Lecture Notes for Discrete Structures[1]

Sandeep Sen[2]

March 28, 2006

[1]

[2]Department of Computer Science and Engineering, IIT Delhi, New Delhi 110016, India.
E-mail:ssen@cse.iitd.ernet.in

Contents

Abstract

The following pages contain preliminary version of the lectures and problems on Discrete Structures. The notes are likely to contain errors, in particular typographic. I will endeavour to update this every week.

Chapter 1

Preliminaries

A *set* is a collection of objects. The objects of a set are called *members* or *elements*. Two sets are equal iff they have the same members. Usually we do not count repeated elements more than once - when we do they are called *multisets*. Sets may contain finite or infinite number of elements. A set that does not have any element is called *empty* and is denoted by ϕ. Some common set identities are

- Idempotency
- Commutativity
- Associativity
- Distributivity
- Absorption
- De Morgan's Laws

The *power set* of a set A is the collection of all distinct subsets of A (including *phi*) and is denoted by 2^A. A *partition* of A is a collection of subsets $A_1, A_2 \ldots$ such that $\cup_i A_i = A$ and $A_i \cap A_j = \Phi$ for all $i \neq j$.

1.1 Relations and Functions

A *Cartesian product* of two sets A and B denoted by $A \times B$ is the set of all ordered pairs (a, b) with $a \in A$ and $b \in B$. A *binary relation* R is a subset of $A \times B$. The definitions for Cartesian product and relations have natural extensions to k-fold Cartesian product and k-ary relation.

Definition 1.1.1 A relation $R \subset A \times A$ is **reflexive** if for all $a \in A$, $(a, a) \in R$. A relation is **symmetric** if $(b, a) \in R$ whenever $(a, b) \in R$. A relation is **anti-symmetric** if $(b, a) \in R$ then $(a, b) \notin R$. A relation is **transitive** if $(a, c) \in R$ whenever $(a, b) \in R$ and $(b, c) \in R$.

Definition 1.1.2 A binary relation that is reflexive, symmetric and transitive is called a **equivalence** relation.
A binary relation that is reflexive, antisymmetric and transitive is called a **partial order**.
A *partial order* is a **total order** if for every pair of distinct elements a, b, either (a, b) or (b, a) belongs to the partial order.

We often use the notation $a \sim b$ to denote that a, b are related under the equivalence relation $\sim$. For $a \in S$, the set of elements $[a] = \{x \in S | x \sim a\}$ is called the equivalence class of a.

Theorem 1.1.3 *The equivalence classes of an equivalence relation on a set S constitute a partition of S.*

Proof: Since $a \sim a$, $a \in [a]$. If $[a]$ and $[b]$ are two distinct equivalence classes where $b \notin [a]$, we must show that $[a] \cap [b] = \phi$. Suppose $c \in [a] \cap [b]$, then $a \sim c$ and $c \sim b$ and therefore from transitivity $a \sim b$. This implies that $b \in [a]$ which is a contradiction. $\Box$

A *function* f is defined from a set of objects called *domain* to another set called *range* or *co-domain*. Intuitively, f associates for **each** element of the domain a **unique element** of range. Often we represent a function by $f : A \to B$ and $f(a)$ to denote the element (of the range) to which $a \in A$ is *mapped* by f. Sometimes $f(a)$ is called the *image* of a (under f) or a as the *inverse image* of f. The definition of a function also naturally extends to k-ary functions, i.e., f has k *arguments*. Another view is to think of A as a set of ordered k tuples.

Definition 1.1.4 A function $f : A \to B$ is **onto** if each element of B is an image of at least one element of A. f is **one-to-one** if for two distinct a, a', $f(a) \neq f(a')$. A function f is a **bijection** if it is one-to-one and onto.

Bijections are especially useful for counting problems, For example, if we can find a bijection between (finite) sets A and B, then the number of elements in A equal that in B. The use of one-to-one functions are even more useful for comparing the number of elements in infinite sets.

1.2 Counting and comparing infinite sets

The motivating question for this topic is "Are there more reals numbers than rationals ?" Both sets $\mathbb{R}$ (set of Real numbers) and $\mathbb{Q}$ (the set of rationals are infinite sets, so how can we distinguish between the sizes of these sets. Similarly, we may want to find the compare the set of integers with rationals.

