLOGIES
TESTING OF HYPOTHESIS

CORE COURSE – I(P16CS11)
MATHEMATICAL FOUNDATIONS FOR COMPUTER SCIENCE

Unit I

Propositions - evaluation - precedence rules -tautologies - reasoning using equivalence transformation - laws of equivalence - substitution rules - a natural deduction system. Deductive proofs - inference rules - proofs - sub proofs.

Unit II

Introduction - Cryptography – Ceaser Cyphor Coding - Matrix encoding - scrambled codes - Hamming metric - Hamming distance - Error detecting capability of an encoding.

Unit III

Assignment problem and its solution by Hungarian method. Project Scheduling by PERT - CPM: Phases of project scheduling - Arrow diagram - Critical path method - Probability and Cost Considerations in project scheduling - Crahing of Networks.

Unit IV

Testing of hypothesis : Tests based on normal population - Applications of chi-square, Student's-t, F-distributions - chi-square Test - goodness of fit - Test based on mean, means, variance, correlation and regression of coefficients.

Unit V

Graph - Directed and undirected graphs - Subgraphs - Chains, Circuits, Paths, Cycles - Connectivity - Relations to partial ordering - adjacency and incidence matrices - Minimal paths - Elements of transport network - Trees - Applications.

Text Books

1. "The Science of Programming", David Gries. Narosa Publishing House, New Delhi, 1993.
2. "Application Oriented Algebra", James L. Fisher, Dun Donnelly Publisher, 1977.

3. "Operation Research - An Introduction", Hamdy A.Taha, Macmillan Publishing Co., 4th Edn., 1987.
4. "Fundamentals of Mathematical Statistics", Gupta,S.C. and V.K.Kapoor, Sultan Chand & Sons, New Delhi, 8th Edn., 1983.
5. "Fundamentals of Applied Statistics", Gupta.S.C. and V.K.Kapoor, Sultan Chand & Sons, New Delhi, 2nd Edn., 1978.

References

1. "Discrete Mathematics", Seymour Lipschutz and Marc Laris Lipson, Second edition, Schuam's Outlines by Tata McGraw- Hill publishing Company Limited, New Delhi 1999.
2. "Operations Research", Kanti Swarup, P.K.Gupta and Man Mohan, Sultan Chand & Sons, New Delhi, 1994.
3. "Introductory Mathematical Statistics", Erwin Kryszig, John Wiley & Sons, New York, 1990.
4. "Probability and Statistics Engineering and Computer Science", Milton, J.S. and J.C.Arnold, McGraw Hill, New Delhi, 1986.

UNIT-1

1. **What is mean by connectives?**

   The simple statements initially are called atomic or primary statement. The new statement can be formed from atomic statement through the use of sentential connectives.

2. **What is negation?**

   The negation of a statement is generally formed by introducing the word "Not" at a proper place in the statement are by prefixing the statement with the phrase "it is not the case that".

3. **What is conjunction?**

   It is a two statement P and Q is the statement PˆQ which is read as "P and Q" the

   statement P and Q has the truth table T whenever both P and Q have the truth value T.

   otherwise it has the truth value F.

4. **What is disjunction?**

   It has two statements P and Q is the statement PˇQ which is read as "P or Q" the

   statement P or Q has the truth value F only when both P and Q have the truth value F.

   otherwise it is true.

5. **Define conditional statement.**

   If P and Q are any two statements then the statements P->Q which is read as "if P then Q" is called a conditional statement.

6. **What is antecedent and consequent?**

   The statement P is called the antecedent and Q the consequent in P->Q again according

   to the definition it is not necessary that there be any kind of relation P and Q in order to form

   P->Q.

7. **Express in English the statement P->Q where**

   P: The sun is shining today

   Q: 2+7>4

   The sun is shining today then 2+7>4

8. Write in symbolic form: the crop will be destroyed if there is a flood.

The proper sentence is: if there is a flood then the crop will be destroyed.

C: crop destroy

F: there is a flood.

Symbolic form: F->C

9. What is biconditional statement?

If P and Q are any two statements the statement P->Q which is read as "P if and only if Q" is called biconditional statement. <-

10. Define well formed formulae

A statement formula is not a statement however a statement can be obtained from it by replacing the variable by statement.

11. What are the rules of well formed formulae?

- a statement variable standing alone is a well formed formulae
- if A is a well formed formulae the ¬A is a well formed formulae
- if A and B are well formed formulae, then (A^B),(AˇB).A->B are well formed formulae
- a string of symbols containing a statement variables, connectives and parenthesis is a well formed formulae. If and only if it can be obtained by finitely many applications of the rules 1, 2 and 3.

12. Define tautology.

A statement formula which is true, regardless of the truth values of the statement which replaces the variable in it is called a universally valid formula or a tautology or a logical truth.

13. What is contradiction?

A statement formula which is false, regardless of the truth values of the statement which replaces the variables in it is called a contradiction the negation of a contradiction is a tautology.

14. State equivalence of formulae.

If the truth value of A=B for every one of the 2ˆ possible truth values assigned to P1, P2,…Pn then A and B are equivalent statement.

15. Verify whether (P ˇQ) -> P is a tautology

P	Q	PˇQ	PˇQ->P
T	T	T	T
T	F	T	T
F	T	F	F
F	F	F	T

The given formula is not a tautology.

16. What is mean by a deduction?

When a conclusion is derived from a set of premises by using the accepted rules of reasoning, then such a process of derivation is called a deduction.

17. What is universal specification in theory of inference?

If the statement of the form [(¥x)p(x)] is assumed to be true when the universal quantifiers may be dropped from the statement to obtain p(x) which is true for each object in the universe, or to obtain p(b) which is true for a specific object of the universe.

18. What is universal generalization in theory of inference?

If a statement p(x) is true for an arbitary x of the universre then the universal quantifier may be prefixed to obtain [(¥x)p(x)] , provided that every existential object in p(x) which depends on x is covered by a quantifier.

19. What is mean by quantifiers?

An analysis of mathematical sentences involving quantifiers indicates that the main two quantifiers are “all” and “some”, where “some is interpreted to mean atleast “one”.

Certain statement involve words that indicate quantity such as “all, some, none,one”. To answer the question “how many?” such words indicate quantity they are called quantifiers.

20. Define truth table technique.

The method to determine whether the conclusion logically follows from the given premises by constructing the relevant truth table is called truth table technique.

21. Define open Statements.

An open statement is a declarative sentence which contains one or more symbols, is not a true false statement, produces a TF statements when each of its symbols is replaced by a specific objects from a designated set.

Eg. P(x): The rational number x is greater than 100

Q(x,y): x+y=10 where x&y are integers then

P(x)&Q(x,y): The rational number x is greater than 100 and x+y=10

22. What is meant by assignment problems?

The problem of assigning 'n'-jobs to 'n' persons such that each job can be assigned to only one persons & each person can be allotted only one job. Suppose all the 'n' persons are capable of doing any one the 'n' jobs but the time taken by them to perform the jobs varies from job to job. The following table represents the time taken by the 'n' persons to perform the 'n' jobs.

22. Define Prohibited Assignments.

Some times due to certain reason, a particular resource way a man or machine, cannot be assigned to perform a particular activity way territory or jobs. In such cases the cost of performing that particular activity by a particular resource in as to prohibit the entry of this pair of resource activity into the final solution.

23. Define Network analysis.

Network analysis is a technique which determines the various sequences of jobs concerning a project & project completion time. Network analysis has been successfully used to a wide range of significant management problems.

23. What are the types of Network analysis?

Two methods of network analysis are

1. Critical Path Method(CPM)
2. The Program Evalution & Review Technique(PERT)

24. Define Total Float(TF).

This is the amount of time a path of activities could be delayed without affecting the overall project duration.

TF=Head event(L)-Tail event(E)-Duration

Total float is also defined as the difference between the two finishing time or the difference between the starting time

TF=LFT-EFT
=LST-EST

25. Define Free Float (FF).

This is the amount of time an activity can be delayed without affecting the commencement of a subsequent activity of its earliest start time, but may affect the float of a previous activity.

FF=Headevent(L)-TailEvent(E)-Duration

FF=TF-Head event slack(L-E)

26. Define PERT.

A PERT Network diagram is drawn. In the same way as a CPM Network. In this we need to estimate the duration of each activity.

27. What are the three types of PERT?

There are three types of time estimates

1. The OptimisticTime(To)
2. The Pessimistic Time(Tp)
3. The most likely time estimate (Tm)

28. Define Optimistic (To).

It is the shortest possible time estimate for finishing and activities the chance of occurance of this is very small.

29. Define Pessimistic Time (Tp).

It is the longest time conceivable for an activity. The chance of this occurance is also very small.

30. Define the most likely time estimate(Tm).

This is the time estimate to be executed under normal conditions of the activity. This is a reasonable time to estimate.

31. What are the rules for drawing an arrow diagram?

The rules for the construction for network are of 2 types namely

1. Logic rules
2. Computer rules

32. Define Logic rules.

- Before an activity begins, all activities preceding it must be completed.
- Arrows indicate logical precedence only. Neither the length of the arrow nor its direction have any significance.
- Loop formation should not occur on a network

33. Define Computer rules.

- Event number must not be repeated on a network.
- Any two events may be directly connected by not more than one activity
- The network should have only one initial & one terminal event.

34. Define Dangling.

- It is a case where activities other than the final activity do not have any successor activity.
- To overcome this situation it is to be remembered that all events other than the initial & terminal event of the network must have at least one entering & one leaving activity.

35. Define Network.

It is a graphical representation of a project plan showing interrelationship of the various activities.

36. Define Project.

It is a combination of interrelated activities all of which must be executed in a certain order to achieve a set goal.

37. Define Activity.

An activity is any portion of a project which consumes time or resources & has a definable beginning & ending.

38. Define Events.

The beginning & ending points of activities are called events an event is an instantaneous pointing time. The events are shown by nodes.

39. What are the types of events?

There are three types of events

- Burst event
- Merge event

- Dual event

40. Define Burst event.

If an event represents the joint initiation of more than one activity, it is called a burst event.

41. Define Merge event.

If an event represents the joint completion of more than one activity, it is called a merge event.

42. Define Dual event.

If an event represents the join completion of more than one activity & also the join initiation of more than one activity. It is called a dual event.

43. What is crashing the program of a project?

For carrying any project effectively all the activities should be carried on without any omission. So extra workers could be appointed & the time schedule may be increased.

Proportionally the cost will also increase co-coordinative approach to maintain considerative cost,time& deployment of personnel's to maintain quality of a project is called crashing the program of a project.

44. Define Cost slope.

Cost slope is defined as crashed cost, normal cost/normal time-crashed time is alse equal to the direct cost per unit of time.

45. What is for crashing the programs & time cost optimization procedure?

The process of shortening a project is called crashing activity project.

46. Define Critical Path.

Find the normal critical path & identify the critical activities.

47. Define Ranking.

Rank the activities in ascending order of the slope.

48. Define Crashing.

Crash the activities in the critical path as per the ranking (ie) activity having lower cost slope could be crashed first to the maximum extent

possible. Calculate the new direct cost by cumulatively adding the cost of crashing to the normal cost.

49. What is Parallel crashing?

As the critical path duration is reduced by the crashing in step3, other path also becomes critical (ie) We get parallel critical path. This means that project duration can be reduced by simultaneous crashing of activities in the parallel circuit paths.

50. What is optimal duration?

Crashing as per step3 & step4, the optimal project duration is determined. It would be the time duration corresponding to which the total cost is a minimum.

51. What is Probability?

When an experiment is conducted, the set of all possible outcomes is called sample space($\sum$).The set of all required events is called event space(E).Therefore the probability of any event A is defined as,
$P(A)=n(E)/n(\sum)$ such that, $o\leq P(A) \leq 1$

52. What is Probability distribution function?

Let 'X' be a continuous random variable then F(x) is defined as a Probability distribution function,such that $F(X)=0\leq P(x) \leq 1$ and $-\infty\leq X\leq\infty$.

53. What is Probability density function?

Let 'X' be a continuous random variable then the Probability density function is defined as $f(x)=F'(x)[\ -\infty\leq X\leq\infty]$. Example:$f(x)=e^{-x}(-\infty\leq x\leq\infty)$.

54. Write properties of pdf.

i)$f(x)\geq 0$

ii) $\int_{-\infty}^{\infty}$

55. Explain Sampling.

"A sample must be representative" means that the sample drawn must contains all the properties and possess all the characteristics of the parent population in the same proportion as the parent population contains and possesses.

56. What are all the classification of sampling method.

It can be classified into two types namely random and Non random sampling.

57. Write any two properties of t-distribution.

1) The variable t-distribution ranges from $-\infty$ to $+\infty$.

2) The constant c is actually a function γ.so that, for a particular value of γ ,the distribution of f(t) is completely specified. Thus f(t) is a family of function one for each value of γ.

58. Describe Applications of the t-distribution.

The following are some of the examples to illustrate the way in which the student distribution is generally used to test the significance of the various results obtained from small samples.

To test the significance of the mean of a random sample

Testing difference between mean of two samples(Independent samples)

59. Define t-statistic.

The t-statistic is defined as $t=|\bar{x} - \mu|/s * \sqrt{n}$

Where $s=\sqrt{\sum d^2-n(d)^2/n-1}$

Where $\bar{x}$ is population mean

μ is sample mean

n is sample size

d is deviation from assumed mean of every sample value.

's' is standard deviation

60. Define Chi-square test.

If O_i (i=1,2..n) is a set of observed (experimental) frequencies of a data and E_i (i=1,2..n) is the corresponding set of expected (theoretical or hypothetical) frequencies, then the value of x^2 is defined as,

$$x^2 = \sum (O_i - E_i)^2 / E_i$$

61. Define Graph or Linear Graph.

A Linear Graph or simply a graph G=[V,E] consist of a set V=(V1,V2…) called vertices and another set E=(e1,e2….) called edjes,such that each edge e_k is identified with an unordered pair of vertices V_i,V_j.

62. Define Self loop or loop.

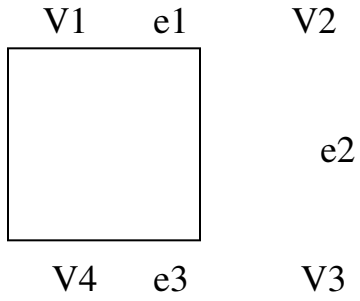
An edge having the same vertex for both of its end vertices is called a loop or self loop. In graph G e1 is the self loop.

63. Define Parallel edges.

If more than one edge have the same pair of end vertices, then those edges called parallel edges. In graph G e4 and e5 are parallel edges.

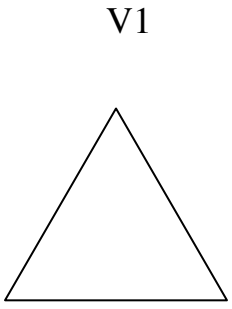
64. Define simple graph.

If a graph has neither self-loop nor parallel edges then it is called a simple graph.



65. Define finite graph.

A graph with a finite number of vertices and finite number of edge is called a finite graph. Otherwise it is an infinitegraph.



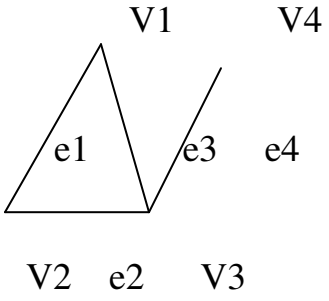
FINITE GRAPH

66. Define Incident relation.

If V_i is an end vertex of some edge e_j, then V_j and e_j said to be incident with ('on' or 'to') each other.

67. Define adjacent relation.

Two vertices are said to be adjacent, if they are the end vertices of the same edge.

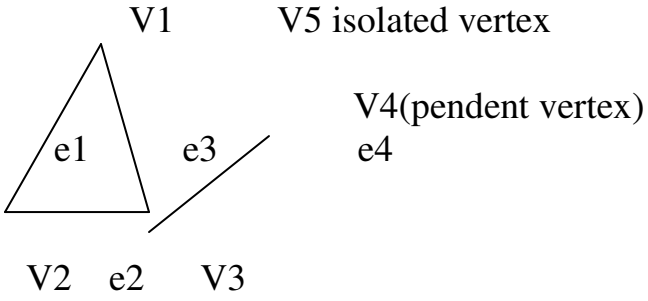


V1, V2 are adjacent vertices V1, V4 are not adjacent vertices

68. Define Isolated vertex.

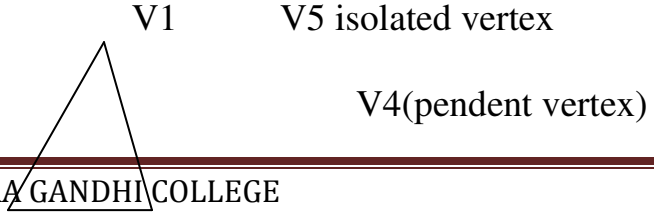
A vertex having no incident edge is called an Isolated vertex.

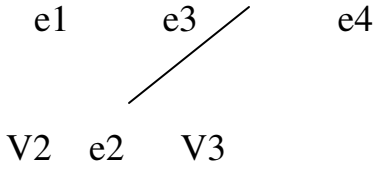
In the below graph V5 is an isolated vertex.



69. Define Pendent Vertex.

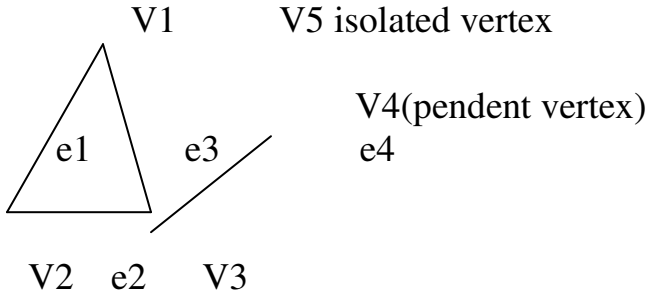
A vertex of degree 1(only one edge) is called a Pendent Vertex or end vertex. In the figure V4 is a Pendent Vertex





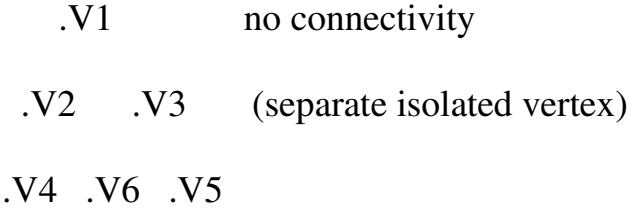
70. Define series.

Two adjacent edges are said to be in series if their common vertex os of degree 2.In the graph below e1, e2, e3, e4 are in series.



71. Define Null graph.

If G= (V, E) where the edge set E=Φ (empty) then G is called a null graph.



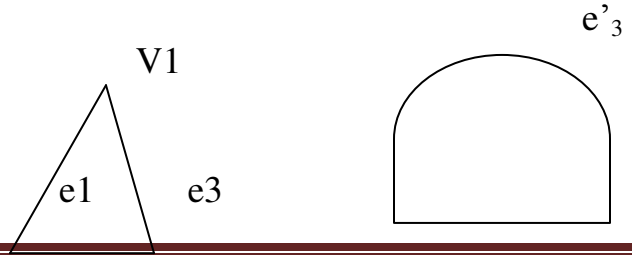
72. Define Regular graph.

A graph in which all vertices are of equal degree is called a regular graph.

73. Define Isomorphism.

Two graph G and G' are said to be isomorphic if there is a one-to-one correspondence between their vertices and their edges, such that the incidence relationship is preserved.

Example,



V'_1 e'_1 V'_2 e'_2 V'_3

V2 e2 V3

Φ1: G-> G' Φ2:G->G'

V1-> V'_1 e1-> e'_1

V2-> V'_2 e2-> e'_2

V3-> V'_3 e3-> e'_3

Then G is isomorphic to G'

74. Define Eulers Line.

An open walk that includes all the edges of a graph without retracing any edge is called an Eulers line.

75. Define Unicursal graph.

A graph that has as unicursal line is called a Unicursal graph

76. Define fusion.

A pair of vertices a,b in G are said to be fused or merged or identified if the two vertices are replaced by a single new vertex such that every edge that was incident on either 'a' or 'b' on both is incident on the new vertex.

Fusion of two vertices does not not alter the number of edges,but reduces the number of vertices by 1.

77. What is component?

Let g be a disconnected graph consisting of two or more connected subgraphs,each of these connected subgraph is called a component.

78. What is a Circuit?

A closed walk in which no vertex except the terminal vertices appears more than once is called a circuit.

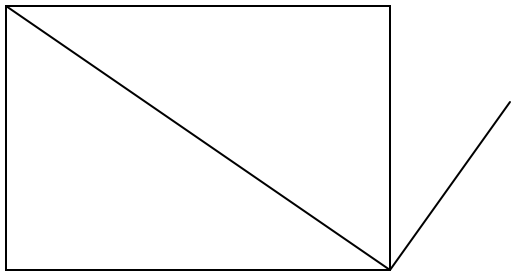
79. What is connected or disconnected graph?

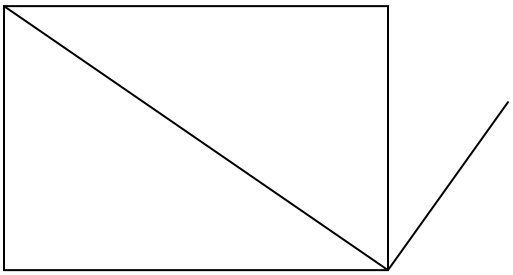
A graph G is said to be connected if there is at least one path between every pair of vertices in G.Otherwise the graph is disconnected.

G G'

V1 e1 V4 V1 e1

V4 .V6





V2 e3 V3 V2 e3

V3

G is a connected graph G' is a dis connected graph

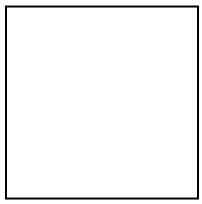
80. Define Length of a path.

The number of edges in a path is called the length of a path.

81. Define Walk.

A walk is a finite alternative sequence of vertices and edges beginning and ending with vertices such that each edge is incident with the vertices preceeding and following it.

V1 e1 V2



e2

V4 e3 V3

V1e1V2e2V3e3V4e4 is a walk

82. Define Critical activity

Critical activity the activity, which can not be delayed without delaying the project duration, is known as critical activity

83. Differentiate between supercritical and subcritical activities

Supercritical an activity having negative float is known as supercritical activity.

Subcritical An activity having positive float is known as subcritical activity This activity may be delayed without any delay m the project

84. Differentiate between slack and float

Slack It is the time by which occurrence of an event can be delayed

85. Enlist four types of floats used in network analysis.

(a) Total float.

(b) Free float

(c) Independent float

(d) Interfering float

86. Define Free Float, Independent float, Interfering float as used in PERT chart.

Freefloat : Portion of the total float within which an activity can be manipulated without affecting the floats of subsequent activities.

Independent float: Portion of the total float within which an activity can be delayed without affecting the floats of proceeding activities.

Interfering float : It is equal to the difference between the total float and the free float of the activity.

87. What do you mean by dummy activity?

Dummy activity : An activity, which only determines the dependency of one activity on the other, but does not consume any time, is called a dummy activity.

88. Define dummy arrow used in network.

Dummy arrow: It represent the dummy activity in the network. It only represents the dependency of one activity on the other. It is denoted by dash/dotted line.

89. Define dangling and looping in net-work models.

Dangling : The disconnection of an activity before the completion of all the activities in a network diagram is known as dangling.

Looping (cycling) : Looping error is also known as cycling error in a network diagram. Drawing an endless loop in a network is known as error of looping.

90. Differentiate between event and activity.

Event: The beginning and end points of an activity are called events or nodes. Event is a point in time and does not consume any resources.

Activity : It is physically identifiable part of a project which require time and resources for its execution. An activity is represented by an arrow, the tail of which represents the start and the head, finish of the activity.

91. Define (i) Network (ii) Path terms used in network.

(i) Network: It is the graphical representation of logically and sequentially connected arrows and nodes representing activities and events of a project.

(ii) Path: An unbroken chain of activity, arrows connecting the initial event to some other event is called path.

92. Differentiate between CPM and PERT.

CPM.

1. CPM is activity oriented i.e., CPM network is built on the basis of activities.

2. CPM is a deterministic model. It does not take into account in uncertainties involved in the estimation of time.

3.

CPM places dual emphasis on project time as well as cost and finds the trade off. between project time and project cost.

4. CPM is primarily used for projects which are repetitive in nature and comparatively small in size.

PERT

1.PERT is event oriented.

2.PERT is a probabilitic model.

3.PERT is primarily concerned with time only.

4. PERT is used for large one time reserach and development type of projects.

93. Define crashing in network mode1s.

Ans. Crashing: The deliberatic reduction of activity normal time by puffing an extra effort is called crashing. The crashing is being done by allocating more manpower or by subcontracts.

94. Differen**tiate between:**

(i) Crash project time and optimum project time

(ii) Normal cost and crash cost.

(i) Crash project time: It is the minimum time by which the prOject may be completed.

Optimum project time: The time corresponding to minimum project cost for completion of the project is known as optimum project time.

(ii) Normal cost : The cost associated when the project completed with normal time

Crash cost: The cost associated when the project completed with crash time is known as crash cost.

95.Define Assignment problem .

Assignment problem is one of the special cases of transportation problems. The goal of the assignment problem is to minimize the cost or time of completing a number of jobs by a number of persons. An important characteristic of the assignment problem is the number of sources is equal to the number of destinations .It is explained in the following way.

1. Only one job is assigned to person.
2. Each person is assigned with exactly one job.

Management has faced with problems whose structures are identical with assignment problems.

Ex: A manager has five persons for five separate jobs and the cost of assigning each job to each person is given. His goal is to assign one and only job to each person in such a way that the total cost of assignment is minimized.

96. Define Balanced assignment problem.

This is an assignment where the number of persons is equal to the number of jobs.

97. Define Unbalanced assignment problem.

This is the case of assignment problem where the number of persons is not equal to the number of jobs. A dummy variable, either for a person or job (as it required) is introduced with zero cost or time to make it a balanced one.

98. Define an infeasible Assignment: Infeasible assignment occurs when a person is incapable of doing certain job or a specific job cannot be performed on a particular machine. These restrictions should be taken in to account when finding the solutions for the assignment problem to avoid infeasible assignment.

99. Define Hungarian method.

Assignment problems can be formulated with techniques of linear programming and transportation problems. As it has a special structure, it is solved by the special method called Hungarian method. This method of assignment problem was developed by a Hungarian mathematician D. Konig and is therefore known as Hungarian method of assignment problem.

