DISCRETE MATHEMATICS

(MTH1C04)

STUDY MATERIAL

I SEMESTER
CORE COURSE

M.SC MATHEMATICS

(2019 Admission onwards)

UNIVERSITY OF CALICUT
SCHOOL OF DISTANCE EDUCATION
Calicut University- P.O, Malappuram- 673635, Kerala.

190554

SCHOOL OF DISTANCE EDUCATION
UNIVERSITY OF CALICUT

STUDY MATERIAL

FIRST SEMESTER

M.SC MATHEMATICS (2019 ADMISSION ONWARDS)

CORE COURSE :

MTH1C04: DISCRETE MATHEMATICS

Prepared by:

SRI.SACHIN CHANDRAN
ASSISTANT PROFESSOR ON CONTRACT (MATHEMATICS)
SCHOOL OF DISTANCE EDUCATION
UNIVERSITY OF CALICUT

Scrutinized By:

SMT. SHIKHI.M
ASSISTANT PROFESSOR
PG DEPARTMENT OF MATHEMATICS
THUNCHAN MEMORIAL GOVERNMENT COLLEGE, TIRUR

Contents

Module 1

Order Relations and Boolean algebra

(Reference text: 'Foundations of Discrete Mathematics' by K. D. Joshi)

1.1 Order Relations

We have already seen equivalence relations that classify the set into disjoint subsets. Now we are going to introduce order relations that will help us compare the elements of the set.

Definition 1. ***Partial order:****A binary R on a set X is called a partial order if it satisfy the following conditions,*
1. Reflexive i.e if $(a, a) \in$ for all $a \in X$.
2. Anti-symmetry i.e if $(a, b) \in R$ and $(b, a) \in R$ then $a = b$, for $a, b \in A$.
3. Transitive i.e if $(a, b) \in R$ and $(b, c) \in R$, then $a, b, c \in X$.

A partial order is usually denoted by $\leq$ and is read as "less than or equal " or $\geq$, read as "greater than or equal". A set X together with a partial ordering R is called a partially ordered set, or poset, and is denoted by (X, R). Members of X are called elements of the poset.

Example : The usual "greater than or equal" relation $(\leq)$ is a partial ordering on the set of integers.

Definition 2. ***Strict partial order:****A binary R on a set X is called a strict partial order if it satisfy the following conditions,*
1. Irreflexive i.e $(a, a) \notin$ for all $a \in X$.
2. Asymmetry i.e if $(a, b) \in R$ then $(b, a) \notin R$, for $a, b \in X$.
3. Transitive i.e if $(a, b) \in R$ and $(b, c) \in R$, then $a, b, c \in X$.

Example : Let R be the relation on the set of people such that xRy if x and y are people and x is older than y.

Proposition 1.1.1. *Let R be a partial order on the set X. Then $R - \Delta X$ is a strict partial order on X. If S is a strict partial order on X, then $S \cup \Delta X$ is a partial order on X.*

Note :ΔX is the set of elements (a, a), for $a \in X$.

Proposition 1.1.2. *Let X be a set and " $\leq$ " be a binary relation of X such that it is reflexive and transitive. Define a binary relation R on the set X such that xRy if $x \leq y$ and $y \leq x$. Then R is an equivalence relation on the set X. Let X/R be the quotient set and $p : X \to X/R$ be the projection map, then there exits an partial order $\preceq$ on X/R such that for all $x, y \in X$, $x \leq y$ implies $p(x) \preceq p(y)$.*

Proof.

- Check that R is an equivalence relation.
- Elements of X/R is of the form $[x]$ for $x \in X$, where $[x]$ denote the R-equivalence class. If $[x]$ and $[y]$ are two elements of X/R, define $[x] \preceq [y]$ if and only if $x \leq y$. Check that this indeed forms a partial order on X/R. To ensure that this gives a well defined binary function on X/R, we must verify that the relation is independent of the choice of the representation of the equivalence class. Let $[x] \preceq [z]$ and $[y] \preceq [w]$. Then we must show that $[x] \preceq [y]$ iff $[z] \preceq [w]$. Suppose, $[x] \preceq [y]$ i.e. $x \leq y$. Since $[x] = [z]$, we have $z \leq x$ (and also $x \leq z$) and similarly $[y] = [w]$ gives $y \leq w$ (and also $w \leq y$). By transitivity of $\leq$, we get $z \leq w$ i.e. $[z] \preceq [w]$. Similarly, $[z] \preceq [w]$ implies $[x] \preceq [y]$. Thus $\preceq$ is a well defined relation on X/R. Now it only remains to verify that $\preceq$ is a partial order on X/R. Reflexivity and transitivity of $\preceq$ follows from the corresponding properties of $\leq$. For anti symmetry, suppose $[x] \preceq [y]$ and $[y] \preceq [x]$, then $x \leq y$ and $y \leq x$, which implies xRy and hence $[x] = [y]$.

□

The significance of this proposition is that whenever we have a reflexive and transitive relation on a set, we can always assume anti symmetry by passing to a suitable quotient set and in this passage we regard as equivalent such elements which would have been equal, had there been anti symmetry.

Definition 3. *Let X be a set and $\leq$ be a order relation on X. Then $x, y \in X$ are comparable if $x \leq y$ or $y \leq x$.*

Definition 4. ***Total order :****A total (simple /linear) order on a set X is a partial order on X which statisfies the law of dichotomy ie. for every $x, y \in X$, x and y are comparable.*

Note : If $\leq$ is a partial order on the set X and $Y \subset X$, then $\leq /Y$ i.e. the rstriction of $\leq$ to Y is also a partial order. If $\leq$ is linear, then so is $\leq /Y$. A linearly ordered subset of a

poset is called a chain, a tower or a nest.

Further examples :
1. Let X be any set and $P(X)$ its power set. Then the inclusion defines a partial order on $P(X)$. Notices that inclusion relation on $P(X)$ is not a total relation.
2. Let $(X_1, \leq_1)$ and $(X_2, \leq_2)$ be posets. Let $X = X_1 \times X_2$. We define a partial order $\leq$ on X by

$$(x_1, x_2) \leq (y_1, y_2)$$

if and only if $x_1 \leq y_1$ or $x_1 = y_1$ and $x_2 \leq y_2$. In other words, we compare the elements coordinate wise. More generally, we can do this for Cartesian product of any finite number of posets. The ordering defined here is called lexicographic or dictionary ordering (for obvious reasons!).
3. Let $\mathbf{R}^* = \mathbf{R} \cup \{-\infty, \infty\}$, where $-\infty$ and ∞ are just two symbols not representing any real numbers. Define $\leq$ on $\mathbf{R}^*$ as $\{(x, y) \in \mathbf{R} \times \mathbf{R} : x \leq y$ in the usual order $\} \cup (\{-\infty\} \times \mathbf{R}^*) \cup (\mathbf{R}^* \times \{\infty\})$ i.e. we retain the usual order on $\mathbf{R}$ but extend it to the points ∞ and $-\infty$ by declaring ∞ is greater than everything else and $-\infty$ is less than everything else. $\mathbf{R}^*$ with this ordering is called the extended real line.

Definition 5. *Let $(X, \leq)$ be a partially ordered set. $A \subset X$ and $x \in X$,*

- *x is called a upper bound of A if for all $a \in A$, $a \leq x$.*
- *x is called a maximum element of A if $x \in A$ and x is an upper bound of A.*
- *x is called a maximal element of A if $x \in A$ and for every $a \in A$, $x \leq a$, implies $x = a$.*
- *x is said to be the least upper bound (supremum) of A if $x \leq a$ for every $a \in U$, where U is the set of all upper bounds of A.*

Question to ponder : What is the distinction between maximum element and maximal element?

Similarly we can define lower bound, minimum element, minimal element and greatest lower bound (infimum) of the set A.

Theorem 1.1.3. *Let $(X. \leq)$ be a poset and A be a non empty finite subset of X. Then A has at least one maximal element. Also A has a maximum element if and only if it has a unique maximal element.(Similar assertions holds for minimal elements.)*

Proof.

- Let $x_1 \in A$. If x_1 is not a maximal element of A, then there exists x_2 such that $x_1 < x_2$. If x_2 is not a maximal element, then there exists x_3 such that $x_2 < x_3$. Continuing this process we get a sequence $x_1 < x_2 < x_3$.... But since A is finite, the sequence has to stop at some stage. But then A would have a maximal element.

- For the second assertion, suppose x be the maximum element of A. Then x is also the unique maximal element of A. For the converse implication, suppose A has a unique maximal element x, then we claim that x is the maximum element of A. If not, then there exits an element $x_1 \in A$ not comparable with x. Now using the same argument as above, we would get a sequence $x_1 < x_2 < x_3$..... Since A is finite, this sequence must terminate and hence x_n would be a maximal element of A different from x, contradicting our assumption.

□

Definition 6. *Let $(X, \leq)$ be a poset and $x, y \in X$. Then y is said to cover x if $x < y$ and there does not exits a $z \in X$ such that $x < z$ and $z < y$.*

Proposition 1.1.4. *Let $(X, \leq)$ be a poset and $x \in X$. Define*

$$A = \{z \in X : x < z\}$$

Then $y \in X$ covers x if and only if y is a maximal element of A.

Theorem 1.1.5. *Let $(X, \leq)$ be a finite poset. Define a binary relation R on X by xRy if and only if y covers x with respect to $' \leq'$. Then $' \leq'$ is generated by R i.e. $' \leq'$ is the smallest order relation containing R.*

Proof. Clearly, as subsets of $X \times X$, R is contained in $\leq$. We have to show that we can get $\leq$ by suitably extending R. First we add ΔX for the sake of reflexivity. So let $R_1 = R \cup \Delta X$. Then R_1 is reflexive. Now we have to show that the order is generated by R_1 is the original relation $\leq$. For this we have to show that given $x, y \in X$ with $x \leq y$, we can find some sequence of elements $x_1, x_2, ..., x_n$ such that $x = x_1$, $y = x_n$ and $(x_i, x_{i+1}) \in R_1$ for all $i = 1, 2, ..., n-1$. If $x = y$, then $(x, y) \in \Delta X$ and hence $(x, y) \in R_1$, so we take $x_1 = x$ and $x_2 = y$. If $x < y$ then we proceed by induction on the number of elements in between x and y. Let r be the number of such elements. If $r = 0$, then y covers x and so $(x, y) \in R$ by definition. So again $(x, y) \in R_1$. Suppose $r > 0$. Let A be the set $\{z \in X : x < z \; and \; z < y\}$. Then, by definition, $|A| = r$. Fix some $z \in A$ and let

$$B = \{w \in X : x < w < z\}$$

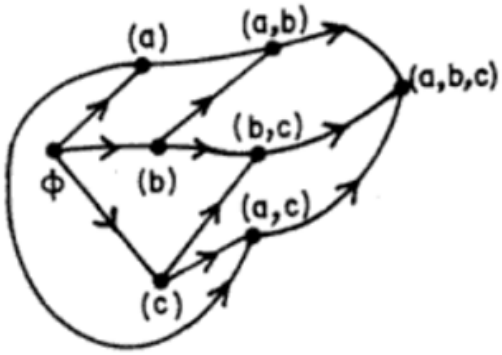
and

$$C = \{w \in X : z < w < y\}.$$

Then B and C are proper subsets of A and so $|B| < r$ and $|C| < r$. So by induction hypothesis, there exist sequences $x_1, x_2, ..., x_n$ and $y_1, y_2, ..., y_m$ such that $x = x_1, z = y_1$ and $y = y_m$ and $(x_i, x_{i+1}) \in R_1$ for $i = 1, 2, ..., n-1$ and $(y_j, y_{j+1}) \in R_1$ for $j = 1, 2, ..., m-1$. Concatenating these two sequences we get a sequence of the desired type with the first term x and last term y. This completes the induction and proves our assertion that if $x < y$ in X then the pair (x, y) belongs to the smallest transitive relation containing R_1. So it follows that $\leq$ is generated by R_1. □

Definition 7. ***Hasse diagram :*** *Hasse diagram of a relation R on set X is the graphical representation of the relation R. It is obtained by viewing the elements of X as points and drawing an arrow from x to y when y covers x.*

Example : Let X be the power set of $\{a, b, c\}$ with inclusion as the partial order. The Hasse diagram of this relation is given below



Definition 8. *Let X be a poset and $x, y \in X$. Then the supremum and infimum of the set (x, y)(if they exists) is respectively called the **join** (denoted by $x \vee y$) and the **meet** (denoted by $x \wedge y$). A poset in which every pair of elements has a join and a meet is called a **lattice** and its Hasse diagram is called **lattice diagram**.*

Note : If x and y are comparable, then obviously $x \vee y$ is the greater of two and $x \wedge y$ is the smaller of two. Thus, every totally ordered set is a lattice.

Example : On the set of non-negative integers define a partial order a/b means a divides b ie. a is related to b if there exits a positive integer n such that $b = na$. Then it is a lattice.

Exercise :

1. Prove that a partial order $\leq$ on a set X is total iff the corresponding strict order $<$

satisfies the following property: for all $x, y \in X$ either $x < y$ or $x = y$ or $y < x$.

2. If A is a chain on a poset X, $|A|$ is called the length of A. if $X = P(B)$, where B is a set with n elements and the partial ordering on X is by set inclusion, prove that
i. for $S, T \in X$, T covers S iff $s \subset T$ and $T \setminus S$ is a singleton set.
ii. the longest chain in X is of length $n + 1$.
iii. the number of such chains is $n!$.

3. Prove that intersection of two chains is a chain but their union need not be a chain.

4. Let m be the largest possible number of mutually incomparable elements of a poset X. Prove that X cannot be expressed as the union of less than m chains.

5. Prove that the poset $(P(\mathbf{N}), \subset)$, where $\mathbf{N}$ is the set of positive integers, contains an uncountable chain.

6. Prove that the Hasse diagram of a poset cannot contain a cycle, i.e. a sequence of consecutive arrows terminating where it starts.

1.2 Boolean algebra

Devices which occur in two states are called binary or two-state devices. We arbitrarily assign the symbols 1 and 0 to these states. These two states are called complementary or opposite states. Two devices that are always in the opposite states are said to be complementary to each other. A conjunction of two binary device is defined as a binary device which is in state 1 iff both of them are in state 1. A disjunction of two binary device is defined as a device which is in state 1 iff at least one of them is in state 1. The structure of Boolean algebra is meant to isolate these basic concepts of complement, conjunction and disjunction about two devices.

Definition 9. *A Boolean algebra is an ordered quadruple of the form $(X, +, ., ')$, where X is a non empty set, $+$ and $.$ are two binary operations on X and $'$ is a unary operation on X satisfying the following conditions:*
1. both $+$ and $.$ are commutative
2. both $+$ and $.$ have identities denoted by 0 and 1 respectively
3. both $+$ and $.$ are distributive over each other
4. for all $x \in X$, $x + x' = 1$ and $x.x' = 0$.

If $x \in X$, then x' is called the complement of x. If $x, y \in X$, then $x + y$ and $x.y$ are called the disjunction and conjunction of x and y respectively.

Examples :

1. Take X as the power set of some set, say S, defining $+$ as $\cup$, . as $\cap$ and $'$ as complementation with respect to S. The empty set ϕ is the identity for $+$ and S is the identity for $'.'$. For any fixed $x \in S$, every subset of S is in two state, because for any subset A of S there are two possibilities, either $x \in A$ or $x \notin A$.

2. Take $S = \phi$, then the Boolean algebra $P(S)$ consists of only one element. This is a trivial Boolean algebra. In this Boolean algebra, 1 and 0 coincides.

3. The simplest non-trivial Boolean algebra has 0 and 1 as its only elements.An example of such a Boolean algebra is he empty set ϕ and X. Such Boolean algebras are very important and are often denoted by $\mathbf{Z}_2$.

Definition 10. *A non empty subset Y of a Boolean algebra $(X, +, ., ')$ is called a sub-algebra if it is closed under the operations $+, ., '$.*

Proposition 1.2.1. *Let Y be a sub-algebra of a Boolean algebra $(X, +, ., ')$. Then, Y with the induced operations is a Boolean algebra with the same identity elements as X.*

Theorem 1.2.2. *Let $(X, +, ., ')$ be a Boolean algebra. Then the following properties hold for all elements x, y, z of X,*
1. $x + x = x$ and $x.x = x$. (Laws of Tautology or idempotency)
2. $x + 1 = 1$ and $x.0 = 0$. ()
3. $x + x.y = x$ and $x.(x + y) = x$. (Laws of Absorption)
4. $x + (y + z) = (x + y) + z$ and $x.(y.z) = (x.y).z$. (Associative laws)
5. If $x + y = 1$ and $x.y = 0$, then $y = x'$. (Uniqueness of Complements)
6. $(x')' = x$. (Law of Double Complementations)
7. $(x + y)' = x'.y'$ and $(x.y)' = x' + y'$ (De Morgan's Laws)
8. $0' = 1$ and $1' = 0$
9. $0 \neq 1$ unless X has only one element.

Proof. Refer to the text. □

Using the above rules, we can simplify an algebraic expression involving elements of a Boolean algebra.