Definition 1.2.1 Two sets A and B are called *cardinally equivalent*, iff there is a bijective function $f : A \to B$ and this will be denoted by $\#(A) = \#(B)$.

Example 1.2.2 : Let A be a finite non-empty set, then there exists a unique integer n such that A is cardinally equivalent to $\{1, 2, \ldots n\}$. Then we say that A has n elements.
Example 1.2.3 : Let E be the set of even positive integers. Then $\#(E) = \#(\mathbb{Z}^+)$ where $\mathbb{Z}^+$ is the set of all positive integers using the function $f : \#(E) \to \#(\mathbb{Z}^+)$ where $f(n) = n/2$. This function is bijective, so intuitively the number of integers is the same as the number of even integers.

Definition 1.2.4 A set S is *countably infinite* iff $\#(S) = \#(\mathbb{Z}^+)$. A set is countable iff S is finite or countably finite.

Theorem 1.2.5 *Every subset of a countable set is countable. A countable union of a countable set is countable.*

Proof: For the first part, renumber the integers whose images are in the subset (i.e. the subsequence of $\{1, 2 \ldots n\}$). For the second part, simply construct a sequence that traverses the subsequences "diagonally." □

Lemma 1.2.6 *The set of reals, $\mathbb{R}$ is uncountable.*

Definition 1.2.7 If there exists an surjective (onto) function $f : A \to B$, then $\#(A) \leq \#(B)$. Equivalently there is an injective (1-1) mapping $g : B \to A$.
If $\#(A) \leq \#(B)$ and $\#(A) \neq \#(B)$, then $\#(A) < \#(B)$.

Example 1.2.8 : If $A \subset B$, then $\#(A) \leq \#(B)$. Consider the subsequence of the identity map(it is an onto map).

Theorem 1.2.9 *If S any set then $\#(S) < \#(2^S)$, i.e. there is no bijection between a set and its powerset.*

1.3 Principle of Induction

One of the most useful proof techniques in discrete structures is the principle of induction. There are two well known (equivalent) formulations of this. To distinguish between these we will give them different names.

Principle of Mathematical Induction
Let $P(i)$ denote a predicate that is defined for an integer i. If $P(0)$ is true and **for all i**, $P(i+1)$ is true whenever $P(i)$ is true, then $P(i)$ is true for all integers i.

Principle of Complete Induction
Let $P(i)$ denote a predicate that is defined for an integer i. If $P(0)$ is true and **for all i** $P(i+1)$ is true whenever $P(j)$ is true for all $j \leq i$, then $P(i)$ is true for all integers i.

Remark Although these are equivalent, we will find the second form easier to apply in most situations.

Problem Set

1. Let $S = \{(x, y)|x, y \text{ are reals}\}$. If (a, b) and (c, d) belong to S, define $(a, b)R(c, d)$ if $a^2 + b^2 = c^2 + d^2$. Prove that R is an equivalence relation.

2. Let S be the set of real numbers. If $a, b \in S$, define $a \sim b$ if $a - b$ is an integer. Show that $\sim$ is an equivalence relation.

3. Let S be a set of integers. If $a, b \in S$, let aRb, if $a \cdot b \geq 0$. Is R an equivalence relation on S? How about the relation R' where $aR'b$ if $a + b$ is even ?

4. Give examples of relations that are

   - reflexive and symmetric but not transitive
   - reflexive and transitive but not symmetric
   - symmetric and transitive but not reflexive

5. Show that for every positive integer n, show that 2^{2n-1} is divisible by 3.

6. Show that for every positive integer n and every real number θ. $(\cos\theta + i\sin\theta)^n = \cos n\theta + i\sin n\theta$.

7. **Fundamental Theorem of Arithmetic** Every integer greater than 1 is a prime or a product of primes and the product is unique up to the order of the factors. Prove the existence part using induction.

8. Show that the set $\mathbb{Q} \times \mathbb{Q}$ is countable.

Chapter 2

Basic Counting

2.1 Permutation and Combinations

A fundamental problem involving discrete structures is counting the number of objects/events satisfying some property. This includes the possibility if such a subset exists at all (*existence problem*). A more difficult version is choosing the *best* according to some criterion, namely *optimization*.

Two elementary rules are extensively used for counting, namely the **Addition Principle** and the **Multiplication principle**.