100. What is Binary Tree?

Binary tree is a tree in which there is exactly one vertex of degree 2 and each of the remaining vertices is of degree one or three.

101. Definition of Symmetric-Key Encryption Scheme.

A symmetric-key encryption scheme with key-length k, plain-text length m and ciphertext-length c is is a pair of probabilistic algorithms (Enc,Dec), such that Enc : $\{0,1\}^k$ X $\{0,1\}^m$ -> $\{0,1\}^c$,
Dec : $\{0,1\}^k$ X $\{0,1\}^m$ -> $\{0,1\}^c$,and for every key K 2 f0; 1gk and every message M, Dec(K;Enc(K;M)) = M] = 1 (2.1) where the probability is taken over the randomness of the algorithms

1. **Construct the truth table for P V ¬ Q**

P	Q	¬Q	PV¬Q
T	T	F	T
T	F	T	T

F	T	F	F
F	F	T	T

2. Construct the truth table for the formula ¬ (P V ¬ Q)⇄(¬PV¬Q)

P	Q	PVQ	¬(P V ¬ Q)	¬P	¬Q	¬PV¬Q	¬(P)
T	T	T	F	F	F	F	T
T	T	F	T	F	T	T	T
F	T	F	T	T	F	T	T
F	F	F	T	T	T	T	T

3. Construct the truth table for the formula (P→Q) ^ (Q→R) →(P→R)

P	Q	R	P→Q	Q→R	(P→Q) ^(Q→R)	P→R	^	→
T	T	T	T	T	T	T	T	T
T	T	F	T	F	F	F	F	T
T	F	T	F	T	F	T	T	T
T	F	F	F	T	F	F	F	T
F	T	T	T	T	T	T	T	T
F	T	F	T	F	F	T	F	T
F	F	T	T	T	T	T	T	T
F	F	F	T	T	T	T	T	T

4. Construct the truth table for the formula (P⇄Q) ⇄(P^Q) V(¬P^¬Q)

P	Q	P⇄Q	P^Q	¬P	¬Q	(P⇄Q)	⇄(P^Q)	¬P^¬Q	V	V
T	T	T	T	F	F		T	F	T	T
T	F	F	F	F	T		T	F	F	T
F	T	F	F	T	F		T	F	F	T
F	F	T	F	T	T		F	T	T	T

5. Define well formed formulae

A statement formula is not a statement however a statement can be obtained from it by replacing the variable by statement

A statement formula is an expression which is a string, consisting of variables, parenthesis and connective symbols.

Not every string of these symbols is a formula. A well formed formula can be generated by the following rules:

1. A statement variable standing alone is a well formed formula
2. If A is a well formed formula then ¬A is a well formed formula
3. If A and B are well formed formula then (A^B),(AVB),A→B,A⇄B are well formed formula
4. A string of symbols containing a statement variable, connectives and parenthesis is a well formed formula. If and only if it can be obtained by

finitely many applications of the rules 1,2 and 3 from the above the following are a well formed formula

¬(P^Q) , ¬ (PVQ) ,(P→(PVQ)

(P→(Q→R)),((P→Q) ^(Q→R) ⇄(P→R))

The following are not well formed formula

1. ¬P^Q – obviously 'P' and 'Q' are well formed formula. a well formed formula would be either (¬P^Q) or ¬(P^Q)
2. (P→Q) →(^Q)-This is not a well formed formula
3. (P→Q – Note that (P→Q) is a well formed formula
4. (P^Q) →Q – the reason for this not being well formed formula in that one of the parenthesis is missing.((P^Q) →Q) is a well formed formula while (P^Q) →Q not a well formed formula

6. Verify whether (PVQ) →P is a tautology

P	Q	PVQ	PVQ→P
T	T	T	T
T	F	T	T
F	T	T	F
F	F	F	T

7. State that QV (P^¬Q) V (¬P^¬Q) is a tautology

P	Q	¬Q	P^¬Q	Q V (P^¬Q)	¬P	¬P^¬Q	( )
T	T	F	F	T	F	F	T

T	F	T	T	T	F	F	T
F	T	F	T	T	T	F	T
F	F	F	F	F	T	T	T

8. Define theory of inference

If an implication A→B is a tautology where A and B are statement formula ,we say that 'B logically follows from A' or 'B is a valid conclusion (consequence) of the premise (the assumption or hypothesis)A'.We say that from a set of premises { H1,H2..Hn}.A Conclusion C follows logically if { H1^H2^..Hn}=>C. When C logically follows from H1,H2..Hn we write

H1
H2
.
.
Hn
___
c
___

Or (H1….Hn)=> C
Or (if H1,H2…Hn then C)
This mean that if we know H1 is true,H2 is true..Hn is true,then we can conclude that C is true.

9. State that (¬P^(¬Q^R)) V (Q^R) V (P^R) <=> R

(¬P^(¬Q^R)) V (Q^R) V (P^R)

<=> ((¬P^¬Q) ^ R) V (Q^R) V (P^R)

<=> (¬(PVQ) ^ R) V ((Q^ R) V (P^ R))

<=> (¬(PVQ) ^ R) V ((Q^ P^ R)

<=> (¬(PVQ) V (P^ Q))^ R)

<=> T^ R

<=> R as (¬^R<=>R)

10. Demonstrate that S is a valid inference from the premises P→¬Q,QVR, ¬S→P and ¬ R

Solution:

[1]	(1)	QVR	premises (or) hypothesis
[2]	(3)	Q	(1), (2) and tautology
[4]	(2)	¬ R	premises (or) hypothesis
[1,2]	4)	P→¬Q	premises (or) hypothesis
[1,2,4]		(5) ¬ P	(3), (4) and tautology
[6]	(6)	¬ S→P	premises (or) hypothesis
[1,2,4,6]	(7)	S	(5), (6) and tautology

Hence S is a valid inference

Condensed form

[1]	(1)	QVR	P
[2]	(2)	¬ R	P
[1,2]	(3)	Q	T,(1),(2)& I_{10}
[4]	(4)	P→¬Q	P
[1,2,4]		(5) ¬ P	T,(3),(4)& I_{12}
[6]	(6)	¬ S→PP	
[1,2,4,6]	(7)	S	T,(2),(6)& I_{12}

11. State that R→S can be derived from the premises P→(Q→S), ¬RVP and Q.

It is enough to include R as an additional premise and derive S.

[1]	(1)	¬RVP	P
[2]	(2)	R	additional premise
[1,2]	(3)	P	T,(1),(2)& I_{10}
[4]	(4)	P→(Q→S)	P
[1,2,4]	(5)	Q→S	T,(3),(4)& I_{11}
[6]	(6)	Q	P
[1,2,4,6]	(7)	S	T,(5),(6)& I_{11}
[8]	(8)	R→S	conditional proof

12. Using indirect method of proof derive P→¬S from P→QVR, Q→¬P,S→¬R,P

RESULT IS P→¬S

¬(P→¬S)=> ¬(¬PV¬S)

= P^S (additional premise)

[1]	(1)	P→QVR	P
[2]	(2)	P	P
[1,2]	(3)	QVR	T,(1),(2)& I_{11}
[4]	(4)	S→¬R	P
[5]	(5)	P^S	P new premise
[5]	(6)	S	T(5) simplification
[4,5]	(7)	¬R	T(4,6) M.P

[1,2,4,5]	(8)	Q	$T(3,7,I_{10})$
[9]	(9)	Q→¬P	P
[1,2,4,5,9]	(10)	¬P	T(8,9,M.P)
[1,2,4,5,9]	(11)	P^¬P	T(2,10) Contradiction

So ¬(P^S) is derivable from P→QVR, Q→¬P,S→¬R,P

13. Briefly explain resource allocations. Distinguish between Resource smoothing and Resource levelling.

The resource allocation procedure mainly consists of two activities:

(i) Resource smoothing

(ii) Resource levelling.

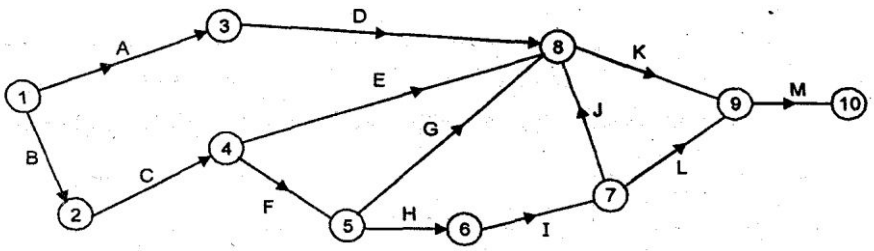
Resource smoothing If the project duration could not be changed then the resource allocation only smoothen the demand on resources in order that the demand for any resource is uniform as possible The periods of maximum demand for resource are located and the activities are according to their float values are shifted for balancing the availability and requirement of resources. The intelligent utilization of floats can smoothen the demand of resources to the maximum possible extent is called resource smoothing.

Resource Levelling : There are various activities in a project demanding varying levels of resources.The demand on certain specified resources should not go beyond the prescribed level is know as resource levelling.

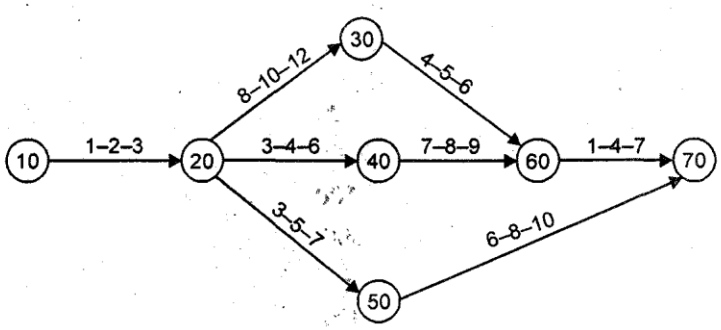
14. Construct the network for the following activity data:

Activity	Preceded by	Activity	Preceded by
A	-	-	-
B	–	H	F
C	B	I	H
D	A	J	I
E	C	K	D,E,G,J
F	C	L	I
G	F	M	K,L

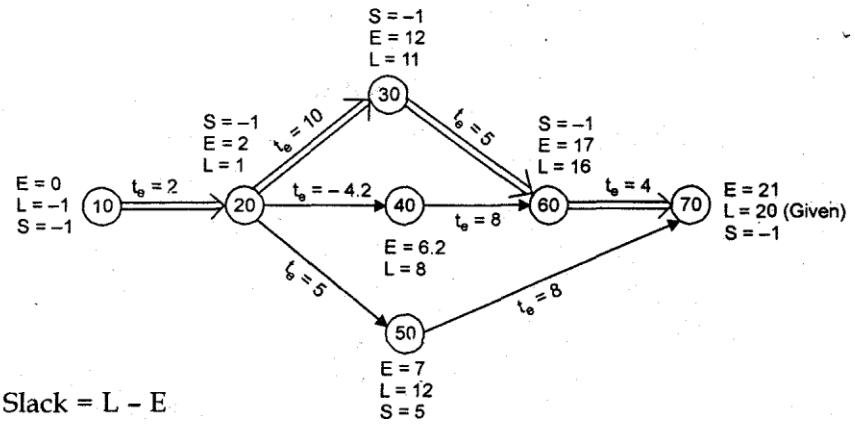
Solution. Network:



15. Consider the PERT network given in fig. Determine the float of each activity and identify the critical path if the scheduled completion time for the project is 20 weeks.



Solution.



(s) Slack = L – E

Activity	$t_e = \frac{t_0 + 4tm + t_p}{6}$	Start Time		Finish Time		Total
		E	T_{ES}	T_{EF}	L	Float
10 – 20	2	0	–1	2	1	–1
20 – 30	10	2	1	12	11	–1
20 – 40	4.2	2	3.8	6.2	8	1.8
20 – 50	5	2	7	7	12	5
30 – 60	5	12	11	17	16	–1
40 – 60	8	6.2	8	14.2	16	1.8
50 – 70	8	7	12	15	20	5
60 – 70	4	17	16	21	20	–1

Critical path 10 - 20- 30 - 60 — 70.

16. Define the Rules for constructing the arrow diagram.

Rule 1:

Each activity is represented by one and only one arrow in the network. No single activity can be represented twice in the network. This is to be differentiated from the case where one activity is broken down into segments. In this case, each segment may be represented by a separate arrow. For example, in laying down a pipe, this may be done in

Sections rather than as one job.

Rule 2:

No two activities can be identified by the same head and tail events. A situation like this may arise when two or more activities can be performed concurrently.

Rule 3:

In order to ensure the correct precedence relationship in the arrow diagram the following questions must be answered as every activity is added to the network.

(i) What activities must be completed immediately before this activity can start ?
(ii)What activities must follow this activity ?
(iii)What activities must occur concurrently with this activity?
This rule is self-explanatory. It actually allows for checking (and rechecking) the precedence relationships as one progresses in the development of the network.

17. Briefly discuss about Hungarian method

Hungarian method: Assignment problems can be formulated with techniques of linear programming and transportation problems. As it has a special structure, it is solved by the special method called Hungarian method. This method of assignment problem was developed by a Hungarian mathematician D. Konig and is therefore known as Hungarian method of assignment problem.

To solve assignment problem using this method, one should know time of completion or cost of making all the possible assignments. Each assignment problem has a table, persons one wishes to assign are represented in the rows and jobs or tasks to be assigned are expressed in the columns. Cost for each particular assignment is in the numbers in the table. It is referred as cost matrix. Hungarian method is based on the principle that if a constant is added to the elements of cost

matrix, the optimum solution of the assignment problem is the same as the original problem. Original cost matrix is reduced to another cost matrix by adding a constant value to the elements of rows and columns of cost matrix where the total completion time or total cost of an assignment is zero. This assignment is also referred as the optimum solution since the optimum solution remains unchanged after the reduction.

18. Describe validity of verbal arguments.

Determine the validity of the following argument if 2 sides of a triangle are equal ,then 2 opposite angles are equal. Two sides of a triangle are not equal therefore the opposite angles are equal.

P: two sides of a triangle are equal

Q:two opposite angle are equal

The given argument in the form

P->Q, ¬P=>¬Q

Now we construct the truth table for (P->Q) ^(¬P)->(¬Q)

P	Q	P->Q	¬P	¬Q	(P->Q) ^(¬P)	(P->Q) ^(¬P)-> (¬Q)
T	T	T	F	F	F	T
T	F	F	F	T	F	T
F	T	T	T	F	T	F
F	F	T	T	T	T	T not a valid statement

Open statement

An open statement is a declarative sentence which contains one or more symbols,is not a true false statement,produces a TF statement when each of its symbols is replaced by a specific object from a designated set.

Ex: P(x):the rational number x is greater than 100

Q(x,y):x+y=10 where x and y are integers

Then P(x) and Q(x,y):the rational number x is greater than 100 and x+y=10

Quantifiers:

An analysis of mathematical sentences involving quantifiers indicates that the main two quantifiers are "all" and "some" where "some" is interpreted to mean atleast "one".

certain statement involve words that indicate quantity such as "all, some, none, one" .to answer the question "how many", such words indicate quantity they are called quantifiers.

19. Verify the validity of the following argument.

Every living thing isa plant or an animal.

John's gold fish is alive and it is not a plant

All animals have heart. Therefore John's gold fish has a heart

Let the universe consists of all living things let

P(x):x is plant

A(x): x is an animal

H(x): x has a heart

G: John's gold fish

Then the inference pattern is

(Vx) (P(x) V A(x))

¬P(g)

(Vx) (A(x)-> H(X))

———————————

H(g)

Argument

[1]	(1)	(P(x) V A(x))		P
[2]	(2)	¬P(g)	P	
[1]	(3)	P(g)vA(g)	us(1)	
[1,2]	(4)	A(g)		T(2)(3)
[5]	(5)	(Vx)(A(x)->H(x))		P
[5]	(6)	A(g)->H(g)	us(5)	
[1,2,5]		(7) H(g)		T(4),(6)

20. Describe Some Equivalence Laws of Propositional Logic

$(P \wedge Q) \vee R \equiv (P \vee R) \wedge (Q \vee R)$ distributivity law

$P \vee P \equiv P$ idempotency law for $\vee$

$P \vee Q \equiv Q \vee P$ commutativity of $\vee$

$P \vee (Q \vee R) \equiv (P \vee Q) \vee R$ associativity of $\vee$

$P \vee$ true $\equiv$ true true is right zero of $\vee$

$true \vee P \equiv true$ true is left zero of $\vee$

$P \vee false \equiv P$ false is right one of $\vee$

$false \vee P \equiv P$ false is left one of $\vee$

$P \wedge P \equiv P$ idempotency law for $\wedge$

$P \wedge Q \equiv Q \wedge P$ commutativity of $\wedge$

$P \wedge (Q \wedge R) \equiv (P \wedge Q) \wedge R$ associativity of $\wedge$

$P \wedge true \equiv P$ true is right one of $\wedge$

$true \wedge P \equiv P$ true is left one of $\wedge$

$P \wedge false \equiv false$ false is right zero of $\wedge$

$false \wedge P \equiv false$ false is left zero of $\wedge$

$\neg \neg P \equiv P$ double negation law

$P \Rightarrow Q \equiv \neg P \vee Q$ implication in terms of $\vee$

$P \Rightarrow Q \equiv \neg Q \Rightarrow \neg P$ contrapositive law

$true \Rightarrow P \equiv P$ true absorbed in implication

$false \Rightarrow P \equiv true$ false implies anything

$P \Rightarrow true \equiv true$ anything implies true

$P \Rightarrow false \equiv \neg P$ implication and negation law

$P \Leftrightarrow Q \equiv (P \wedge Q) \vee (\neg P \wedge \neg Q)$ bi-implication in terms of $\vee$ and $\wedge$

$P \Leftrightarrow Q \equiv (P \Rightarrow Q) \wedge (Q \Rightarrow P)$ bi-implication in terms of implication

$P \wedge Q \equiv \neg (\neg P \vee \neg Q)$ De Morgan's law

$P \vee Q \equiv \neg (\neg P \wedge \neg Q)$ De Morgan's law

$P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$ distributivity law

$(P \vee Q) \wedge R \equiv (P \wedge R) \vee (Q \wedge R)$ distributivity law

$P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$ distributivity law

$(P \wedge Q) \vee R \equiv (P \vee R) \wedge (Q \vee R)$ distributivity law

$(P \vee Q) \Rightarrow R \equiv (P \Rightarrow R) \wedge (Q \Rightarrow R)$ distributivity law

$P \Rightarrow (Q \wedge R) \equiv (P \Rightarrow Q) \wedge (P \Rightarrow R)$ distributivity law

21. Discuss the Equivalence Laws of Predicate Logic

$\exists\ x : X \cdot P \equiv P$ provided x\P

$\exists\ x : X \cdot P \vee Q \equiv (\exists\ x : X \cdot P) \vee (\exists\ x : X \cdot Q)$ existential quantification and disjunction

$\exists\ x : X \cdot P \wedge x = e \equiv P[e/x]$ one point rule

$\forall\ x : X \cdot P \equiv P$ provided x\P

$\forall\ x : X \cdot P \wedge Q \equiv (\forall\ x : X \cdot P) \wedge (\forall\ x : X \cdot Q)$ existential quantification and disjunction

$\exists\ x : X \cdot P \equiv \neg\ \forall\ x : X \cdot \neg\ P$ existential and universal quantification

$\forall\ x : X \cdot P \equiv \neg\ \exists\ x : X \cdot \neg\ P$ existential and universal quantification

22. State about Laws of Equivalence

Laws of Equivalence

Idempotence p ^ p is logically equivalent to p

p _ p is logically equivalent to p

Identity p ^ true is logically equivalent to p

p _ false is logically equivalent to p

Domination p ^ false is logically equivalent to false

p _ true is logically equivalent to true

Commutativity p ¤ q is logically equivalent to q * p,

where * can be either ^, _, ©, or $, but not !

Associativity p * (q * r) is logically equivalent to (p * q) * r,

where * can be either ^, _, ©, or $, but not !

Distributivity p ^ (q _ r) is logically equivalent to (p ^ q) _ (p ^ r)

p _ (q ^ r) is logically equivalent to (p _ q) ^ (p _ r)

De Morgan's Laws :(p ^ q) is logically equivalent to :p _ :q

:(p _ q) is logically equivalent to :p ^ :q

Absorption p _ (p ^ q) is logically equivalent to p

p ^ (p _ q) is logically equivalent to p

Double Negation ::p is logically equivalent to p

Excluded Middle p ^ :p is logically equivalent to false

p _ :p is logically equivalent to true

De¯nitions p ! q is logically equivalent to :p _ q

p $ q is logically equivalent to (p ! q) ^ (q ! p)

p © q is logically equivalent to (p _ q) ^ :(p ^ q)

23. Which laws of Equivalence says P V (~P ∧ q) =P V Q?

1.Commutative laws: $p \wedge q \equiv q \wedge p$ $p \vee q \equiv q \vee p$

2. Associative laws: $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$ $(p \vee q) \vee r \equiv p \vee (q \vee r)$

3. Distributive laws: $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$ $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$

4. Identity laws: $p \wedge t \equiv p$ $p \vee c \equiv p$

5. Negation laws: $p \vee \sim p \equiv t$ $p \wedge \sim p \equiv c$

6. Double negative law: $\sim(\sim p) \equiv p$

7. Idempotent laws: $p \wedge p \equiv p$ $p \vee p \equiv p$

8. Universal bound laws: $p \vee t \equiv t$ $p \wedge c \equiv c$

9. De Morgan's laws: $\sim(p \wedge q) \equiv \sim p \vee \sim q$ $\sim(p \vee q) \equiv \sim p \wedge \sim q$

10. Absorption laws: $p \vee (p \wedge q) \equiv p$ $p \wedge (p \vee q) \equiv p$

11. Negations of t and c: $\sim t \equiv c$ $\sim c \equiv t$

24. Prove: $(p \wedge \neg q) \vee q \Leftrightarrow p \vee q$

$(p \wedge \neg q) \vee q$	Left-Hand Statement
$\Leftrightarrow q \vee (p \wedge \neg q)$	Commutative
$\Leftrightarrow (q \vee p) \wedge (q \vee \neg q)$	Distributive
$\Leftrightarrow (q \vee p) \wedge T$	Or Tautology
$\Leftrightarrow q \vee p$	Identity
$\Leftrightarrow p \vee q$	Commutative

Begin with exactly the left-hand side statement

End with exactly what is on the right

Justify EVERY step with a logical equivalence

25. Prove: $(p \wedge \neg q) \vee q \Leftrightarrow p \vee q$

$(p \wedge \neg q) \vee q$	Left-Hand Statement
$\Leftrightarrow q \vee (p \wedge \neg q)$	Commutative
$\Leftrightarrow (q \vee p) \wedge (q \vee \neg q)$	Distributive

Why did we need this step?

Our logical equivalence specified that $\vee$ is distributive on the right. This does not guarantee the equivalence works on the left!

Ex.: Matrix multiplication is not always commutative

(Note that whether or not $\vee$ is distributive on the left is not the point here.)

26. Prove: $p \rightarrow q \Leftrightarrow \underline{\neg q \rightarrow \neg p}$

Contrapositive

$p \rightarrow q$

$\Leftrightarrow \neg p \vee q$ Implication Equivalence

$\Leftrightarrow q \vee \neg p$ Commutative

$\Leftrightarrow \neg(\neg q) \vee \neg p$ Double Negation

$\Leftrightarrow \neg q \rightarrow \neg p$ Implication Equivalence

27. Prove: $p \rightarrow p \vee q$ is a tautology

Must show that the statement is true for any value of p,q.

$p \rightarrow p \vee q$

$\Leftrightarrow \neg p \vee (p \vee q)$ Implication Equivalence

$\Leftrightarrow (\neg p \vee p) \vee q$ Associative

$\Leftrightarrow (p \vee \neg p) \vee q$ Commutative

$\Leftrightarrow T \vee q$ Or Tautology

$\Leftrightarrow q \vee T$ Commutative

$\Leftrightarrow T$ Domination

This tautology is called the addition rule of inference.

28. Prove: $(p \wedge q) \rightarrow p$ is a tautology

$(p \wedge q) \rightarrow p$

$\Leftrightarrow \neg(p \wedge q) \vee p$ Implication Equivalence

$\Leftrightarrow (\neg p \vee \neg q) \vee p$ DeMorgan's

$\Leftrightarrow (\neg q \vee \neg p) \vee p$ Commutative

$\Leftrightarrow \neg q \vee (\neg p \vee p)$ Associative

$\Leftrightarrow \neg q \vee (p \vee \neg p)$ Commutative

$\Leftrightarrow \neg q \vee T$ Or Tautology

$\Leftrightarrow T$ Domination

29. Prove or Disprove $p \rightarrow q \Leftrightarrow p \wedge \neg q$???

$p \rightarrow q \Leftrightarrow p \wedge \neg q$???

To prove that something is not true it is enough to provide one counter-example. (Something that is true must be true in every case.)

p	q	p→q	p∧¬q
F	T	T	F

The statements are not logically equivalent

30. Prove:$\neg p \leftrightarrow q \Leftrightarrow p \leftrightarrow \neg q$

$\neg p \leftrightarrow q$

$\Leftrightarrow (\neg p \rightarrow q) \wedge (q \rightarrow \neg p)$ Biconditional Equivalence

$\Leftrightarrow (\neg\neg p \vee q) \wedge (\neg q \vee \neg p)$ Implication Equivalence (x2)

$\Leftrightarrow (p \vee q) \wedge (\neg q \vee \neg p)$ Double Negation

$\Leftrightarrow (q \vee p) \wedge (\neg p \vee \neg q)$ Commutative

$\Leftrightarrow (\neg\neg q \vee p) \wedge (\neg p \vee \neg q)$ Double Negation

$\Leftrightarrow (\neg q \rightarrow p) \wedge (p \rightarrow \neg q)$ Implication Equivalence (x2)

$\Leftrightarrow p \leftrightarrow \neg q$ Biconditional Equivalence

31. Discuss about The Chi-Square Distribution

The *chi-square distribution* can be used to perform the *goodness-of-fit test*, which compares the observed values of a categorical variable with the expected values of that same variable.

Research Question: Do 11^{th} grade students prefer a certain type of lunch?