Definition 11. *Let $(X, +, ., ')$ be a Boolean algebra. If $x, y \in X$, we say $x \leq y$ if $x.y' = 0$.*

Theorem 1.2.3. *The relation $\leq$ defined above makes the underlying set of a Boolean algebra into lattices. Moreover, 0 and 1 are the minimum and maximum elements of this lattices.*

Proof.

- Let $(X, +, .,')$ be a Boolean algebra. For $x, y, \in X$, we have $x \leq y$ if $x.y' = 0$. We want to show that $(X, \leq)$ is a lattice.
- First verify that $\leq$ is a partial order on X.
- To verify the lattice properties, let $x, y \in X$. We claim that join of x and y is $x + y$. $x.(x + y)' = x.x'.y' = 0$, thus $x \leq x + y$ and similarly $y \leq x + y$. Hence $x + y$ is an upper bound of x and y. Now, to prove that $x + y$ is the least upper bound, suppose z be an upper bound of x and y i.e $x \leq z$ and $y \leq z$. So, $(x + y)z' = xz' + yz' = 0$. This proves our claim. The proof that meet of x and y is obtained by duality.
- If $x \in X$, then $0.x' = 0$, showing that 0 is the minimum element of X. Similarly, $x.1' = 0$. Thus 1 is the maximum element of X.

□

The above theorem shows that every Boolean algebra gives rise to a lattice. Then, naturally the question raises whether the converse is true. Obviously the answer is no as we have seen that the lattices obtained from the Boolean algebra is always bounded. But it turns out that a bounded lattice satisfying a couple of properties indeed arises from a Boolean algebra.

Definition 12. *Let $(X, \leq)$ be a bounded lattice with 0 and 1 as its minimum and maximum elements respectively. Then X is called complemented if for every $x \in X$, there exists some $y \in X$ such that $x \vee y = 1$ and $x \wedge y = 0$. Any such y is called a complement of x.*

Theorem 1.2.4. *Let $(X, +, .,')$ be a Boolean algebra. Then the corresponding lattice $(X, \leq)$ is complemented and distributive. Conversely, if $(X, \leq)$ is a bounded, complemented and distributive lattice then there exists a Boolean algebra structure on $X, (X, +, .,')$ such that the partial order relation defined by this structure coincides with the given relation $\leq$.*

Proof.

- First part is straight forward since $\vee$ and $\wedge$ are precisely $+$ and . respectively.
- For the converse, suppose $(X, \leq)$ is a bounded, complemented, distributive lattice with 0 and 1 as the smallest and largest elements. For $x, y \in X$, define

$$x + y = x \vee y$$

and

$$x.y = x \wedge y$$

The operations $+$ and $.$ are commutative with 0 and 1 as their respective identities and are distributive over each other. Now, for each $x \in X$, select any one complement of x denote it as x'. This gives a function $' : X \to X$ such that $(X, +, .,')$ is a Boolean algebra.

- Now let $\preceq$ be the partial order on X induced by this Boolean algebra i.e. for $x, y \in X$, $x \preceq y$ iff $xy' = 0$. We have to show that $x \preceq y$ iff $x \leq y$. Suppose $x \preceq y$, then $x.y' = 0$. So $x = x.1 = x(y + y') = xy + xy' = x.y$. This mean $x = x \wedge y$. But $x \wedge y \leq y$, which gives $x \leq y$. Conversely suppose $x \leq y$. Then $x = x \wedge y = x.y$. Hence $xy' = xyy' = 0$. So $x \preceq y$.

□

Definition 13. *Let $(X, +, .,')$ be a Boolean algebra. Then a minimal element of the set $X \setminus \{0\}$ is called an atom of X.*

Atoms of a power set Boolean algebra $P(S)$ are precisely the singletons subsets of S.

Proposition 1.2.5. *Let $(X, +, .,')$ be a finite Boolean algebra. Then,*
1. every non zero element of X contains at least one atom
2. every two distinct atoms of X are mutually disjoint
3. every element of X can be uniquely expressed as a sum of atoms, specifically if $x \in X$, then x is the sum of all atoms contained in x (with the understanding that an empty sum is 0).

Theorem 1.2.6. *Every finite Boolean algebra is isomorphic to a power set Boolean algebra, specifically to the power set Boolean algebra of the set of all its atoms.*

Proof. Refer the text. □

Corollary 1.2.6.1. *If X is a finite Boolean algebra then $|X| = 2^n$ for some non-negative integer n.*

Exercise :

1. Let S be any set and $\mathbf{Z}_2$ a Boolean algebra with two elements 0 and 1. Prove that the Boolean algebra of all functions from S to $\mathbf{Z}_2$ (under point-wise multiplication) is isomorphic to the power set Boolean algebra of S.
2. Suppose $S$and T are disjoint set. Prove that the product Boolean algebra of $P(S) \times P(T)$ is isomorphic to the Boolean algebra $P(S \cup T)$.
3. Given a Boolean algebra $(X, +, .,')$ prove that for any $x, y, z \in X$, if $x + y = x + z$ and

$x.y = x.z$ then $y = z$.
4. If $x_1, x_2, , , , x_n$ are elements of a Boolean algebra prove that

$$x_1 + x_2 + ... + x_n = 0$$

iff $x_i = 0$ for all $i = 1, 2, , , , n$ and

$$x_1.x_2....x_n = 1$$

iff $x_i = 1$ for all $i = 1, 2, , , , n$.
5. If x, y are elements of Boolean algebra, prove that $x = y$ iff $xy' + x'y = 0$.
6. Suppose $(X, \leq)$ is a lattice with a maximum element 1. Then a maximal element of $X - \{1\}$ is called a co-atom. Prove that an element a of a Boolean algebra is a co-atom iff its complement a' is an atom. Prove that every element of a finite Boolean algebra can be uniquely expressed as a product of co-atoms.

1.3 Boolean function

Definition 14. *A Boolean variable is a variable which assumes only two possible values,* 0 *or* 1. *In other words, Boolean variable takes values from* $\mathbf{Z}_2$. *Since* $\mathbf{Z}_2$ *is a Boolean algebra, we can define addition, multiplication and complements of Boolean variables. Two Boolean variables* x *and* y *are said to be independent if each cab assume values independently of the other.*

Definition 15. *A Boolean function of* n *variables is any function from* $\mathbf{Z}_2^n$ *to* $\mathbf{Z}_2$.

For example, suppose $n = 3$ and $f(x_1, x_2, x_3) = x_1x_2 + x_2'x_3$. Then f is a Boolean function of three variables. If we substitute the values $x_1 = 1, x_2 = 0$ and $x_3 = 1$, then we get

$$f(1, 0, 1) = 1.0 + 1.1 = 1$$

There are two ways to denote a Boolean function f. The first method is to write the table of values of f, in which we actually list down the values of f for all possible elements of the domain. Shown below is the table of values of the function $f(x_1, x_2, x_3) = x_1x_2 + x_2'x_3$,

Row No.	x_1	x_2	x_3	f
1	1	1	1	1
2	1	1	0	1
3	1	0	1	1
4	1	0	0	0
5	0	1	1	0
6	0	1	0	0
7	0	0	1	1
8	0	0	0	0

The other way to write the table of values of a Boolean function, which is a little more concise. The idea is to replace several (say k) Boolean variables by a single variable taking 2^k possible values. The concise table for Boolean function $f(x_1, x_2, x_3) = x_1x_2 + x_2'x_3$ is given below,

(x_1, x_2) \ x_3	1	0
(1,1)	1	1
(1,0)	1	0
(0,1)	0	0
(0,0)	1	0

Definition 16. *Let $x_1, x_2, , , , x_n$ be mutually independent Boolean variable. For $i = 1, 2, , , , n$, let $f_i : \mathbf{Z}_2^n \to \mathbf{Z}_2$ be the Boolean function defined by,*

$$f_i(x_1, x_2, , , , x_n) = x_i.$$

The function f_i is called projection function on the ith factor and denoted by π_i.

Theorem 1.3.1. *Let $x_1, x_2, , , , x_n$ be mutually independent Boolean variable. Then there are 2^{2^n} Boolean functions of these n variables. The totality of such functions constitutes a Boolean algebra. The atoms of this algebra are the 2^n functions of the form*

$$x_1^{\epsilon_1} x_2^{\epsilon_2} .....x_n^{\epsilon_n}$$

where each $x_i^{\epsilon_i}$ is either x_i or x_i'.

Proof.

- Let X be the set of all Boolean functions of n variables i.e. elements of X are functions from $\mathbf{Z}_2^n$ to $\mathbf{Z}_2$. We know that $|\mathbf{Z}_2^n| = 2^n$, thus we have $|X| = 2^{2^n}$.

- We will first show that all the functions of the form

$$x_1^{\epsilon_1} x_2^{\epsilon_2} .....x_n^{\epsilon_n}$$

are atoms. Let g be one such function and for simplicity we suppose $g = x_1 x_2 ....x_k x'_{k+1} ...x'_n$, for some k.Note that g vanishes for all point except $(1, 1, , , 1, 0, 0, , , , 0)$, where the first k entries are 1 and remaining entries are all 0. Show that g is a atom of X. Thus every function of the form

$$x_1^{\epsilon_1} x_2^{\epsilon_2} .....x_n^{\epsilon_n}$$

is an atom of X and we have 2^n such elements.

- By1.2.6.1, the number of atoms of a Boolean algebra with 2^{2^n} elements is precisely 2^n. Thus, there can be no other atom.

□

Theorem 1.3.2. *Every Boolean function of n variables $x_1, x_2, , , , x_n$ can be uniquely expressed as a sum of terms of the form*

$$x_1^{\epsilon_1} x_2^{\epsilon_2} .....x_n^{\epsilon_n}$$

where each $x_i^{\epsilon_i}$ is either x_i or x'_i. Specifically, for every element

$$\overline{y} = (y_1, y_2, , , , y_n)$$

of $\mathbf{Z}_2^n$, let $f_{\overline{y}}$ be the Boolean function from $\mathbf{Z}_2^n \to \mathbf{Z}_2$ defined by $f_{\overline{y}}(\overline{y}) = 1$ and $f_{\overline{y}}(\overline{z}) = 0$, for all $\overline{z} \in \mathbf{Z}_2^n \setminus \{\overline{y}\}$ and let $x_{\overline{y}}$ be the term

$$x_1^{\epsilon_1} x_2^{\epsilon_2} .....x_n^{\epsilon_n}$$

where $x_i^{\epsilon_i} = x_i$ if $y_i = 1$ and $x_i^{\epsilon_i} = x'_i$ if $y_i = 0$. Then every function

$$f : \mathbf{Z}_2^n \to \mathbf{Z}_2$$

equals $\sum x_{\overline{y}}$ where the sum extends over all $\overline{y}$ such that $f(\overline{y}) = 1$.

Proof. Proof left to the reader. □

Example : We have already seen the table of values of the Boolean function $f(x_1, x_2, x_3) = x_1 x_2 + x'_2 x_3$. The function takes value 1 at four points, namely,

$$(1, 1, 1), (1, 1, 0), (1, 0, 1), (0, 0, 1)$$

Thus the corresponding atoms are

$$x_1x_2x_3, x_1x_2x_3', x_1x_2'x_3, x_1'x_2'x_3$$

hence f equals the sum of there four atoms i.e.

$$\begin{aligned} f &= x_1x_2x_3 + x_1x_2x_3' + x_1x_2'x_3 + x_1'x_2'x_3 \\ &= x_1x_2(x_3 + x_3') + (x_1 + x_1')x_2'x_3 \\ &= x_1x_2 + x_2'x_3 \end{aligned}$$

Thus, we have shown that even if we do not know the original formula for f, but simply the points where f takes the value 1 then we can always get an algebraic expression for f by adding the corresponding atoms.

Definition 17. *The algebraic expression for a Boolean function of n variables as s sum of atoms is called the disjunctive normal form (D.N.F) of that function. The disjunctive normal form of the function which identically equals 1 is called the complete disjunctive normal form in n variables.*

Sometimes we are already given some function f in an algebraic form and we want to cast it into its D.N.F. One method is to write the table of values of f. Other method is to write f as a sum of monomials, where by a monomial we mean a product in which every factor is some variable or its complement. Because $x_ix_i' = 0$, we can ignore monomials in which some variable and its complement occur. Also because $x_ix_i = x_i$, if the same variable occur more than once in a monomial, we retain only its first occurrence. If neither x_i or x_i' occur in a monomial, then we multiply the monomial by $x_i + x_i'$. We repeat this process until all the monomial in the original expression of f is expressed as a sum of monomials in which all the variables occur. The resulting expression is the D.N.F of the given function.

Examples : Write the following functions in D.N.F

1. $f(x_1, x_2, x_3) = (x_1 + x_2')x_3' + x_2x_1'(x_2 + x_1x_3)$

$$\begin{aligned} f(x_1, x_2, x_3) &= x_1x_3' + x_2'x_3' + x_2x_1' + x_2x_1'x_1'x_3 \\ &= x_1(x_2 + x_2')x_3' + (x_1 + x_1')x_2'x_3' + x_1'x_2(x_3 + x_3') + x_1'x_2x_3 \\ &= x_1x_2x_3' + x_1x_2'x_3' + x_1'x_2x_3 + x_1'x_2x_3' + x_1'x_2'x_3' \end{aligned}$$

2. $g(a, b, c) = (a + b + c)(a' + b + c')(a + b' + c')(a' + b' + c')(a + b + c')$

We use that fact that addition is distributive over multiplication i.e. $a+bc=(a+b)(a+c)$.

$$\begin{aligned}
g(a,b,c) &= (a+b+c)(a'+b+c')(a+b'+c')(a'+b'+c')(a+b+c') \\
&= (a+b+c)(a+b+c')(a'+b+c')(a+b'+c')(a'+b'+c')(a'+b'+c')(law\ of\ tautology) \\
&= (a+b+c.c')(a'+b+c')(a'+b'+c')(a+b'+c')(a'+b'+c') \\
&= (a+b)(a'+c')(b'+c') \\
&= (a+b)(a'b'+c') \\
&= ac'+bc' \\
&= a(b+b')c'+(a+a')bc' \\
&= abc'+ab'c'+a'bc'
\end{aligned}$$

Proposition 1.3.3. *Let two functions f and g of the same n Boolean variable be expressed in their respective disjunctive normal form. Then the D.N.F of $f+g$ is obtained by taking the sum f those terms which appear in the D.N.F of at least one of f and g; D.N.F of $f.g$ is obtained by summing terms common to D.N.F of both f and g and the D.N.F of f' is obtained by omitting from the complete D.N.F the terms which appear in the D.N.F of f.*

Definition 18. *A Boolean function f of n variables $x_1,x_2,,,,x_n$ is said to be symmetric w.r.t a pair of variables x_i and x_j if the values of f is unaffected by interchanging x_i and x_j i.e. if for all values of $x_1,x_2,,,,,x_n$, we have*

$$f(x_1,,,x_{i-1},x_i,x_{i+1},,,x_{j-1},x_j,x_{j+1},,,x_n)=f(x_1,,,x_{i-1},x_j,x_{i+1},,,x_{j-1},x_i,x_{j+1},,,x_n)$$

If f is symmetric w.r.t every pair of variables, then f is called symmetric among these variables.

For example, the function $x_1x_2'+x_2x_3'+x_3x_1'$ is symmetric in three variables. Its not quite obvious, but when we write its D.N.F,

$$x_1x_2'x_3+x_2x_3'x_1+x_3x_1'x_2+x_1x_2'x_3'+x_1'x_2'x_3+x_1'x_2x_3'$$

it becomes pretty obvious.

Definition 19. *Let f be a symmetric function of n Boolean variables. Then an integer $r(0\le r\le n)$ is called a characteristics number of f if $f(e_r)=1$, where e_r is the binary sequence of length n whose first r entries are 1 and remaining entries are 0. The identically zero function has no characteristic number while the function which is identically 1 has every integer between 0 and n as a characteristic number.*

Theorem 1.3.4. *The characteristic numbers of a symmetric Boolean completely determine*

it. In other words, given integers

$$0 \leq r_1 \leq r_2 \leq .... \leq r_k \leq n$$

there exists one and only one symmetric function of n Boolean variables whose characteristics number are $r_1, r_2, , , , r_k$.

Proof. For an integer r, $0 \leq r \leq n$ let S_r be the set of all $\binom{(}{n}, r)$ terms of the form $x_1^{\epsilon_1}.....x_n^{\epsilon_n}$ where exactly r of the n variables occur without primes(') and rest $n-r$ occur with primes('). Let f be a symmetric Boolean function of $x_1, , , , , , x_n$ with characteristic numbers $r_1, , , , , r_k$. Then because of symmetry, f contains all the terms of $S_{r_1}, , , , , , S_{r_k}$ and none of the terms of S_r for $r \neq r_1, , , , , r_k$. Therefore the D.N.F of f must be sum of the terms in $S_{r_1}, , , , , , S_{r_k}$. Since two distinct functions cannot have the same D.N.F, f is uniquely determined by this expression. □

Theorem 1.3.5. *The set of all symmetric Boolean functions of n Boolean variables $x_1, x_2, , , , x_n$ is a sub-algebra of the Boolean algebra of all Boolean functions of these variables. As a Boolean algebra, it is isomorphic to the power set Boolean algebra of the set $\{0, 1, , , , n\}$.*

Proof.