Definition 2.1.1 [Addition Principle] If one event can occur in m ways and another in n ways then there are $m + n$ ways in which **one** of the two events can occur.

Note that the two events cannot occur simultaneously.

Definition 2.1.2 [Multiplication Principle] If one event can occur in m ways and another event can occur in n, independently of each other, then there are $m \times n$ ways in which **both** events can occur.

Example 2.1.3 : To choose two books of different languages among 5 books in Latin, seven books in Greek and 10 books in Sanskrit, there are $5 \times 7 + 5 \times 10 + 7 \times 10 = 155$ ways since there are 5×7 ways to choose a Latin **and** a Greek book (multiplication principle), 5×10 ways to choose a Latin and a Sanskrit book and 7×10 ways to choose a Greek and a Sanskrit book. Finally, we must choose only one of the pairs, so the answer follows from the Addition principle.

It is not difficult to formally prove the two principles. We can use inducion in the following manner to prove the Addition Principle.
Induction Hypothesis For m of events of type 1 and i events of type 2, $i \geq 0$, the

number of events of either type is $m + i$.

Proof: By induction on i. The base case is clearly true, i.e. when there are no events of type 1. Suppose it is true for k events of type 2, namely there are $m + k$ eents. When there are $k+1$ events of type 2, then there are two distinct possiblities - either $k + 1$st event occurs or it doesn't. Theefore, by invoking the induction hypothesis, the total number of possibiliies is $n + k + 1$. □

Similarly one can prove the Multiplication Principle as well as the following generalizations given in exercises.

Definition 2.1.4 [**Permutation** and **Combination**] A **permutation** of n *distinct* objects is an arrangement or ordering of the n objects. An **r-permutation** of n distinct objects is an arrangement using r out of the n objects. An **r-combination** of n distinct objects is an unordered selection (subset) of size r.
We will denote r-permutation and r-combination of n objects by $P(n, r)$ and $C(n, r)$ respectively.

From the multiplication principle, we obtain

$$
\begin{array}{ll}
P(n,2) = n(n-1) & P(n,3) = n(n-1)(n-2) \\
P(n,n) = n(n-1)(n-2)\ldots 1 & = n! \text{ (n factorial)} \\
P(n,r) = n(n-1)\ldots(n-r+1) = & \frac{n!}{(n-r)!}
\end{array}
$$

To obtain a formula for r-combination, we will make use of an indirect technique. For every distinct r-subset, there are $P(r, r)$ distinct arrangements. Let us number the distinct r-permutations in some order say $\Pi_1, \Pi_2 \ldots \Pi_t$ where $t = P(n, r)$. We can group them in a way such that each group corresponds to a distinct r-subset. From the previous observation, each group has $P(r, r)$ members. Therefore

$$C(n,r) = \frac{P(n,r)}{P(r,r)} = \frac{n!}{(n-r)! \times r!}$$

Caveat We could have invoked multiplication principle, but we have to be careful about setting up the events appropriately. One of the most common pitfalls of counting problems is the temptation of applying the formulae carelessly. The formulae are usually simple but one must be careful about the applicability in a specific situation.

2.2 Distribution problems

The problem of counting the number of ways to distribute r objects in n cells shows up in different contexts and is also referred to as *occupancy* problems. Let us consider the following cases depending on distinguishable and distinguishable objects. The cells are distinct and can be numbered as $C_1, C_2 \dots C_n$.

We will first consider the distinguishable objects case and further separate out the situations where there can be at most one object per cell or the unrestricted case (any number). In the unrestricted case, there are n choices for every ball and since these are independent, the number of possibilities is n^r from the multiplication principle. In the restricted case, i.e. at most one object per cell, suppose $r \leq n$. Any distribution corresponds to an arrangement of r labels from the set $\{1, 2 \dots n\}$, namely, the labels of the occupied cells, where the i-th label indicates the placement of the i-th object. This is clearly $P(n, r)$. If $n \leq r$, then each distribution corresponds to an arrangement of n labels from the set $\{1, 2, \dots r\}$, namely the objects that are alloted to cells $C_1 \dots C_n$. This is the same as $P(r, n)$.