Using a sample of 100 11^{th} grade students, we recorded the following information:

Frequency of Type of School Lunch Chosen by Students

Type of Lunch	**Observed Frequency**	**Expected Frequency**
Salad	21	25
Sub Sandwich	29	25
Daily Special	14	25
Brought Own Lunch	36	25

If there is no difference in which type of lunch is preferred, we would expect the students to prefer each type of lunch equally. To calculate the expected frequency of each category when assuming school lunch preferences are distributed equally, we divide the number of observations by the number of categories. Since there are 100 observations and 4 categories, the expected frequency of each category is $\frac{100}{4}$, or 25.

32. The Chi-Square Statistic

The value that indicates the comparison between the observed and expected frequency is called the *chi-square statistic*. The idea is that if the observed frequency is close to the expected frequency, then the chi-square statistic will be small. On the other hand, if there is a substantial difference between the two frequencies, then we would expect the chi-square statistic to be large.

To calculate the chi-square statistic, χ^2, we use the following formula:

$\chi^2 = \sum \frac{(O-E)^2}{E}$

where:

χ^2 is the chi-square test statistic.

O is the observed frequency value for each event.

E is the expected frequency value for each event.

We compare the value of the test statistic to a tabled chi-square value to deter mine the probability that a sample fits an expected pattern.

33. Discuss Features of the Goodness-of-Fit Test

As mentioned, the goodness-of-fit test is used to determine patterns of distinct categorical variables. The test requires that the data are obtained through a random sample. The number of *degrees of freedom* associated with a particular chi-square test is equal to the number of categories minus one. That is, $df = c - 1$.

Using our example about the preferences for types of school lunches, we calculate the degrees of freedom as follows:

$df = \text{number of categories} - 1$

$3 = 4 - 1$

There are many situations that use the goodness-of-fit test, including surveys, taste tests, and analysis of behaviors. Interestingly, goodness-of-fit tests are also used in casinos to determine if there is cheating in games of chance, such as cards or dice. For example, if a certain card or number on a die shows up more than expected (a

high observed frequency compared to the expected frequency), officials use the goodness-of-fit test to determine the likelihood that the player may be cheating or that the game may not be fair.

34. Discuss about Regression.

Regression analysis involves identifying the relationship between a dependent variable and one or more independent variables. A model of the relationship is hypothesized, and estimates of the parameter values are used to develop an estimated regression equation. Various tests are then employed to determine if the model is satisfactory. If the model is deemed satisfactory, the estimated regression equation can be used to predict the value of the dependent variable given values for the independent variables.

Regression model.

In simple linear regression, the model used to describe the relationship between a single dependent variable y and a single independent variable x is $y = a0 + a1x + k$. a0and a1 are referred to as the model parameters, and is a probabilistic error term that accounts for the variability in y that cannot be explained by the linear relationship with x. If the error term were not present, the model would be deterministic; in that case, knowledge of the value of x would be sufficient to determine the value of y.

Least squares method.

Either a simple or multiple regression model is initially posed as a hypothesis concerning the relationship among the dependent and independent variables. The least squares method is the most widely used procedure for developing estimates of the model parameters.

35. Discuss about Correlation

Correlation and regression analysis are related in the sense that both deal with relationships among variables. The correlation coefficient is a measure of linear association between two variables. Values of the correlation coefficient are always between -1 and +1. A correlation coefficient of +1 indicates that two variables are perfectly related in a positive linear sense, a correlation coefficient of -1 indicates that two variables are perfectly related in a negative linear sense, and a correlation coefficient of 0 indicates that there is no linear relationship between the two variables. For simple linear regression, the sample correlation coefficient is the square root of the coefficient of determination, with the sign of the correlation coefficient being the same as the sign of b1, the coefficient of x1 in the estimated regression equation.

36. The following data characteristics of a project

Activity	Immediate predecessors	Duration in days
A	-	2
B	A	3
C	A	4
D	B,C	6
E	-	2
F	E	8

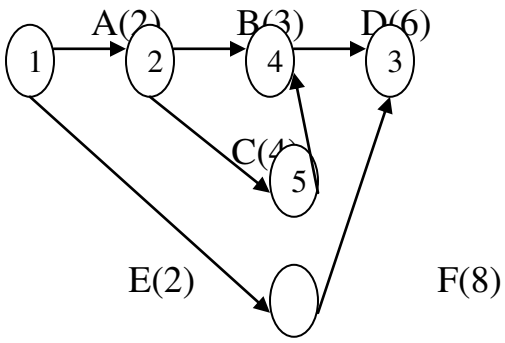
i)Draw the network diagram for the above project

ii) find the maximum project completion time and the critical path

solution:

A,E has no immediate predecessor.so A,E is a start event

D,F->terminal event



THE VARIOUS PATHS ARE

1->2->4->6=11

1->2->5->4->6=12

1->3->6=10

Critical path=1->2->5->4->6

Minimum project completion time:12

37. Discuss about Directed and undirected Graph.

Directed and Undirected Graphs

A graph is a mathematical structure consisting of a set of vertices and a set of edges connecting the vertices.

Formally: G = (V; E), where V is a set and E V V

G = (V; E) undirected if for all v; w 2 V :

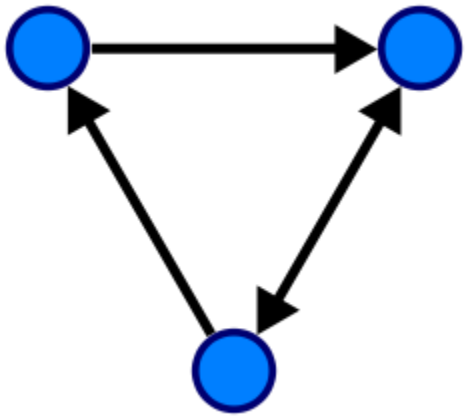
(v; w) 2 E () (w; v) 2 E

In mathematics, a **directed graph** or **digraph** is a graph, or set of nodes connected by edges, where the edges have a direction associated with them. In formal terms a digraph is a pair $G = (V, A)$ (sometimes $G = (V, E)$) of:[1]

- a set V, whose elements are called *vertices* or *nodes*,
- a set A of ordered pairs of vertices, called *arcs*, *directed edges*, or *arrows* (and sometimes simply *edges* with the corresponding set named E instead of A).

It differs from an ordinary or undirected graph, in that the latter is defined in terms of unordered pairs of vertices, which are usually called edges.

Sometimes a digraph is called a *simple digraph* to distinguish it from a *directed multigraph*, in which the arcs constitute a multiset, rather than a set, of ordered pairs of vertices. Also, in a simple digraph loops are disallowed. (A loop is an arc that pairs a vertex to itself.) On the other hand, some texts allow loops, multiple arcs, or both in a digraph.



undirected graph

An undirected graph is one in which edges have no orientation. The edge (a, b) is identical to the edge (b, a), i.e., they are not ordered pairs, but sets $\{u, v\}$ (or 2-multisets) of vertices.

38. Write about F-Distribution.

F-DISTRIBUTION

In probability theory and statistics, the **F-distribution** is a continuousprobability distribution. It is also known as **Snedecor's F distribution** or the **Fisher-Snedecor distribution** (after R.A. Fisher andGeorge W. Snedecor). The F-distribution arises frequently as the null distribution of a test statistic, most notably in the analysis of variance; see F-test.

There is a different F distribution for each combination of the degrees of freedom of the numerator and denominator. Since there are so many F distributions, the F tables are organized somewhat differently than the tables for the other distributions. The three tables which follow are organized by the level of significance. The first table gives F values for that are associated with ® = 0:10 of the area in the right tail of the distribution. The second table gives the F values for ® = 0:05 of the area in the right tail, and the third table gives F values for the ® = 0:01 level of significance. In each of these tables, the F values are given for various combinations of degrees of freedom.

39. Write about t-test.

A ***t*-test** is any statistical hypothesis test in which the test statistic follows a Student's *t* distribution if the null hypothesis is supported. It is most commonly

applied when the test statistic would follow a normal distribution if the value of a scaling term in the test statistic were known. When the scaling term is unknown and is replaced by an estimate based on the data, the test statistic (under certain conditions) follows a Student's *t* distribution.

Among the most frequently **used *t*-tests** are:

- A one-sample location test of whether the mean of a normally distributed population has a value specified in a null hypothesis.
- A two-sample location test of the null hypothesis that the means of two normally distributed populations are equal. All such tests are usually called **Student's *t*-tests**, though strictly speaking that name should only be used if the variances of the two populations are also assumed to be equal; the form of the test used when this assumption is dropped is sometimes called Welch's *t*-test. These tests are often referred to as "unpaired" or "independent samples" *t*-tests, as they are typically applied when the statistical units underlying the two samples being compared are non-overlapping.
- A test of the null hypothesis that the difference between two responses measured on the same statistical unit has a mean value of zero. For example, suppose we measure the size of a cancer patient's tumor before and after a treatment. If the treatment is effective, we expect the tumor size for many of the patients to be smaller following the treatment. This is often referred to as the "paired" or "repeated measures" *t*-test:[5][6] see paired difference test.
- A test of whether the slope of a regression line differs significantly from 0.

40. Describe Ceaser Cipher

A cipher is a system which transforms plaintext into ciphertext by applying a set of

transformations to each character (or letter) in the plaintext. The particular transformations employed at any time are controlled by a "key" used at that time. Security of the ciphertext rests heavily on the secrecy of the key; it is the objective of
the cryptanalyst to find the key and consequently break the system.

Caesar Ciphers. One of the earliest known cryptographic systems was used by Julius
Caesar and is appropriately referred to as a Caesar Cipher [26]. Around 50 B.C., Julius
Caesar wrote to Marcus Cicero, using a cipher that shifts the alphabet three places to the
right and wraps the last three letters X, Y, Z back onto the first three letters:

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

Thus, the plaintext message
MEET YOU IN ORLANDO
is transformed into the ciphertext
PHHW BRX LQ RUODQGR.

Such a transformation, using modular arithmetic, can also be performed by a computer.
Any message can be expressed digitally based on the one-to-one correspondence,
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
To encipher the plaintext message, we use the transformation
where M is the numeric equivalent of a plaintext letter. To decipher, we utilize the

transformation where C is the numeric equivalent of a cipher text letter. Hence:

O R L A N D O

14 17 11 00 13 03 14

E: D:

DsCd ; sC 1 23dsmodulo 26d,

EsMd ; sM 1 3dsmodulo 26d,

217 20 14 03 16 06 17

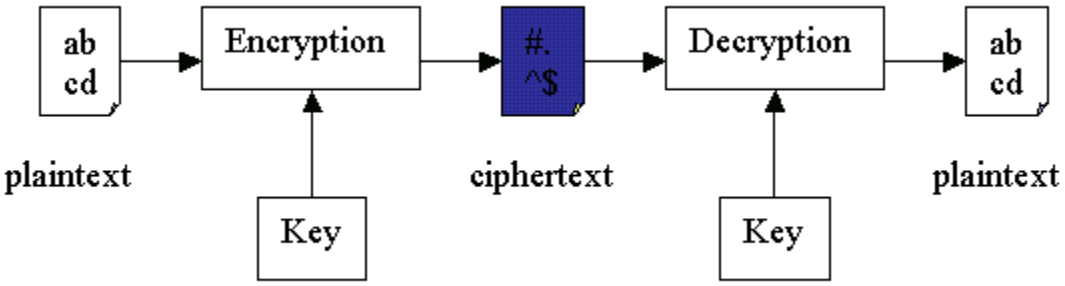
R U O D Q G R

41. Write about Cryptography

When a message is transferred from one place to another, it contents are readily available to an eavesdropper. A simple network-monitoring tool can expose the entire message sent from one computer to another in a graphical way. For an N-Tier or distributed application to be secure, all messages sent on the network should be scrambled in a way that it is computationally impossible for any one to read it.

In cryptography world the message that needs to be secured is called plaintext or cleartext. The scrambled form of the message is called ciphertext. The process of converting a plaintext to ciphertext is called encryption. The process of reconverting the ciphertext into plaintext is called decryption. Cryptography algorithms (ciphers) are mathematical functions used for encryption and decryptions.

For cryptography to be used in practical solutions algorithms used for encryption and decryption should be made public. This is possible by using a byte stream called Key. For the algorithm to encipher a plaintext to ciphertext and to decipher it



42. Discuss about Symmetric Key Encryption

Symmetric key encryption uses same key, called secret key, for both encryption and decryption. Users exchanging data keep this key to themselves. Message encrypted with a secret key can be decrypted only with the same secret key.

The algorithm used for symmetric key encryption is called secret-key algorithm. Since secret-key algorithms are mostly used for encrypting the content of the message they are also called content-encryption algorithms.

The major vulnerability of secret-key algorithm is the need for sharing the secret-key. One way of solving this is by deriving the same secret key at both ends from a user supplied text string (password) and the algorithm used for this is called password-based encryption algorithm. Another solution is to securely send the secret-key from one end to other end. This is done using another class of encryption called asymmetric algorithm, which is discussed later.

Strength of the symmetric key encryption depends on the size of the key used. For the same algorithm, encrypting using longer key is tougher to break than the one done using smaller key. Strength of the key is not liner with the length of the key but doubles with each additional bit.

Following are some of popular secret-key algorithms and the key size that they use

RC2	-	64	bits
DES	64		bits
3DES	192		bits
AES	256		bits

IDEA 128 bits

CAST 128 bits (CAST256 uses 256 bits key)

Algorithm Parameters:

1. EncryptionMode

   There are two types of secret-key ciphers, block ciphers and stream ciphers Block Ciphers convert fixed-length block of plain text into cipher text of the same length. Most of the block ciphers use a block size of 64 bits. When the message size is more than that of the block size, then the message is broken into multiple blocks and each block is encrypted separately. There are different modes in which a Block cipher can encrypt these blocks. Viz., Cipher Block Chaining (CBC), Electronic Codebook (ECB) or Cipher Feedback (CFB). Of these CBC mode is more commonly used. In CBC Mode, each plain text block is XORed with the previous cipher text block before being encrypted. Some famous Block ciphers are DES, 3DES, IDEA, SAFER, Blowfish and Skipjack (used by US National SecurityAgency). Stream Ciphers operate on small group of bits, typically applying bitwise XOR operations to them using the key as a sequence of bits. Some famous stream ciphersincludeRC4andSEAL.

2. Initialization Vector

   Since Block ciphers working on CBC modes XOR each block with the previous encrypted block, the first block of the message needs a byte array, of same block size, with which it will be XORed This byte array is called IV or Initialization Vector.
   Following are some block ciphers with their normal block size
   DES - 64 bits

3DES	64	bits
AES	128	bits

3. Padding

Often last block of the message will be smaller that the expected block size in which case a predetermined string will be repeatedly added to the end of the block to make it to the expected size. For instance if the block size is 64 bits and the last block has only 40 bits then 24 bits of padding will be added to it. There were two ways to add the pad, either by adding zeros or the number of the bytes that needs to be added (in this case it will be 3).

43. Discuss about Asymmetric Key Encryption

Asymmetric key encryption uses different keys for encryption and decryption. These two keys are mathematically related and they form a key pair. One of these two keys should be kept private, called private-key, and the other can be made public (it can even be sent in mail), called public-key. Hence this is also called Public Key Encryption.

A private key is typically used for encrypting the message-digest; in such an application private-key algorithm is called message-digest encryption algorithm. A public key is typically used for encrypting the secret-key; in such a application private-key algorithm is called key encryption algorithm.

Popular private-key algorithms are RSA (invented by Rivest, Shamir and Adlemen) and DSA (Digital Signature Algorithm). While for an ordinary use of RSA, a key size of 768 can be used, but for corporate use a key size of 1024 and for extremely valuable information a key size of 2048 should be used.

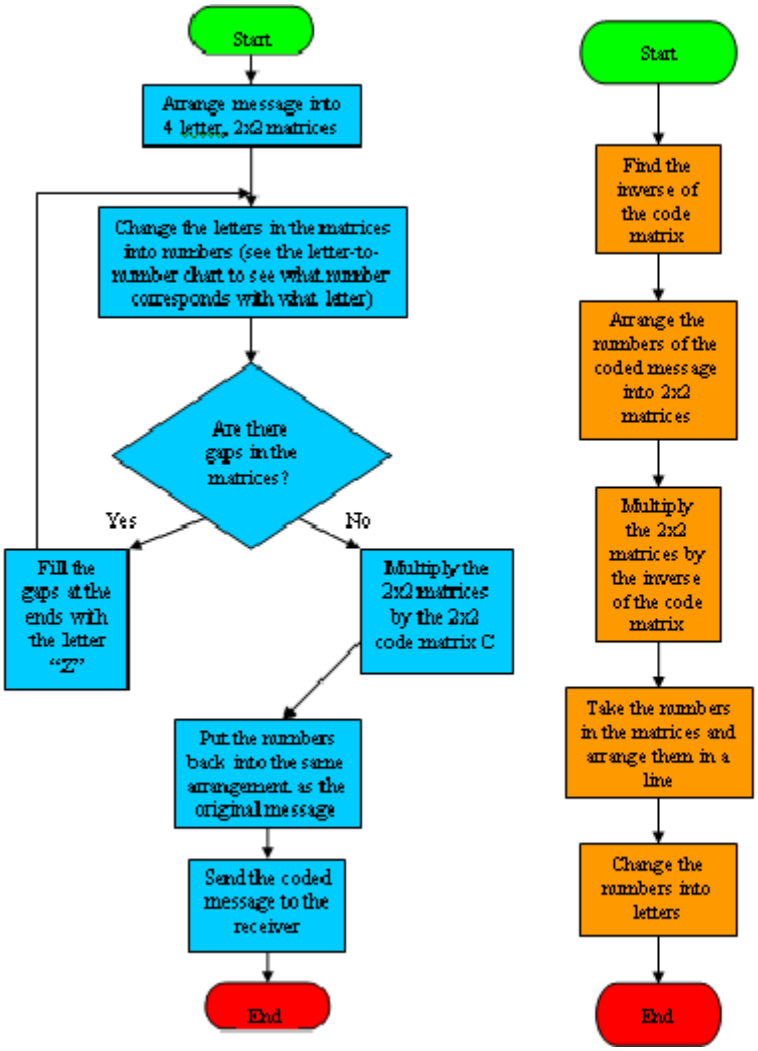
Asymmetric key encryption is much slower than symmetric key encryption and hence they are only used for key exchanges and digital signatures.

44. Illustrate Encoding and Decoding a Message

Encoding and decoding using matrices in cryptography

Matrix multiplication can be used to "encode" and "decode" messages. For that a coding matrix is required which is known to both the sender and the recipient. The following flowchart illustrates this in a vivid way:-

Encoding a Message & Decoding a Message



45. Write an example for encoding a message

Coding Letters

A	B	C	D	E	F	G	H	I	J	K	L	M
1	2	3	4	5	6	7	8	9	10	11	12	13

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
14	15	16	17	18	19	20	21	22	23	24	25	26

Example:

Table 1

Encoding

An example for encoding a message is as follows:

To encode the message FMG ROCKS the following steps are taken:

A) Write the message using 2x2 matrices.

$$\begin{bmatrix} F & M \\ G & R \end{bmatrix}, \begin{bmatrix} O & C \\ K & S \end{bmatrix}$$

If any gaps are left, we use a filler instead, adding a Z.

B) Replace each letter with its corresponding number in the alphabet.

A=1, B=2, C=3 and so on.

$$\begin{bmatrix} 6 & 13 \\ 7 & 18 \end{bmatrix}, \begin{bmatrix} 15 & 3 \\ 11 & 19 \end{bmatrix}$$

Now the matrix is encoded by multiplying it to the coding matrix, which is known only by the sender and the receiver. $C = \begin{bmatrix} 3 & 4 \\ 5 & 6 \end{bmatrix}$

$$\begin{bmatrix} 3 & 4 \\ 5 & 6 \end{bmatrix} x \begin{bmatrix} 6 & 13 \\ 7 & 18 \end{bmatrix} = \begin{bmatrix} (3x6)+(4x7)=46 & (3x13)+(4x18)=111 \\ (5x6)+(6x7)=72 & (5x13)+(6x18)=173 \end{bmatrix}$$

$$\begin{bmatrix} 3 & 4 \\ 5 & 6 \end{bmatrix} x \begin{bmatrix} 15 & 3 \\ 11 & 19 \end{bmatrix} = \begin{bmatrix} (3x15)+(4x11)=89 & (3x3)+(4x19)=85 \\ (5x15)+(6x11)=141 & (5x3)+(6x19)=129 \end{bmatrix}$$

Therefore, you would send the message as 46,111,72,173,89,85,141,129.

46. Write an example for decoding a message

Coding Letters

A	B	C	D	E	F	G	H	I	J	K	L	M
1	2	3	4	5	6	7	8	9	10	11	12	13

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
14	15	16	17	18	19	20	21	22	23	24	25	26

To decode a message you multiply the coded message with the INVERSE of the coding matrix.

First, write the coded message as a 2×2 matrix.

$$A = \begin{bmatrix} 46 & 111 \\ 72 & 173 \end{bmatrix}$$

$$B = \begin{bmatrix} 89 & 85 \\ 141 & 129 \end{bmatrix}$$

The decoding matrix, or key, is the inverse matrix of the coding matrix.

$$C = \begin{bmatrix} 3 & 4 \\ 5 & 6 \end{bmatrix}$$

$$C^{-1} = \frac{1}{ad-bc}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

$$C^{-1} = \frac{1}{(3)(6)-(4)(5)}\begin{bmatrix} 6 & -4 \\ -5 & 3 \end{bmatrix}$$

$$C^{-1} = -\frac{1}{2}\begin{bmatrix} 6 & -4 \\ -5 & 3 \end{bmatrix}$$

$$C^{-1} = \begin{bmatrix} -3 & 2 \\ 2.5 & -1.5 \end{bmatrix}$$

Multiply the inverse key by the coded message.(key X message)

$$C^{-1}A = \begin{bmatrix} -3 & 2 \\ 2.5 & -1.5 \end{bmatrix}\begin{bmatrix} 46 & 111 \\ 72 & 173 \end{bmatrix} = \begin{bmatrix} 6 & 13 \\ 7 & 18 \end{bmatrix}$$

$$C^{-1}B = \begin{bmatrix} -3 & 2 \\ 2.5 & -1.5 \end{bmatrix}\begin{bmatrix} 89 & 85 \\ 141 & 129 \end{bmatrix} = \begin{bmatrix} 15 & 3 \\ 11 & 19 \end{bmatrix}$$

47. Discuss about Scrambler.

In telecommunications, a **scrambler** is a device that transposes or inverts signals or otherwise encodes a message at the transmitter to make the message unintelligible at a receiver not equipped with an appropriately set descrambling device. Whereas encryption usually refers to operations carried out in the digital domain, scrambling usually refers to operations carried out in the analog domain. Scrambling is accomplished by the addition of components to the original signal or the changing of some important component of the original signal in order to make extraction of the original signal difficult. Examples of the latter might include removing or changing vertical or horizontal sync pulses in television signals; televisions will not be able to display a picture from such a signal. Some modern scramblers are actually encryption devices, the name remaining due to the similarities in use, as opposed to internal operation.

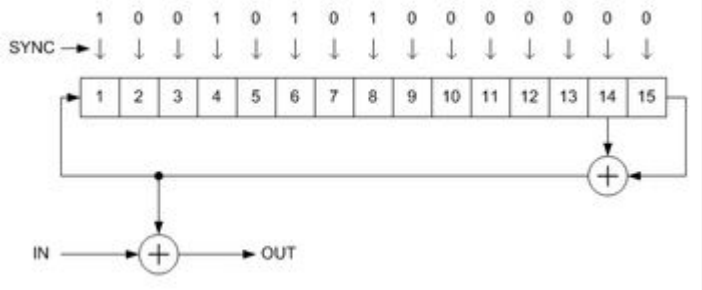
In telecommunications and recording, a **scrambler** (also referred to as a **randomizer**) is a device that manipulates a data stream before transmitting. The manipulations are reversed by a**descrambler** at the receiving side. Scrambling is widely used in satellite, radio relay communications and PSTN modems. A scrambler can be placed just before a FEC coder, or it can be placed after the FEC, just before the modulation or line code. A scrambler in this context has nothing to do with encrypting, as the intent is not to render the message unintelligible, but to give the transmitted data useful engineering properties.

A scrambler replaces sequences into other sequences without removing undesirable sequences, and as a result it changes the probability of occurrence of vexatious sequences. Clearly it is not foolproof as there are input sequences that yield all-

zeros, all-ones, or other undesirable periodic output sequences. A scrambler is therefore not a good substitute for a line code, which, through a coding step, removes unwanted sequences.

48. Write about Additive Scrambler.

Additive (synchronous) scramblers



An additive scrambler (descrambler) used in DVB

Additive scramblers (they are also referred to as *synchronous*) transform the input data stream by applying a pseudo-random binary sequence (PRBS) (by modulo-two addition). Sometimes a pre-calculated PRBS stored in the Read-only memory is used, but more often it is generated by a linear feedback shift register (LFSR).

In order to assure a synchronous operation of the transmitting and receiving LFSR (that is, *scrambler* and *descrambler*), a *sync-word* must be used.

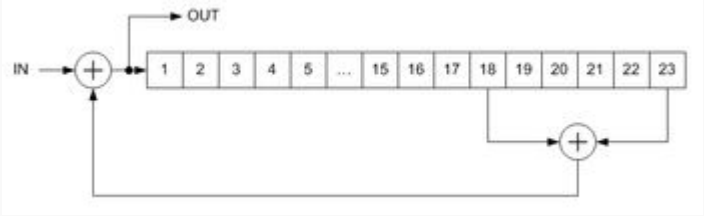
A sync-word is a pattern that is placed in the data stream through equal intervals (that is, in each frame). A receiver searches for a few sync-words in adjacent frames and hence determines the place when its LFSR must be reloaded with a pre-defined *initial state*.

The *additive descrambler* is just the same device as the additive scrambler.

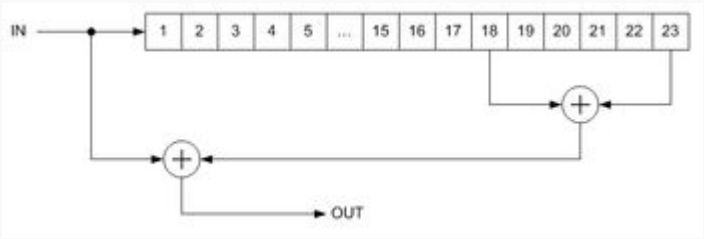
Additive scrambler/descrambler is defined by the polynomial of its LFSR (for the scrambler on the picture above, it is $1 + x^{-14} + x^{-15}$) and its *initial state*.