- Let X denote the Boolean algebra of all the Boolean functions of $x_1, x_2, , , , , x_n$ under the point-wise operation and let Y be the set of all symmetric functions of $x_1, x_2, , , , , x_n$. Let $f, g \in Y$, then f, g are unaffected under the interchange of any two variable. So obviously the same is true for $f+g$. Hence $f+g$ is symmetric. So Y is closed under $+$. Similarly, Y is closed under . and $'$. Also Y always contains the constant functions 1 and 0. Thus, Y is a sub-algebra of X.

- Let T be the set $\{1, 2, 3, , , , , n\}$. We will now give an explicit isomorphism between Y and $P(T)$. For a symmetric function of $x_1, x_2, , , , x_n$, let $C(f)$ denote the set of characteristics numbers of f. Then $C(f) \in P(T)$. Define a function $\theta : Y \to P(T)$ by $\theta(f) = C(f)$. By previous theorem, θ is bijective. Now to show that θ is an isomorphism, we have to show that θ is compatible with the operations i.e for all $f, g \in Y$,
  i. $C(f+g) = C(f) \cup C(g)$
  ii. $C(f.g) = C(f) \cap C(g)$
  iii. $C(f') = T \setminus C(f)$
  This part is left to the reader to work out.

□

Exercise :

1. Prepare the table of values of the following functions and and obtain their D.N.F
i. $x_1'x_2(x_1' + x_2 + x_1x_3)$
ii. $a + b + c'$
iii. $(xy + x'y + x'y')(x + y)$

2. Let f, g be two Boolean functions of n variables. Prove that f is a summand of g (i.e. there exists a Boolean function h such that $f + h = g$) iff for all $\overline{y} = \{y_1, y_2, , , , y_n \in \mathbf{Z}_2^n$, $f(\overline{y}) = 1$ implies $g(\overline{y}) = 1$.

3. Decide which of the following functions are symmetric. In case of symmetric functions find their characteristic number
i. $(a + b'c)(b + c'a) + (c + a'b)$
ii. $x_1x_2x_3x_4' + x_2x_3x_4x_1' + x_3x_4x_2x_1' + x_4x_1x_2x_3'$
iii. $x_1x_2'x_3x_4' + x_2x_3'x_4x_1'$

4. A Boolean function $f(x_1, x_2, , , , x_n)$ is said to be cyclically symmetric if for all $(x_1, , , , x_n) \in \mathbf{Z}_2^n$,

$$f(x_1, x_2, , , , x_n) = f(x_2, x_3, , , , x_n, x_1)$$

Prove that f is cyclically symmetric then

$$f(x_1, x_2, , , , x_n) = f(x_2, x_3, , , , x_n, x_1) = f(x_3, x_4, , , , x_n, x_1, x_2) = ... = f(x_n, x_1, , , , x_{n-1})$$

5. Prove that every symmetric function is cyclically symmetric and that the converse is not true in general.

6. Prove that for $n = 1, 2, 3$ every cyclically symmetric Boolean function is symmetric.

Module 2

Graph theory

(Reference text: 'A textbook of graph theory' by R. Balakrishnan and K. Ranganathan)

2.1 Basic definitions

2.1.1 Introduction

Graph theory is the study of graphs, first studied by the mathematician Leonhard Euler for the famous Konigsberg bridge problem. In this we course, we will begin with the basic definitions of a graph and isomorphisms of graphs and later introduce the concepts of tree, Eulerian graphs, and planar and non-planar graphs.

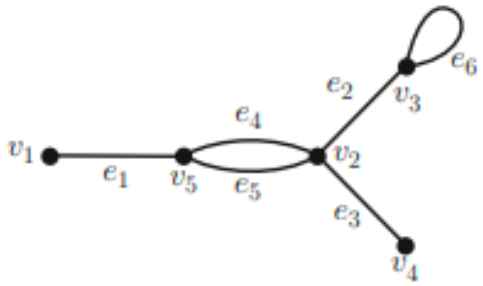
Definition 20. *A graph is an ordered triple $G = (V(G), E(G), I(G))$, where $V(G)$ is a nonempty set, EG is a set disjoint from $V(G)$ and I_G is an "incidence" relation that associates with each element of $E(G)$ an unordered pair of elements of $V(G)$. Elements of $V(G)$ are called the vertices (or nodes or points) of G; and elements of $E(G)$ are called the edges (or lines) of G. $V(G)$ and $E(G)$ are the vertex set and edge set of G, respectively. If, for the edge e of G $I_G(e) = \{u, v\}$, we write $I_G(e) = uv$.*

Diagrammatic Representation of a Graph : Each graph can be represented by a diagram in the plane, where each vertex of the graph is represented by a point and distinct vertices are represented by distinct points. Each edge is represented by a simple arc joining two (not necessarily distinct) vertices.

Definition 21. *Let $G = (V(G), E(G), I(G))$ be graph and $I_G(e) = \{u, v\}$, then the vertices u and v are called the end vertices or ends of the edge e. A set of two or more edges of a graph G is called a set of multiple or parallel edges if they have the same pair of distinct ends. If e is an edge with end vertices u and v; we write $e = uv$. An edge for which the two ends are the same is called a loop at the common vertex. A vertex u is a neighbor of v in G if uv is an edge of G. and $u \neq v$. The set of all neighbors of v is the open neighborhood*

of v or the neighbor set of v, and is denoted by $N_G(v)$. The set $N_G[v] = N_G(v) \cup \{v\}$ is called the closed neighborhood of v in G. Vertices u and v are adjacent to each other in G if and only if there is an edge of G with u and v as its ends. Two distinct edges e and f are said to be adjacent if and only if they have a common end vertex. A graph is simple if it has no loops and no multiple edges. Thus, for a simple graph G; the incidence function I_G is one-to-one.

Example : $V(G) = \{v_1, v_2, v_3, v_4, v_5\}$ and $E(G) = \{e_1, e_2, e_3, e_4, e_5, e_6\}$ and I_G given by $I_G(e_1) = \{v_1, v_5\}, I_G(e_2) = \{v_2, v_3\}, I_G(e_3) = \{v_2, v_4\}, I_G(e_4) = \{v_2, v_5\}, I_G(e_5) = \{v_2, v_5\}, I_G(e_6) = \{v_3, v_3\}$.



- v_1 and v_5 are the end vertices of the edge e_1.
- The edges $\{e_4, e_5\}$ is an example of multiple or parallel edges.
- The edge e_6 is a loop at the vertex v_3.
- $N_G(v_2) = \{v_4, v_5, v_3\}$ and $N_G[v_2] = \{v_4, v_5, v_3, v_2\}$.
- e_4 and e_2 are adjacent edges.

Definition 22. *A graph is called finite if both $V(G)$ and $E(G)$ are finite otherwise it is called an infinite graph. $n(G)$ denotes the number of vertices of the graph G and is called the order of G. $m(G)$ denotes the number of edges of the graph G and is called the size of G. A graph is said to be labeled if its vertices are distinguished from one another by labels such as $\{v_1, v_2, , , , , v_n...\}$.*

Definition 23. ***Isomorphism of graphs :*** *Let $G = \{V(G), E(G), I_G\}$ and $H = \{V(H), E(H), I_H\}$ be two graphs. A graph isomorphism from G to H is a pair (ϕ, θ), where $\phi : V(G) \to V(H)$ and $\theta : E(G) \to E(H)$ are bijections with the property that for $e \in E(G)$ and $u, v \in V(G)$, $I_G(e) = \{u, v\}$ if and only if $I_H(\theta(e)) = \{\phi(u), \phi(v)\}$. Isomorphism between graphs is denoted by the symbol $\cong$.*

Note : For two simple graphs G and H, a bijection $\phi : V(G) \to V(H)$ such that u and v are adjacent vertices if and only if $\phi(u)$ and $\phi(v)$ are adjacent vertices, induces a bijection $\theta : E(G) \to E(H)$ satisfying the condition $I_G(e) = \{u, v\}$ if and only if $I_H(\phi(e)) = \{\theta(u), \theta(v)\}$. Thus,for simple graphs G and h, an isomorphism from G to H is a bijection $\phi : V(G) \to V(H)$ such that u and v are adjacent in G if and only if $\theta(u)$ and $\theta(v)$ are adjacent in H.

Definition 24. *A simple graph G is said to be complete if every pair of distinct vertices of G are adjacent in G: Any two complete graphs each on a set of n vertices are isomorphic and each such graph is denoted by K_n.*

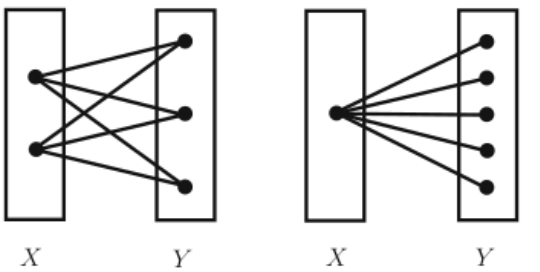
Examples for complete graphs :

K_3 K_4

Question to ponder : What is the maximum number of edges a simple graph of n distinct vertices have?

Definition 25. *A graph is trivial if its vertex set is a singleton and it contains no edges. A graph is bipartite if its vertex set can be partitioned into two nonempty subsets X and Y such that each edge of G has one end in X and the other in Y. The pair (X, Y) is called a bipartition of the bipartite graph. The bipartite graph G with bipartition (X, Y) is denoted by $G(X, Y)$. A simple bipartite graph $G(X, Y)$ is complete if each vertex of X is adjacent to all the vertices of Y. If $G(X, Y)$ is complete with $|X| = p$ and $|Y| = q$ then $G(X, Y)$ is denoted by $K_{p,q}$. A complete bipartite graph of the form $K_{1,q}$ is called a star.*

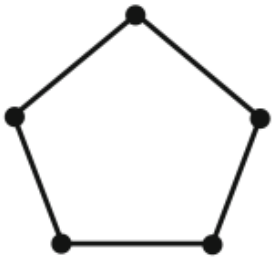
Examples of complete bipartite graphs.



The graph $K_{2,3}$ The star graph $K_{1,5}$

Definition 26. *Let G be a simple graph, then the complement G^c of G is defined by taking $V(G^c) = V(G)$ and making two vertices u and v adjacent in G^c if and only if they are non adjacent in G. A simple graph G is called self-complementary if $G = G^c$.*

Check whether the following graph is self-complementary.



2.1.2 Subgraphs

Definition 27. *A graph H is called a subgraph of G if $V(H) \subseteq V(G)$,$E(H) \subseteq E(G)$ and I_H is the restriction of I_G to $E(H)$. If H is a subgraph of G, then G is called the supergraph of H. A subgraph H of a graph G is a proper subgraph of G if either $V(H) \neq V(G)$ or $E(H) \neq E(G)$. A subgraph H of G is said to be an induced subgraph of G if any edge of G having its ends in $V(H)$ is also an edge of H. A subgraph H of G is a spanning subgraph of G if $V(H) = V(G)$. The induced subgraph of G with vertex set $S \subseteq V(G)$ is called the subgraph of G induced by S and is denoted by $G[S]$. Let E' be a subset of E and let S denote the subset of V consisting of all the end vertices in G of edges in E'. Then the graph $\{S, E', I_G|_{E'}\}$ is the subgraph of G induced by the edge set E' of G and is denoted by $G[E']$. Let u and v be vertices of a graph G, then $G + uv$ is the the graph obtained by adding a new edge uv to G.*

Definition 28. *A clique of G is a complete subgraph of G. A clique of G which is not properly contained in another clique of G is called a maximal clique of G.*

Definition 29. *Let G be a graph, S a proper subset of the vertex set V and E' a subset of E. The subgraph $G[V \setminus S]$ is said to be obtained from G by the deletion of S and is denoted by $G \setminus S$.The spanning subgraph of G with the edge set $E \setminus E'$ is the subgraph obtained from G by deleting the edge subset E' and is denoted by $G \setminus E'$.*

Note : When a vertex is deleted from G, all the edges incident to it are also deleted from G but whereas the deletion of an edge from G does not affect the vertices of G.

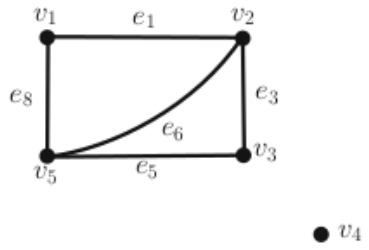
2.1.3 Degree of vertices

Definition 30. *Let G be a graph and $v \in V$. The number of edges incident at v in G is called the degree of the vertex v in G and is denoted by $d_G(v)$. A loop at v is to be counted*

twice in computing the degree of v. The minimum and the maximum of the degrees of the vertices of a graph G is denoted by $\delta(G)$ and $\delta(G)$ respectively. A graph G is called k-regular if every vertex of G has degree k. A graph is said to be regular if it is k-regular for some non negative integer k.

Definition 31. *A spanning 1-regular subgraph of G is called a 1-factor or a perfect matching of G. A vertex of degree 0 is an isolated vertex of G. A vertex of degree 1 is called a pendant vertex of G and the unique edge of G incident to such a vertex of G is a pendant edge of G.*

Example : Consider the following graph,



- $d_G(v_1) = 2$
- $d_G(v_2) = 3$
- $d_G(v_3) = 2$
- v_4 is an isolated vertex.
- $d_G(v_5) = 3$

Theorem 2.1.1. ***Euler :****The sum of the degrees of the vertices of a graph is equal to twice the number of its edges.*

Proof. Let the degrees of the vertices be $d_1, d_2, , , d_n$. Let $n = \sum_{i=1}^{n} d_i$ and m denote the total number of edges in the graph. Notice that each edge contributes 2 to the sum of degrees of the vertices, thus $n = 2m$. □

Corollary 2.1.1.1. *In any graph G the number of vertices of odd degree is even.*

Proposition 2.1.2. *Let V_1 and V_2 be the subsets of vertices of graph G with vertices of odd and even degrees respectively. Then from the previous theorem, we have*

$$\begin{aligned} 2m = n &= \sum_{v \in V} d_G(v) \\ &= \sum_{v \in V_1} d_G(v) + \sum_{v \in V_2} d_G(v) \end{aligned}$$

Both $2m$ and $\sum_{v \in V_2} d_G(v)$ are even, which forces $\sum_{v \in V_1} d_G(v)$ to be even.

Definition 32. *A sequence of non negative integers $d = (d_1, d_2, , , , , dn)$ is called graphical if there exists a simple graph whose degree sequence is d. A necessary condition for d to be a graphical is that $\sum d_1$ is even and $d_i \geq 0$ for $0 \leq i \leq n$. But a word of caution, these are not the sufficient conditions.*

Examples : The sequence $d = (1, 2, 2, 2, 3)$ is graphical but $d = (5, 4, 1, 1, 1)$ is not.

Note : For any sequence of non negative integers $d = (d_1, d_2, , , , , d_n)$ with $\sum d_1$ even, we can always construct a graph with d as its degree sequence. (we can always add loops to obtain the required degree for a particular vertex)

Theorem 2.1.3. *For a simple graph of n vertices, $(n \geq 2)$, there exists at least two vertices with same degree.*

Proof. Let G be a simple graphs with vertices $(v_1, v_2, , , , v_n)$. Suppose that the degree of all the vertices are non zero, then the possible degrees of the vertices are $\{1, 2, 3, , , , (n-1)\}$. There are n vertices and $n-1$ possible values for the degrees of the vertices, which implies that degrees of at least two vertices must me same. Now suppose there is a isolated vertex, then the possible degrees for the remaining vertices are $\{1, 2, 3, , , , n-2\}$ and we apply the same argument as before. □

Exercise:
1. Show that if G and H are isomorphic graphs, then each pair of corresponding vertices of G and H has the same degree.
2. Let $d_1, d_2, , , , , d_n$ be the degree sequence of a graph and r be any positive integer. Show that $\sum_{i=1}^{n} d_i^r$is even.
3. Let G be a graph with n vertices and m edges. Assume that each vertex of G is of degree either k or $k+1$. Show that the number of vertices of degree k in G is $(k+1)n - 2m$.