If the objects are indistinguishable, then in the unrestricted case, two distributions are equivalent, if the number of objects in each cell remains same (although the labels of the balls may be different). Each configuration can be described as a vector $(x_1, x_2, \dots x_n)$ where x_i denotes the content of the i-th cell. Moreover $\sum_i^n x_i = r$. Consider $n-1$ *markers* and look at any configuration of r balls and $n-1$ markers. Interpret the number of objects between the j and $j+1$st marker as the content of the j-th cell (make appropriate adjustments for the end markers). Notice that by permuting the (indistinguishable) markers among themselves and the (indistinguishable) objects among themselves the cell-contents do not change. So the number of distinct distributions (where two distributions are different if they differ in one or more cell contents) is given by $\frac{P(n+r-1,n+r-1)}{P(n-1,n-1)\times P(r,r)} = C(n+r-1, r)$. The above argument can be made more rigorous by invoking the Addition and Multiplication Principles.

In the restricted case (at most one per cell), the corresponding formulae for $r \leq n$ is $C(n, r)$. Let us formalise the arguments since these have a very intuitive connections with the corresponding figures of the distinguishable case. Let us consider the set of distinct distributions for the distinguishable objects when $r \leq n$. As noted above, these can be represented as strings of length r over the labels $\{1, 2 \dots n\}$. Two strings $s_1, s_2 \dots s_r$ and $s'_1, s'_2 \dots s'_r$ represent identical distribution for indistinguishable objects if one can be permuted into the other, i.e., they contain the same labels. Therefore, we can group the strings into *equivalence* classes, where a class corresponds to strings over the same labels.

Claim 2.2.1 *The number of classes is equal to the number of distribution of indistinguishable objects. Moreover each class contains exactly $r!$ strings.*

Proof: Since the set of labels are different for two classes, the number of possible distributions is not less than the number of classes. Moreover, each distribution corresponds to set of labels, so the number of distribution cannot exceed the number of classes. Therefore they are equal. (Basically $x \leq y$ and $y \leq x$ implies $x = y$).

For the second part of the claim, note that each distinct permutation of (a fixed set of) labels represent a different distribution for distinguishable objects. □

From the above claim it follows that the number of distributions for distinguishable objects equals $r!$ times the number of distributions for indistinguishable objects. This gives us the required result as $P(n,r) = C(n,r) \times r!$.

A new situation emerges if we fix the number of each type of objects (as opposed to having an unlimited number of each object). Suppose we have r_i objects of type i, such that $\sum_i^k r_i \leq n$. Note that the r_i objects are indistinguishable. Then the possible distributions is equal to

$$C(n; r_1, r_2, \ldots r_k) = \frac{n!}{r_1! \times r_2! \times \ldots r_k! \times (n - r_1 - r_2 \ldots r_k)!}$$

This follows from the observation that for the r_1 objects of type 1, the number of choices for placements is $C(n, r_1)$, for object 2, the number of choices is $C(n - r_1, r_2)$ and so on. The result follows from the Multiplication principle.

Remark 2.2.2 Note that the value of the above expression does not depend on the order in which the types are chosen.

Problem Set

1. If there are k events $E_1, E_2 \ldots E_k$, where E_i has n_i possibilities then show that
   - There are $n_1 + n_2 \ldots n_k$ ways in which one of E_i can occur.
   - There are $n_1 \times n_2 \ldots n_k$ ways in which all the events can occur (if they are independent of each other).
2. Show that $C(n,r) = C(n-1, r-1) + C(n-1, r)$.
   Instead of applying the formula, you may want to argue using the addition principle. Consider all distinct subsets that contain a fixed object and those subsets that do not contain this fixed object. The above is an example of a recurrence equation that will be addressed later in the course. The above identity can be used to derive a formula for $C(n,r)$ thus inverting the process.
3. In how many ways can you choose r objects out of n different kinds where there are unlimited number of objects of each type ?
4. How many ways are there to place two identical queens on an 8×8 chess board so that the queens are not in a common row, column or diagonal.

5. How many different rectangles can be drawn on an 8×8 chess board (rectangles can have lengths 1 through 8 and two rectangles are different if they contain a different subset of squares).

6. What is the probability that a 4-digit telephone number has one or more repeated digits ?

7. There are six French books, eight Russian books, and five different Spanish books. How many ways are there to arrange the books in a row with all books of the same language consecutively arranged ?