49. Write about multiplicative Scrambler.

Multiplicative (self-synchronizing) scramblers



A multiplicative scrambler used in V.34 recommendation



A multiplicative descrambler used in V.34 recommendation

Multiplicative scramblers (also known as *feed-through*) are called so because they perform a *multiplication* of the input signal by the scrambler's transfer function in Z-space. They are discrete linear time-invariant systems. A multiplicative scrambler is recursive and a multiplicative descrambler is non-recursive. Unlike additive scramblers, multiplicative scramblers do not need the frame synchronization, that is why they are also called *self-synchronizing*. Multiplicative scrambler/descrambler is defined similarly by a polynomial (for the scrambler on the picture it is $1 + x^{-18} + x^{-23}$), which is also a *transfer function* of the descrambler.

50. Write the draw backs of Scrambler.

Scramblers have certain drawbacks:

- Both types may fail to generate random sequences under worst case input conditions.
- Multiplicative scramblers lead to error multiplication during descrambling (i.e. a single bit error at the descrambler's input will result into *w* errors at its output, where *w* equals the number of the scrambler's feedback taps).
- Additive scramblers must be reset by the frame sync; if this fails massive error propagation will result as a complete frame cannot be descrambled.
- The effective length of the random sequence of an additive scrambler is limited by the frame length, which is normally much shorter than the period of the PRBS. By adding frame numbers to the frame sync, it is possible to extend the length of the random sequence, by varying the random sequence in accordance with the frame number.

51. Write about the usage of using Scrambler.

There are two main reasons why scrambling is used:

It facilitates the work of a timing recovery circuit (see also Clock recovery), an automatic gain control and other adaptive circuits of the receiver (eliminating long sequences consisting of '0' or '1' only).

It eliminates the dependence of a signal's power spectrum upon the actual transmitted data, making it more dispersed to meet maximum power spectral density requirements (because if the power is concentrated in a narrow frequency

band, it can interfere with adjacent channels due to the cross modulation and the intermodulation caused by non-linearities of the receiving tract).

52. Discuss about connectives.

Connectives are used for making compound propositions. The main ones are the following (p and q represent given propositions):

Name	Represented	Meaning
Negation	$\neg p$	"not p"
Conjunction	$p \wedge q$	"p and q"
Disjunction	$p \vee q$	"p or q (or both)"
Exclusive Or	$p \oplus q$	"either p or q, but not both"
Implication	$p \rightarrow q$	"if p then q"
Biconditional	$p \leftrightarrow q$	"p if and only if q"

The truth value of a compound proposition depends only on the value of its components. Writing F for "false" and T for "true", we can summarize the meaning of the connectives in the following way:

1.1. PROPOSITIONS

p	q	$\neg p$	$p \wedge q$	$p \vee q$	$p \oplus q$	$p \rightarrow q$	$p \leftrightarrow q$
T	T	F	T	T	F	T	T
T	F	F	F	T	T	F	F
F	T	T	F	T	T	T	F
F	F	T	F	F	F	T	T

Note that _ represents a non-exclusive or, i.e., p $\oplus$ q is true when any of p, q is true and also when both are true. On the other hand $\oplus$ represents an exclusive or, i.e., p $\oplus$ q is true only when exactly one of

p and q is true.

53. Discuss about Tautology, Contradiction, and Contingency.

1. A proposition is said to be a tautology if its truth value is T for any assignment of truth values to its components.
Example: The proposition p _ ¬p is a tautology.

2. A proposition is said to be a contradiction if its truth value is F for any assignment of truth values to its components.
Example: The proposition p ^ ¬p is a contradiction.

3. A proposition that is neither a tautology nor a contradiction is called a contingency.

p	$\neg p$	$p \vee \neg p$	$p \wedge \neg p$
T	F	T	F
T	F	T	F
F	T	T	F
F	T	T	F

tautology contradiction

54. Write about Conditional Propositions.

A proposition of the form “if p then q” or “p implies q”, represented “p ! q” is called a conditional proposition. For instance: “if John is from Chicago then John is from

Illinois". The proposition p is called hypothesis or antecedent, and the proposition q is the conclusion or consequent.

Note that p ! q is true always except when p is true and q is false. So, the following sentences are true: "if 2 < 4 then Paris is in France" (true ! true), "if London is in Denmark then 2 < 4" (false ! true),

"if 4 = 7 then London is in Denmark" (false ! false). However the following one is false: "if 2 < 4 then London is in Denmark" (true ! false).In might seem strange that "p ! q" is considered true when p is false, regardless of the truth value of q. This will become clearer when we study predicates such as "if x is a multiple of 4 then x is a multiple of 2". That implication is obviously true, although for the particular case x = 3 it becomes "if 3 is a multiple of 4 then 3 is a multiple of 2".
The proposition p $ q, read "p if and only if q", is called biconditional. It is true precisely when p and q have the same truth value, i.e., they are both true or both false.

55. Describe Pseudorandom Functions

A function $F : \{0,1\}^k \times \{0,1\}^m \to \{0,1\}^m$ *is a* (t,ϵ)*-secure pseudorandom function if for every oracle algorithm* T *that has complexity at most* t *we have*

$$\left| \mathop{\mathbb{P}}_{K\in\{0,1\}^k}[T^{F_K}() = 1] - \mathop{\mathbb{P}}_{R:\{0,1\}^m\to\{0,1\}^m}[T^R() = 1] \right| \leq \epsilon$$

Intuitively, this means that an adversary wouldn't be able to distinguish outputs from a purely random function and a pseudorandom function (upto a certain ϵ additive error). Typical parameters are $k = m = 128$, in which case security as high as $(2^{60}, 2^{-40})$ is conjectured to be possible.

As usual, it is possible to give an asymptotic definition, in which $\epsilon(k)$ is required to be negligible, $t(k)$ is allowed to be any polynomial, and F is required to be computable in polynomial time.

56.Write about The Randomized Counter Mode

Recall that a pseudorandom function is a function $F\colon \{0,1\}^k \times \{0,1\}^m \to \{0,1\}^m$ which looks approximately like a random function $R\colon \{0,1\}^m \to \{0,1\}^m$. With the encryption method from the previous lecture (in which the ciphertext is a random $r \in \{0,1\}^m$ followed by $F_K(r) \oplus M$) the encryption of a message is twice as long as the original message. We now define an encryption method which continues to use a pseudorandom function, but whose ciphertext overhead is marginal.

Suppose we have a pseudorandom function $F\colon \{0,1\}^k \times \{0,1\}^m \to \{0,1\}^m$. We describe an encryption scheme that works for messages of variable length. We assume without loss of generality that the length of the message is a multiple of m, and we write a plaintext M of length cm as $M_1, \ldots, M_c$, a sequence of c blocks of length m.

- $Enc(K, M_1, \ldots, M_c)$:
  - pick a random $r \in \{0,1\}^m$
  - output

$$(r, F_K(r) \oplus M_1, F_K(r+1) \oplus M_2, \ldots, F_K(r+(c-1)) \oplus M_c)$$

- $Dec(K, C_0, \ldots, C_c) := C_1 \oplus F_K(C_0), \ldots, C_c \oplus F_K(C_0 + (c-1))$

57. Describe Pseudorandom Permutations

Definition

Denote by $\mathcal{P}_n$ the set of permutations $P\colon \{0,1\}^n \to \{0,1\}^n$.

A pair of functions $F\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$, $I\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ *is a* (t,ϵ)*-secure pseudorandom permutation if:*

- *For every* $r \in \{0,1\}^k$, *the functions* $F_r(\cdot)$ *and* $I_r(\cdot)$ *are permutations (i.e. bijections) and are inverses of each other.*
- *For every oracle algorithm* T *that has complexity at most* t

$$\left| \mathbb{P}_K[T^{F_K, I_K}() = 1] - \mathbb{P}_{P \in \mathcal{P}_n}[T^{P, P^{-1}}() = 1] \right| \leq \epsilon$$

That is, to any algorithm T that doesn't know K, the functions F_K, I_K look like a random permutation and its inverse.

In applied cryptography literature, pseudorandom permutations are called block ciphers.

58. How do construct pseudorandom permutations

There are a number of block cipher proposals, including the AES standard, that have been studied extensively and are considered safe for the time being. We shall prove later that any construction of pseudorandom functions can be turned into a construction of pseudorandom permutations; also, every construction of pseudorandom generators can be turned into a pseudorandom function, and every one-way function can be used to construct a pseudorandom generator. Ultimately, this will mean that it is possible to construct a block cipher whose security relies, for example, on the hardness of factoring random integers. Such a construction, however, would not be practical.

59. State and prove $|(a)-(d)| \leq 2(\epsilon + ct/2^m) = O(\epsilon + ct/2^m)$, i.e.

PROOF: Recall the proof from last time in which we defined $\overline{Enc}(R,\cdot)$, where R is a truly random function. Given messages M, M' and a cryptanalytic algorithm T, we considered:

- (a) $\mathbb{P}_K[T^{Enc(K,\cdot)}(Enc(K,M)) = 1]$
- (b) $\mathbb{P}_R[T^{\overline{Enc}(R,\cdot)}(\overline{Enc}(R,M)) = 1]$
- (c) $\mathbb{P}_R[T^{\overline{Enc}(R,\cdot)}(\overline{Enc}(R,M')) = 1]$
- (d) $\mathbb{P}_K[T^{Enc(K,\cdot)}(Enc(K,M')) = 1]$

We were able to show in the previous proof that $|(a)-(b)| \leq \epsilon$, $|(c)-(d)| \leq \epsilon$, and $|(b)-(c)| \leq t/2^m$, thus showing that $|(a)-(d)| \leq 2\epsilon + t/2^m$. Our proof will follow similarly.

We will first show that for any M

$$\left| \mathbb{P}_K[T^{Enc(K,\cdot)}(Enc(K,M)) = 1] - \mathbb{P}_R[T^{\overline{Enc}(R,\cdot)}(\overline{Enc}(R,M)) = 1] \right| \leq \epsilon$$

hence showing $|(a)-(b)| \leq \epsilon$ and $|(c)-(d)| \leq \epsilon$. Suppose for a contradiction that this is not the case, i.e. $\exists M = (M_1, \ldots, M_c)$ and $\exists T$ where T is of complexity $\leq t - O(cm)$ such that

$$\left| \mathbb{P}_K[T^{Enc(K,\cdot)}(Enc(K,M)) = 1] - \mathbb{P}_R[T^{\overline{Enc}(R,\cdot)}(\overline{Enc}(R,M)) = 1] \right| > \epsilon$$

Define $T'^{O(\cdot)}(\cdot)$ as a program which simulates $T(O(M))$. (Note that T' has complexity $\leq t$). Noting that $T'^{Enc(K,\cdot)}() = T^{Enc(K,\cdot)}(Enc(K,M))$ and $T'^{\overline{Enc}(R,\cdot)}() = T^{\overline{Enc}(R,\cdot)}(\overline{Enc}(R,M))$, this program T' would be a counterexample to F being (t,ϵ)-secure.

Now we want to show that $\forall M = M_1, \ldots, M_c$, $\forall M' = M'_1, \ldots, M'_c$, and $\forall T$ such that the complexity of T is $\leq t - O(cm)$,

$$\left| \mathbb{P}_R[T^{\overline{Enc}(R,\cdot)}(\overline{Enc}(R,M)) = 1] - \mathbb{P}_R[T^{\overline{Enc}(R,\cdot)}(\overline{Enc}(R,M')) = 1] \right| \leq 2ct/2^m$$

As in the previous proof, we consider the requests T may make to the oracle $\overline{Enc}(R,\cdot)$. The returned values from the oracle would be $r_k, R(r_k) \oplus M_1^k, R(r_k+1) \oplus M_2^k, \ldots, R(r_k + (c-1)) \oplus M_c^k$, where k ranges between 1 and the number of requests to the oracle. Since T has complexity limited by t, we can assume $1 \leq k \leq t$. As before, if none of the $r_k + i$ overlap with $r + j$ (for $1 \leq i, j \leq c$) then T only sees a random stream of bits from the oracle. Otherwise, if $r_k + i = r + j$ for some i, j, then T can recover, and hence distinguish,

M_j and M'_j. Hence the probability of T distinguishing M, M' is ϵ plus the probability of a collision.

Note that the kth oracle request will have a collision with some $r+j$ iff $r-c < r_k \leq r+(c-1)$. If we have $r \leq r_k \leq r+(c-1)$ then obviously there is a collision, and otherwise $r-c < r_k < r$ so $r-1 < r_k + (c-1) \leq r + (c-1)$ so there is a collision with $r_k + (c-1)$. If r_k is outside this range, then there is no way a collision can occur. Since r_k is chosen randomly from the space of 2^m, there is a $(2c-1)/2^m$ probability that the kth oracle request has a collision. Hence $2ct/2^m$ is an upper bound on the probability that there is a collision in at least one the oracle requests.

Combining these results, we see that $|(a) - (d)| \leq 2(\epsilon + ct/2^m) = O(\epsilon + ct/2^m)$, i.e.

$$\left|\mathbb{P}_K[T^{Enc(K,\cdot)}(Enc(K,M)) = 1] - \mathbb{P}_K[T^{Enc(K,\cdot)}(Enc(K,M')) = 1]\right| = O(\epsilon + ct/2^m)$$

60.Discuss Encryption Using Pseudorandom Permutations

Here are two ways of using Pseudorandom Functions and Permutations to perform encryption. Both are used in practice.

ECB Mode

The Electronic Code-Book mode of encryption works as follows

- $Enc(K, M) := F_K(M)$
- $Dec(K, M) := I_K(M)$

CBC Mode

In its simplest instantiation the Cipher Block-Chaining mode works as follows:

- $Enc(K, M)$: pick a random string $r \in \{0,1\}^n$, output $(r, F_K(r \oplus M))$
- $Dec(K, (C_0, C_1)) := C_0 \oplus I_K(C_1)$

Note that this similar to (but a bit different from) the scheme based on pseudorandom functions that we saw last time. In CBC, we take advantage of the fact that F_K is now a permutation that is efficiently invertible given the secret key, and so we are allowed to put the $\oplus M$ inside the computation of F_K.

There is a generalization in which one can use the same random string to send several

messages. (It requires synchronization and state information.)

- $Enc(K, M_1, \ldots, M_c)$:
  - pick a random string $C_0 \in \{0,1\}^n$
  - output $(C_0, C_1, \ldots, C_c)$ where $C_i := F_K(C_{i-1} \oplus M_i)$
- $Dec(K, C_0, C_1, \ldots, C_c) := M_1, \ldots, M_c$ where $M_i := I_K(C_i) \oplus C_{i-1}$

61.Discuss Message Authentication

The goal of message authentication is for two parties (say, Alice and Bob) who share a

secret key to ensure the integrity and authenticity of the messages they exchange. When Alice wants to send a message to Bob, she also computes a tag, using the secret key, which she appends to the message. When Bob receives the message, he veri_es the validity of the tag, again using the secret key.

The syntax of an authentication scheme is the following.

An authentication scheme is a pair of algorithms $(Tag, Verify)$, where $Tag(\cdot,\cdot)$ takes in input a key $K \in \{0,1\}^k$ and a message M and outputs a tag T, and $Verify(\cdot,\cdot,\cdot)$ takes in input a key, a message, and a tag, and outputs a boolean answers. We require that for every key K, and very message M

$$Verify(K, M, Tag(K, M)) = True$$

if $Tag(\cdot,\cdot)$ is deterministic, and we require

$$\mathbb{P}[Verify(K, M, Tag(K, M)) = True] = 1$$

if $Tag(\cdot,\cdot)$ is randomized.

In defining security, we want to ensure that an adversary who does not know the private

key is unable to produce a valid tag. Usually, an adversary may attempt to forge a tag for a message after having seen other tagged messages, so our definition of security must ensure that seeing tagged messages does not help in producing a forgery. We provide a very strong definition of security by making sure that the adversary is able to tag no new messages, even after having seen tags of any other messages of her choice.

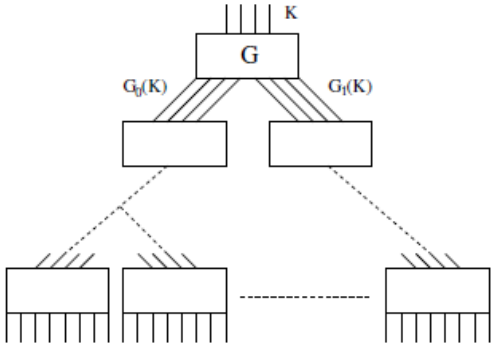
62. Write about the Construction of Pseudorandom Functions.

We now describe the construction of a pseudorandom function from a pseudorandom generator. Let $G : \{0,1\}^n \to \{0,1\}^{2n}$ be a length-doubling pseudorandom generator. Define $G_0 : \{0,1\}^n \to \{0,1\}^n$ such that $G_0(x)$ equals the first n bits of $G(x)$, and define $G_1 : \{0,1\}^n \to \{0,1\}^n$ such that $G_1(x)$ equals the last n bits of $G(x)$.

The the GGM pseudorandom function based on G is defined as follows: for key $K \in \{0,1\}^n$ and input $x \in \{0,1\}^n$:

$$F_K(x) := G_{x_n}(G_{x_{n-1}}(\cdots G_{x_2}(G_{x_1}(K))\cdots)) \tag{11.1}$$

The evaluation of the function F can be visualized by considering a binary tree of depth n, with a copy of the generator G at each node. The root receives the input K and passes the outputs $G_0(K)$ and $G_1(K)$ to its two children. Each node of the tree, receiving an input z, produces the outputs $G_0(z)$ and $G_1(z)$ which are passed to its children if its not a leaf. The input x to the function F_K, then selects a path in this tree from the root to a leaf, and produces the output given by the leaf.



We will prove that if $G : \{0,1\}^n \to \{0,1\}^{2n}$ is a (t, ϵ) pseudorandom generator running in time t_g, then F is a $(t/O(n \cdot t_g), \epsilon \cdot nt)$ secure pseudorandom function.

63.Discuss about Public-Key Cryptography

So far, we have studied the setting in which two parties, Alice and Bob, share a secret key K and use it to communicate securely over an unreliable channel. In many cases, it is not difficult for the two parties to create and share the secret key; for example, when we connect a laptop to a wireless router, we can enter the same password both into the router and into the laptop, and, before we begin to do online banking, our bank can send us a password in the physical mail, and so on.

In many other situations, however, the insecure channel is the only communication device available to the parties, so it is not possible to share a secret key in advance. A general problem of private-key cryptography is also that, in a large network, the number of required secret keys grows with the square of the size of the network.

In public-key cryptography, every party generates two keys: a secret key SK and a public key PK. The secret key is known only to the party who generated it, while the public key is known to everybody.

(For public-key cryptosystems to work, it is important that everybody is aware of, or has

secure access to, everybody else's public key. A mechanism for the secure exchange of public keys is called a Public Key Infrastructure (PKI). In a network model in which adversaries are passive, meaning that they only eavesdrop on communication, the parties can just send each other's public keys over the network. In a network that has active adversaries, who can inject their own packets and drop other users' packets, creating a public-key infrastructure is a very difficult problem, to which we may return when we talk about network protocols.

For now we assume that either the adversary is passive or that a PKI is in place.)

As in the private-key setting, we will be concerned with two problems: privacy, that is

the communication of data so that an eavesdropper can gain no information about it, and

authentication, which guarantees to the recipient the identity of the sender. The first task

is solved by public-key encryption and the second task is solved by signature schemes.

64. Write about El Gamal Encryption.

The El Gamal encryption scheme works as follows. Let $\mathcal{D}$ be a distribution over $(\mathbb{G}, g, q)$ that satisfies the Decision Diffie-Hellman assumption:

- $G()$ samples $(\mathbb{G}, g, q)$, and picks a random number $x \in \{0, \ldots, q-1\}$.
  - $PK = (\mathbb{G}, g, q, g^x)$
  - $SK = (\mathbb{G}, g, q, x)$
- $E((\mathbb{G}, g, q, a), m)$:
  - pick at random $r \in \{0, \ldots, q-1\}$
  - output $(g^r, a^r \cdot m)$
- $D((\mathbb{G}, g, q, x), (c_1, c_2))$
  - Compute $b := c_1^x$
  - Find the multiplicative inverse b' of b
  - output $b' \cdot c_2$

The decryption algorithm works as follows. c_1 is g^r (as returned by E), so $b = g^{rx}$. c_2, as returned by E, is $a^r \cdot m$, where a is g^x. This means that $c_2 = g^{rx} \cdot m$. We see that $c_2 = b \cdot m$, which is why multiplying c_2 by b^{-1} correctly yields m.

65.State Protocol for 3-Coloring

We assume we have a (t, ϵ)-secure commitment scheme (C, O) for messages in the set $\{1, 2, 3\}$.

The prover P takes in input a 3-coloring graph $G = ([n], E)$ (we assume that the set of vertices is the set $\{1, \ldots, n\}$ and use the notation $[n] := \{1, \ldots, n\}$) and a proper 3-coloring $\alpha : [n] \to \{1, 2, 3\}$ of G (that is, α is such that for every edge $(u, v) \in E$ we have $\alpha(u) \neq \alpha(v)$). The verifier V takes in input G. The protocol, in which the prover attempts to convince the verifier that the graph is 3-colorable, proceeds as follows:

- The prover picks a random permutation $\pi : \{1, 2, 3\} \to \{1, 2, 3\}$ of the set of colors, and defines the 3-coloring $\beta(v) := \pi(\alpha(v))$. The prover picks n keys $K_1, \ldots, K_n$ for (C, O), constructs the commitments $c_v := C(K_v, \beta(v))$ and sends $(c_1, \ldots, c_n)$ to the verifier;
- The verifier picks an edge $(u, v) \in E$ uniformly at random, and sends (u, v) to the prover;

- The prover sends back the keys K_u, K_v;
- If $O(K_u, c_u)$ and $O(K_v, c_v)$ are the same color, or if at least one of them is equal to $FAIL$, then the verifier rejects, otherwise it accepts

66. State and prove The protocol is complete and it has soundness error at most (1 -1/ |E|).

PROOF: The protocol is easily seen to be complete, since if the prover sends a valid 3-coloring, the colors on endpoints of every edge will be different.

To prove the soundness, we first note that if any commitment sent by the prover opens to an invalid color, then the protocol will fail with probability at least $1/|E|$ when querying an edge adjacent to the corresponding vertex (assuming the graph has no isolated vertices - which can be rivially removed). If all commitments open to valid colos, then the commitments define a 3-coloring of the graph. If the graph is not 3-colorable, then there must be at least one edge e both of whose end points receive the same color. Then the probability of the verifier rejecting is at least the probability of choosing e, which is $1/|E|$. □

Repeating the protocol k times sequentially reduces the soundness error to $(1 - 1/|E|)^k$; after about $27 \cdot |E|$ repetitions the error is at most about 2^{-40}.

67. State Simulability

We now describe, for every verifier algorithm V^*, a simulator S^* of the interaction between V^* and the prover algorithm.

The basic simulator is as follows:

Algorithm S^*_{1round}

- Input: graph $G = ([n], E)$
- Pick random coloring $\gamma : [n] \to \{1, 2, 3\}$.
- Pick n random keys $K_1, \ldots, K_n$
- Define the commitments $c_i := C(K_i, \gamma(i))$
- Let (u, v) be the 2nd-round output of V^* given G as input and $c_1, \ldots, c_n$ as first-round message
- If $\gamma(u) = \gamma(v)$, then output FAIL
- Else output $((c_1, \ldots, c_n), (u, v), (K_u, K_v))$

And the procedure $S^*(G)$ simply repeats $S^*_{1round}(G)$ until it provides an output different from $FAIL$.

It is easy to see that the output distribution of $S^*(G)$ is always *different* from the actual distribution of interactions between P and V^*: in the former, the first round is almost always a commitment to an invalid 3-coloring, in the latter, the first round is always a valid 3-coloring.

We shall prove, however, that the output of $S^*(G)$ and the actual interaction of P and V^* have *computationally indistinguishable* distributions provided that the running time of V^* is bounded and that the security of (C, O) is strong enough.

For now, we prove that $S^*(G)$ has efficiency comparable to V^* provided that security of (C, O) is strong enough.

68. State and prove the following

Suppose that (C, O) is $(t + O(nr), \epsilon/(n \cdot |E|))$-secure and C is computable in time $\leq r$ and that V^* is a verifier algorithm of complexity $\leq t$.

Then the algorithm S^*_{1round} as defined above has probability at most $\frac{1}{3} + \epsilon$ of outputting $FAIL$.

The proof of the theorem relies on the following result

Lemma

Fix a graph G and a verifier algorithm V^* of complexity $\leq t$.

Define $p(u, v, \alpha)$ to be the probability that V^* asks the edge (u, v) at the second round in an interaction in which the input graph is G and the first round is a commitment to the coloring α.

Suppose that (C, O) is $(t + O(nr), \epsilon/n)$-secure, and C is computable in time $\leq r$.

Then for every two colorings α, β and every edge (u, v) we have

$$|p(u, v, \alpha) - p(u, v, \beta)| \leq \epsilon$$

PROOF: If $p(u, v, \alpha)$ and $p(u, v, \beta)$ differ by more than ϵ for any edge (u, v), then we can define an algorithm which distinguishes the n commitments corresponding to α from the n commitments corresponding to β. A simply runs the verifier given commitments for n colors and outputs 1 if the verifier selects the edge (u, v) in the second round.

Then, by assumption, A ϵ-distinguishes the n commitments corresponding to α from the n commitments corresponding to β in time $t + O(nr)$. However, by Theorem 85, this means that (C, O) is not $(t + O(nr), \epsilon/n)$-secure which is a contradiction.

Given the lemma, we can now easily prove the theorem.

PROOF: (of Theorem 87) The probability that S^*_{1round} outputs $FAIL$ is given by

$$\mathbb{P}\left[S^*_{1round} = FAIL\right] = \frac{1}{3^n} \cdot \sum_{c \in \{1,2,3\}^n} \sum_{\substack{(u,v) \in E \\ c(u) \neq c(v)}} p(u, v, c)$$

Let $\mathbf{1}$ denote the coloring which assigns the color 1 to every vertex. Then using Lemma we bound the above as

$$\begin{aligned}
\mathbb{P}\left[S^*_{1round} = FAIL\right] &\leq \frac{1}{3^n} \cdot \sum_{c \in \{1,2,3\}^n} \sum_{\substack{(u,v) \in E \\ c(u) \neq c(v)}} (p(u, v, \mathbf{1}) + \epsilon) \\
&= \sum_{(u,v) \in E} p(u, v, \mathbf{1}) \left(\sum_{c : c(u) \neq c(v)} \frac{1}{3^n} \right) + \epsilon \\
&= \frac{1}{3} \sum_{(u,v) \in E} p(u, v, \mathbf{1}) + \epsilon \\
&= \frac{1}{3} + \epsilon
\end{aligned}$$

where in the second step we used the fact that $c(u) \neq c(v)$ for a 1/3 fraction of all the colorings and the last step used that the probability of V^* selecting some edge given the coloring $\mathbf{1}$ is 1.