2.1.4 Path and connectedness

Definition 33. *A walk in a graph G is an alternating sequence $W : v_0e_1v1e2v2, , , , , e_nv_p$ of vertices and edges, beginning and ending with vertices in which v_{i-1} and v_i are the ends of e_i. $v0$ is the origin and v_p is the terminus of $W$and the walk W is said to join v_0 and v_p. If the G is simple, then a walk is determined by the sequence of its vertices. The walk is closed if $v_0 = v_p$ and otherwise it is open . A walk is called a trail if all the edges appearing in the walk are distinct and it is called a path if all the vertices are distinct. Thus, every path in G is automatically a trail in G. When writing a path, we usually omit the edges. A cycle is a closed trail in which the vertices are all distinct. A cycle is odd or even depending on whether its length is odd or even. The length of a walk is the number of edges in it. A walk of length 0 consists of just a single vertex.*

Definition 34. *A cycle of length k is denoted by C_k and P_k denotes a path on k vertices. In particular, $C3$ is often referred to as a triangle, C_4 as a square, and $C5$ as a pentagon. If $P = v0e1v1e2v2,,,,,,ekvk$ is a path, then the inverse of the path P, denoted as P^{-1} is defined as $P^{-1} = v_k e_k v_{k-1} ek-1 v_{k-2},,,,,v_1 e_1 v_0$ is also a path. The subsequence $v_i e_{i+1} v_{i+1},,,,,e_j v_j$ of P is called the $v_i - v_j$ section of P.*

Definition 35. ***Connectedness :*** *Let G be a graph. Two vertices u and v in G are said to be connected if there exits a $u-v$ path in G. The relation "connected" forms a equivalence relation on $V(G)$. Let $V_1, V_2,,,V_\omega$ be the equivalence classes, then $G[V_1], G[V_2],,,,,G[V_\omega]$ are called the connected components of G. If $\omega = 1$, the the graph G is said to be connected or else G is disconnected with ω connected components. Number of components of G is denoted by $\omega(G)$.*

Note : The connected components of G also are the maximal connected subgraphs of G.

Definition 36. *For vertices u and v in G, define a function d on graph G as follows:*
1. $d(u, v)$= length of the shortest $u - v$ path if u and v belongs to the same connected component of G.
2. $d(u, v) = \infty$, otherwise.
If G is a connected graph, then d forms a metric function on G.

Proposition 2.1.4. *If G is simple and $\delta \geq \frac{n-1}{2}$, then G is connected.*

Proof. Suppose that G is not connceted, then g has atleast two connected components G_1 and G_2. Let v be a vertex in G_1. since $\delta \geq \frac{n-1}{2}$, $d(v) \geq \frac{n-1}{2}$. Thus, G_1 has atleast $\frac{n-1}{2} + 1$ vertices. Similarly, G_2 has atleast $\frac{n-1}{2} + 1$ vertices. Therefore G has atleast $\frac{n-1}{2} + 1 + \frac{n-1}{2} + 1 = n + 1$ vertices, which is a contradiction. □

Note : In a group of six people, there must be three people who are mutually acquainted or three people who are mutually non acquainted.
This a interesting theorem in an area of mathematics called Ramsey theory and proof of the theorem only requires logic.

Theorem 2.1.5. *If a simple graph G is not connected, then G^c is connected.*

Proof. Take two vertices u and v in G. Suppose u and v belong to different components of G, then u and v will be adjacent in G^c. Therefore, u and v is connected. Suppose u and v belongs to the same component. Take a vertex w from a different component of G, then uw and vw are not edges of G, therefore, they are edges of G^c. Then uwv is a path in G^c, thus G^c is connected. □

Theorem 2.1.6. *The number of edges of a simple graph of order n having ω components cannot exceed .* $\frac{(n-\omega)(n-\omega+1)}{2}$.

Proof. Let $G_1, G_2, , , , G_\omega$ be the components of the simple graph G and $n_1, n_2, , , n_\omega$ be the vertices of the components G_i, $1 \leq i \leq \omega$ respectively. Then $m(G_i) \leq \frac{n_i(n_i-1)}{2}$, hence $m(G) \leq \sum_{i=1}^{\omega} \frac{n_i(n_i-1)}{2}$. Since $v_1 \geq 1$,for $1 \leq n_i$, we have $n_i \leq n - \omega + 1$. Therefore,

$$\begin{aligned} m(G) &\leq \sum_{i=1}^{\omega} \frac{n_i(n_i-1)}{2} \\ &\leq \sum_{i=1}^{\omega} \frac{(n-\omega+1)(n_i-1)}{2} \\ &\leq \frac{n-\omega-1}{2} \sum_{i=1}^{\omega} (n_i - 1) \\ &= \frac{(n-\omega+1)(n-\omega)}{2} \end{aligned}$$

□

Theorem 2.1.7. *A graph is bipartite if and only if it contains no odd cycle.*

Proof. Let G be a bipartite graph with the bipartition (X, Y). Let $v_1e_1v_2e_2....v_ke_kv_1$ be a cycle in G. Without loss of generality, assume that $v_1 \in X$. Then, since v_2 is adjacent to v_2, $v_2 \in Y$. Similarly, $v_3 \in X$, $v_4 \in Y$ and so on. Thus, $v_i \in X$ or $v_i \in Y$ depending on whether i is odd or even. Now, v_kv_1 is an edge in G and $v_k \in Y$, $v_1 \in X$. Therefore, k is even and the cycle is an even cycle.

Suppose G be a graph with no odd cycles. Assume that G is connected and let u be a vertex in G. Define $X = \{v \in V | d(u, v) iseven\}$ and $Y = \{v \in V | d(u.v) isodd\}$ ($u \in Xasd(u, u) = 0$). We will prove that (X, Y) forms a bipartition for G. We need to show there exits no edge between two vertices in X or Y. Let v, w be two vertices in X. Then $d(u, v)$ and $d(u, w)$ are even.Let P be a $u - v$ shortest path of length $d(u, v)$ and Q a $u - w$ shortest path of length $d(u, w)$. Let w' be a vertex common to the path P and Q such that the $w' - v$ section of P and the $w' - w$ section of Q dose not have any common vertex. Then the $u - w'$ section of P and Q are of same length. The $w' - v$ section of P and $w' - w$ section of Q are both odd or even. Now, if vw is an edge of G, then the $w' - v$ section of path P followed by the edge vw and $w - w'$ section of Q^{-1} forms a odd cycle in G, which is a contradiction to our hypothesis. Thus, no two vertices of X are adjacent. Similarly, no two vertices of Y are adjacent. Therefore, G is a bipartite graph with bipartition (X, Y).

If G is a not a connected graph, let $G_1, G_2, , , G_n$ be the components of G. Then by our hypothesis, no component of G contains an odd cycle. Then with the same argument as before, each component G_i is a bipartite graph and let the bipartition of each G_i be (X_i, Y_i). Define $X = \cup_{i=1}^{n} X_i$ and $Y = \cup_{i=1}^{n} Y_i$. Then G is a bipartite graph with bipartition (X, Y).

□

Exercise:

1. Prove that the function d defined in this section indeed form a metric on $V(G)$.

2. Show that in a group of six people there must be three people who are mutually acquainted or three people who are mutually nonacquainted.
3. Prove that a simple nontrivial graph G is connected if and only if for any partition of V into two nonempty subsets V_1 and V_2, there is an edge joining a vertex of V_1 to a vertex of V_2.
4. Prove that in a simple graph G, the union of two distinct paths joining two distinct vertices contains a cycle.
5. If a simple connected graph G is not complete, prove that there exist three vertices u, v, w of G such that uv and vw are edges of G, but uw is not an edge of G.

2.1.5 Automorphism of a Simple Graph

Definition 37. *An automorphism of a graph G is an isomorphism of G onto itself. Recall that two simple graphs G and H are isomorphic if and only if there exists a bijection $\phi : V(G) \to V(H)$ such that uv is an edge of G if and only if $\phi(u)\phi(v)$ is an edge of H.*

Theorem 2.1.8. *The set $Aut(G)$ of all automorphisms of a simple graph G is a group with respect to the composition $\circ$ of mappings as the group operation.*

Proof. i. Let ϕ_1 and ϕ_2 be bijections on $V(G)$ preserving adjacency and nonadjacency. Clearly, the mapping $\phi_1 \circ \phi_2$ is a bijection on $V(G)$. If u and v are adjacent in G, then $\phi_2(u)$ and $\phi_2(v)$ are adjacent in G. But $\phi_1 \circ \phi_2(u) = \phi_1(\phi_2(u))$ and $\phi_1 \circ \phi_2(v) = \phi_1(\phi_2(v))$. Hence, $\phi_1 \circ \phi_2(u)$ and $\phi_1 \circ \phi_2(v)$ are adjacent in G. That is, $\phi_1 \circ \phi_2$ preserves adjacency. A similar argument shows that $\phi_1 \circ \phi_2$ preserves nonadjacency. Thus, $\phi_1 \circ \phi_2$ is an automorphism of G.
ii. It is a well-known result that the composition of mappings of a set onto itself is associative.
iii. The identity mapping I of $V(G)$ onto itself is an automorphism of G and it satisfies the condition $\phi \circ I = I \circ \phi = \phi$ for every $\phi \in Aut(G)$. Hence, I is the identity element of $Aut(G)$.
iv. Finally, if ϕ is an automorphism of G, the inverse mapping ϕ^{-1} is also an automorphism of G. □

Theorem 2.1.9. *For any simple graph G, $Aut(G) = Aut(G^c)$.*

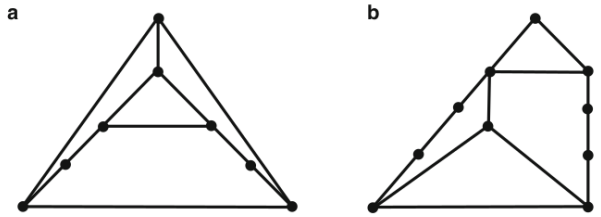
Proof. Since $V(G^c = V(G)$, every bijection on $V(G)$ is also a bijection on $V(G^c)$. As an automorphism of G preserves the adjacency and nonadjacency of vertices of G, it also preserves the adjacency and nonadjacency of vertices of G^c. Hence, every element of $Aut(G)$ is also an element of $Aut(G^c)$, and vice versa. □

Exercise:
1. Show that the automorphism group of K_n (or K_n^c) is isomorphic to the symmetric group

S_n of degree n.

2. Let G be a simple connected graph with n vertices such that $Aut(G) \cong S_n$. Show that G is the complete graph K_n.

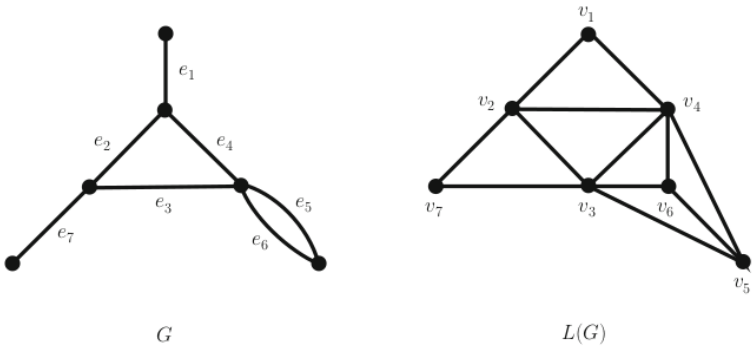
3. Find the automorphism groups of the following graphs



2.1.6 Line graph

Definition 38. *Let G be a graph without any loops. We construct a graph $L(G)$ in the following way:*
The vertex set of $L(G)$ is in one-to-one correspondence with the edge set of G and two vertices of $L(G)$ are joined by an edge if and only if the corresponding edges of G are adjacent in G: The graph $L(G)$ is called the line graph or the edge graph of G. A line graph is always simple graph.

Example :A graph G and its line graph $L(G)$



G $\qquad$ $L(G)$

Properties of line graph $L(G)$:

1. G is connected if and only if $L(G)$ is connected.

2. If H is a subgraph of G, then $L(H)$ is a subgraph of $L(G)$.

3. The edges incident at a vertex of G give rise to a maximal complete subgraph of $L(G)$.

4. If $e = uv$ is an edge of a simple graph G, the degree of e in $L(G)$ is the same as the number of edges of G adjacent to e in G. This number is $d_G(u) + d_G(v) - 2$. Hence, $d_{L(G)}(e) = d_G(u) + d_G(v) - 2$.

5. If G is a simple graph with the degree sequence $(d_1, d_2, , , , , d_n)$ and let $m(G) = m$, then

$$\begin{aligned}\sum_{e \in V(L(G))} d_{L(G)}(e) &= \sum_{uv \in E(G)} (d_G(u) + d_G(v) - 2) \\ &= (\sum_{u \in V(G)} d_G(u)^2) - 2m \\ &= (\sum_{i=1}^{n} d_i^2) - 2m\end{aligned}$$

Therefore, by Euler's theorem, we get

$$m(L(G)) = \frac{1}{2}(\sum_{i=1}^{n} d_i^2) - m$$

Theorem 2.1.10. *The line graph of a simple graph G is a path if and only if G is a path.*

Proof. Let G be a path P_n with n vertices, then $L(G)$ is clearly a path P_{n-1} with $n-1$ vertices. Conversely, let $L(G)$ be a path. Suppose G has a vertex v with degree greater then 2, then the edges incident on v forms a complete subgraph of $L(G)$ with three or more vertices. Thus G cannot have a vertex with degree more than 2, which implies G is a path or a cycle. But since $L(G)$ is not a cycle, it forces G to be a path. □

Theorem 2.1.11. *If the simple graphs G_1 and G_2 are isomorphic, then $L(G_1)$ and $L(G_2)$ are isomorphic.*

Proof. Let (ϕ, θ) be an isomorphism of graphs G_1 and G_2. θ is a bijection between $E(G_1)$ and $E(G_2)$ preserving the adjacent vertices and non adjacent vertices. Recall that $L(G)$ is always a simple graph. We will prove that θ is an isomorphism of $L(G_1)$ and $L(G_2)$. Let e_1 and e_2 be two adjacent vertices in $L(G_1)$ and let v be the vertex in G_{e_1} and e_2. Then, $\phi(v)$ is incident to the vertices $\theta(e_1)$ and $\theta(e_2)$. Therefore, e_1 and e_2 are adjacent vertices in $L(G_2)$.
Now, let $\theta(e_1)$ and $\theta(e_2)$ be adjacent vertices in $L(G_2)$. This means that they are adjacent edges in G_2 and hence there exists a vertex v' of G_2 incident to both $\theta(e_i)$ and $\theta(e_j)$ in G_2. Then $\phi^{-1}(v')$ is a vertex of $G1$ incident to both e_i and e_j , so that e_i and e_j are adjacent vertices of $L(G_1)$. Therefore, vertices e_1 and e_2 of $L(G_1)$ are adjacent if and only if $\theta(e_1)$ and $\theta(e_2)$ are adjacent in $L(G_2)$, which imlies θ is an isomorphism of $L(G_1)$ and $L(G_2)$. □

Theorem 2.1.12. *(H. Whitney) Let G and G' be simple connected graphs with isomorphic line graphs. Then G and G' are isomorphic unless one of them is $K_{1,3}$ and the other is K_3*

Proof. Refer textbook for proof. □

2.1.7 Operations on Graphs

Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two simple graphs.

Definition 39. *(Union of two graphs) The graph $G = (V, E)$, where $V = V_1 \cup V_2$ and $E = E_1 \cup E_2$ is called the union of G_1 and G_2 and is denoted by $G_1 \cup G_2$. When G_1 and G_2 are vertex disjoint, $G_1 \cup G_2$ is denoted by $G_1 + G_2$ and is called the sum of the graphs G_1 and G_2. The finite union of graphs is defined by means of associativity. In particular, if $G_1, G_2, , , , , , G_r$ are pairwise vertex-disjoint graphs, each of which is isomorphic to G. Then $G_1 + G_2 + ... + G_r$ is denoted by rG.*

Definition 40. *(Intersection of two graphs) If $V_1 \cap V_2 \neq \emptyset$, the graph $G = (V, E)$, where $V = V_1 \cap V_2$ and $E = E_1 \cap E_2$ is the intersection of G_1 and G_2 and is written as $G_1 \cap G_2$.*

Definition 41. *(Join of two graphs) Let G_1 and G_2 be two vertex-disjoint graphs. Then the join $G_1 \vee G_2$ of G_1 and G_2 is the supergraph of $G + G_2$ in which each vertex of G_1 is also adjacent to every vertex of G_2.*

From the definitions we get the following,
i. $n(G_1 \cup G_2) = n(G_1) + n(G_2) - n(G_1 \cap G2)$, $m(G_1 \cup G_2) = m(G_1) + m(G_2) - m(G_1 \cap G2)$
ii. $n(G_1 + G_2) = n(G_1) + n(G_2)$, $m(G_1 + G_2) = m(G_1) + m(G_2)$
iii. $n(G_1 \vee G_2) = n(G_1) + n(G_2)$, $m(G_1 \vee G_2) = m(G_1) + m(G_2) + n(G_1)n(G_2)$

2.2 Connectivity

The connectivity of a graph is a measure of connectedness of a graph. We can characterise some graphs as 'loosly connected' in the sense that deletion of a vertex or an edge destroys the connectedness of the graph. On the other hand we have complete graphs $k_n, n \geq 2$, which remain connected even after deletion of k, $1 \leq k \leq n - 1$. We can represent a communication network as a graph in which the the vertices represent the communication centres and the edges represent the communication channels. Needless to say, it is important to maintain a reliable communication network and the reliability of a communication network is depends directly on its connectivity.