8. How many ways are there to assign 10 students to 10 out of 20 sections ?

9. A man has n friends and invites a different subset of four of them to his house for a year (365 nights). How large must n be ?

10. What is the probability that the difference between the largest and the smallest numbers is k in a subset of four different numbers chosen from 1 to 20 ($3 \leq k \leq 19$) ?

11. How many points of intersection are formed by the chords of an n-gon (a regular polygon with n sides) assuming that no three chords meet at a common point ? How many line segments are formed by the intersections - note that if a chord has k intersection points then it has $k+1$ segments.

12. How many integer solutions are there to the equation $x_1 + x_2 + x_3 + x_4 = 12$, with $x_i \geq 0$? How many solutions are there with $x_i \geq 1$?

13. In how many ways can you distribute 20 distinct flags into 12 distinct flagpoles if in arranging the flags on the poles, the order from the ground up makes a difference ?

14. In how many ways can you distribute r identical balls into n distinct boxes with the first m boxes collectively containing at least s balls ?

15. Eleven scientists are working on a secret project. They wish to lock up the documents in a cabinet such that the cabinet can be opened if and only if six or more scientists are present. What is the smallest number of locks required ? What is the smallest number of keys that each scientist must carry ?

16. In how many ways can three numbers be selected from the numbers $1, 2 \ldots 300$ such that their sum is divisible by 3 ?

17. Show that $(k!)!$ is divisible by $(k)!^{(k-1)!}$.

18. A binary string is a sequence of 0's and 1's. How many binary strings of length n contain an even number of 0's ? If strings are over the alphabet $\{0, 1, 2\}$, then show that the number of strings where 0 appears an even number of times is $(3^n + 1)/2$.

19. A boolean function can be represented using a tabular form where all the n-digit binary numbers are listed along with the function values. How many boolean functions are possible ?
A *self-dual* boolean function is a table which remains unchanged if all the 0's and 1's are swapped. How many self-dual boolean functions are there ?
A *symmetric* boolean function is one that remains unchanged for any permutation of the n input columns. How many symmetric boolean functions are there ?

20. A system consists of four identical particles. The total energy in the system is $4E_o$ where E_o is a positive constant. Each of the particles can have an energy level equal to kE_o ($k = 0, 1..4$). A particle with energy kE_o can occupy one of the $k^2 + 1$ distinct energy states at that energy level. How many different configurations (in terms of energy states occupied by the particles) can the system have ?

Chapter 3

Introduction to Graphs

A *graph* $G = (V, E)$ consists of a finite set V of *vertices* and a set E of *edges* which are ordered pairs of vertices. Schematically, we represent graphs using a set of points that denote vertices and edges by an arc joining the two defining vertices with an arrow indicating the ordering of the vertices. An *undirected* graph doesn't have directions associated with an edge. If we think about the edges as roads connecting vertices then in the undirected case we can traverse the edge in either direction where as the (directed) graph is like one-way streets. Unless stated otherwise a graph will be used to imply the undirected version.

There are several generalization of the basic definition. If the set of edges form a *multiset*, i.e., some edges have multiple instances, then it is a *multigraph*. One way to represent a multigraph is to label the edges with an integer denoting the number of occurrences of the edge. This may be regarded as a *weighted* graph, where each edge has an associated (integral) weight. In some cases, we will allow weights to be arbitrary real numbers.

A more complicated structure is a *hypergraph* where the edges correspond to arbitrary subsets of vertices (and not necessarily pairs of vertices). The choice of a certain class of graphs depends on the application.

Graphs can be used to model very complex problems and some of the most intuitive examples are problems related to communication networks. A flowchart can be thought of as a graph where the nodes represent instructions and the edges indicate the flow of control.

3.1 Representation of graphs

Graphs can be represented as a list of edges associated with every vertex. If there are $m = |E|$ edges and $n = |V|$ vertices then the size of the representation is roughly

$m + n$ (Why ?).

Another representation is using matrices of dimensions $n \times n$. If A_G is the matrix corresponding to graph $G = (V, E)$, then $A_{i,j} = 1$ if $(i, j) \in E$ and 0 otherwise. Here we are assuming that the vertex set is $\{1, 2, \ldots n\}$. The size of this representation is n^2 irrespective of the number of edges.