69.Define Computational Zero Knowledge

(Computational Zero Knowledge) *We say that a protocol (P,V) for 3-coloring is (t,ϵ) computational zero knowledge with simulator overhead $so(\cdot)$ if for every verifier algorithm V^* of complexity $\leq t$ there is a simulator S^* of complexity $\leq so(t)$ on average such that for every algorithm D of complexity $\leq t$, every graph G and every valid 3-coloring α we have*

$$|\mathbb{P}[D(P(G,\alpha) \leftrightarrow V^*(G)) = 1] - \mathbb{P}[D(S^*(G)) = 1]| \leq \epsilon$$

Suppose that (C,O) is $(2t + O(nr), \epsilon/(4 \cdot |E| \cdot n))$-secure and that C is computable in time $\leq r$.

Then the protocol defined above is (t,ϵ) computational zero knowledge with simulator overhead at most $1.6 \cdot t + O(nr)$.

70. Discuss about the structure of arrow diagram.

Arrow diagrams illustrate a large variety of semantics, which increases the difficulty of

their interpretation. To tackle this problem, we first develop a method for making rough

interpretation of an arrow diagram from its structural pattern alone. As the foundation

of this method, this section summarizes the formal structures of arrow diagrams developed by Kurata and Egenhofer (2005).

The components of an arrow diagram are diagrammatic elements that an arrow symbol originates from, traverse, or points to. We classify the components of arrow diagrams into the following five types:

• An object takes an action, either independently (e.g., a person in Fig. 4a) or as a result of interaction (e.g., a bag in Fig. 4a).

• An event occurs in time, and is characterized by a set of changes. An event occurs

over an interval (e.g., snow in Fig. 4b) or at an instant (e.g., a traffic accident in Fig. 4b).

• A location is a position in space. It may be a point (e.g., a place in Maine in Fig. 4c) or a homogeneous area (e.g., Maine).

• A moment is a position in time. It may be an instant (e.g., 8:20 in Fig. 5a) or a homogeneous interval (e.g., morning).

• A note is a short description that modifies an arrow symbol (e.g., send in Fig. 4a) or another component (e.g., Mr. K in Fig. 4a and You are here in Fig. 4c). A note and the modified component are placed adjacently to each other or connected with each other by an arrow symbol.

For simplification, an object, an event, a moment, a location, and a note are sometimes
denoted as O, E, M, L, and N, respectively.

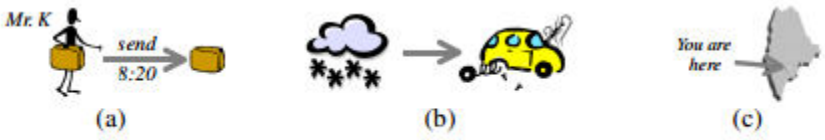


Fig. 4. Arrow diagrams that contain various types of components

Although a component may be mentioned by an icon, a text, or a specific position in
a background drawing, this classification is not concerned with such a descriptive style
of the component. We assume that automated interpreters of arrow semantics will distinguish these component types based on their knowledge base.

Components of an arrow diagram are located in front of the arrow's head, behind the
arrow's tail, or along the arrow's body. We, therefore, consider that an arrow symbol

identifies three different areas where the components of the arrow diagram are located

(Fig. 5). These areas are called component slots, and they are further classified into a

tail slot, a head slot, and a body slot. Each component in an arrow diagram is uniquely

assigned to one of these three slots, thereby making the distinction of tail components,

body components, and head components. The component slots need to bedistinguished, because the same symbols, used in different slots, illustrate significantly

different semantics (Kurata and Egenhofer 2005).

An arrow diagram has three slots, where five types of components may be located. The combination of the types of components in the three slots composes a certain pattern that is specific to every arrow diagram. These patterns are described as

Tail slot Body slot Head slot

Fig. 5. Three component slots associated with an arrow symbol

([M|E|L|O|N]*, [M|E|L|O|N]*, [M|E|L|O|N]*), where [x]* means empty or a sequence

of any number of x, x|y means x or y but not both, and the three elements in parentheses

indicate the types of components in tail, body, and head slot, respectively. For example,

the structures of arrow diagrams in Fig. 4a-c are described as (ON, MN, O), (E, -, E),

and (N, -, L), respectively. These patterns capture fundamental structures of arrow diagrams, since they capture the alignment of components while abstracting the individual difference and the absolute location of the components.

71. Explain Arrow Diagram.

Arrows are major components of diagrams. Arrows appear in various types of diagrams,
such as traffic signs, guideboards, route maps, flowcharts, and illustrations. One reason for such popularity is that arrows capture a large variety of semantics with their simple shape. Another reason is that the existence of arrows encourages people to interpret causal and functional aspects in a diagram.

For instance, Fig. 1 contains only a few words and some
arrow symbols over a background map, but people easily read the mechanism of a spatio-temporal process—the El Niño effect in the Southeastern Pacific Ocean indirectly
influences the rise of tofu price in Japan. In this way, arrows are powerful tools that
facilitate the communication of spatial and temporal knowledge in a static diagram.

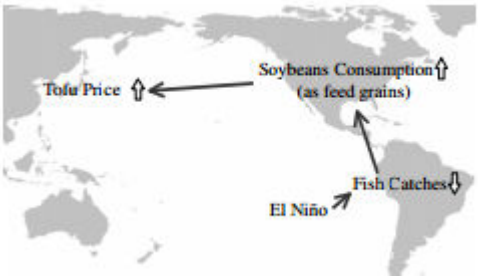


Fig. 1. A diagram with arrows, which illustrates a spatio-temporal process that the El Niño effect (i.e., sea temperature rise in the Southeastern Pacific Ocean) indirectly influence the rise of the tofu price in Japan

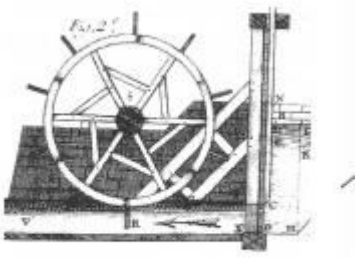
The combination of arrow symbols and the related elements is considered as a unit of
syntax, and called an arrow diagram. Then, these arrow-related elements are called the
components of the arrow diagram. An arrow diagram must have at least one arrow symbol and one component. As a first step, we consider the arrow diagram that contains only a uni-directional arrow symbol and its related elements. Bi-directional arrows are
not considered, because they are regarded as a synthesis of two oppositely-directed arrow symbols. Also, independent arrow symbols, such as arrow-shaped traffic signs
indicating curving roads and map symbols indicating north direction, are not discussed
in this paper, because they have no components.

72. Explain the usage of Arrow diagram

One of the primary usages of arrow diagrams is to express a direction. Gombrich

(1990) reported a very early example of an arrow diagram, where an arrow symbol was

used to represent the direction of a water stream (Fig.).



(a)

Arrow diagrams may also refer to metaphorical directions. Upward directions are metaphorically associated with increase or improvement, whereas downwards directions are associated with decrease or debasement (Lakoff and Johnson 1980). Accordingly, upward and downward arrow symbols are used to illustrate those semantics.

An arrow symbol illustrating a direction has a diagrammatic freedom of length. This

freedom allows the representation of a directed quantity, which is called a vector. A

vector is a quantity that is specified by a direction and a magnitude.

Another traditional usage of arrow diagrams is to illustrate a spatial movement.

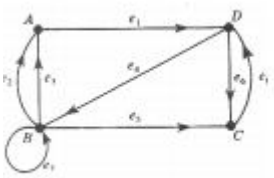
Spatial movement is an event where an entity changes its spatial position continuously.

Arrow diagrams are used not only in a spatial context, but also in a temporal context.

For instance, timetables often contain arrow diagrams, each of which illustrates that

something continues over a certain period

An arrow diagram visualizes the presence of a directed relation between two components. In mathematics, a set of directed binary relations is modeled as a directed graph, which is often visualized using arrow diagrams.



73. Explain Huffman Codes

Usually characters are represented in a computer with fix length bit strings. Huffman codes provide an alternative representation with variable length bit strings, so that shorter strings are used for the most frequently used characters. As an example assume that we have an alphabet with four symbols: A = {a, b, c,d}.Two bits are enough for representing them, for instance a = 11, b = 10, c = 01, d = 00 would be one such representation. With this encoding n-character words will have 2n bits. However assume that they do not appear with the same frequency, instead some are more frequent that others, say a appears with a frequency of 50%, b 30%, c 15% and d 5%. Then the following enconding would be more efficient than the fix length encoding: a = 1, b = 01, c = 001, d = 000.

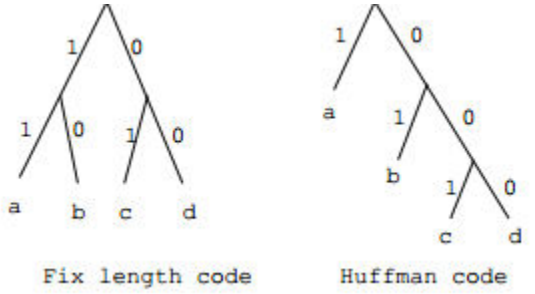
Now in average an n-character word will have 0.5n a's, 0.3n b's, 0.15n c's and 0.05n d's, hence its length will be $0.5n \cdot 1+0.3n \cdot 2+0.15n \cdot 3+0.05n \cdot 3 = 1.7n$, which is shorter than 2n. In general the length per character of a given encoding with characters a1,a2,...,an whose frequencies are f1,f2,...,fn is

$$\frac{1}{F}\sum_{k=1}^{n} f_k\, l(a_k)\,,$$

where $l(a_k)$ = length of a_k and F = $\sum_{k=1}^{n} f_k\, l(a_k)$. The problem now is, given an alphabet and the frequencies of its characters, find an optimal encoding that provides minimum average length for words.

74. Explain Constructing an Optimal Huff man Code

An optimal Huff man code is a Huff man code in which the average length of the symbolsis minimum. In general an optimal Huff man code can be made



as follows. First we list the frequencies of all the codes and represent the symbols as vertices (which at the end will be leaves of a tree).

Then we replace the two smallest frequencies f1 and f2 with their sum f1+ f2, and join the corresponding two symbols to a common vertex above them by two edges, one labeled 0 and the other one labeled 1.

Than common vertex plays the role of a new symbol with a frequency equal to f1+f2. Then we repeat the same operation with the resulting shorter list of frequencies until the list is reduced to one element and the graph obtained becomes a tree.

75. Write about Adjacency Matrix.

Let G be an n-vertex directed graph. Let A be the $n \times n$ *adjacency matrix* of the graph G. Element $a_{ij} = 1$ if and only if the edge $(i, j) \in G$. All other elements are zero. A row of A lists the nodes at the tip of the *outgoing edges* while a column of A lists the nodes at the tail of the *incoming edges*.

Powers of an Adjacency Matrix

Consider the power A^2 of an adjacency matrix. What does element $(A^2)_{ij}$ mean? Element $(A^2)_{ij}$ is the dot product $A(i,:)A(:,j)$. The k-th term in dot product equals 1 if and only if $A(i,k) = A(k,j) = 1$. In other words, it is 1 if and only if there is a path from $i \to k \to j$. The dot product thus *counts the number of paths of length exactly* 2 *from* i *to* j. This generalizes to arbitrary nonnegative powers: $(A^r)_{ij}$ is the number of *paths of length exactly* r *from* i *to* j.

Relevant questions:

- **What if A is symmetric?**
  If A is symmetric, then there is an edge (i, j) and an edge (j, i), so A is basically undirected.
- **If $A^r = 0$, then what can we say about the structure of G? And if $A^r \neq 0$ for all r?**
  If $A^r = 0$, then there are *no paths of length exactly* r, so the *longest path is* $< r$ and the graph has *no cycles*. Conversely, $A^r \neq 0$ for all r if and only if G has a cycle.
- **What is $I + A + A^2 + \cdots + A^r$?**
  The sum represents the number of paths of any length less than or equal to r between every pair of vertices. Drop the identity and get only the nontrivial paths.
- $I + A + A^2 + \cdots + A^r - (A + A^2 + \cdots + A^r) = I$.
  This implies that $(I - A)^{-1} = I + A + A^2 + \cdots + A^r$.
- **How can we interpret a symmetric permutation $P^T AP$?**
  It is a relabeling of the vertices and does not change the structure of the graph. Two graphs are isomorphic if and only if there is a symmetric permutation from the adjacency graph of one to the adjacency graph of the other.
- **What if A is (now or after symmetric permutation) strictly upper triangular?**
  The vertices have been labeled such that all arcs go from lower to higher labels. This is what is known as a topological ordering and it also means that the graph has no cycles.
- **What if A is (now or after symmetric permutation) banded?**
  The vertices have been labeled so that the edges are local. The graph is in a sense long and thin.

76. Write about Incidence Matrix.

An undirected or directed graph G with n vertices and m edges can be represented as an *incidence matrix* A. Arbitrarily number the edges $1, 2, \ldots, m$ and the vertices $1, 2, \ldots, n$. For edge i: (j, k), there is an entry $a_{ij} = -1$ and an entry $a_{ik} = 1$. All other elements are zero.

Interpretation of Ax

The matrix-vector product Ax can be understood in the following way. The vector x has one component for each vertex in G. Each component in the product Ax contains the difference between the components at the head of the edge and at the tail of the edge. The matrix-vector product Ax therefore maps values on the edges to differences on the edges.

Interpretation of Null space

The null space of A are those vectors that give zero difference over all edges.

Interpretation of $A^T x$

The matrix-vector product $A^T x$ can be understood in the following way. The vector x has one component for each edge in G. The entry $(A^T)_{ij} = -1$ iff edge j is outgoing from vertex i. Similarly, the entry $(A^T)_{ij} = 1$ iff edge j is incoming to vertex i. Therefore, the matrix-vector product $A^T x$ sums the incoming flow from each edge and subtracts the outgoing flow. In other words, each component is the *net flow* to/from that vertex.

Interpretation of Left Nullspace

The left nullspace of A are those vectors that give zero net flow on all vertices. The only nontrivial flows are constant flows around cycles of G.

Interpretation of $A^T Ax$

Since Ax computes the difference over each edge, the product $A^T(Ax)$ computes the net flow induced by the values at the vertices.

77. Tree

Trees

Mathematically speaking trees are a special class of a graph.

The relationship of a trees to a graph is very important in solving many problems in Maths and Computer Science

However, in computer science terms it is sometimes convenient to think of certain trees (especially rooted trees — more soon) as separate data structures.

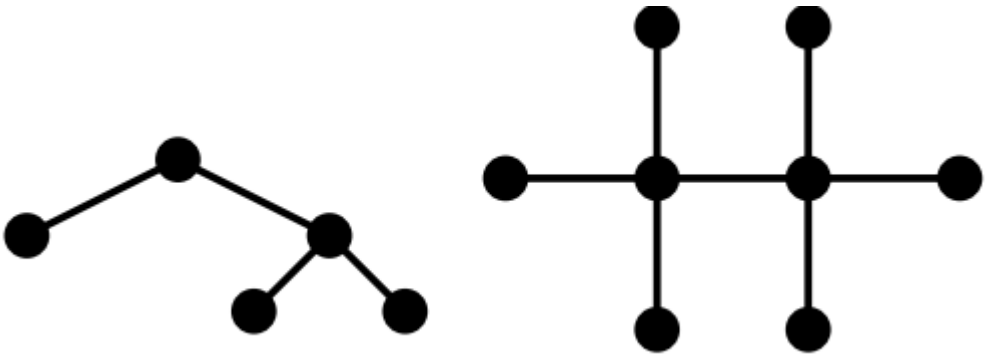
They have they own variations of data structure

They have many specialised algorithms to traverse, search etc.

Very common data structure in almost all areas of computer science.

A tree T is a connected graph that has no cycles.

(Simple Trees).



Let T be a graph with n vertices.
Then the following statetments are equivalent.
_ T is connected and has no cycles.
_ T has n - 1 edges and has no cycles.
_ T is connected and has n - 1 eges.
_ T is connected and the removal of any edge disconnects T.
_ Any two vertices of T are connected by exactly one path.
_ T contains no cycles, but the additon of any new edge creates a cycle.

1. **Find the optimal solution for the assignment problem with the following cost matrix**

salesman	W	X	Y	Z
A	11	17	8	16
B	9	7	12	6
C	13	16	15	12
D	14	10	12	11

STEP 1:

Choose the least element in each row and subtract it from all the el;ements of that row.

3	9	0	8
3	1	6	0
1	4	3	0
4	0	2	1

STEP 2:

Choose the least element in each column and subtract it from all the el;ements of that column.

2	9	0	8
2	1	6	0
0	4	3	0
3	0	2	1

STEP 3:

Test whether we can choose only one zero from each column and from each row.draw a circle covering these chosen zeros.

		0	
			0

0			
	0		

The Optimal assignment is

Area	cost(rs)	
4	8	salesman
Z	6	
W	13	
x	10	

total min cost

2. A project work consist of 4 major jobs for which 4 major contractors have submitted tendors. The tendor document quoted in thousands of rupees are given with the matrix as

	J1	J2	J3	J4
C	15	27	35	20
C	21	29	33	17
C	17	25	37	15
C	14	31	39	21

Find the assignment which minimizes the total of the project cost.

Step 1:
0 12 29 5
4 12 16 0
2 10 22 0
0 17 25 7

Step 2:
0 2 4 5
4 2 0 0

step3:
0 2 4 5
4 2 0 0

step 4:
0 0 2 3
6 2 0 0

2 0 6 0 2 0 6 0 4 0 6 0
0 7 9 7 0 7 9 7 0 5 7 5

The optimal assignment
Contractor job cost(rs)
C1-> J1 27
C2-> J2 33
C3-> J3 15
C4-> J4 14

89

3. Four jobs can be processed on 4 different machines, one job on one machine. Resulting times in minutes vary with assignments. They are given below

Machines

	A	B	C	D
I	42	35	28	21
II	30	25	20	15
III	30	25	20	15
IV	24	20	16	12

FIND THE OPTIMUM ASSIGNMENT

Step1:

21 14 7 0
15 10 5 0
15 10 5 0
12 8 4 0

step2:

9 6 3 0
3 2 1 0
3 2 1 0
0 0 0 0

8 5 2 0
2 1 0 0
2 1 0 0

Step3:

Step4:

Step5:

The optimal	7 2 2 0 1 0 0 0 1 0 0 0 0 0 0 0

I D 21
II C 20
III B 25
IV A 24

———

90

4. Maximization case in assignment assignment problem

The payoff elements of the assignment problem may represent revenue or profits instead of costs to that the objective will be to maximize the total revenue or profit. The problem of maximization can be converted into a maximization care by selecting the largest element among all elements of the profit matrix 2 then subtracting it from all other elements in the matrix we can then proceed as usual

and obtain the optimum solution by adding the original values of these cells. through which the assignment have been made.

Prohibited assignments:

Sometimes due to certain reason a particular resource say a man or machine cannot be assigned to perform a particular activity say territory or job. in such cases the cost of performing that particular activity by a particular resource is considered to be very large writes as M orD. so as to prohibit the entry of this pair of resource activity into the final solution.

Maximization

	A	B	C	D
P	6	10	14	12
Q	7	5	3	4
R	6	7	10	10
S	20	10	15	15

Find out the assignment of operator which will maximize the profit.

Step 1:

	A	B	C	D
P	14	10	6	8
Q	13	15	17	16
R	14	13	10	10
S	0	10	5	5

Step 2:

	A	B	C	D
P	6	2	2	0
Q	0	2	4	3
R	4	3	0	0
S	0	10	5	5

Step 3:

6 0 4 0
0 0 4 3
4 1 0 0
0 7 5 5

5. PERT –CPM

Network analysis is a technique which determines the various sequences of jobs concerning a project and the project completion time. Network analysis has been successfully used to a wide range of significant management problem. Two methods of this technique which are widely used.

1. critical path method (CPM)
2. The program evaluation and review technique(PERT)

Total float (TF)

This is the amount of time a path of activities could be delayed without affecting the overall project duration

TF=Head event (L)-Tail event (E)-duration

Total float is also defined as the difference between the two finishing time or the difference between the starting time.

TF=LFT-EFT

=LST-EST

Free float (FF)

This is the amount of time an activity can be delayed without affecting the commencement of asubsequent activity of its earliest start time, but may affect the float of a previous activity

FF=Headevent(L)-Tailevent(E)-duration

FF=TF-Head event slack (L-E)

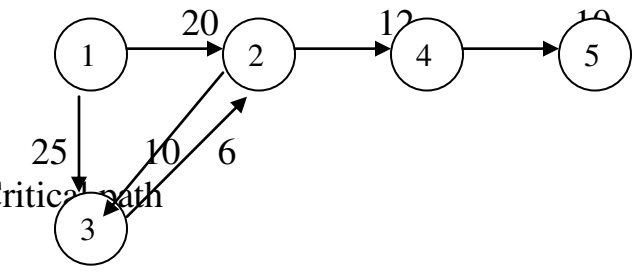
Independent Float (IF)

IF=FF-Tailevent slack (L-E)

6. Activity 1-1 1-3 2-3 2-4 3-4 4-5
Duration 20 25 10 12 6 10

i) Draw the network for the project
ii) Find the critical path and project duration
iii) Find TF for each activity

i)



ii) Critical path

to find ES(earliest start time)

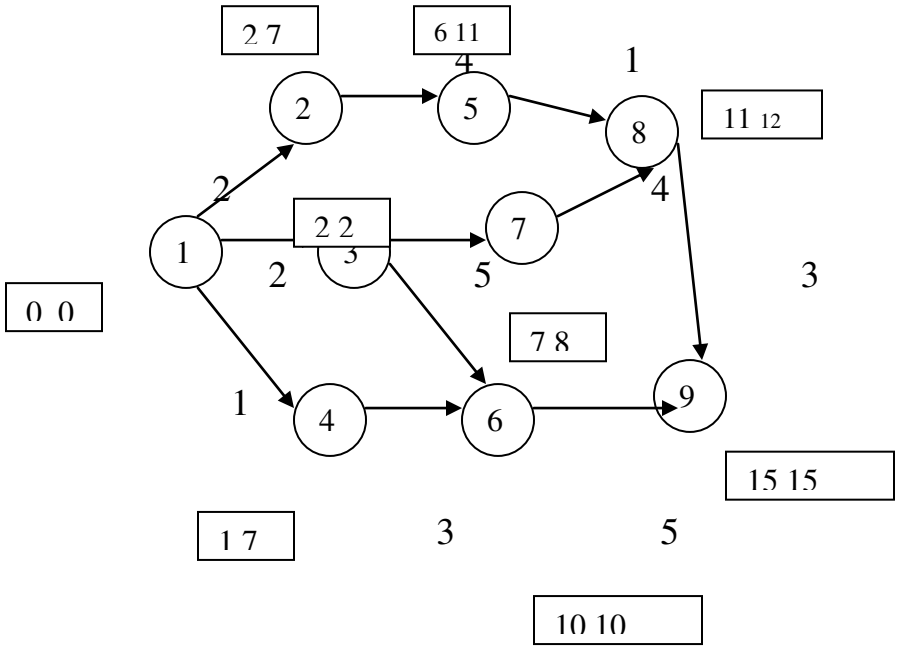
ES1=0
ES2=ES!+20=0+20=20
ES3=max{ES1+25,ES2+10}=max{0+25,20+10}
Max{25,30}=30
ES4=max{ES2+12,ES3+6}
=max{20+12,30+6}=max{32,36}
ES4=36
ES5=ES4+10=36+10=46
To find LF(Latest finishing time)
LF5=46
LF4=LF5-10=46-10=36
LF3=LF4-6=36-6=30
LF2=min(LF4-12,LF3-10)=min{36-12,30-10}
=min{24,20}=20
LF1=min{LF2-20,LF3-25}
=min{0,5}=0

Activity	Duration	EST	EFT	LST	LFT	TF
1-2	20	0	20	0	20	0
1-3	25	0	25	5	30	5
2-3	10	20	30	20	30	0
2-4	12	20	32	24	36	4
3-4	6	30	36	30	36	0
4-5	10	36	46	36	46	0

Therefore CP 1->2->3->4->5
20+10+6+10=46
Project duration=46

7. **Activity 1-2 1-3 1-4 2-5 3-6 3-7 4-6 5-8 6-9 7-8 8-9**
Time 2 2 1 4 8 5 3 1 5 4 3

i. construct the network
ii. find the TF for each activity
iii. find the cpd project duration



ii) to find the critical path(EST)

ES1=0

ES2=ES1+2=0+2=2

ES3=ES1+2=0+2=2

ES4=ES1+1=0+1=1

ES5=ES2+4=2+4=5

ES6=max{ES3+8,ES4+3}

=max{2+8,1+3}={10,4}

ES6=10

ES7=ES3+5=2+5=7

ES8=max{ES5+1,ES7+4}

=max{6+1,7+4} ={7,11}

ES8=11

ES9=max{ES6+5,ES8+3}

=max{10+5,11+33}={15,14}

ES9=15

To find the last finishing time

LS9=15

LS8=LS9-3=15-3=12

LS7=LS8-4=12-4=8
LS6=LS9-5=15-5=10
LS5=LS8-1=12-1=10
LS4=LS6-3=10-3=7
LS3=min{LS6-8,LS7-5}
=min{10-8,8-5}={2,3}
LS3=2
LS2=LS5-4=11-4=7
LS1=min{LS2-2,LS3-2,LS4-1}
=min{7-2,2-2,7-1}=min{5,0,6}
LS1=0

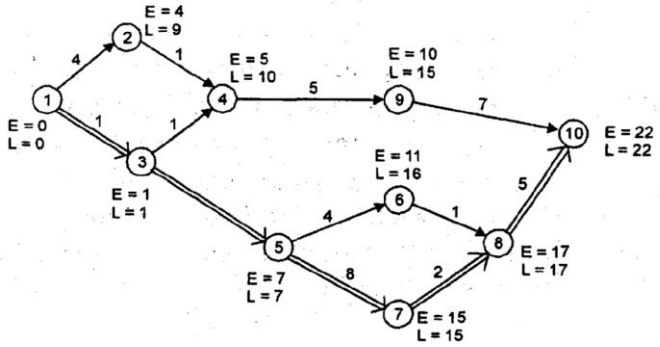
Activity	Duration	EST	EFT	LST	LFT	TF
1-2	2	0	2	5	7	5
1-3	2	0	2	0	2	0
1-4	1	0	1	6	7	6
2-5	4	2	6	7	11	5
3-6	8	2	10	2	10	0
3-7	5	2	7	3	8	1
4-6	3	1	4	7	10	6
5-8	1	6	7	11	12	5
6-9	5	10	15	10	15	0
7-8	4	7	11	8	12	1
8-9	3	11	14	12	15	1

Problem 11.16. A project has the following time schedule :

Activity	1 – 2	1 – 3	2 – 4	3 – 4	3 – 5	4 – 9	5 – 6	5 – 7	6 – 8	7 – 8	8 – 9	8 – 10	9 – 10
Time weeks	4	1	1	1	6	5	4	8	1	2	1	5	7

1. Draw Network diagram and find the critical paths.
2. Calculate float on each activity.
Solution. (*i*)

8.