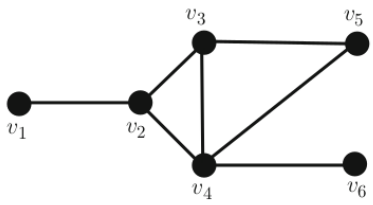
2.2.1 Vertex cuts and edge cuts

Definition 42. *A subset V' of the vertex set $V(G)$ of a connected graph G is a vertex cut of G if $G - V'$ is disconnected; it is a k-vertex cut if $|V'| = k$. V' is then called a separating set of vertices of G. A vertex v of G is a cut vertex of G if $\{v\}$ is a vertex cut of G.*

Let G be a nontrivial connected graph with vertex set $V(G)$ and let S be a nonempty subset of $V(G)$. For $\overline{S} = \{VG \setminus S \neq \emptyset\}$ and let $[S, \overline{S}]$ denote the set of all edges of G that have one end vertex in S and the other in $\overline{S}$.The set of edges of G of the form $[S, \overline{S}]$ is

called an edge cut of G. An edge e is a cut edge of G if $\{e\}$ is an edge cut of G. An edge cut of cardinality k is called a k-edge cut of G.

Example: Consider the graph



- vertex cuts: $\{v_2\}$ and $\{v_3, v_4\}$.
- cut vertex: $\{v_2\}$
- edge cuts: $\{v_3v_5, v_4v_5\}$, $\{v_1v_2\}$ and $v_4v_6\}$
- cut edge: $\{v_1v_2\}$ and $v_4v_6\}$

Theorem 2.2.1. *A vertex v of a connected graph G with at least three vertices is a cut vertex of G if and only if there exist vertices u and w of G distinct from v such that v is in every $u - w$ path in G.*

Proof. Suppose G be a connected graph and v a cut edge. Then $G - v$ will have at least two components G_1 and G_2. Take $u \in V(G_1)$ and $w \in V(g_2)$, then every $u - w$ path must contain v or otherwise both u and w will belong in the same component. Conversely, if there exist vertices u and w of G distinct from v such that v is in every $u - w$ path in G, then deletion of v will destroy every $u-w$ path in G. Thus, u and w will belong to different components in $G - v$. □

Theorem 2.2.2. *An edge $e = xy$ of a connected graph G is a cut edge of G if and only if e belongs to no cycle of G.*

Proof. Let e be the cut edge of G and let $[S, \overline{S}] = e$ be the partition defined by $G - e$. With out loss of generality, suppose $x \in S$ and $y \in \overline{S}$. If e belongs to a a cycle, then $[S, \overline{S}]$ will contain at least two edges, which is a contradiction to $[S, \overline{S}] = e$.

Conversely, suppose e is not a cut edge of G. Then $G - e$ is connected, and there exists a $x - y$ path, say P. Thus, $P \cup e$ is a cycle containing e. □

Theorem 2.2.3. *An edge $e = xy$ is a cut edge of a connected graph G if and only if there exist vertices u and v such that e belongs to every $u - v$ path in G.*

Proof. Straightforward. □

Theorem 2.2.4. *A connected graph G with at least two vertices contains at least two vertices that are not cut vertices.*

Proof. Suppose $n(G) \geq 3$. Take vertices u and v such that $d(u, v)$ is maximum. Then neither u or v can be a cut vertex. Suppose u be a cut vertex, Then $G - u$ is disconnected with at least two components. Then v will belong to one of these components and take w from a component not containing v. Then every $v - w$ path in G will contain u and thus $d(v, w) > d(v, w)$, which is a contradiction to our choice. Thus, u is not a cut vertex. Similarly, v is not a cut vertex. For $n(G) = 2$, K_2 is the spanning subgraph of G and so no vertex of G is a cut vertex.

□

Proposition 2.2.5. *A simple cubic (i.e., 3-regular) connected graph G has a cut vertex if and only if it has a cut edge.*

Proof. Let v_0 be a cut vertex of G and let v_1, v_2 and v_3 are the adjacent vertices of v_0 in G. Then $G - v_0$ has three or two components. If $G - v_0$ has three components then no two of v_1,v_2 and v_3 can belong to the same component. Thus v_0v_1,v_0v_2 and v_0v_3 and all cut edges of G. If $G - v_0$ has two edges, then of the vertex, say v_1 belongs to one component and other two, v_2 and v_3 belongs to the other. In this case v_0v_1 is the cut edge.

Conversely, let $e = uv$ be a cut edge of G. Then the deletion of vertex u causes the deletion of edge uv. Since G is cubic. $G - u$ is disconnected. Thus, u is cut vertex of G. □

Exercise:
1. If $\{x, y\}$ is a 2-edge cut of a graph G, show that every cycle of G that contains x must also contain y.
2. Show that in a graph, the number of edges common to a cycle and an edge cut is even.

2.2.2 Connectivity and edge connectivity

Definition 43. *For a nontrivial connected graph G having a pair of nonadjacent vertices, the minimum k for which there exists a k-vertex cut is called the vertex connectivity or simply the connectivity of G. It is denoted by $\kappa(G)$ or simply by κ when G is understood. A set of vertices and/or edges of a connected graph G is said to disconnect G if its deletion results in a disconnected graph.*

When a connected graph G ($n \geq$ 3 vertices) does not contain K_n as a spanning subgraph, κ is the connectivity of G if there exists a set of κ vertices of G whose deletion results in a disconnected subgraph of G while no set of $\kappa - 1$ (or fewer) vertices has this property.

Note:

1. If G is trivial or disconnected, then $\kappa(G)$ is taken as zero.
2. If G contains K_n as a spanning subgraph, $\kappa(G)$ is taken to be $n - 1$.

Definition 44. *The edge connectivity of a connected graph G is the smallest k for which there exists a k-edge cut. The edge connectivity of G is denoted by $\lambda(G)$. If λ is the edge connectivity of a connected graph G, then there exists a set of λ edges whose deletion results in a disconnected graph, and no subset of edges of G of size less than λ has this property.*

Definition 45. *A graph G is r-connected if $\kappa(G) \geq r$. Also, G is r-edge connected if $\kappa(G) \geq r$. An r-connected (respectively, r-edge-connected) graph is also l-connected (respectively, l-edge connected) for each $0 \leq l \leq r - 1$.*

Theorem 2.2.6. *For any loopless connected graph G, $\kappa(G) \leq \lambda(G) \leq \delta(G)$.*

Proof. Observe that $\kappa = 0$ if and only if $\lambda = 0$. Also, if $\delta = 0$, then κ and λ are both zero. Thus, we can assume that κ, λ and δ are all atleast 1. Suppose E is an edge cut of G. Let u and v be end vertices of an edge in E. For each edge in E that does not have both u and v, removes an end vertex different from u and v. Let there be t such vertices and at most t of the vertices have been removed. If the resulting graph, say H, is disconnected then $\kappa \leq t < \lambda$. If not, then there remains a setset of edges of E having u and v are the end vertices, removal of which from H will disconnect G. Thus, with an additional removal of u or v results in a disconnected graph or a trivial graph. In this process, we have removed $t + 1$ vertices to disconnect the graph and we get $\kappa \leq t + 1 \leq \lambda$.

If v is a vertex with $d_G(v) = \delta$, then the set $[\{v\}, V - \{v\}]$ of δ edges forms an edge cut of G. Thus, $\lambda \leq \delta$. □

Theorem 2.2.7. *The connectivity and edge connectivity of a simple cubic graph G are equal.*

Proof. It is enough to consider the case of connected cubic graph G. Since, we have$\kappa \leq \lambda \leq \delta = 3$, we need to only consider the cases when $\kappa = 1, 2$ or 3. It is left to the reader to work put the argument for each case. □

Definition 46. *A family of two or more paths in a graph G is said to be internally disjoint if no vertex of G is an internal vertex of more than one path of the family.*

Theorem 2.2.8. *(Whitney) A graph G with at least three vertices is 2-connected if and only if any two vertices of G are connected by at least two internally disjoint paths.*

Proof. Refer to the textbook. □

Remark: Edge form of Whitney's theorem: For a graph G with $n \geq 3$ is 2-edge connected if and only if any two distinct vertices of G are connected by at least two edge-disjoint paths in G.

Theorem 2.2.9. *A graph G with at least three vertices is 2-connected if and only if any two vertices of G lie on a common cycle.*

Proof. Let u and v be two vertices of a 2-connected graph G. then, by the above theorem, there exist two internally distinct path in G joining u and v. Union of there path gives a cycle containing u and v.

Conversely, if vertices u and v lie on a cycle C, then it is an union of two internally disjoint path. Therefore, by above theorem the graph is 2-connected. □

Remark: A graph is 2-connected if and only if for every pair of disjoint connected subgraphs G_1 and G_2, there exist two vertex disjoint paths P_1 and P_2 of G between $G1$ and G_2. This statement can be proved using arguments similar to that used to prove Whitney's theorem. We will be using this particular fact in proving the next theorem.

Theorem 2.2.10. *For a 2-connected graph G, any two longest cycles have at least two vertices in common.*

Proof. Let $C_1 = u_1u_2....u_ku_1$ and $C_2 = v_1v_2...v_kv_1$ be two longest cycle in G.
Case 1: Suppose C_1 and C_2 are disjoint. Then there exits two disjoint paths connecting C_1 and C_2, say P_1, connecting u_i and v_j and P_2, connecting u_l and v_p. Then, u_i and u_l divides C_1 into two subpath. Let L_1 be the longer subpath. Similarly, define L_2 for C_2. Then, $L_1 \cup P_1 \cup L_2 \cup P_2$ is a cycle in G larger then C_1 or C_2. Hence. C_1 and C_2 cannot be disjoint.
Case 2: □

Theorem 2.2.11. *A connected simple graph G is 3-edge connected if and only if every edge of G is the (exact) intersection of the edge sets of two cycles of G.*

Proof. Refer the textbook. □

Exercise:
1. Prove that a simple graph G with n vertices, $n \geq 2$, is complete if and only if $\kappa(G) = n - 1$.
2. Prove that the deletion of edges of a minimum-edge cut of a connected graph G results in a disconnected graph with exactly two components.
3. Determine $\kappa(K_n)$.
4. a. Show that a graph G with at least three vertices is 2-connected if and only if any vertex and any edge of G lie on a common cycle of G.

b. Show that a graph G with at least three vertices is 2-connected if and only if any two edges of G lie on a common cycle.

5. Prove that a graph is 2-connected if and only if for every pair of disjoint connected subgraphs G_1 and G_2, there exist two internally disjoint paths P_1 and P_2 of G between G_1 and G_2.

2.2.3 Trees

A graph without any cycles is called an acyclic graph or a forest. A tree is connected graph without any cycles. So each connected component of a forest is a tree. From definition it is clear that a forest (hence a tree) is a simple graph. A subgraph of a tree is a forest and connected subgraph of a tree T is called subtree of T.

Theorem 2.2.12. *A simple graph is a tree if and only if any two distinct vertices are connected by a unique path.*

Proof. Let T be a tree. Suppose two distinct vertices u and v are connected by two distinct $u - v$ paths in T. Then union of there two path gives a cycle, which is an contradiction. Hence, any two distinct vertices in T are connected by a unique path.

Conversely, suppose that any two distinct vertices in a graph G be connected by a unique path. Then, obviously G is connected. Also, G contains no cycle as every distinct vertices are connected by a unique path. Hence G is a tree. □

A spanning subgraph of a graph G, which is also a tree, is called a spanning tree of G. A tree T may have more than one spanning tree.

Theorem 2.2.13. *Every connected graph contains a spanning tree.*

Proof. Let G be a connected graph and $\mathcal{C}$ denote the set of all connected spanning subgraphs of G. $\mathcal{C}$ is not empty as $G \in \mathcal{C}$. Let $T \in \mathcal{C}$ be a spanning subgraph of G having the least of edges. Then, T must be a spanning tree of G. Suppose not, then T contain a cycle. Then, deletion of any edge of this cycles gives a spanning subgraph with less number of edges than T, which is a contradiction to our choice. Thus, T must be a spanning tree of G. □

Theorem 2.2.14. *The number of edges in a tree on n vertices is $n - 1$. Conversely, a connected graph on n vertices and $n - 1$ edges is a tree.*

Proof. We will prove this using induction (strong) on n to prove that $m = n-1$. For $n = 2$, the argument is straightforward.

Assume that the result holds true for all trees on $n - 1$ or fewer vertices, for $n \geq 3$. Let T be a tree on n vertices. Let $e = uv$ be an edge of T¿ then, e is an unique path connecting u and v. Thus, the deletion of e will disconnect T into two components, say T_1 and T_2. Now, notice that $n(T_1)$ and $n(T_2)$ are less than n. Therefore, by induction hypothesis,

$m(T_1) = n(T_1) - 1$ and $m(T_2) = n(T_2) - 1$. Thus, $m(T) = m(T_1) + m(T_2) + 1 = n(T_1) + n(T_2) - 1 = n(T) - 1$. Hence, the result hold true for T.

Conversely, let G be a connected graph with n vertices and $n - 1$ edges. then, by previous theorem, G has a spanning tree T. T has n vertices and being a tree hass $n - 1$ edges. Thus, $G = T$. □

Theorem 2.2.15. *A tree with at least two vertices contains at least two pendant vertices (i.e., end vertices or vertices of degree 1).*

Proof. Let P be the longest path of a tree T. Then, the end vertices of P must be a pendant vertices of T, otherwise at least one of the end vertices will have a second neighbour in P, which yields a cycle, a contradiction. □

Corollary 2.2.15.1. *If $\delta(G) \geq 2$, G contains a cycle.*

Theorem 2.2.16. *A connected graph G is a tree if and only if every edge of G is a cut edge of G.*

Proof. If G is a tree, then there are no cycles in G. Hence, no edge of G belongs to a cycle and therefore, each edge of G is a cut edge. Conversely, if every edge of G is a cut edge, then there is no cycle in G. Hence, G is a tree. □

Theorem 2.2.17. *A connected graph G with at least two vertices P is a tree if and only if its degree sequence $d_1, d_2, , , , , d_n$ satisfies the condition $\sum_{i=1}^{n} d_i = 2(n-1)$, with $d_i > 0$ for each i.*

Proof. Let G be a tree. Since G is connected and non trivial, every term of the degree sequence is non zero. Thus,

$$\sum_{i=1}^{n} d_i = 2m = 2(n-1)$$

Conversely, if the graph satisfies $\sum_{i=1}^{n} d_i = 2m = 2(n-1)$, that implies $m = n - 1$. Thus, we get that G is a tree. □

Lemma 2.2.18. *If u and v are nonadjacent vertices of a tree T , then $T + uv$ contains a unique cycle.*

Exercise:

1. Give an example of a graph with n vertices and $n - 1$ edges that is not a tree.
2. Show that a simple graph with ω components is a forest if and only if $m = n - \omega$.
3. Prove that every tree is a bipartite graph.

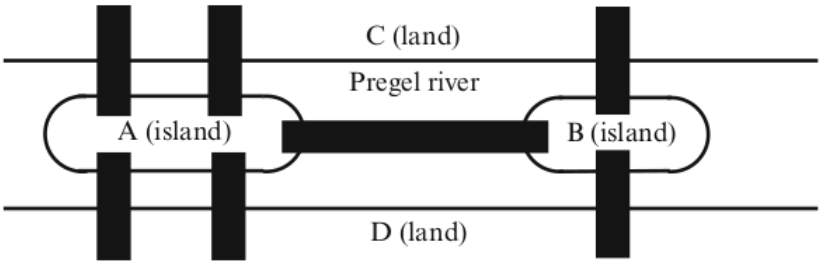
2.3 Eulerian Graphs

An Euler trail in a graph G is a spanning trail in G that contains all the edges of G. An Euler tour of G is a closed Euler trail of G. G is called Eulerian if G has an Euler tour. It was Euler who first considered these graphs, and hence their name. Notice that every Eulerian graph is connected.

Theorem 2.3.1. *For a nontrivial connected graph G, the following statements are equivalent:*
i. G is Eulerian.
ii. The degree of each vertex of G is an even positive integer.
iii. G is an edge-disjoint union of cycles.

Proof. i. $\implies$ ii. Let T be an Euler tour of G described from some vertex $v_0 \in V(G)$. For $v \in V(G)$ and $v \neq v_0$, each time T enters v, T has to leave v so that it can move back to v_0. Therefore, $d(v)$ is even. For v_0, every time T move out of v_0, it has to come back to v_0. Therefore, the degree of each vertex is even.
ii. $\implies$ iii. As $\delta(G) \geq 2$, G contains a cycle C_1. In $G \setminus E(C_1)$, remove the isolated vertices if there are any. Let the resulting subgraph of G be G_1. If G_1 is nonempty, each vertex of G_1 is again of even positive degree. Hence $\delta(G_1) \geq 2$, and so G_1 contains a cycle C_2. It follows that after a finite number, say r, of steps, $G \setminus E(C_1 \cup C_2....C_r)$ is totally disconnected. Then G is the edge- disjoint union of the cycles $C_1, C_2, , , , , C_r$.
iii. $\implies$ i. Assume that G is an edge-disjoint union of cycles. Since any cycle is Eulerian, G contains an Eulerian subgraph. Let G_1 be a longest closed trail in G. Then G_1 must be G. Suppose not, let $G_2 = G - E(G_1)$. Since G is an edge-disjoint union of cycles, every vertex of G is of even degree ≥ 2. Also, since G_1 is Eulerian, each vertex of G_1 is of even degree ≥ 2. Hence each vertex of G_2 is of even degree. Since G_2 is not totally disconnected and G is connected, G_2 contains a cycle C having a vertex v in common with G_1. Describe the Euler tour of G_1 starting and ending at v and follow it by C . Then $G \cup C$ is a closed trail in G longer than G_1 . This contradicts the choice of G_1 and so G_1 must be G. Hence G is Eulerian. □

Königsberg bridge problem: The city of Königsberg has seven bridges linking two islands A and B and the banks C and D of the Pregel, as shown in the figure below.