2.

Activity	Duration (weeks)	Start Time		Finish Time		Total Float
		E	T_{LS}	T_{EF}	L	
1 – 2	4	0	5	4	9	5
1 – 3	1	0	0	1	1	0
2 – 4	1	4	9	5	10	5
3 – 4	1	1	9	2	10	8
3 – 5	6	1	1	7	7	0
4 – 9	5	5	10	10	15	5
5 – 6	4	7	12	11	16	5
5 – 7	8	7	7	15	15	0
6 – 8	1	11	16	12	17	5
7 – 8	2	15	15	17	17	0
8 – 10	5	17	17	22	22	0
9 – 10	7	10	15	17	22	5

Critical path 1—3—5—7—8—10 with project duration of 22 weeks.

9. The time estimate for the activities of a PERT network are given below :

Activity	t_0	t_m	t_p
1 – 2	1	1	7
1 – 3	1	4	7
1 – 4	2	2	8
2 – 5	1	1	1
3 – 5	2	5	14
4 – 6	2	5	8
5 – 6	3	6	15

(a) Draw the project network and identify all the path through it.

(b) Determine the expected project length.

(c) Calculate the standard deviation and variance of the project length.

(d) What is the probability that the project will be completed

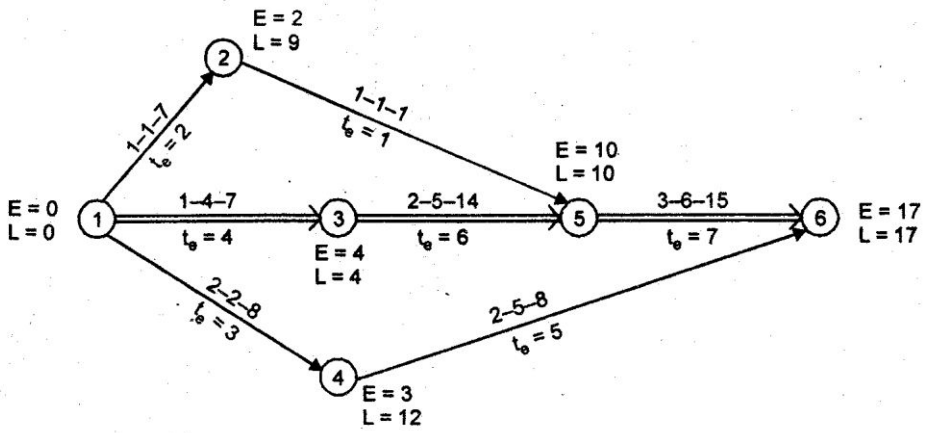
1. At least 4 weeks earlier than expected time.

2. No more than 4 weeks later than expected time.

(e) The probability that the project will be completed on schedule if the schedule completion time is 20 weeks.

(f) What should be the scheduled completion time for the probability of completion to be 90%.

Solution. (a) Network



Activity	t_0	t_m	t_p	$t_e = \frac{t_0 + 4t_m + t_p}{6}$	$\sigma^2 = \frac{(t_p - t_0)^2}{6}$
1 – 2	1	1	7	2	1
1 – 3	1	4	7	4	1
1 – 4	2	2	8	3	1
2 – 5	1	1	1	1	0
3 – 5	2	5	14	6	4
4 – 6	2	5	8	5	1
5 – 6	3	6	15	7	4

Critical path—1 —3—5—6

Project duration = 17 weeks.

(c) Variance of the project length is the sum of the variance of the activities on the critical.

$V_{cp} = V_{1-3} + V_{3-5} + V_{5-6} = 1 + 4 + 4 = 9$

$\sigma^2 = V \Rightarrow \sigma^2 = 9 \Rightarrow \sigma = 3$ weeks.

(d) (i) Probability that the project will be completed at least 4 week earlier than expected time

Expected time (E_p) = 17 weeks

Scheduled time = 17 – 4 = 13 weeks

$$Z = \frac{13-17}{3} = -1.33$$

P(– 1.33) = 1 – 0.9082 = 0.0918

2. Probability that the project will be completed at least 4 weeks later than expected

Time

Expected time = 17 weeks Scheduled time =17+ 4 =21 weeks

$$Z = \frac{21-17}{3} = 1.33$$

P(1.33) = 0.9082 = 90.8%.

(e) Scheduled time = 20 weeks

$$Z = \frac{20-17}{3} = 1$$

P(1) = 84.13%

(f) Value of Z for P = 0.9 is 1.28 (from probability table)

$$1.28 = \frac{T-17}{3}$$

$$T = 17 + 3.84 = 20.84 \text{ weeks.}$$

10. Table below show jobs, their normal thne and and cost estimates for cost and crash time the project

Job	Normal time	Cost	Crash time	Cost
1 – 2	6	1400	4	1900
1 – 3	8	2000	5	2800
2 – 3	4	1100	2	1500
2 – 4	3	800	2	1400
3 – 4	Dummy	–	–	–
3 – 5	6	900	3	1600
4 – 6	10	2500	6	3500
5 – 6	3	500	2	800
		9200		

Indirect cost for the project is Rs. 300 per day.

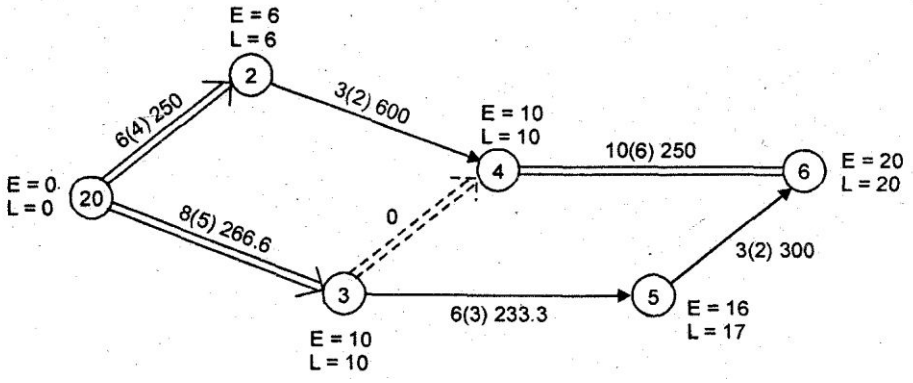
1. Draw the network of the project.

2 What is normal duration and cost of the project'

3 If all activities are crashed, what will be the minimum project duration and corresponding cost.

4 Find the optimum duration and minimum project cost

Solution.

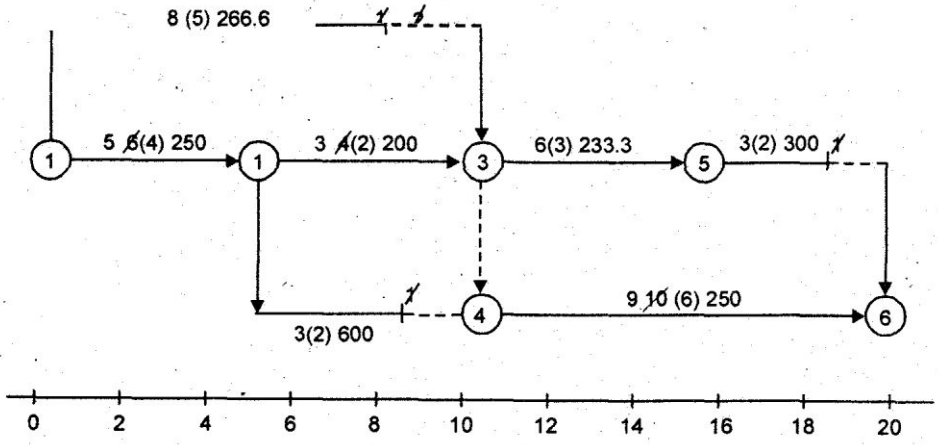


2. Critical path I — 2— 3 — 4 — 6

Normal duration of project = 20 days

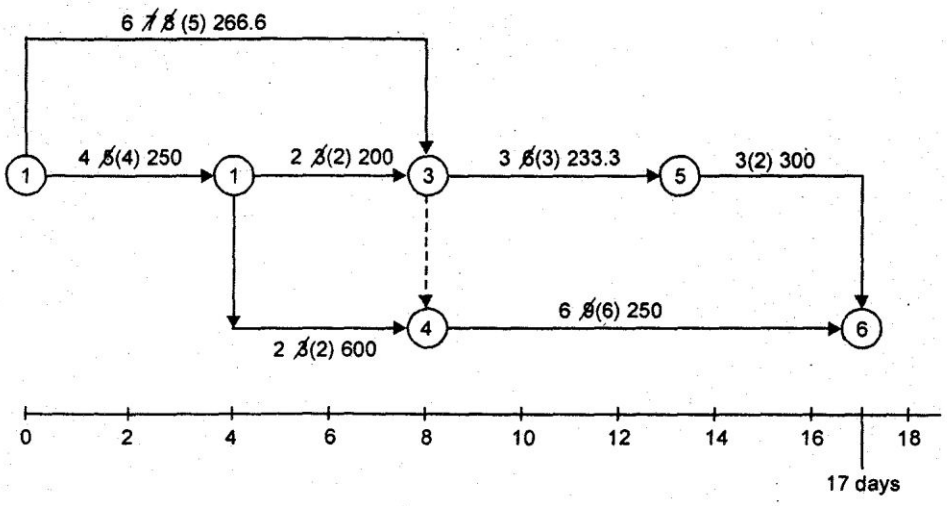
Normal cost of project = Rs 9200.

3. Crashing



Crash following activities one day

Activity	crash cost/day	Activity	Crash cost/day	Activity	crash cost/day
1 – 2	250	2 – 3	200	4–6	250
1 – 3	NIL	2 – 4	NIL	3–5/5–6	NIL
		1 – 3	NIL		
	250		200		250



Activity	Crash cost	Activity	Crash cost	Activity	Crash cost
1 – 2	250	2 – 3	200	4 – 6	250
1 – 3	266.6	1 – 3	266.6	3–5/5–6	233.3
		2 – 4	600		
	516.6		1066.6		483.3
Crahs above activities for one day reduce project duration to 13 days.		Crash above activities for one day project duration = 12 days.		Crash above activities for 3 day reduce project duration to 14 days.	

Minimum project duration 12 days

Duration (days)	Normal cost	Crash cost	Indirect cost	Total cost
20	9200	–	6000	15200
19	9200	200	5700	15100
18	9200	450	5400	15050
17	9200	700	5100	15000
16	9200	1183.3	4800	15183.3
15	9200	1666.6	4500	15366.6
14	9200	2149.9	4200	15549.9
13	9200	2666.5	3900	15766.5
12	9200	3733.1	3600	16533.1

Cost corresponding to mm. project duration 16533.1

Minimum cost of project 15000

Optimum duration = 17 days.

11. The following table gives the activities in a construction project and other relevant information.

Activity	Preceding activity	Normal time (days)	Crash time (days)	Normal cost (Rs.)	Crash cost (Rs.)
1 – 2	–	20	17	600	720
1 – 3	–	25	25	200	200
2 – 3	1-2	10	8	300	440
2 – 4	1-2	12	6	400	700
3 – 4	1-3, 2-3	5	2	300	420
4 – 5	2-4, 3-4	10	5	300	600

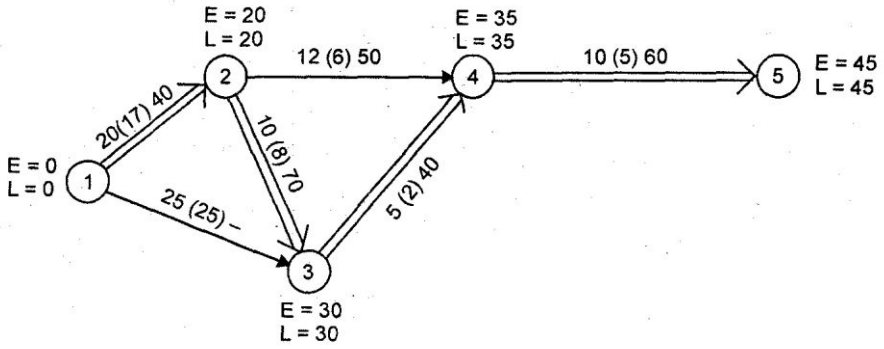
Draw activity network of the project.

(b)Find total float and free float of each activity.

(c) Using the above information "Crash" or shorten the activity step by step untill the shortest duration is reached.

Solution.

1.Activity Network.

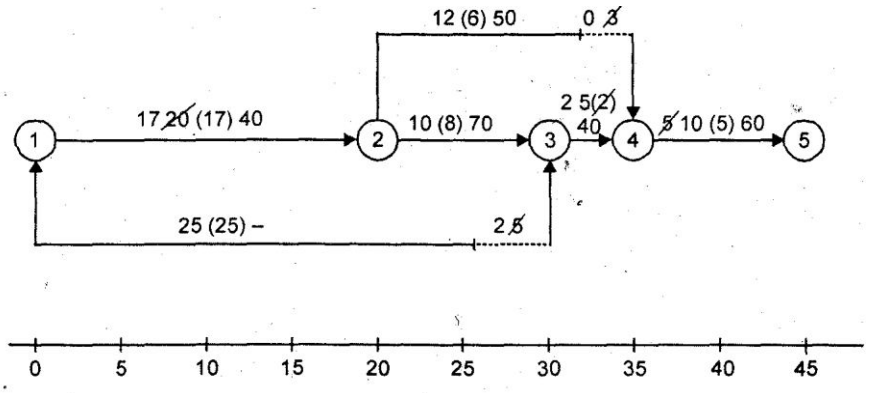


(b)

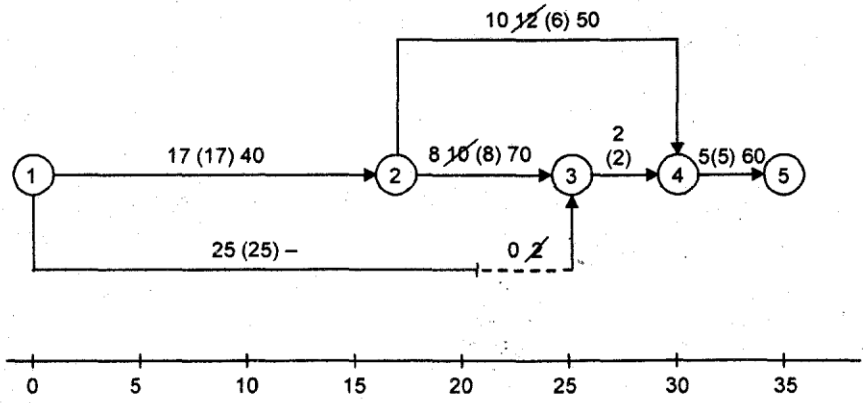
Activity	Duration	Start time		Finish Time		Total Float	Free Float
		E	T_{LS}	T_{EF}	L		
1 – 2	20	0	0	20	20	0	0
1 – 3	25	0	5	25	30	5	5
2 – 3	10	20	20	30	30	0	0
2 – 4	12	20	23	32	35	3	3
3 – 4	5	30	30	35	35	0	0
3 – 5	10	30	35	45	45	0	0

Free float = Total float - Head event slack. Critical path 1-2-3—4-5.

(c) Crashing



Activity	cost	Activity	cost	Activity	cost	Activity	cost
1 – 2	40	2 – 3	70	3 – 4	40	4 – 6	60
1 – 3	NIL	1 – 3	NIL	2 – 4	NIL		
		2 – 4	NIL				
	40		70		40		60
Crash above activities for 3 days Project duration = 42. Crash cost = Rs. 120.				Crash above activities for 3 days project duration = 39 days. Crash cost = Rs. 120.		Crahs activities for 5 days project duration = 34 days Crash cost = Rs. 300.	



Activity	cost
2 - 3	70
1 - 2	NIL
2 - 4	50
	120

Crash above activities for 2 days project duration = 32 days.
Crash cost = Rs. 240.

So fully crashed project minimum project duration = 32 days.
Crash cost = 120 + 120 + 300 + 240 = Rs. 780.

12. The activities, of a project are tabulated below with immediate predecessors and normal and crash time cost.

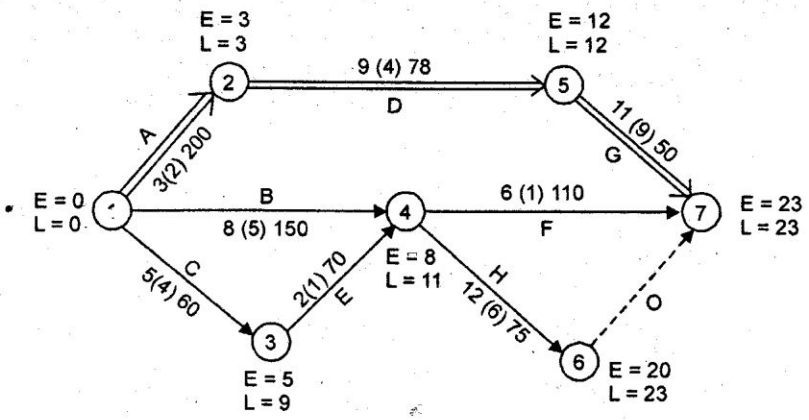
Activity	Immediate predecessor	Normal		Crash	
		cost	time	cost	time
A	–	200	3	400	2
B	–	250	8	700	5
C	–	320	5	380	4
D	A	410	9	800	4
E	C	600	2	670	1
F	B,E	400	6	950	1
H	B,E	550	12	1000	6
G	D	300	11	400	9

1. Draw the network corresponding to normal time

2 Determine the critical path and normal duration and cost of project, if the indirect cost per day is Rs. 8.

3. Suitably reduce the activities so that the normal duration may be reduced by 2 days at minimum cost. Also find the. project cost for this shortened duration.

Solution.



2. Critical path = 1 – 2 – 5 –7

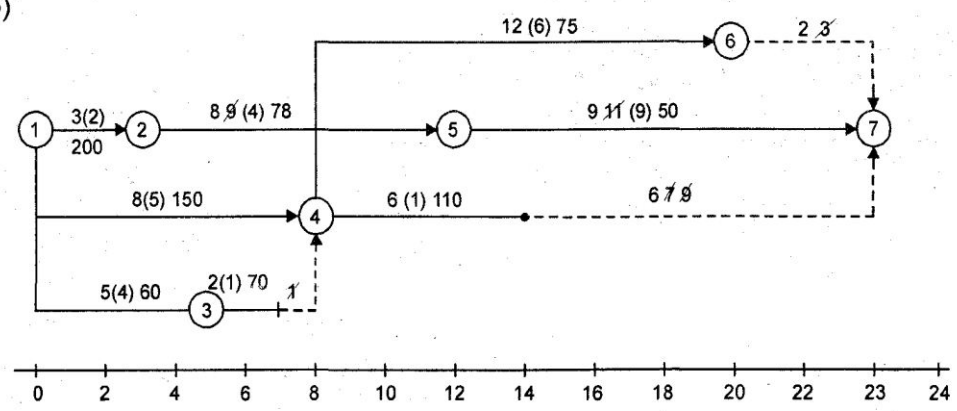
Normal duration = 23 days.

Normal cost = Direct cost + Indirect cost (corresponding to normal time)

= (200 + 250 + 320 + 410 + 600+ 400 + 550 +300 + 8 × 23

= (3030) + (184) = Rs. 3114.

(3)



Activity	Cost	Activity	Cost	Activity	Cost
1 – 2	200	2 – 5	78	5 – 7	50
1 – 3/3–4	NIL	1 – 4/4 – 6/6 – 7	NIL	4 – 7	NIL
1 – 4/4 – 6/6 – 7	NIL	1 – 4/4 – 7	NIL	4 – 6/6 – 7	NIL
		1–3 / 3 – 4	NIL		
	200		78		50
		Reduce above activities for 1 day. Crash cost = 78/- Project duration = 20 days.		Reduce above activities for 2 day. Crash cost = Rs. 100/- Project duration = 21 days.	

Total project cost for reducing 3 days duration.

= Direct cost + Indirect cost + crash cost

= 3030 + 160 + 178 = Rs. 3368.

13. Discuss briefly about Directed Graph

A directed graph (digraph) is a tuple G = (V; E) where V is a (finite) set of vertices and E is a collection of elements contained in V X V .

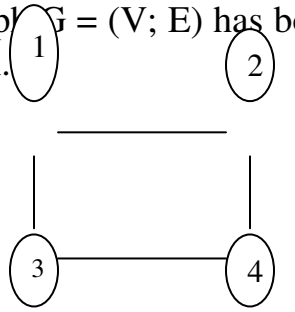
That is, E is a collection of ordered pairs of vertices. The edges in E are called directed

edges to distinguish them from those edges in graph

Let G = (V; E) be a directed graph. The source (or tail) of the (directed) edge e = (v1; v2) is v1 while the destination (or sink or head) of the edge is v2.

A directed graph (digraph) differs from a graph only insofar as we replace the concept of an edge as a set with the idea that an edge as an ordered pair in which the ordering gives some notion of direction of flow. In the context of a digraph, a self-loop is an ordered pair with form (v; v). We can define a multi-digraph if we allow the set E to be a true collection (rather than a set) that contains multiple copies of an ordered pair.

It is worth noting that the ordered pair (v1; v2) is distinct from the pair (v2; v1). Thus if a digraph G = (V; E) has both (v1; v2) and (v2; v1) in its edge set, it is not a multi-digraph.



We can modify the figures in Example 2.8 to make it directed. Suppose

we have the directed graph with vertex set V = f1; 2; 3; 4g and edge set:

E = {(1; 2);(2; 3);(3; 4);(4; 1)}

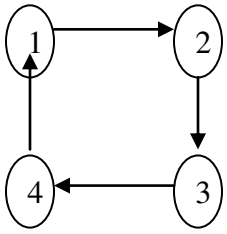
This digraph is visualized in Figure 2.7(a). In drawing a digraph, we simply append arrow-heads to the destination associated with a directed edge.

We can likewise modify our self-loop example to make it directed. In this case, our edge

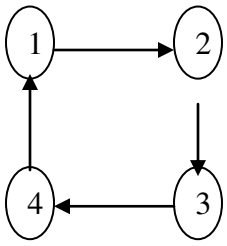
set becomes:

E = {(1; 2);(2; 3);(3; 4);(4; 1);(1; 1)}

This is shown in Figure 2.7(b).



(a)



(b)

Figure 2.7. (a) A directed graph. (b) A directed graph with a self-loop. In a directed graph, edges are directed; that is they are ordered pairs of elements drawn from the vertex set. The ordering of the pair gives the direction of the edge.

Consider the (simple) graph from Example Suppose that the vertices represent islands (just as they did) in the Bridges of K□onigsburg Problem and the edges represent bridges. It is very easy to see that a tour of these islands exists in which we cross each bridge exactly once. (Such a tour might start at Island 1 then go to Island 2, then 3,then 4 and finally back to Island 1.)

(Underlying Graph). If G = (V; E) is a digraph, then the underlying graph of G is the (multi) graph (with self-loops) that results when each directed edge (v1; v2) is replaced by the set fv1; v2g thus making the edge non-directional. Naturally if the directed edge is a directed self-loop (v; v) then it is replaced by the singleton set {v}

Notions like edge and vertex adjacency and neighborhood can be extended to digraphs by simply defining them with respect to the underlying graph of a digraph. Thus the neighborhood of a vertex v in a digraph G is N(v) computed in the underlying graph.

Whether the underlying graph of a digraph is a multi-graph or not usually has no bearing on relevant properties. In general, an author will state whether two directed edges (v1; v2) and (v2; v1) are combined into a single set fv1; v2gor two sets in a multiset. As a rule-of-thumb, multi-digraphs will have underlying multigraphs, while digraphs generally have underlying graphs that are not multi-graphs.

It is possible to mix (undirected) edges and directed edges together into a very general definition of a graph with both undirected and directed edges. Situations requiring such a model almost never occur in modeling and when they do, the

undirected edges with form fv1; v2g are usually replaced with a pair of directed edges (v1; v2) and (v2; v1).

Thus, for remainder of these notes, unless otherwise stated:

(1) When we say graph we will mean simple graph as in Remark 2.19. If we intend the

result to apply to any graph we'll say a general graph.

(2) When we say digraph we will mean a directed graph G = (V; E) in which every edge

is a directed edge and the component E is a set and in which there are no self-loops.

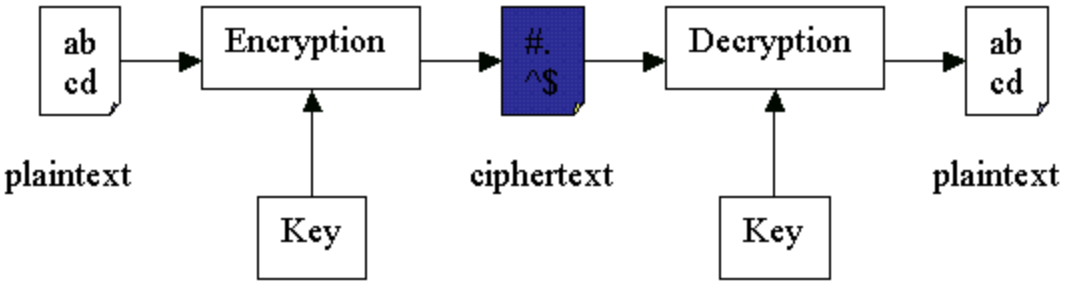
14. Briefly discuss about Cryptography?

Cryptography is the art and science of keeping messages secure.

When a message is transferred from one place to another, it contents are readily available to an eavesdropper. A simple network-monitoring tool can expose the entire message sent from one computer to another in a graphical way. For an N-Tier or distributed application to be secure, all messages sent on the network should be scrambled in a way that it is computationally impossible for any one to read it.

In cryptography world the message that needs to be secured is called plaintext or cleartext. The scrambled form of the message is called ciphertext. The process of converting a plaintext to ciphertext is called encryption. The process of reconverting the ciphertext into plaintext is called decryption. Cryptography algorithms (ciphers) are mathematical functions used for encryption and decryptions.

For cryptography to be used in practical solutions algorithms used for encryption and decryption should be made public. This is possible by using a byte stream called Key. For the algorithm to encipher a plaintext to ciphertext and to decipher it



Symmetric Key Encryption

Symmetric key encryption uses same key, called secret key, for both encryption and decryption. Users exchanging data keep this key to themselves. Message encrypted with a secret key can be decrypted only with the same secret key.

The algorithm used for symmetric key encryption is called secret-key algorithm. Since secret-key algorithms are mostly used for encrypting the content of the message they are also called content-encryption algorithms.

The major vulnerability of secret-key algorithm is the need for sharing the secret-key. One way of solving this is by deriving the same secret key at both ends from a user supplied text string (password) and the algorithm used for this is called password-based encryption algorithm. Another solution is to securely send the secret-key from one end to other end. This is done using another class of encryption called asymmetric algorithm, which is discussed later.

Strength of the symmetric key encryption depends on the size of the key used. For the same algorithm, encrypting using longer key is tougher to break than the one done using smaller key. Strength of the key is not liner with the length of the key but doubles with each additional bit.

Following are some of popular secret-key algorithms and the key size that they use

RC2	-	64 bits
DES	64	bits
3DES	192	bits
AES	256	bits
IDEA	128	bits

CAST 128 bits (CAST256 uses 256 bits key)
Algorithm Parameters:

4. **EncryptionMode**

There are two types of secret-key ciphers, block ciphers and stream ciphers Block Ciphers convert fixed-length block of plain text into cipher text of the same length. Most of the block ciphers use a block size of 64 bits. When the message size is more than that of the block size, then the message is broken into multiple blocks and each block is encrypted separately. There are different modes in which a Block cipher can encrypt these blocks. Viz., Cipher Block Chaining (CBC), Electronic Codebook (ECB) or Cipher Feedback (CFB). Of these CBC mode is more commonly used. In CBC Mode, each plain text block is XORed with the previous cipher text block before being encrypted. Some famous Block ciphers are DES, 3DES, IDEA, SAFER, Blowfish and Skipjack (used by US National Security Agency). Stream Ciphers operate on small group of bits, typically applying bitwise XOR operations to them using the key as a sequence of bits. Some famous stream ciphersincludeRC4andSEAL.

5. **InitializationVector**

Since Block ciphers working on CBC modes XOR each block with the previous encrypted block, the first block of the message needs a byte array, of same block size, with which it will be XORed This byte array is called IV or Initialization Vector.

Following are some block ciphers with their normal block size

DES	-	64 bits
3DES		64 bits
AES		128 bits

6. **Padding**

Often last block of the message will be smaller that the expected block size in which case a predetermined string will be repeatedly added to the end of the block to make it to the expected size. For instance if the block size is 64 bits and the last block has only 40 bits then 24 bits of padding will be added to it. There were two ways to add the pad, either by adding zeros or the number of the bytes that needs to be added (in this case it will be 3).

Asymmetric Key Encryption

Asymmetric key encryption uses different keys for encryption and decryption. These two keys are mathematically related and they form a key pair. One of these two keys should be kept private, called private-key, and the other can be made public (it can even be sent in mail), called public-key. Hence this is also called Public Key Encryption.

A private key is typically used for encrypting the message-digest; in such an application private-key algorithm is called message-digest encryption algorithm. A public key is typically used for encrypting the secret-key; in such a application private-key algorithm is called key encryption algorithm.

Popular private-key algorithms are RSA (invented by Rivest, Shamir and Adlemen) and DSA (Digital Signature Algorithm). While for an ordinary use of RSA, a key size of 768 can be used, but for corporate use a key size of 1024 and for extremely valuable information a key size of 2048 should be used.

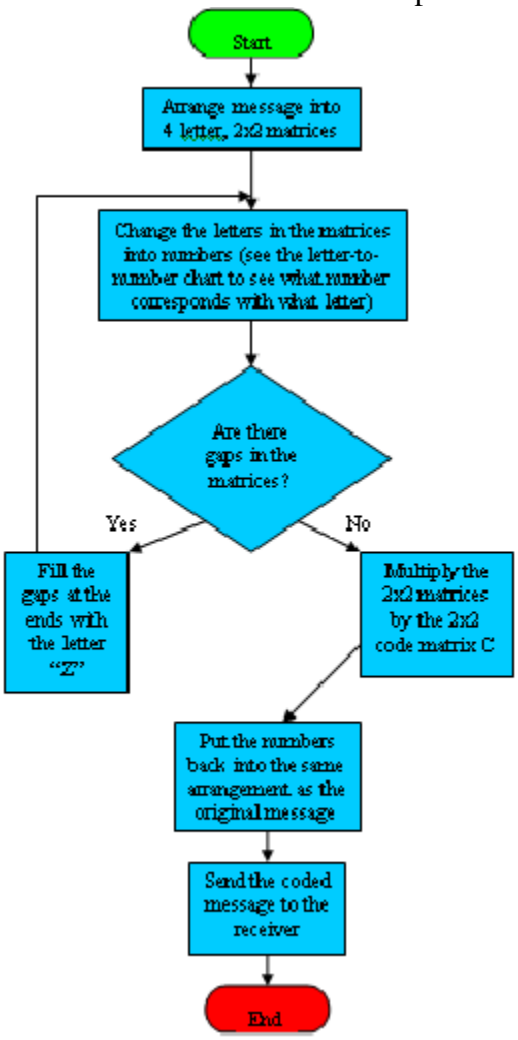
Asymmetric key encryption is much slower than symmetric key encryption and hence they are only used for key exchanges and digital signatures.

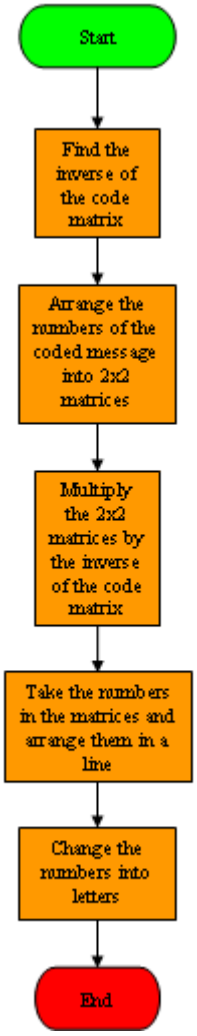
15. Encoding and decoding using matrices in cryptography

Encoding and decoding using matrices in cryptography

Matrix multiplication can be used to “encode” and “decode” messages. For that a coding matrix is required which is known to both the sender and the recipient. The following flowchart illustrates this in a vivid way:-

Encoding a Message & Decoding a Message





Coding Letters

A	B	C	D	E	F	G	H	I	J	K	L	M
1	2	3	4	5	6	7	8	9	10	11	12	13

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
14	15	16	17	18	19	20	21	22	23	24	25	26

Example:

Encoding

An example for encoding a message is as follows:

To encode the message FMG ROCKS the following steps are taken:

A) Write the message using 2x2 matrices.

$$\begin{bmatrix} F & M \\ G & R \end{bmatrix}, \begin{bmatrix} O & C \\ K & S \end{bmatrix}$$

If any gaps are left, we use a filler instead, adding a Z.

B) Replace each letter with its corresponding number in the alphabet.

A=1, B=2, C=3 and so on.

$$\begin{bmatrix} 6 & 13 \\ 7 & 18 \end{bmatrix}, \begin{bmatrix} 15 & 3 \\ 11 & 19 \end{bmatrix}$$

Now the matrix is encoded by multiplying it to the coding matrix, which is known only by the sender and the receiver. $C = \begin{bmatrix} 3 & 4 \\ 5 & 6 \end{bmatrix}$

$$\begin{bmatrix} 3 & 4 \\ 5 & 6 \end{bmatrix} \times \begin{bmatrix} 6 & 13 \\ 7 & 18 \end{bmatrix} = \begin{bmatrix} (3\times6)+(4\times7)=46 & (3\times13)+(4\times18)=111 \\ (5\times6)+(6\times7)=72 & (5\times13)+(6\times18)=173 \end{bmatrix}$$

$$\begin{bmatrix} 3 & 4 \\ 5 & 6 \end{bmatrix} \times \begin{bmatrix} 15 & 3 \\ 11 & 19 \end{bmatrix} = \begin{bmatrix} (3\times15)+(4\times11)=89 & (3\times3)+(4\times19)=85 \\ (5\times15)+(6\times11)=141 & (5\times3)+(6\times19)=129 \end{bmatrix}$$

Therefore, you would send the message as 46,111,72,173,89,85,141,129.

To decode a message you multiply the coded message with the INVERSE of the coding matrix.

First, write the coded message as a 2×2 matrix.

$$A = \begin{bmatrix} 46 & 111 \\ 72 & 173 \end{bmatrix}$$

$$B = \begin{bmatrix} 89 & 85 \\ 141 & 129 \end{bmatrix}$$

The decoding matrix, or key, is the inverse matrix of the coding matrix.

$$C = \begin{bmatrix} 3 & 4 \\ 5 & 6 \end{bmatrix}$$

$$C^{-1} = \frac{1}{ad-bc}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

$$C^{-1} = \frac{1}{(3)(6)-(4)(5)}\begin{bmatrix} 6 & -4 \\ -5 & 3 \end{bmatrix}$$

$$C^{-1} = -\frac{1}{2}\begin{bmatrix} 6 & -4 \\ -5 & 3 \end{bmatrix}$$

$$C^{-1} = \begin{bmatrix} -3 & 2 \\ 2.5 & -1.5 \end{bmatrix}$$

Multiply the inverse key by the coded message.(key X message)

$$C^{-1}A = \begin{bmatrix} -3 & 2 \\ 2.5 & -1.5 \end{bmatrix}\begin{bmatrix} 46 & 111 \\ 72 & 173 \end{bmatrix} = \begin{bmatrix} 6 & 13 \\ 7 & 18 \end{bmatrix}$$

$$C^{-1}B = \begin{bmatrix} -3 & 2 \\ 2.5 & -1.5 \end{bmatrix}\begin{bmatrix} 89 & 85 \\ 141 & 129 \end{bmatrix} = \begin{bmatrix} 15 & 3 \\ 11 & 19 \end{bmatrix}$$

16. Discuss briefly about Scrambler.

In telecommunications, a **scrambler** is a device that transposes or inverts signals or otherwise encodes a message at the transmitter to make the message unintelligible at a receiver not equipped with an appropriately set descrambling device. Whereas encryption usually refers to operations carried out in the digital domain, scrambling usually refers to operations carried out in theanalog domain. Scrambling is accomplished by the addition of components to the original signal or the changing of some important component of the original signal in order to make extraction of the original signal difficult. Examples of the latter might include removing or changing vertical or horizontal sync pulses in television signals; televisions will not be able to display a picture from such a

signal. Some modern scramblers are actually encryption devices, the name remaining due to the similarities in use, as opposed to internal operation.

In telecommunications and recording, a **scrambler** (also referred to as a **randomizer**) is a device that manipulates a data stream before transmitting. The manipulations are reversed by a**descrambler** at the receiving side. Scrambling is widely used in satellite, radio relay communications and PSTN modems. A scrambler can be placed just before a FEC coder, or it can be placed after the FEC, just before the modulation or line code. A scrambler in this context has nothing to do with encrypting, as the intent is not to render the message unintelligible, but to give the transmitted data useful engineering properties.

A scrambler replaces sequences into other sequences without removing undesirable sequences, and as a result it changes the probability of occurrence of vexatious sequences. Clearly it is not foolproof as there are input sequences that yield all-zeros, all-ones, or other undesirable periodic output sequences. A scrambler is therefore not a good substitute for a line code, which, through a coding step, removes unwanted sequences.

Purposes of scrambling

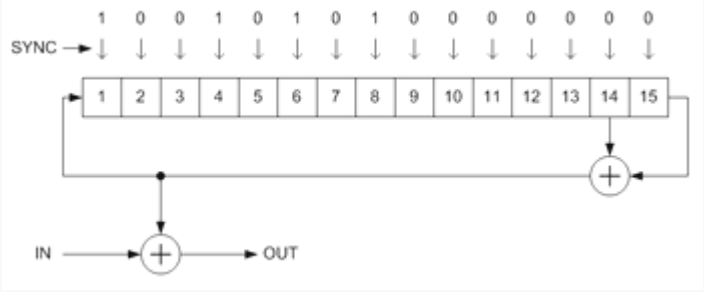
There are two main reasons why scrambling is used:

- It facilitates the work of a timing recovery circuit (see also Clock recovery), an automatic gain control and other adaptive circuits of the receiver (eliminating long sequences consisting of '0' or '1' only).
- It eliminates the dependence of a signal's power spectrum upon the actual transmitted data, making it more dispersed to meet maximum power spectral density requirements (because if the power is concentrated in a narrow frequency band, it can interfere with adjacent channels due to the cross modulation and the intermodulation caused by non-linearities of the receiving tract).

Types of scramblers

- Additive (synchronous) scramblers
- Multiplicative (self-synchronizing) scramblers

Additive (synchronous) scramblers



An additive scrambler (descrambler) used in DVB

Additive scramblers (they are also referred to as *synchronous*) transform the input data stream by applying a pseudo-random binary sequence (PRBS) (by modulo-two addition). Sometimes a pre-calculated PRBS stored in the Read-only memory is used, but more often it is generated by a linear feedback shift register (LFSR).

In order to assure a synchronous operation of the transmitting and receiving LFSR (that is, *scrambler* and *descrambler*), a *sync-word* must be used.

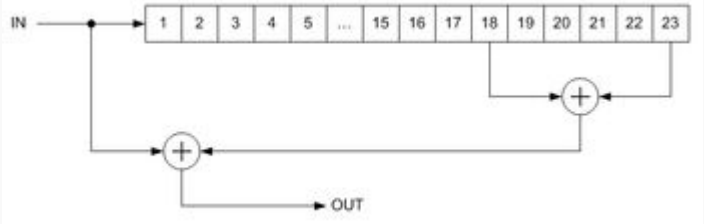
A sync-word is a pattern that is placed in the data stream through equal intervals (that is, in each frame). A receiver searches for a few sync-words in adjacent frames and hence determines the place when its LFSR must be reloaded with a pre-defined *initial state*.

The *additive descrambler* is just the same device as the additive scrambler.

Additive scrambler/descrambler is defined by the polynomial of its LFSR (for the scrambler on the picture above, it is $1 + x^{-14} + x^{-15}$) and its *initial state*.

Multiplicative (self-synchronizing) scramblers

A multiplicative scrambler used in V.34 recommendation



A multiplicative descrambler used in V.34 recommendation

Multiplicative scramblers (also known as *feed-through*) are called so because they perform a *multiplication* of the input signal by the scrambler's transfer function in Z-space. They are discrete linear time-invariant systems. A multiplicative scrambler is recursive and a multiplicative descrambler is non-recursive. Unlike additive scramblers, multiplicative scramblers do not need the frame synchronization, that is why they are also called *self-synchronizing*. Multiplicative scrambler/descrambler is defined similarly by a polynomial (for the scrambler on the picture it is $1 + x^{-18} + x^{-23}$), which is also a *transfer function* of the descrambler.

[edit]**Comparison of scramblers**

Scramblers have certain drawbacks:

- Both types may fail to generate random sequences under worst case input conditions.
- Multiplicative scramblers lead to error multiplication during descrambling (i.e. a single bit error at the descrambler's input will result into *w* errors at its output, where *w* equals the number of the scrambler's feedback taps).
- Additive scramblers must be reset by the frame sync; if this fails massive error propagation will result as a complete frame cannot be descrambled.
- The effective length of the random sequence of an additive scrambler is limited by the frame length, which is normally much shorter than the period of the PRBS. By adding frame numbers to the frame sync, it is possible to extend the length of the random sequence, by varying the random sequence in accordance with the frame number.

17. ENCRYPTION USING PSEUDORANDOM FUNCTIONS

$$
\begin{aligned}
&|\,\mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(K,M)) = 1] - \mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(K,M')) = 1]| \leq \\
&\quad |\,\mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(M)) = 1 \wedge REPEAT] - \mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(M')) = 1 \wedge REPEAT]| + \\
&\quad |\,\mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(M)) = 1 \wedge \neg REPEAT] - \mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(M')) = 1 \wedge \neg REPEAT]|
\end{aligned}
$$

Now the _rst di_erence is the di_erence between two numbers which are both between 0 and P[REPEAT], so it is at most P[REPEAT], which is at most t/2m .

The second di_erence is zero, because with a purely random function there is a 1-1 mapping between every random choice (of R; r; r1; : : : ; rt) which makes the _rst event happen and every random choice that makes the second event happen.

We have shown that with a purely random function, the above encryption scheme is CPA-secure. We can now turn our eyes to the pseudorandom scheme (Enc;Dec), and prove Theorem 18.

Proof: Consider the following four probabilities, for messages M, M', and algorithm T :

1. $\mathbb{P}_K[T^{Enc(K,\cdot)}(Enc(K,M)) = 1]$
2. $\mathbb{P}_K[T^{Enc(K,\cdot)}(Enc(K,M')) = 1]$
3. $\mathbb{P}_R[T^{\overline{Enc}(\cdot)}(\overline{Enc}(M)) = 1]$
4. $\mathbb{P}_R[T^{\overline{Enc}(\cdot)}(\overline{Enc}(M')) = 1]$

From the previous proof, we have $|3-4| \leq \frac{t}{2^m}$. If we are able to show that $|1-3| \leq \epsilon$, $|2-4| \leq \epsilon$, then we have $|1-2| \leq 2\epsilon + \frac{t}{2^m}$.

So, it remains to show that

$$|\mathop{\mathbb{P}}_{K}[T^{Enc(K,\cdot)}(Enc(K,M)) = 1] - \mathop{\mathbb{P}}_{R}[T^{\overline{Enc}(\cdot)}(\overline{Enc}(M)) = 1]| \leq \epsilon$$

Suppose, by contradiction, this is not the case. We will show that such a contradiction
implies that F is not secure, by constructing an oracle algorithm T' that distinguishes F
from a truly random function.

For an oracle G, we define T'^G to be the following algorithm:

- pick a random $r \in \{0,1\}^m$ and compute $C := (r, G(r) \oplus M)$
- simulate $T(C)$; every time C makes an oracle query M_i, pick a random r_i and respond to the query with $(r_i, G(r_i) \oplus M)$

Note that if T' is given the oracle F_K, then the computation T'^{F_K} is exactly the same as the computation $T^{Enc}(Enc(M))$, and if T' is given the oracle R, where R is a random function, then the computation $T^{\overline{Enc}}(\overline{Enc}(M))$.

18. Briefly discuss about Encryption Using Pseudorandom Functions

Suppose $F : \{0,1\}^k \times \{0,1\}^m \to \{0,1\}^m$ is a pseudorandom function. We define the following encryption scheme.

- $Enc(K, M)$: pick a random $r \in \{0,1\}^m$, output $(r, F_K(r) \oplus M)$
- $Dec(K, (C_0, C_1)) := F_K(C_0) \oplus C_1$

This construction achieves CPA security.

Theorem 18 *Suppose F is a (t, ϵ) secure pseudorandom function. Then the above scheme is $(\frac{t}{O(m)}, 2\epsilon + t \cdot 2^{-m})$-secure against CPA.*

The proof of Theorem 18 will introduce another key idea that will often reappear in this course: to first pretend that our pseudorandom object is truly random, and perform our analysis accordingly. Then extend the analysis from the pseudorandom case to the truly random case.

Let us therefore consider a modified scheme $(\overline{Enc}, \overline{Dec})$, where instead of performing $F_K(r) \oplus M$, we do $R(r) \oplus M$, where $R : \{0,1\}^m \to \{0,1\}^m$ is a truly random function. We need to look at how secure this scheme is. In fact, we will actually prove that

Lemma 19 *$(\overline{Enc}, \overline{Dec})$ is $(t, \frac{t}{2^m})-CPA$ secure.*

PROOF:

In the computation $T^{\overline{Enc}}(\overline{Enc}(r, C))$ of algorithm T given oracle $\overline{Enc}$ and input the ciphertext (r, C), let us define REPEAT to be the event where T gets the messages $(r_1, C_1), \ldots, (r_t, C_t)$ from the oracle, such that r equals one of the r_i.

Then we have

$$\begin{aligned} \mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(M)) = 1] &= \mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(M)) = 1 \wedge REPEAT] \\ &\quad + \mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(M)) = 1 \wedge \neg REPEAT] \end{aligned}$$

similarly,

$$\begin{aligned} \mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(M')) = 1] &= \mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(K, M')) = 1 \wedge REPEAT] \\ &\quad + \mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(M')) = 1 \wedge \neg REPEAT] \end{aligned}$$

$$|\mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(K,M))=1]-\mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(K,M'))=1]| \leq$$
$$|\mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(M))=1 \wedge REPEAT]-\mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(M'))=1 \wedge REPEAT]| +$$
$$|\mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(M))=1 \wedge \neg REPEAT]-\mathbb{P}[T^{\overline{Enc}}(\overline{Enc}(M'))=1 \wedge \neg REPEAT]|$$

Now the first difference is the difference between two numbers which are both between 0 and $P[REPEAT]$, so it is at most $P[REPEAT]$, which is at most $\frac{t}{2^m}$.

The second difference is zero, because with a purely random function there is a 1-1 mapping between every random choice (of $R, r, r_1, \ldots, r_t$) which makes the first event happen and every random choice that makes the second event happen. □

We have shown that with a purely random function, the above encryption scheme is CPA-secure. We can now turn our eyes to the pseudorandom scheme (Enc, Dec), and prove Theorem 18.

PROOF: Consider the following four probabilities, for messages M, M', and algorithm T :

1. $\mathbb{P}_K[T^{Enc(K,\cdot)}(Enc(K,M))=1]$
2. $\mathbb{P}_K[T^{Enc(K,\cdot)}(Enc(K,M'))=1]$
3. $\mathbb{P}_R[T^{\overline{Enc}(\cdot)}(\overline{Enc}(M))=1]$
4. $\mathbb{P}_R[T^{\overline{Enc}(\cdot)}(\overline{Enc}(M'))=1]$

From the previous proof, we have $|3-4| \leq \frac{t}{2^m}$. If we are able to show that $|1-3| \leq \epsilon$, $|2-4| \leq \epsilon$, then we have $|1-2| \leq 2\epsilon + \frac{t}{2^m}$.

So, it remains to show that

$$|\mathbb{P}_K[T^{Enc(K,\cdot)}(Enc(K,M))=1]-\mathbb{P}_R[T^{\overline{Enc}(\cdot)}(\overline{Enc}(M))=1]| \leq \epsilon \qquad (4.1)$$

Suppose, by contradiction, this is not the case. We will show that such a contradiction implies that F is not secure, by constructing an oracle algorithm T' that distinguishes F from a truly random function.

For an oracle G, we define T'^G to be the following algorithm:

- pick a random $r \in \{0,1\}^m$ and compute $C := (r, G(r) \oplus M)$
- simulate $T(C)$; every time C makes an oracle query M_i, pick a random r_i and respond to the query with $(r_i, G(r_i) \oplus M)$

Note that if T' is given the oracle F_K, then the computation T'^{F_K} is exactly the same as the computation $T^{Enc}(Enc(M))$, and if T' is given the oracle R, where R is a random function, then the computation $T^{\overline{Enc}}(\overline{Enc}(M))$.

Thus, we have

$$\mathop{\mathbb{P}}_{K\in\{0,1\}^k}[T^{vF_K}() = 1] = \mathop{\mathbb{P}}_{K}[T^{Enc(K,\cdot)}(Enc(K,M)) = 1] \tag{4.2}$$

$$\mathop{\mathbb{P}}_{R:\{0,1\}^m\to\{0,1\}^m}[T^{vR}() = 1] = \mathop{\mathbb{P}}_{R}[T^{\overline{Enc}(\cdot)}(\overline{Enc}(M)) = 1] \tag{4.3}$$

which means that

$$\left|\mathop{\mathbb{P}}_{K\in\{0,1\}^k}[T^{vF_K}() = 1] - \mathop{\mathbb{P}}_{R:\{0,1\}^m\to\{0,1\}^m}[T^{vR}() = 1]\right| > \epsilon \tag{4.4}$$

The complexity of T' is at most the complexity of T times $O(m)$ (the time needed to translate between oracle queries of T and oracle queries of T'), and so if T has complexity $t/O(m)$ then T' has complexity $\leq t$. This means that (4.4) contradicts the assumption that F is $(t,\epsilon)-$secure. □

19.Discuss about Public Key Encryption

A public-key encryption scheme is defined by three efficient algorithms (G,E,D) such that

- G takes no input and outputs a pair of keys (PK,SK)
- E, on input a public key PK and a plaintext message m outputs a ciphertext E(PK,M).

(Typically, E is a probabilistic procedure.)

- D, on input a secret key SK and ciphertext C, decodes C. We require that for every message m

$$\mathop{\mathbb{P}}_{\substack{(PK,SK)=G()\\ \text{randomness of } E}}[D(SK,E(PK,m)) = m] = 1$$

A basic definition of security is message-indistinguishability for one encryption.

We say that a public-key encryption scheme (G, E, D) is (t, ϵ) message-indistinguishable if for every algorithm A of complexity $\leq t$ and for every two messages m_1, m_2,

$$\left| \mathop{\mathbb{P}}_{\substack{(PK,SK)=G() \\ randomness\ of\ E}} [A(PK, E(PK, m_1)) = 1] - \mathop{\mathbb{P}}_{\substack{(PK,SK)=G() \\ randomness\ of\ E}} [A(PK, E(PK, m_2)) = 1] \right| \leq \epsilon$$

20. Briefly discuss about the definition of security

There are three ways in which the definitions of security given in class differ from the way they are given in the textbook. The first one applies to all definitions, the second to definitions of encryption, and the third to CPA and CCA notions of security for encryption:

1. Our definitions usually refer to schemes of fixed key length and involve parameters t, ϵ, while the textbook definitions are asymptotic and parameter-free.