The problem was to start from any one of the four land areas, take a stroll across the seven bridges, and get back to the starting point without crossing any bridge a second time. This problem can be converted into one concerning the graph obtained by representing each land area by a vertex and each bridge by an edge. The Königsberg bridge problem will have a solution provided that this graph H is Eulerian. But this is not the case since it has vertices of odd degrees.

2.4 Planarity

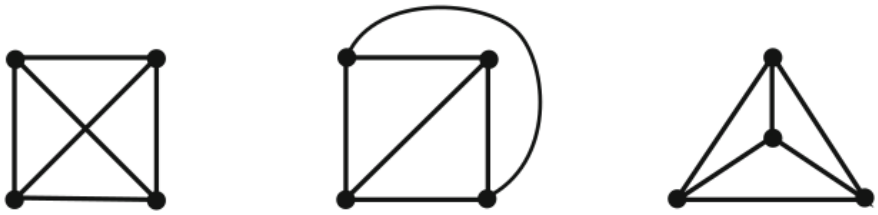
2.4.1 Planar and Nonplanar Graphs

In this section we will look into some basic results on planar graphs. Before that, we will do a quick revision on Jordan arc and the famous Jordan curve theorem. A Jordan arc in the plane is the image of an injective continuous map of a closed and bounded interval $[a, b]$ into the plane. In other words, the map has no "intersection points". A Jordan curve is a plane curve which is homeomorphic to a the unit circle, i.e., it is simple and closed. If J is any closed Jordan curve in the plane, the complement of J (with respect to the plane) is partitioned into two disjoint open connected subsets of the plane, one of which is bounded and the other unbounded. The bounded subset is called the interior of J and is denoted by $intJ$. The unbounded subset is called the exterior of J and is denoted by $extJ$. The Jordan curve theorem (of topology) states that if J is any closed Jordan curve in the plane, any arc joining a point of $intJ$ and a point of $extJ$ must intersect J at some point.

Definition 47. *A graph G is planar if there exists a drawing of G in the plane in which no two edges intersect in a point other than a vertex of G i.e each edge is a Jordan arc. Such a drawing of a planar graph G is called a plane representation of G. We also say that G has been embedded in the plane. A plane graph is a planar graph that has already been embedded in the plane.*

Example:

1. All trees
2. Al cycles
3. K_4



Planar graph K_4 Two plane embeddings of K_4

4. K_5 (try to work out the planar representation of K_5)

Let G be a plane graph. Then the union of the edges (as Jordan arcs) of a cycle C of G form a closed Jordan curve, which we also denote by C. A plane graph G divides the rest of the plane, say π into one or more face. We define an equivalence relation $\sim$ on π.

Definition 48. *For points A and B of π, we say that $A \sim B$ if and only if there exists a Jordan arc from A to B in π. Check that $\sim$ indeed forms an equivalence relation on π. The equivalence classes of $\sim$ are called the faces of G.*

Remark :
1. A connected graph is a tree if and only if it has only one face.
2. Any plane graph has exactly one unbounded face. (It is important to note that the distinction between bounded and unbounded face of plane graph G is unnecessary as there exists a plane representation of G such that any specified vertex or edge belongs to the unbounded face.)

Definition 49. *A graph G is embeddable on a sphere S if it can be drawn on the surface of S so that its edges intersect only at its vertices. Such a drawing, if it exists, is called an embedding of G on S. Embeddings on a sphere are called spherical embeddings.*

Before moving onto the next theorem, let us recollect stereographic projection of sphere S. Let S be a sphere resting on a plane P so that P is a tangent plane to S. Let N be the "north pole", the point on the sphere diametrically opposite the point of contact of S and P. Let the straight line joining N and a point s of $S \setminus N$ meet P at p. Then the mapping $\eta : S \setminus N \to P$ defined by $\eta(s) = p$ is called the stereographic projection of S from N.

Theorem 2.4.1. *A graph is planar if and only if it is embeddable on a sphere.*

Proof. Let a graph G be embeddable on a sphere and let G' be a spherical embedding of G. The image of G' under the stereographic projection η of the sphere from a point N of the sphere not on G' is a plane representation of G on P. Conversely, if G'' is a plane

embedding of G on a plane P, then the inverse of the stereographic projection of G'' on a sphere touching the plane P gives a spherical embedding of G. □

Theorem 2.4.2. *a. Let G be a plane graph and f be a face of G. Then there exists a plane embedding of G in which f is the exterior face.*
b. Let G be a planar graph. Then G can be embedded in the plane in such a way that any specified vertex (or edge) belongs to the unbounded face of the resulting plane graph.

Proof. a. Let n be a point of $intf$. Let $G' = \sigma(G)$ be a spherical embedding of G and let $N = \sigma(n)$. Let η be the stereographic projection of the sphere with N as the north pole. Then the map $\eta\sigma$ gives a plane embedding of G that maps f onto the exterior face of the plane representation $\eta\sigma$ of G.
b. Let f be a face containing the specified vertex (or edge) in a plane representation of G. Now, by part (a) of the theorem, there exists a plane embedding of G in which f becomes the exterior face. The specified vertex (or edge) then becomes a vertex (or edge) of the new unbounded face. □

Remark:

1. Let G be a connected plane graph. Each edge of G belongs to one or two faces of G. A cut edge of G belongs to exactly one face, and conversely, if an edge belongs to exactly one face of G, it must be a cut edge of G.
2. The union of the vertices and edges of G incident with a face f of G is called the boundary of f and is denoted by $b(f)$. The vertices and edges of a plane graph G belonging to the boundary of a face of G are said to be incident with that face. If G is connected, the boundary of each face is a closed walk in which each cut edge of G is traversed twice. When there are no cut edges, the boundary of each face of G is a closed trail in G. However, if G is a disconnected plane graph, then the edges and the vertices incident with the exterior face will not define a trail.
3. The number of edges incident with a face f is defined as the degree of f. In counting the degree of a face, a cut edge is counted twice. Thus, each edge of a plane graph G contributes two to the sum of the degrees of the faces. It follows that if $\mathcal{F}$ denotes the set of faces of a plane graph G, then $\sum_{f\in(F)} d(f) = 2m(G)$, where $d(f)$ denotes the degree of the face f.

Exercise:
1. Show that every graph with at most three cycles is planar.
2. Find a simple graph G with degree sequence $(4, 4, 3, 3, 3, 3)$ such that
a. G is planar.
b. G is nonplanar.

2.4.2 Euler Formula and Its Consequences

Theorem 2.4.3. *For a connected plane graph G, $n - m + \mathbf{f} = 2$, where n, m, and $\mathbf{f}$ denote the number of vertices, edges, and faces of G respectively.*

Proof. We prove by induction on $\mathbf{f}$. For, $\mathbf{f} = 1$, $m = n - 1$ and thus $n - m + \mathbf{f} = 2$. Suppose the statement holds for all plane graphs with $f = k - 1$, for $k \geq 2$. Suppose G be a plane graph with k faces. Since $k \geq 2$, G is not a tree. let C be a cycle in G and e and edge of C. Then, e belongs to exactly two faces, say f_1 and f_2. The deletion of e causes the formation of a single face from f_1 and f_2. Since e is not a cut edge, $G - e$ is connected with $k-1$ faces. By applying induction hypothesis, we get $n-(m-1)+k-1 = 2$, which gives $n - m + k = 2$. Hence proved. □

Corollary 2.4.3.1. *All plane embeddings of a given planar graph have the same number of faces.*

Corollary 2.4.3.2. *If G is a simple planar graph with at least three vertices, then $m \leq 3n - 6$.*

Proof. Without loss of generality, assume that G is a simple connected plane graph. Since G is simple and $n \geq 3$, each face of G has degree at least 3. Hence, if $\mathcal{F}$ denotes the set of faces of G, $\sum_{f\in\mathcal{F}} d(f) \geq 3\mathbf{f}$. But $\sum_{f\in\mathcal{F}} d(f) = 2m$. Thus, $2m \geq 3\mathbf{f}$, so that $\mathbf{f} \leq \frac{2m}{3}$. By the Euler formula, $m = n + \mathbf{f} - 2$. That implies, $m \leq n + \frac{2m}{3} - 2$, which gives $m \leq 3n - 6$. □

Corollary 2.4.3.3. *For any simple planar graph G, $\delta(G) \leq 5$.*

Proof. If $n \leq 6$, then $\Delta(G) \leq 5$. Hence $\delta(G) \leq \Delta(G) \leq 5$ proving the result for such graphs. Now, assume that $n \geq 7$. Then, by the above corollary, $m \leq 3n - 6$. Now, $\delta n \leq \sum_{v\in V(G)} d_G(v) = 2m \leq 2(3n - 6) = 6n - 12$. Hence, $n(\delta - 6) = -12$. Thus, $\delta - 6$ is negative, implying that $\delta \leq 5$. □

Recall that the girth of a graph G is the length of a shortest cycle in G.

Theorem 2.4.4. *If the girth k of a connected plane graph G is at least 3, then $m \leq \frac{k(n-2)}{k-2}$.*

Proof. Let $\mathcal{F}$ denote the set of faces and $\mathbf{f}$ denote the number of faces of G. If $f \in \mathcal{F}$, then $d(f) \geq k$. Since $2m = \sum_{f\in\mathcal{F}} d(f)$, we get $2m \geq \mathbf{f}k$. Then,$\mathbf{f} = 2 - n + m$. Hence, $2m \geq k(2 - n + m)$, implying that $m \leq \frac{k(n-2)}{k-2}$. □

Exercise:

1. Show that every simple bipartite cubic planar graph contains a C_4.
2. Let G be a simple plane cubic graph having eight faces. Determine $n(G)$.
3. Prove that a simple planar graph (with at least four vertices) has at least four vertices

each of degree 5 at most.
4. If G is a nonplanar graph, show that it has either five vertices of degree at least 4, or six vertices of degree at least 3.
5. Show that there is no 6-connected planar graph.

2.4.3 K_5 and $K_{3,3}$ are Nonplanar Graphs

Theorem 2.4.5. *K_5 is nonplanar.*

Proof. Assume the contrary that $K5$ is planar. Let v_1, v_2, v_3, v_4 and v_5 be the vertices of K_5 in a plane representation of K_5. The cycle $v_1v_2v_3v_4v_1$ (as a closed Jordan curve) divides the plane into two faces, namely, the interior and the exterior of C. The vertex v_5 must belong either to $intC$ or to $extC$. Suppose that v_5 belongs to $intC$. Draw the edges v_5v_1, v_5v_2 , v_5v_3, and v_5v_4 in $intC$. Now there remain two more edges v_1v_3 and v_2v_4 to be drawn. None of these can be drawn in $intC$, since it is assumed that K_5 is planar. Thus, v_1v_3 lies in $extC$. Then one of v_2 and v_4 belongs to the interior of the closed Jordan curve $C_1 = v_1v_5v_3v_1$ and the other to its exterior. Hence, v_2v_4 cannot be drawn without violating planarity. A similar proof holds if v_5 belongs to $extC$. Hence, K_5 is non planar. □

Alternate proof. If K_5 were planar, it follows that $10 \leq \frac{3(5-2)}{3-2}$, which is not true. Hence K_5 is nonplanar.

Theorem 2.4.6. *$K_{3,3}$ is nonplanar.*

Proof. Suppose that $K_{3,3}$ is planar. Let $U = \{u_1, u_2, u_3\}$ and $V = \{v_1, v_2, v_3\}$ be the bipartition of $K_{3,3}$ in a plane representation of the graph. Consider the cycle $C = u_1v_1u_2v_2u_3v_3u_1$. Since the graph is assumed to be planar, the edge u_1v_2 must lie either in the interior of C or in its exterior. For the sake of definiteness, assume that it lies in $intC$. Two more edges remain to be drawn, namely, u_2v_3 and u_3v_1. None of these can be drawn in $intC$ without crossing the edge u_1v_2. Hence, both of them are to be drawn in $extC$. Now draw u_2v_3 in $extC$. Then one of v_1 and u_3 belongs to the interior of the closed Jordan curve $C_1 = u_1v_2u_2v_3u_1$ and the other to the exterior of C_1. Hence, the edge v_1u_3 cannot be drawn without violating planarity. A similar proof holds if one assumes that the edge u_1v_2 lies in $extC$. This shows that $K_{3,3}$ is nonplanar. □

Alternate proof. Suppose $K_{3,3}$ is planar. Let $\mathbf{f}$ be the number of faces of $G =_{3,3}$ in a plane embedding of G and $\mathcal{F}$, the set of faces of G. As the girth of $K_{3,3}$ is 4, we have $m = \frac{1}{2}\sum_{f\in\mathcal{F}} d(f) \geq \frac{4\mathbf{f}}{2} = 2\mathbf{f}$. We have, $n - m + \mathbf{f} = 2$. For $K_{3,3}$, $n = 6$ and $m = 9$, $\mathbf{f} = 5$. Thus, $9 \geq 10$, a contradiction.

Remark: K_5 and $K_{3,3}$ have some features in common.

1. Both are regular graphs.
2. The removal of a vertex or an edge from each graph results in a planar graph.
3. Contraction of an edge results in a planar graph.
4. K_5 is a nonplanar graph with the smallest number of vertices, whereas $K_{3,3}$ is a nonplanar graph with the smallest number of edges. Hence, any nonplanar graph must have at least five vertices and nine edges.

2.4.4 Dual of a Plane Graph

Let G be a plane graph. One can form out of G a new graph H in the following way. Corresponding to each face f of G take a vertex f^* and corresponding to each edge e of G, take an edge e^*. Then edge e^* joins vertices f^* and g^* in H if and only if edge e is common to the boundaries of faces f and g in G. The graph H is then called the dual. If e is a cut edge of G embedded in face f of G, then e^* is a loop at f^*. H is a planar graph and there exists a natural way of embedding H in the plane. Vertex f^* corresponding to face f is placed in face f of G. Edge e^* joining f^* and g^*, is drawn so that e^* crosses e once and only once and crosses no other edge. This embedding is the canonical embedding of H. H with this canonical embedding is denoted by G^*. Any two embeddings of H, as described above, are isomorphic.

Module 3

Introduction to the theory of Computation

(Reference text: 'An Introduction to Formal Languages and Automata' by Peter Linz)

Formal languages, automata, computability, and other related topics form a major part of what is known as the theory of computation. The study of the theory of computation has several purposes, most importantly (i) to familiarize students with the foundations and principles of computer science, (ii) to teach material that is useful in subsequent courses, and (iii) to strengthen students' ability to carry out formal and rigorous mathematical arguments.

3.1 Three Basic Concepts

3.1.1 Language

Informally speaking, language is a system suitable for the expression of certain ideas, facts, or concepts, including a set of symbols and rules for their manipulation. We will now try to give a precise definition for language.

Alphabet are defined as a finite, nonempty set Σ of symbols. From the individual symbols we construct strings, which are finite sequences of symbols from the alphabet. For example, if the alphabet $\Sigma = \{a, b\}$, then $abab$ and $aaabbba$ are strings on Σ. With few exceptions, we will use lowercase letters $a, b, c, ...$ for elements of Σ and $u, v, w, ...$ for string names. For example,

$$w = abaaa$$

indicates that the string named w has the specific value $abaaa$.

The concatenation of two strings w and v is the string obtained by appending the sym-

bols of v to the right end of w, that is, if

$$w = abaaa$$

and

$$v = baa$$

then the concatenation of w and v, denoted by wv, is

$$abaaabaa$$

The reverse of a string is obtained by writing the symbols in reverse order, that is if $w = abaaa$, then its reverse w^R is $aaaba$. The length of a string w, denoted by $|w|$, is the number of symbols in the string. Empty string is a string with no symbols at all and is be denoted by λ. The following simple relations hold for all w.

$$|\lambda| = 0$$

$$w\lambda = \lambda w = w$$

Any string of consecutive symbols in some w is said to be a substring of w. If $w = vu$, then the substrings v and u are said to be a prefix and a suffix of w, respectively. For example, if $w = abbab$, then $\{\lambda, a, ab, abb, abba, abbab\}$ is the set of all prefixes of w, while bab, ab, b are some of its suffixes.