   Generally, one obtains the textbook definition by considering a family of constructions with arbitrary key length (or, more abstractly "security parameter") k, and allowing t to grow like any polynomial in k and requiring ϵ to be negligible. (Recall that a non-negative function $\nu(k)$ is negligible if for every polynomial p we have $\lim_{k\to\infty} p(k) \cdot \nu(k) = 0$.)

   The advantage of the asymptotic definitions is that they are more compact and make it easier to state the result of a security analysis. (Compare "if one-way permutations exist, then length-increasing pseudorandom generators exist" with "if $f : \{0,1\}^n \to \{0,1\}^n$ is a (t, ϵ) one-way permutation computable in time $\leq r$, then there is a generator $G : \{0,1\}^{2n} \to \{0,1\}^{2n+1}$ computable in time $r + O(n)$ that is $(t\epsilon^4/(n^2 \log n) - r\epsilon^{-4}n^3 \log n, \epsilon/3)$ pseudorandom")

   The advantage of the parametric definitions is that they make sense for fixed-key constructions, and that they make security proofs a bit shorter.

   (Every asymptotic security proof starts from "There is a polynomial time adversary A, an infinite set N of input lengths, and a polynomial $q()$, such that for every $n \in N \ldots$.)

2. Definitions of security for an encryption algorithm $E()$, after the proper quantifications, involve an adversary A (who possibly has oracles, etc.) and messages m_0, m_1; we require

   $$|\mathbb{P}[A(E(m_0)) = 1] - \mathbb{P}[A(E(m_1)) = 1]| \leq \epsilon$$

   while the textbook usually has a condition of the form

   $$\mathop{\mathbb{P}}_{b\in\{0,1\}} [A(E(m_b)) = b] \leq \frac{1}{2} + \frac{\epsilon}{2}$$

   The two conditions are equivalent since

$$\mathbb{P}[A(E(m_b)) = b] = \frac{1}{2}\mathbb{P}[A(E(m_0)) = 0] + \frac{1}{2}\mathbb{P}[A(E(m_1)) = 1$$

and the absolute value in (13.1) may be removed without loss of generality (at the cost of increasing the complextiy parameter by one).

3. Definitions of CPA and CCA security, as well as all definitions in the public-key setting, have a different structure in the book. The difference is best explained by an example. Suppose we have a public-key scheme (G, E, D) such that, for every valid public key pk, $E(pk, pk)$ has some distinctive pattern that makes it easy to distinguish it from other ciphertexts. This could be considered a security weakness because an eavesdropper is able to see if a party is sending a message that concerns the public key.

This would not be a concern if the encryption mechanism were separate from the transport mechanism. For instance, if the application of this scheme occurred in such a way they two parties are securely communicating over an instant messaging client which exists in the application layer and encryption were occurring layers below in the transport layer. This abstraction of layers and separation of the encryption mechanism from the application abstracts away the notion that the public key could be encrypted with the public key. The messaging client is aware of the interface, but it never exposes the actual public or private key to the user, which prevents incorrectly using the cryptographic primitives.

You can show as an exercise that if a secure public-key encryption scheme exists, then there is a public-key encryption scheme that is secure according to our definition from last lecture but that has a fault of the above kind.

The textbook adopts a two-phase definition of security, in which the adversary is allowed to choose the two messages m_0, m_1 that it is going to try and distinguish, and the choice is done *after* having seen the public key. A random bit b is chosen and then the ciphertext of m_b is computed and given to the adversary. The adversary continues to have access to the Encryption function with the given public key. When the adversary is done, it outputs a guess called b' The output of this procedure is 1 when $b = b'$. A cryptosystem in which $E(pk, pk)$ can be distinguished from other ciphertexts violates this definition of security.

21. Briefly discuss about Arrow diagram

Arrows are major components of diagrams. Arrows appear in various types of diagrams,
such as traffic signs, guideboards, route maps, flowcharts, and illustrations. One reason for such popularity is that arrows capture a large variety of semantics with their simple shape. Another reason is that the existence of arrows encourages people to interpret causal and functional aspects in a diagram.
For instance, Fig. 1 contains only a few words and some
arrow symbols over a background map, but people easily read the mechanism of a

spatio-temporal process—the El Niño effect in the Southeastern Pacific Ocean indirectly
influences the rise of tofu price in Japan. In this way, arrows are powerful tools that
facilitate the communication of spatial and temporal knowledge in a static diagram.

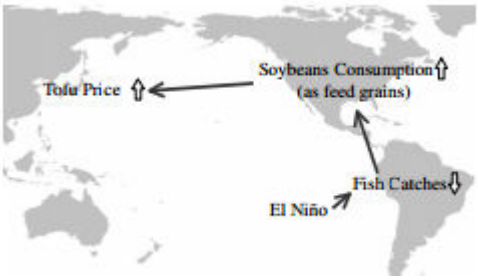
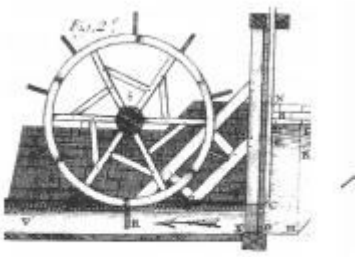


Fig. 1. A diagram with arrows, which illustrates a spatio-temporal process that the El Niño effect (i.e., sea temperature rise in the Southeastern Pacific Ocean) indirectly influence the rise of the tofu price in Japan

The combination of arrow symbols and the related elements is considered as a unit of
syntax, and called an arrow diagram. Then, these arrow-related elements are called the
components of the arrow diagram. An arrow diagram must have at least one arrow symbol and one component. As a first step, we consider the arrow diagram that contains only a uni-directional arrow symbol and its related elements. Bi-directional arrows are
not considered, because they are regarded as a synthesis of two oppositely-directed arrow symbols. Also, independent arrow symbols, such as arrow-shaped traffic signs
indicating curving roads and map symbols indicating north direction, are not discussed
in this paper, because they have no components.

One of the primary usages of arrow diagrams is to express a direction. Gombrich (1990) reported a very early example of an arrow diagram, where an arrow symbol was
used to represent the direction of a water stream (Fig.).



(a)

Arrow diagrams may also refer to metaphorical directions. Upward directions are metaphorically associated with increase or improvement, whereas downwards directions are associated with decrease or debasement (Lakoff and Johnson 1980). Accordingly, upward and downward arrow symbols are used to illustrate those semantics.

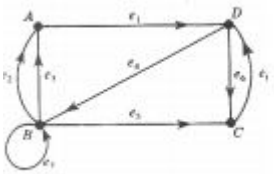
An arrow symbol illustrating a direction has a diagrammatic freedom of length. This
freedom allows the representation of a directed quantity, which is called a vector. A
vector is a quantity that is specified by a direction and a magnitude.

Another traditional usage of arrow diagrams is to illustrate a spatial movement.
Spatial movement is an event where an entity changes its spatial position continuously.

Arrow diagrams are used not only in a spatial context, but also in a temporal context.

For instance, timetables often contain arrow diagrams, each of which illustrates that
something continues over a certain period

An arrow diagram visualizes the presence of a directed relation between two components. In mathematics, a set of directed binary relations is modeled as a directed graph, which is often visualized using arrow diagrams.



22. Briefly explain Structure of Arrow Diagram.

Arrow diagrams illustrate a large variety of semantics, which increases the difficulty of

their interpretation. To tackle this problem, we first develop a method for making rough
interpretation of an arrow diagram from its structural pattern alone. As the foundation
of this method, this section summarizes the formal structures of arrow diagrams developed by Kurata and Egenhofer (2005).
The components of an arrow diagram are diagrammatic elements that an arrow symbol originates from, traverse, or points to. We classify the components of arrow
diagrams into the following five types:
• An object takes an action, either independently (e.g., a person in Fig. 4a) or as a result of interaction (e.g., a bag in Fig. 4a).
• An event occurs in time, and is characterized by a set of changes. An event occurs
over an interval (e.g., snow in Fig. 4b) or at an instant (e.g., a traffic accident in Fig. 4b).
• A location is a position in space. It may be a point (e.g., a place in Maine in Fig. 4c) or a homogeneous area (e.g., Maine).
• A moment is a position in time. It may be an instant (e.g., 8:20 in Fig. 5a) or a homogeneous interval (e.g., morning).
• A note is a short description that modifies an arrow symbol (e.g., send in Fig. 4a) or another component (e.g., Mr. K in Fig. 4a and You are here in Fig. 4c). A note and the modified component are placed adjacently to each other or connected with each other by an arrow symbol.
For simplification, an object, an event, a moment, a location, and a note are sometimes
denoted as O, E, M, L, and N, respectively.

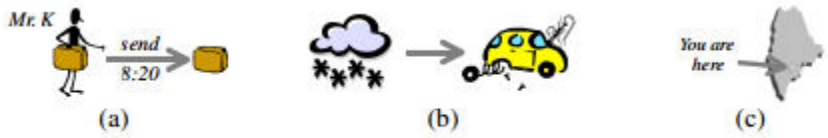


Fig. 4. Arrow diagrams that contain various types of components

Although a component may be mentioned by an icon, a text, or a specific position in
a background drawing, this classification is not concerned with such a descriptive style
of the component. We assume that automated interpreters of arrow semantics will distinguish these component types based on their knowledge base.

Components of an arrow diagram are located in front of the arrow's head, behind the
arrow's tail, or along the arrow's body. We, therefore, consider that an arrow symbol
identifies three different areas where the components of the arrow diagram are located
(Fig. 5). These areas are called component slots, and they are further classified into a
tail slot, a head slot, and a body slot. Each component in an arrow diagram is uniquely
assigned to one of these three slots, thereby making the distinction of tail components,
body components, and head components. The component slots need to bedistinguished, because the same symbols, used in different slots, illustrate significantly
different semantics (Kurata and Egenhofer 2005).
An arrow diagram has three slots, where five types of components may be located.
The combination of the types of components in the three slots composes a certain
pattern that is specific to every arrow diagram. These patterns are described as
Tail slot Body slot Head slot
Fig. 5. Three component slots associated with an arrow symbol
([M|E|L|O|N]*, [M|E|L|O|N]*, [M|E|L|O|N]*), where [x]* means empty or a sequence
of any number of x, x|y means x or y but not both, and the three elements in parentheses
indicate the types of components in tail, body, and head slot, respectively. For example,
the structures of arrow diagrams in Fig. 4a-c are described as (ON, MN, O), (E, -, E),
and (N, -, L), respectively. These patterns capture fundamental structures of arrow
diagrams, since they capture the alignment of components while abstracting the
individual difference and the absolute location of the components.

Table 1. Classification of the semantics of arrow diagrams

Example of semantics	Class	Definition
Direction, vector	Property	modification of a component by attaching an arrow symbol to the component
labeling	Annotation	modification of a component by connecting a description to the component
Spatial movement, temporal continuity, interaction	Action	a motion of one component that may be caused or cause an interaction with another component.
temporal order, conditional relation, causal relation, change, ordered relation	conjunction	association of components, where an arrow symbol does not express a motion

23. Find the optimal Huff man code for the following table of symbols:

character	frequency
a	2
b	3
c	7
d	8
e	12

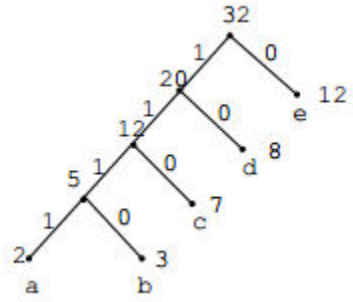
Answer: : The successive reductions ofthe list of frequencies are as follows:

$$\underbrace{2,3}_{5},7,8,12 \rightarrow \underbrace{5,7}_{12},8,12 \rightarrow 12,8,12$$

Here we have a choice, we can choose to add the first 12 and 8, or 8 and the second 12. Let's choose the former:

$$\underbrace{12,8}_{20},12 \rightarrow \underbrace{20,12}_{32} \rightarrow 32$$

The tree obtained is the following:



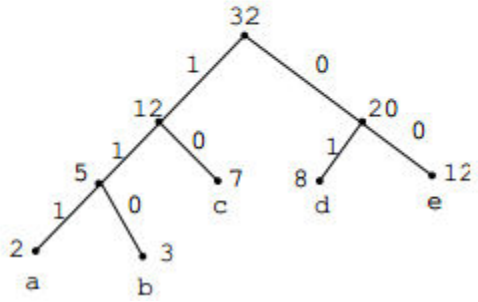
Optimal Huffman code 1

The resulting code is as follows:

character	code
a	1111
b	1110
c	110
d	10
e	0

The other choice yields the following:

$$12, \underbrace{8, 12}_{20} \rightarrow \underbrace{20, 12}_{32} \rightarrow 32$$



Optimal Huffman code 2

character	code
a	111
b	110
c	10
d	01
e	00

24. Discuss briefly about Huffman code.

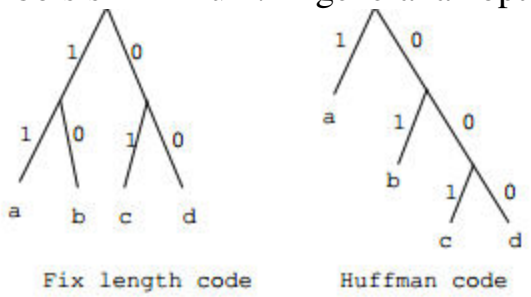
Usually characters are represented in a computer with fix length bit strings. Huffman codes provide an alternative representation with variable length bit strings, so that shorter strings are used for the most frequently used characters. As an example assume that we have an alphabet with four symbols: A = {a, b, c,d}.Two bits are enough for representing them, for instance a = 11, b = 10, c = 01, d = 00 would be one such representation. With this encoding n-character words will have 2n bits. However assume that they do not appear with the same frequency, instead some are more frequent that others, say a appears with a frequency of 50%, b 30%, c 15% and d 5%. Then the following enconding would be more efficient than the fix length encoding: a = 1, b = 01, c = 001, d = 000.
Now in average an n-character word will have 0.5n a's, 0.3n b's, 0.15n c's and 0.05n d's, hence its length will be $0.5n \cdot 1+0.3n \cdot 2+0.15n \cdot 3+0.05n \cdot 3 = 1.7n$, which

is shorter than 2n. In general the length per character of a given encoding with characters a1, a2... an whose frequencies are f1, f2... fn is

$$\frac{1}{F}\sum_{k=1}^{n} f_k\, l(a_k)\,,$$

where $l(a_k)$ = length of a_k and F = $\sum_{k=1}^{n} f_k\, l(a_k)\,,$ .The problem now is, given an alphabet and the frequencies of its characters, find an optimal encoding that provides minimum average length for words.

An optimal Huff man code is a Huff man code in which the average length of the symbolsis minimum. In general an optimal Huff man code can be made



as follows. First we list the frequencies of all the codes and represent the symbols as vertices (which at the end will be leaves of a tree).

Then we replace the two smallest frequencies f1 and f2 with their sum f1+ f2, and join the corresponding two symbols to a common vertex above them by two edges, one labeled 0 and the other one labeled 1.

Than common vertex plays the role of a new symbol with a frequency equal to f1+f2. Then we repeat the same operation with the resulting shorter list of frequencies until the list is reduced to one element and the graph obtained becomes a tree.

25. Discuss about adjacency and incidence matrix

Let G be an n-vertex directed graph. Let A be the $n \times n$ *adjacency matrix* of the graph G. Element $a_{ij} = 1$ if and only if the edge $(i, j) \in G$. All other elements are zero. A row of A lists the nodes at the tip of the *outgoing edges* while a column of A lists the nodes at the tail of the *incoming edges*.

Powers of an Adjacency Matrix

Consider the power A^2 of an adjacency matrix. What does element $(A^2)_{ij}$ mean? Element $(A^2)_{ij}$ is the dot product $A(i,:)A(:,j)$. The k-th term in dot product equals 1 if and only if $A(i,k) = A(k,j) = 1$. In other words, it is 1 if and only if there is a path from $i \to k \to j$. The dot product thus *counts the number of paths of length exactly* 2 *from* i *to* j. This generalizes to arbitrary nonnegative powers: $(A^r)_{ij}$ is the number of *paths of length exactly* r *from* i *to* j.

Relevant questions:

- **What if A is symmetric?**
  If A is symmetric, then there is an edge (i,j) and an edge (j,i), so A is basically undirected.
- **If $A^r = 0$, then what can we say about the structure of G? And if $A^r \neq 0$ for all r?**
  If $A^r = 0$, then there are *no paths of length exactly* r, so the *longest path is* $< r$ and the graph has *no cycles*. Conversely, $A^r \neq 0$ for all r if and only if G has a cycle.
- **What is $I + A + A^2 + \cdots + A^r$?**
  The sum represents the number of paths of any length less than or equal to r between every pair of vertices. Drop the identity and get only the nontrivial paths.
- $I + A + A^2 + \cdots + A^r - (A + A^2 + \cdots + A^r) = I$.
  This implies that $(I - A)^{-1} = I + A + A^2 + \cdots + A^r$.
- **How can we interpret a symmetric permutation $P^T AP$?**
  It is a relabeling of the vertices and does not change the structure of the graph. Two graphs are isomorphic if and only if there is a symmetric permutation from the adjacency graph of one to the adjacency graph of the other.
- **What if A is (now or after symmetric permutation) strictly upper triangular?**
  The vertices have been labeled such that all arcs go from lower to higher labels. This is what is known as a topological ordering and it also means that the graph has no cycles.
- **What if A is (now or after symmetric permutation) banded?**
  The vertices have been labeled so that the edges are local. The graph is in a sense long and thin.

Incidence Matrix.

An undirected or directed graph G with n vertices and m edges can be represented as an *incidence matrix* A. Arbitrarily number the edges $1, 2, \ldots, m$ and the vertices $1, 2, \ldots, n$. For edge i: (j,k), there is an entry $a_{ij} = -1$ and an entry $a_{ik} = 1$. All other elements are zero.

Interpretation of Ax

The matrix-vector product Ax can be understood in the following way. The vector x has one component for each vertex in G. Each component in the product Ax contains the difference between the components at the head of the edge and at the tail of the edge. The matrix-vector product Ax therefore maps values on the edges to differences on the edges.

Interpretation of Null space

The null space of A are those vectors that give zero difference over all edges.

Interpretation of $A^T x$

The matrix-vector product $A^T x$ can be understood in the following way. The vector x has one component for each edge in G. The entry $(A^T)_{ij} = -1$ iff edge j is outgoing from vertex i. Similarly, the entry $(A^T)_{ij} = 1$ iff edge j is incoming to vertex i. Therefore, the matrix-vector product $A^T x$ sums the incoming flow from each edge and subtracts the outgoing flow. In other words, each component is the *net flow* to/from that vertex.

Interpretation of Left Nullspace

The left nullspace of A are those vectors that give zero net flow on all vertices. The only nontrivial flows are constant flows around cycles of G.

Interpretation of $A^T Ax$

Since Ax computes the difference over each edge, the product $A^T(Ax)$ computes the net flow induced by the values at the vertices.

7. What is mean by connectives?

The simple statements initially are called atomic or primary statement. The new statement can be formed from atomic statement through the use of sentential connectives.

2. What is mean by quantifiers?

An analysis of mathematical sentences involving quantifiers indicates that the main two quantifiers are “all” and “some”, where “some is interpreted to mean atleast “one”.

Certain statement involve words that indicate quantity such as “all, some, none,one”. To answer the question “how many?” such words indicate quantity they are called quantifiers.

3. **Define tautology.**

A statement formula which is true, regardless of the truth values of the statement which replaces the variable in it is called a universally valid formula or a tautology or a logical truth.

4. **What are the rules of well formed formulae?**

- a statement variable standing alone is a well formed formulae
- if A is a well formed formulae the ¬A is a well formed formulae
- if A and B are well formed formulae, then (A^B),(AˇB).A->B are well formed formulae

- a string of symbols containing a statement variables, connectives and parenthesis is a well formed formulae. If and only if it can be obtained by finitely many applications of the rules 1, 2 and 3.

5. **What is meant by assignment problems?**

The problem of assigning 'n'-jobs to 'n' persons such that each job can be assigned to only one persons & each person can be allotted only one job. Suppose all the 'n' persons are capable of doing any one the 'n' jobs but the time taken by them to perform the jobs varies from job to job. The following table represents the time taken by the 'n' persons to perform the 'n' jobs.

6. **Define Network analysis.**

Network analysis is a technique which determines the various sequences of jobs concerning a project & project completion time. Network analysis has been successfully used to a wide range of significant management problems.

7. **Define Crashing.**

Crash the activities in the critical path as per the ranking (ie) activity having lower cost slope could be crashed first to the maximum extent possible. Calculate the new direct cost by cumulatively adding the cost of crashing to the normal cost.

8. **Define Total Float(TF), Free Float (FF).**

This is the amount of time a path of activities could be delayed without affecting the overall project duration.

TF=Head event(L)-Tail event(E)-Duration

Total float is also defined as the difference between the two finishing time or the difference between the starting time

TF=LFT-EFT =LST-EST

FF This is the amount of time an activity can be delayed without affecting the commencement of a subsequent activity of its earliest start time, but may affect the float of a previous activity.

FF=Headevent(L)-TailEvent(E)-Duration

FF=TF-Head event slack(L-E)

9. What are the rules for drawing an arrow diagram?

The rules for the construction for network are of 2 types namely

3. Logic rules
4. Computer rules

10. **Define Hungarian method.**

Assignment problems can be formulated with techniques of linear programming and transportation problems. As it has a special structure, it is solved by the special method called Hungarian method. This method of assignment problem was developed by a Hungarian mathematician D. Konig and is therefore known as Hungarian method of assignment problem.

11. **Define Chi-square test.**

If O_i (i=1,2..n) is a set of observed (experimental) frequencies of a data and E_i (i=1,2..n) is the corresponding set of expected (theoretical or hypothetical) frequencies, then the value of x^2 is defined as,

$$x^2 = \left[\sum (O_i - E_i)^2 / E_i \right]$$

12. **Define t-statistic.**

The t-statistic is defined as $t=|\bar{x} - \mu|/s * \sqrt{n}$

Where $s=\sqrt{\sum d^2 - n(d)^2/n-1}$

Where $\bar{x}$ is population mean

μ is sample mean

n is sample size

d is deviation from assumed mean of every sample value.

's' is standard deviation

13. **Write any two properties of t-distribution.**

1) The variable t-distribution ranges from $-\infty$ to $+\infty$.

2) The constant c is actually a function γ.so that, for a particular value of γ ,the distribution of f(t) is completely specified. Thus f(t) is a family of function one for each value of γ.

14. **Describe Applications of the t-distribution.**

The following are some of the examples to illustrate the way in which the student distribution is generally used to test the significance of the various results obtained from small samples.

To test the significance of the mean of a random sample

Testing difference between mean of two samples(Independent samples)

15. **Define Graph or Linear Graph.**

A Linear Graph or simply a graph G=[V,E] consist of a set V=(V1,V2…) called vertices and another set E=(e1,e2….) called edjes,such that each edge e_k is identified with an unordered pair of vertices V_i,V_j.

16. **What is a Circuit?**

A closed walk in which no vertex except the terminal vertices appears more than once is called a circuit.

17. **What is Binary Tree?**

Binary tree is a tree in which there is exactly one vertex of degree 2 and each of the remaining vertices is of degree one or three.

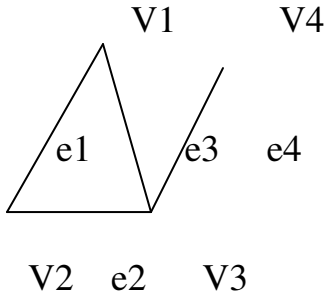
18. **Define Incident relation, adjacent relation.**

Incident relation.

If V_i is an end vertex of some edge e_j, then V_j and e_j said to be incident with (‘on’ or ‘to’) each other.

adjacent relation.

Two vertices are said to be adjacent, if they are the end vertices of the same edge.



V1, V2 are adjacent vertices V1, V4 are not adjacent vertices

19. What is meant by Cryptography.

Cryptography is the process of transforming plain text or original information into an unintelligible form (cipher text) so that it may be sent over unsafe channels or communications. The transformer process is controlled by a data string (key).

20. Define Hamming Metic.

Geormetrically, two codewords are ``far'' from each other if there are ``a lot'' of coordinates where they differ. This notion is made more precide in the following definition.

Definition 3.4.1 If $\mathbf{v} = (v_1, v_2, ..., v_n)$ $\mathbf{w} = (w_1, w_2, ..., w_n)$, are vectors in $V = F^n$ then we define

$$d(\mathbf{v}, \mathbf{w}) = |\{i \mid 1 \leq i \leq n,\ v_i \neq w_i\}|$$

to be the **Hamming distance** between $\mathbf{v}$ and $\mathbf{w}$. The function $d : V \times V \to \mathbb{N}$ is called the **Hamming metric**. The **weight** of a vector (in the Hamming metric) is $d(\mathbf{v}, \mathbf{0})$.

21. Define Hamming Distance

the **Hamming distance** between two strings of equal length is the number of positions at which the corresponding symbols are different. In another way, it measures the minimum number of *substitutions* required to change one string into the other, or the minimum number of *errors* that could have transformed one string into the other.

22. Define Ceaser Cipher Coding.

In cryptography, a **Caesar cipher**, also known as a **Caesar's cipher**, the **shift cipher**, **Caesar's code** or **Caesar shift**, is one of the simplest and most widely known encryption techniques. It is a type of substitution cipher in which each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet.

23. Define Correlation and regression

Correlation is a measure of association between two variables. The variables are not designated as dependent or independent.

Simple regression is used to examine the relationship between one dependent and one independent variable.

24.Define Tree.

A **tree** is an undirected simple graph *G* that satisfies any of the following equivalent conditions:

- *G* is connected and has no cycles.
- *G* has no cycles, and a simple cycle is formed if any edge is added to *G*.
- *G* is connected, but is not connected if any single edge is removed from *G*.
- *G* is connected and the 3-vertex complete graph K_3 is not a minor of *G*.
- Any two vertices in *G* can be connected by a unique simple path.