Proposition 3.1.1. *For any string u and v,*

$$|uv| = |u| + |v|$$

Proof. We need to give a formal definition for length of a string before moving on with the proof. For all $a \in \Sigma$ and any string w on Σ, we define

$$|a| = 1$$

$$|wa| = |u| + 1$$

By definition, the proposition holds for all u of any length and all v of length 1.We now proceed by using induction on v. Assume that the proposition holds for all u of any length and all v of length $1, 2, ..., n$. Now take any v of length $n + 1$ and write it as $v = wa$. Then,

$$|v| = |w| + 1$$

$$|uv| = |uwa| = |uw| + 1$$

Then, by induction hypothesis,

$$|uw| = |u| + |w|$$

Thus, $|uv| = |uw| + 1 = |u| + |w| + 1 = |u| + |v|$. Hence proved.

□

If w is a string, then w^n stands for the string obtained by repeating n times. As a special case, we define $w^0 = \lambda$ for all w. If Σ is an alphabet, then we use Σ^* to denote the set of strings obtained by concatenating zero or more symbols from Σ. The set Σ^* always contains λ. To exclude the empty string, we define $\Sigma^+ = \Sigma^* \setminus \lambda$. While Σ is finite by assumption, Σ^* and Σ^+ are always infinite since there is no limit on the length of the strings in these sets. A language is defined very generally as a subset of Σ^*. A string in a language L will be called a sentence of L. This definition is quite broad, any set of strings on an alphabet Σ can be considered a language.

Since languages are sets, we can define the union, intersection, and difference of two languages. The complement of a language is defined with respect to Σ^*. The complement of L is

$$\overline{L} = \Sigma^* \setminus L$$

The reverse of a language is the set of all string reversals,

$$L^R = \{w^R | w \in L\}$$

The concatenation of two languages L_1 and L is the set of all strings obtained by concatenating any element of L_1 with any element of L_2,

$$L_1 L_2 = \{w_1 w_2 | w_1 \in L_1, w_2 \in L_2\}$$

For any language L, L^n is defined as L concatenated with itself n times and $L^0 = \lambda$.

We define the star-closure of a language L as $L^* = L_0 \cup L_1 \cup L_2 \cup ....$ and the positive closure as $L^+ = L_1 \cup L_2 \cup ....$

3.1.2 Grammar

To study languages mathematically, we need a mechanism to describe them. Hence, we introduce a common and powerful notion of a grammar.

Definition 50. *A grammar G is defined as a quadruple $G = (V, T, S, P)$, where V is a finite set of objects called variables,*
T is a finite set of objects called terminal symbols,
$S \in V$ is a special symbol called the start variable,
P is a finite set of productions.

It will be assumed without further mention that the sets V and T are nonempty and disjoint.

The production rules forms the heart of a grammar. They specify how the grammar transforms one string into another and through this they define a language associated with the grammar. In our discussion we will assume that all production rules are of the form

$$x \to y$$

where x is an element of $(V \cup T)^+$ and y is in $(V \cup T)^*$. Given a string w of the form $w = uxv$, we say the production $x \to y$ is applicable to this string, and we apply the production by to replacing x with y, thereby obtaining a new string $z = uyv$. This is denote by,

$$w \implies z$$

We say that w derives z or that z is derived from w. Successive strings are derived by applying the productions of the grammar in arbitrary order. A production can be used whenever it is applicable, and it can be applied as often as desired. If

$$w_1 \implies w_2 \implies w_3 \implies ..... \implies w_n$$

we say that w_1 derives w_n and denote it by $w_1 \overset{*}{\Rightarrow} w_n$, where $*$ indicates that an unspecified number of steps can be taken to derive w_n from w_1 . By applying the production rules in a different order, a given grammar can normally generate many strings. The set of all such terminal strings is the language defined or generated by the grammar.

Definition 51. *Let $G = (V, T, S, P)$ be a grammar. Then the set*

$$L(G) = \{w \in T^* | S \overset{*}{\Rightarrow} w\}$$

is the language generated by G.

If $w \in L(G)$, then the sequence

$$S \implies w_1 \implies w_2 \implies ..... \implies w_n \implies w$$

is a derivation of the sentence w. The strings $S, w_1, w_2, ..., w_n$, which contain variables as well as terminals, are called sentential forms of the derivation.

Example. Consider the grammar

$$G = \{\{S\}, \{a, b\}, S, P\}$$

with P given by

$$S \to aSb$$

$$s \to \lambda$$

Then, $S \implies aSb \implies aaSbb \implies aabb$. Thus, the string $aabb$ is a sentence in the language generated by the language and $aaSbb$ is a sentential form. We conjecture that $L(G) = \{a^n b^n | n \geq 0\}$. To prove this, we will first show that every sentential must be of the form

$$w_i = a^i S b^i \qquad (*)$$

Suppose $(*)$ holds true for all sentential of length less than or equal to $2i + 1$. Then, to obtain another sentential form, we have to apply the production $S \to aSb$.

$$a^i S b^i \implies a^{i+1} S b^{i+1}$$

Thus, every sentential of length of length $2i + 3$ is of the form $(*)$. $i = 1$ also satisfies $(*)$. Thus, by induction $(*)$ holds true. Finnaly, to obtain a sentence, we must apply the production $S \to \lambda$, giving us

$$S \stackrel{*}{\Rightarrow} a^n S b^n \implies a^n b^n$$

Thus, G can derive only strings of the form $a^n b^n$.

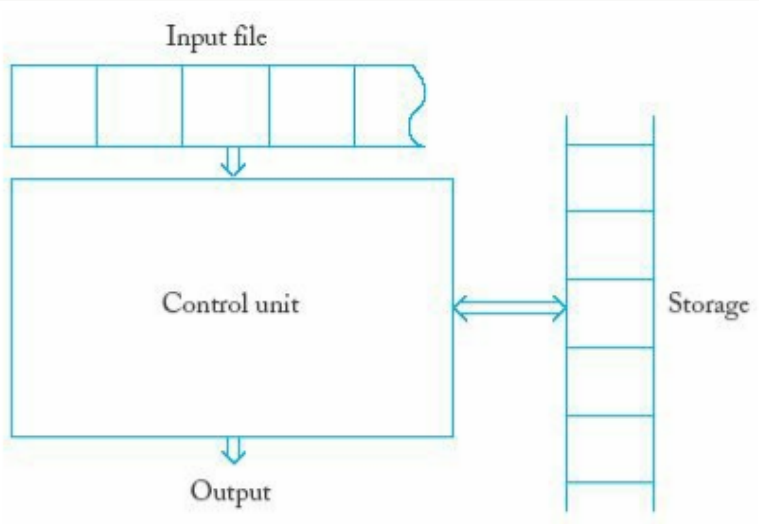
Definition 52. *We say that two grammars G_1 and G_2 are equivalent if they generate the same language, that is, if*

$$L(G_1) = L(G_2)$$

3.1.3 Automata

An automaton is an abstract model of a digital computer. Every automaton includes some essential features. It has a mechanism for reading input. It will be assumed that the input is a string over a given alphabet, written on an input file, which the automaton can read but not change. The input file is divided into cells, each of which can hold one symbol. The input mechanism can read the input file from left to right, one symbol at a time. The input mechanism can also detect the end of the input string. The automaton can produce output of some form. It may have a temporary storage device, consisting of an unlimited number of cells, each capable of holding a single symbol from an alphabet. The automaton can read and change the contents of the storage cells. Finally, the automaton has a control unit, which can be in any one of a finite number of internal states, and which can change state in

some defined manner.



An automaton is assumed to operate in a discrete timeframe. At any given time, the control unit is in some internal state, and the input mechanism is scanning a particular symbol on the input file. The internal state of the control unit at the next time step is determined by the next-state or transition function. This transition function gives the next state in terms of the current state, the current input symbol, and the information currently in the temporary storage. During the transition from one time interval to the next, output may be produced or the information in the temporary storage changed. The term configuration will be used to refer to a particular state of the control unit, input file, and temporary storage. The transition of the automaton from one configuration to the next will be called a move.

A deterministic automaton is one in which each move is uniquely determined by the current configuration. If we know the internal state, the input, and the contents of the temporary storage, we can predict the future behavior of the automaton exactly. In a nondeterministic automaton, this is not so. At each point, a nondeterministic automaton may have several possible moves, so we can only predict a set of possible actions. The relation between deterministic and nondeterministic automata of various types will play a significant role in our study.

An automaton whose output response is limited to a simple “yes” or “no” is called an accepter. Presented with an input string, an accepter either accepts the string or rejects it. A more general automaton, capable of producing strings of symbols as output, is called a transducer.

3.2 Applications

Although we stress the abstract and mathematical nature of formal languages and automata, it turns out that these concepts have widespread applications in computer science and are, in

fact, a common theme that connects many specialty areas. Formal languages and grammars are used widely in connection with programming languages.

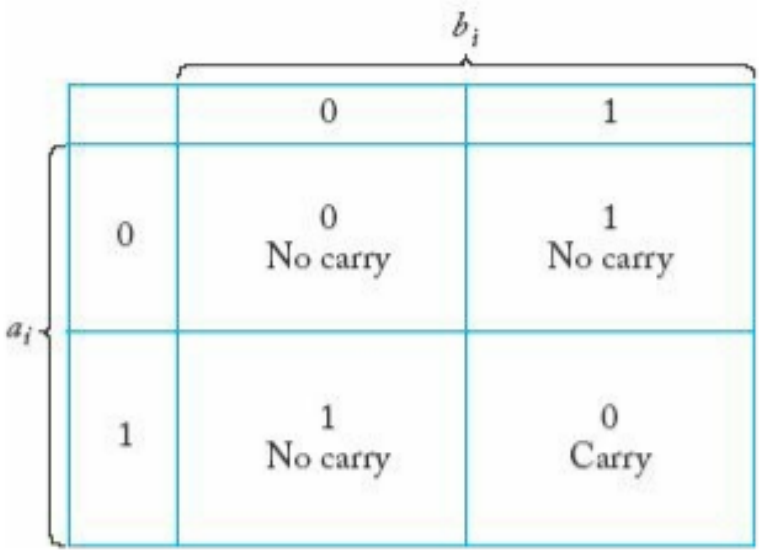
Example. A binary adder is an integral part of any general-purpose computer. Such an adder takes two bit strings representing numbers and produces their sum as output. For simplicity, let us assume that we are dealing only with positive integers and that we use a representation in which

$$x = a_0 a_1 a_3 ..... a_n$$

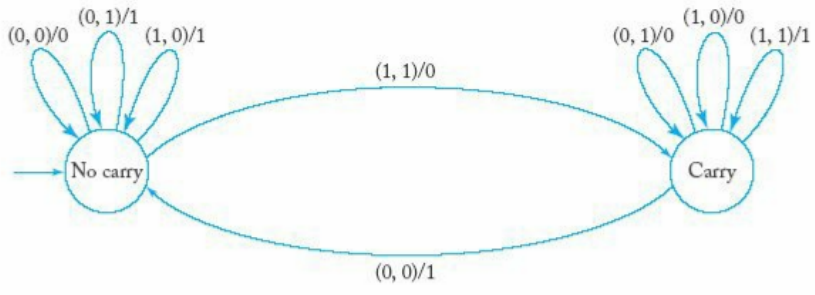
stands for the integer

$$v(x) = \sum_{i=0}^{n} a_i 2^i$$

A serial adder processes two such numbers $x = a_0 a_1 .... a_n$, and $y = b_0 b_1 .... b_n$, bit by bit, starting at the left end. Each bit addition creates a digit for the sum as well as a carry digit for the next higher position. The block diagram given below summarises the process.



The block diagram describes what an adder does, but explains little about its internal workings. An automaton (a transducer in this case) can make this much more explicit. The input to the transducer are the bit pairs (a_i, b_i), the output will be the sum bit d_i . Again, we represent the automaton by a graph now labeling the edges $(a_i, b_j)/d_i$. The carry from one step to the next is remembered by the automaton via two internal states labeled "carry" and "no carry." Initially, the transducer will be in state "no carry." It will remain in this state until a bit pair $(1, 1)$ is encountered. This will generate a carry that takes the automaton into the "carry" state. The presence of a carry is then taken into account when the next bit pair is read. A complete picture of a serial adder is given by the diagram below.



As this example indicates, the automaton serves as a bridge between the very high-level, functional description of a circuit and its logical implementation through transistors, gates, and flip- flops. The automaton clearly shows the decision logic, yet it is formal enough to lend itself to precise mathematical manipulation. For this reason, digital design methods rely heavily on concepts from automata theory.

3.3 Finite Automata

We have seen a brief introduction to automata. Now, we will explore further on, give formal definitions and develop rigorous results. We begin with finite acceptors. This type of automaton is characterized by having no temporary storage. Since an input file cannot be rewritten, a finite automaton is severely limited in its capacity to "remember" things during the computation. A finite amount of information can be retained in the control unit by placing the unit into a specific state. But since the number of such states is finite, a finite automaton can only deal with situations in which the information to be stored at any time is strictly bounded.

3.3.1 Deterministic Finite Accepters

Definition 53. *A deterministic finite accepter or dfa is defined by the quintuple*

$$M = (\mathcal{Q}, \Sigma, \delta, q_0, F)$$

, where $\mathcal{Q}$ is a finite set of internal states,
Σ is a finite set of symbols called the input alphabet,
$\delta : \mathcal{Q} \times \Sigma \to \mathcal{Q}$ is a total function called the transition function,
$q_0 \in \mathcal{Q}$ is the initial state,
$F \subseteq \mathcal{Q}$ is a set of final states.

Working : At the initial time, it is assumed to be in the initial state q_0 , with its input mechanism on the leftmost symbol of the input string. During each move of the automaton,

the input mechanism advances one position to the right, so each move consumes one input symbol. When the end of the string is reached, the string is accepted if the automaton is in one of its final states. Otherwise the string is rejected. The input mechanism can move only from left to right and reads exactly one symbol on each step. The transitions from one internal state to another are governed by the transition function δ. For example, if $\delta(q_0, a) = q_1$, then if the dfa is in state q_0 and the current input symbol is a, the dfa will go into state q_1.

A transition graph helps us to visualize and represent a finite automata, thereby giving a more intuitive picture. In a transition graph, the vertices represent states and the edges represent transitions. The labels on the vertices are the names of the states, while the labels on the edges are the current values of the input symbol. For example, if q_0 and q_1 are internal states of some dfa M, then the graph associated with M will have one vertex labeled q_0 and another labeled q_1 . An edge (q_0, q_1) labeled a represents the transition $\delta(q_0, a) = q_1$. The initial state will be identified by an incoming unlabeled arrow not originating at any vertex. Final states are drawn with a double circle.

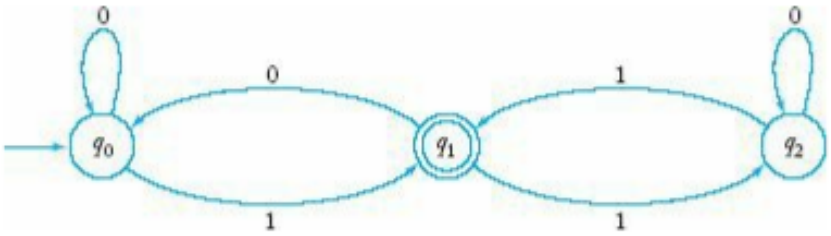
Example. Given a dfa

$$M = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_1\}),$$

where δ is given by

$\delta(q_0, 0) = q_0$ $\quad$ $\delta(q_0, 1) = q_1$
$\delta(q_1, 0) = q_0$ $\quad$ $\delta(q_1, 1) = q_2$
$\delta(q_2, 0) = q_2$ $\quad$ $\delta(q_2, 1) = q_1$

Then the graph given below represents the dfa.



This dfa accepts the string 01. Starting in state q_0 , the symbol 0 is read first. Looking at the edges of the graph, we see that the automaton remains in state q_0 . Next, the 1 is read and the automaton goes into state q_1 . We are now at the end of the string and, at the same time, in a final state q_1 . Therefore, the string 01 is accepted. The dfa does not accept the string 00, since after reading two consecutive 0's, it will be in state q_0 . By similar reasoning, we

see that the automaton will accept the strings $101, 0111$, and 11001, but not 100 or 1100.

We will give a quick introduction to the extended transition function δ^*.

$\delta^* : \mathcal{Q} \times \Sigma \to \mathcal{Q}$ and its value gives the state the automaton will be in after reading that string. For example, if $\delta(q_0, a) = q_1$ and $\delta(q_1, b) = q_2$, then $\delta^*(q_0, ab) = q_2$.

We have already seen the precise definition of an accepter. Let us now define an associated language.

Definition 54. *The language accepted by a dfa $M = (\mathcal{Q}, \Sigma, \delta, q_0, F)$ is the set of all strings on Σ accepted by M. In formal notation,*

$$M = \{w \in \Sigma^* | \Sigma^*(q_0, w) \in F\}$$

Example. Consider the automation given below. The automaton remains in its initial state q_0 until the first b is encountered. If this is also the last symbol of the input, then the string is accepted. If not, the dfa goes into state q_2 , from which it can never escape. The state q_2 is a trap state. We see clearly from the graph that the automaton accepts all strings consisting of an arbitrary number of a's, followed by a single b. All other input strings are rejected. In set notation, the language accepted by the automaton is

$$L = \{a^n b : n \geq 0\}$$

Theorem 3.3.1. *Let $M = (\mathcal{Q}, \Sigma, \delta, q_0, F)$ be a deterministic finite accepter, and let G_M be its associated transition graph. Then for every q_i , $q_j \in \mathcal{Q}$, and $w \in \Sigma^+$, $\Sigma^*(q_i, w) = q_j$ if and only if there is in G_M a walk with label w from q_i to q_j .*

Proof. We will use induction on length of w to prove the theorem. The result obviously holds for $|w| = 1$. Assume the result holds for all strings w with length $|w| \leq n$. Consider any string w of length $n + 1$. Then, w can be written as

$$w = va$$

, where v is a string of length n and $a \in \Sigma$. Suppose that $\delta^*(q_i, v) = q_k$. Then, by induction

hypothesis there must be a walk in G_M labeled v from q_i to q_k . But if $\Sigma^*(q_i, w) = q_j$, then M must have a transition $\delta(q_k, a) = q_j$, so that by construction G_M has an edge (q_k, qj) with label a. Thus, there is a walk in G_M labeled $va = w$ between q_i and q_j. Hence proved. □

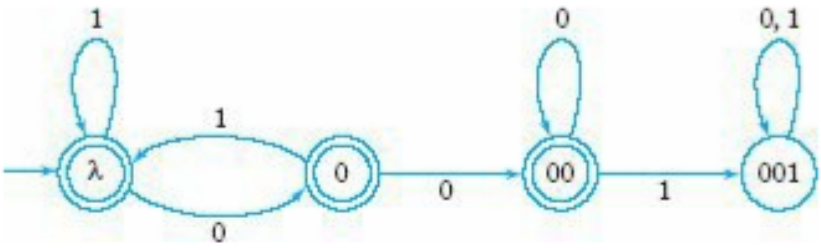
Example. Find a dfa that accepts all the strings on $\{0, 1\}$, except those containing the substring 001.

To work out a solution, we need to keep track of the current and preceding states of the automation. We can keep track of this by putting the automaton into specific states and labeling them accordingly. For example, the state in which two 0's were the immediately preceding symbols can be labeled simply 00.

If the string starts with 001, then it must be rejected. This implies that there must be a path labeled 001 from the initial state to a nonfinal state. For convenience, this nonfinal state is labeled 001. This state must be a trap state, because later symbols do not matter. All other states are accepting states. We also need to reject strings with 001 occurring as substring. We must define $\mathcal{Q}$ and δ so that whatever we need to make the correct decision is remembered by the automaton. In this case, when a symbol is read, we need to know some part of the string to the left, for example, whether or not the two previous symbols were 00. If we label the states with the relevant symbols, it is very easy to see what the transitions must be. For example,

$$\delta(00, 0) = 00$$

because this situation arises only if there are three consecutive 0's. We are only interested in the last two, a fact we remember by keeping the dfa in the state 00. The solution is as given below.



Every finite automaton accepts some language. If we consider all possible finite automata, we get a set of languages associated with them. We will call such a set of languages a **family**.

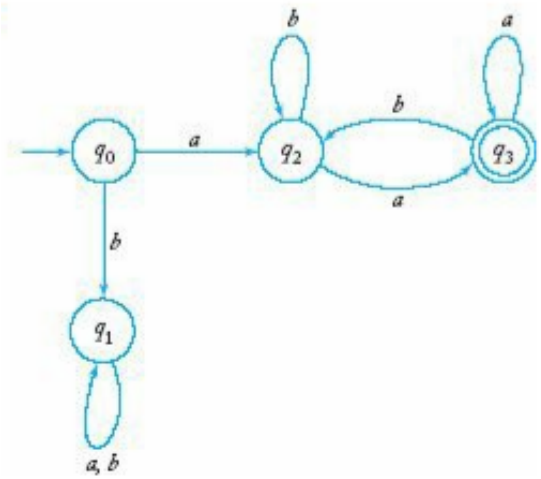
Definition 55. *(Regular language) A language L is called regular if and only if there exists some deterministic finite accepter M such that*

$$L = L(M)$$

Example. Show that the language L, defined as

$$L = \{awa : w \in \{a, b\}^*\}$$

is regular.
To show that this or any other language is regular, all we have to do is find a dfa for it. The dfa must check whether a string begins and ends with an a, what is between is immaterial. The solution is complicated by the fact that there is no explicit way of testing the end of the string. This difficulty is overcome by simply putting the dfa into a final state whenever the second a is encountered. If this is not the end of the string, and another b is found, it will take the dfa out of the final state. Scanning continues in this way, each a taking the automaton back to its final state. The final solution is given below.



3.3.2 Nondeterministic Finite Accepter

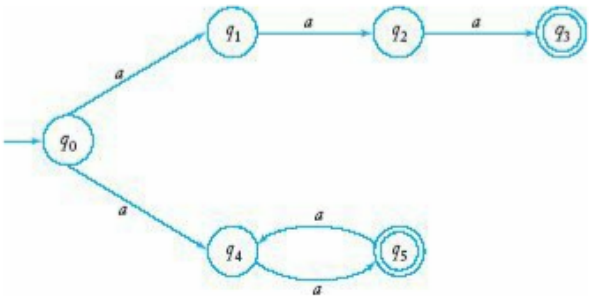
Definition 56. *A nondeterministic finite accepter or nfa is defined by the quintuple $M = (\mathcal{Q}, \Sigma, \delta, q_0, F)$,where $\mathcal{Q}, \Sigma, q_0, F$ are defined as for deterministic finite accepters, but*

$$\delta : \mathcal{Q} \times (\Sigma \cup \lambda) \rightarrow 2^{\mathcal{Q}}$$

Unlike determininstic accepter, in a nondeterministic accepter, the range of δ is in the powerset $2^{\mathcal{Q}}$, so that its value is not a single element of $\mathcal{Q}$ but a subset of it. This subset defines the set of all possible states that can be reached by the transition. Also, we allow λ as the second argument of δ. This means that the nfa can make a transition without consuming an input symbol. Although we still assume that the input mechanism can only travel to the right, it is possible that it is stationary on some moves. Finally, in an nfa, the set $\delta(q_i, a)$ may be empty, meaning that there is no transition defined for this specific situation.

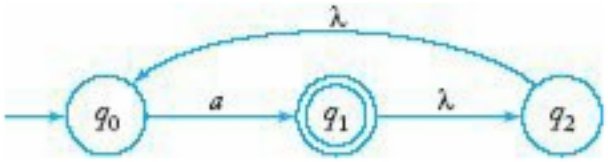
Like dfa's, nondeterministic accepters can be represented by transition graphs. The vertices are determined by $\mathcal{Q}$, while an edge (q_i, q_j) with label a is in the graph if and only if $\delta(q_i, a)$ contains q_j. Note that since a may be the empty string, there can be some edges labeled λ. A string is accepted by an nfa if there is some sequence of possible moves that will put the machine in a final state at the end of the string.

Example.



Definition 57. *For an nfa, the extended transition function is defined so that $\Sigma^*(q_i, w)$ contains q_j if and only if there is a walk in the transition graph from q_i to q_j labeled w. This holds for all q_i , $q_j \in \mathcal{Q}$, and $w \in \Sigma^*$.*

Example . Given an ufa,



Find $\delta^*(q_1, a)$ and $\delta^*(q_2, \lambda)$
There is a walk labeled a involving two λ-transitions from q_1 to itself. By using some of the λ-edges twice, we see that there are also walks involving λ-transitions to q_0 and q_2. Thus,

$$\delta^*(q_1, a) = \{q_0, q_1, q_3\}$$

Since there is a λ-edge between q_2 and q_0 , we have immediately that $\Sigma^*(q_2, \lambda)$ contains q_0 . Also, any state can be reached from itself by making no move. Thus,

$$\delta^*(q_2, \lambda) = \{q_0, q_2\}$$

Definition 58. *The language L accepted by an nfa $M = (\mathcal{Q}, \Sigma, \delta, q_0, F)$ is defined as*

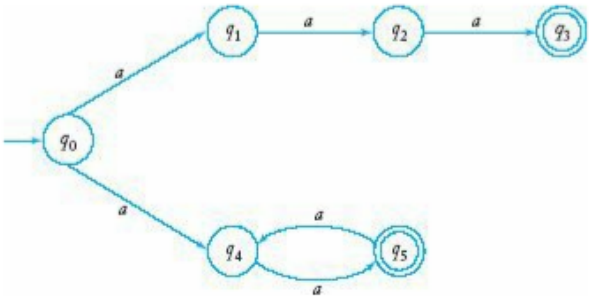
$$L(M) = \{w \in \Sigma^* : \delta^*(q_0, w) \cap F \neq \emptyset\}$$

In words, the language consists of all strings w for which there is a walk labeled w from the initial vertex of the transition graph to some final vertex.

Why Nondeterminism?

- Many deterministic algorithms require that one make a choice at some stage. A typical example is a game-playing program. Frequently, the best move is not known, but can be found using an exhaustive search with backtracking. When several alternatives are possible, we choose one and follow it until it becomes clear whether or not it was best. If not, we retreat to the last decision point and explore the other choices. A nondeterministic algorithm that can make the best choice would be able to solve the problem without backtracking, but a deterministic one can simulate nondeterminismwith some extra work. For this reason, nondeterministic machines can serve as models of search-and-backtrack algorithms.

- Nondeterminism is sometimes helpful in solving problems easily. Observe the figure given below. It is clear that there is a choice to be made. The first alternative leads to the acceptance of the string a^3, while the second accepts all strings with an even number of a's. The language accepted by the nfa is $\{a^3\} \cup \{a^{2n} : n \geq 1\}$. While it is possible to find a dfa for this language, the nondeterminism is quite natural. The language is the union of two quite different sets, and the nondeterminism lets us decide at the outset which case we want. The deterministic solution is not as obviously related to the definition, and so is a little harder to find. Thus, nondeterminism is an effective mechanism for describing some complicated languages concisely.



- Finally, there is a technical reason for introducing nondeterminism. Certain theoretical results are more easily established for nfa's than for dfa's. In the next section, we will see that there is no essential difference between these two types of automata. Consequently, allowing nondeterminism often simplifies formal arguments without affecting the generality of the conclusion.

3.3.3 Equivalence of Deterministic and Nondeterministic Finite Accepters

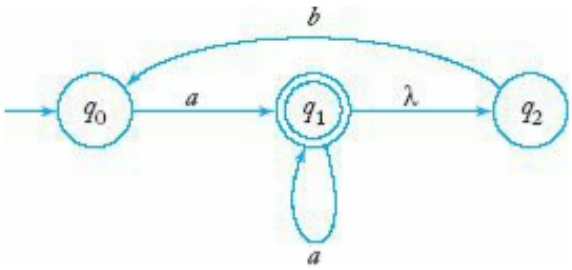
Definition 59. *Two finite accepters, M_1 and M_2 , are said to be equivalent if that is, if they both accept the same language*

$$L(M_1) = L(M_2)$$

When we compare different classes of automata, the question invariably arises whether one class is more powerful than the other. By "more powerful" we mean that an automaton of one kind can achieve something that cannot be done by any automaton of the other kind. Let us look at this question for finite accepters. Since a dfa is in essence a restricted kind of nfa, it is clear that any language that is accepted by a dfa is also accepted by some nfa. But the converse is not so obvious. It turns out that the classes of dfa's and nfa's are equally powerful: For every language accepted by some nfa there is a dfa that accepts the same language.

How to construct a dfa equivalent to a given nfa? The rationale for the construction is the following. After an nfa has read a string w, we may not know exactly what state it will be in, but we can say that it must be in one state of a set of possible states, say $\{q_i, q_j, ..., q_k\}$. An equivalent dfa after reading the same string must be in some definite state. How can we make these two situations correspond? The answer is a nice trick: Label the states of the dfa with a set of states in such a way that, after reading w, the equivalent dfa will be in a single state labeled $\{q_i, q_j, ..., q_k\}$. Since for a set of $|Q|$ states there are exactly $2^{|Q|}$ subsets, the corresponding dfa will have a finite number of states.

Example. Convert the following nfa to an equivalent dfa.



The nfa starts in state q_0 , so the initial state of the dfa will be labeled $\{q_0\}$. After reading an a, the nfa can be in state q_1 or, by making a λ-transition, in state q_2 . Therefore, the corresponding dfa must have a state labeled $\{q_1, q_2\}$ and a transition $\delta(\{q_0\}, a) = \{q_1, q_2\}$. In state q_0 , the nfa has no specified transition when the input is b. Therefore, $\delta(\{q_0\}, b) = \emptyset$. A state labeled $\emptyset$ represents an impossible move for the nfa and, therefore, means nonacceptance of the string. Consequently, this state in the dfa must be a non final trap state.

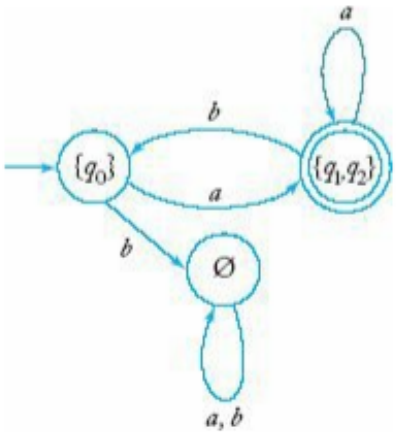
We have now introduced into the dfa the state $\{q_1, q_2\}$, so we need to find the transitions out of this state. Remember that this state of the dfa corresponds to two possible states of the nfa, so we must refer back to the nfa. If the nfa is in state q_1 and reads an a, it can go to q_1 . Furthermore, from q_1 the nfa can make a λ-transition to q_2. If, for the same input, the nfa is in state q_2 , then there is no specified transition. Therefore,

$$\delta(\{q_1, q_2\}, a) = \{q_1, q_2\}$$

Similarly,

$$\delta(\{q_1, q_2\}, b) = \{q_0\}$$

The equivalent dfa is shown below



Theorem 3.3.2. *Let L be the language accepted by a nondeterministic finite accepter $M_N = (\mathcal{Q}_N, \Sigma, \delta_N, q_0, F_N)$. Then there exists a deterministic finite accepter $M_D = (\mathcal{Q}_D, \Sigma, \delta_D, \{q_0\}, F_D)$ such that $L = L(M_D)$.*

Proof. Given M_N, we use the procedure $nfa - to - dfa$ below to construct the transition graph G_D for M_D. To understand the construction, remember that G_D has to have certain properties. Every vertex must have exactly $|\Sigma|$ outgoing edges, each labeled with a different element of Σ. During the construction, some of the edges may be missing, but the procedure continues until they are all there.

Procedure:

1. Create a graph G_D with vertex $\{q_0\}$. Identify this vertex as the initial vertex.
2. Repeat the following steps until no more edges are missing.

Take any vertex $\{q_i, q_j, \ldots, q_k\}$ of G_D that has no outgoing edge for some $a \in \Sigma$ Compute $\delta_N^*(q_i, a), \delta_N^*(q_j, a), ..., \delta_N^*(q_k, a)$. If

$$\delta_N^*(q_i, a) \cup \delta_N^*(q_j, a) \cup ... \cup \delta_N^*(q_k, a) = \{q_l, q_m, ..., q_n\},$$

create a vertex for G_D labeled $\{q_l, q_m, \ldots, q_n\}$ if it does not already exist. Add to G_D an edge from $\{q_i, q_j, \ldots, q_k\}$ and label it with a.
3. Every state of G_D whose label contains any $q_f \in F_N$ is identified as a final vertex.
4. If M_N accepts λ, the vertex $\{q_0\}$ in G_D is also made a final vertex.

It is clear that this procedure always terminates. Each pass through the loop in Step 2 adds an edge to G_D. But G_D has at most $2^{|Q_N|}|\Sigma|$ edges, so that the loop eventually stops. To show that the construction also gives the correct answer, we argue by induction on the length of the input string.
Assume that for every v of length less than or equal to n, the presence in G_N of a walk labeled v from q_0 to q_i implies that in G_D there is a walk labeled v from $\{q_0\}$ to a state $Q_i = \{..., q_i, ...\}$. Consider now any $w = va$ and look at a walk in G_N labeled w from q_0 to q_1. There must then be a walk labeled v from q_0 to q_i and an edge (or a sequence of edges) labeled a from q_i to q_l. By the inductive assumption, in G_D there will be a walk labeled v from $\{q_0\}$ to Q_i. But by construction, there will be an edge from Q_i to some state whose label contains q_l Thus, the inductive assumption holds for all strings of length $n+1$. As it is obviously true for $n = 1$, it is true for all n. The result then is that whenever contains a final state q_f , so does the label of $\delta^*_D(q_0, w)$. To complete the proof, we reverse the argument to show that if the label of $\delta^*_D(q_0, w)$ contains q_f, so must $\delta^*_D(q_0, w)$. □