The motivation for having a good representation of graphs is to use computer programs for solving graph problems. The above two representations can be easily converted into appropriate data-structures.

3.2 Reachability in graphs

The *neighbourhood* of a vertex $v \in V$ is the set of vertices $W \subset V$ such that for all $w \in W$, $(v, w) \in E$. The number of vertices in the neighbourhood $N(v)$ of a vertex v is called the degree of v.

Definition 3.2.1 A **path** is a sequence of vertices $(x_1, x_2 \ldots x_k)$ such that x_i, x_{i+1} is an edge of the graph. A path is **simple** if there is no repetition of vertices. If $x_1 = x_k$ then the path is called a **cycle**.

3.2.1 Tours and cycles

A cycle that visits every vertex exactly once is called a Hamiltonian cycle. It is an extremely hard algorithmic problem to detect if a Hamiltonian cycle exists.

A cycle that visits every edge exactly once is called a Euler's path. Historically, the origin of the problem is known as the Konigsberg bridge problem. Two islands and two banks of the river Pregel were connected by seven bridges (see Figure **??**) and the problem is to make a tour passing through every bridge exactly once. Euler gave a very simple necessary and sufficient condition for such a tour to be feasible, namely every vertex should be of even degree. In the case of directed graphs, the equivalent condition is that for every vertex, the indegree equals the outdegree.

3.2.2 Connectivity

One of the basic problems in graphs is *connectivity*, namely if there exists a path between every pair of vertices. We will assume that a vertex is connected to itself.

Definition 3.2.2 A set of vertices C form a **connected component** if for every $u, v \in C$ there is a path from u to v. Moreover for all $x \notin C$, $C \cup \{x\}$ is not a connected component, i.e. C is maximal. If C includes all vertices in the graph, then the underlying graph is *connected*.

Remark Note that for directed graphs, a path of u to v is not the same as a path from v to u.

There are several algorithms for verifying if a given graph is connected, the most notable being *Depth First Search* and *Breadth First Search*. Among other consequences of these search techniques, they produce *Spanning Forest*, which is a special kind of a *sub-graph*.

Definition 3.2.3 A **subgraph** $S = (W, F)$ of a graph $G = (V, E)$ is graph such that $W \subset V$ and $F \subset E$. A subgraph is a **tree** if it is connected and removal of any one edge disconnects some pairs of vertices, i.e. it is a minimal connected graph. A set of disjoint trees is called a **forest**.

Lemma 3.2.4 *The number of edges in a tree, m is related to the number of vertices n by the formula $m = n - 1$.*

Corollary 3.2.5 *If there are k trees in a forest with m edges and n vertices then $m = n - k$.*

Lemma 3.2.6 *In a tree, there is a unique path between every pair of vertices.*

Remark This is equivalent to saying that there are no cycles in a tree.

3.2.3 k-connectivity

A measure of how well-connected a graph is related to the following question -

Does the graph remain connected if any subset of k vertices is removed ?

This is clearly motivated by the problem of node-failures in a communication network where we may have to find alternate routes. The same question can be posed with respect to a set of edges.

Definition 3.2.7 A graph is k vertex-connected if removal of any $k - 1$ vertices does not disconnect the graph. A graph is k edge-connected if the graph remains connected after removing any set of $k - 1$ edges.

A classic theorem on k-connectivity can be stated as follows

Theorem 3.2.8 (Menger) *Let s and t be distinct vertices of a graph G. Then the minimal number of vertices that must be removed to separate s from t is the maximum number of* vertex-disjoint *paths between s and t.*

Remark The same holds true for edge-disjoint paths and edge-connectivity. The minimum number of vertices (edges) that must be removed to disconnect a graph is called the vertex (edge) connectivity of the graph and is usually denoted by κ (λ).

3.3 Some special classes of graphs

A graph is called *bipartite*, if its vertices can be partitioned into two sets V_1, V_2 such that there there are no edges between the vertices in V_1 (respectively V_2).

Lemma 3.3.1 *A graph is bipartite if and only if all the cycles are of even length.*

A *matching* in a graph $G = (V, E)$ is subset $M \subset E$ such that no two edges share an endpoint. A matching M is *maximal* is there is no matching M' such that $M \subset M'$. A matching is *maximum* id there is no larger matching. A matching is *perfect* if all vertices are matched.

Let M be a matching. A path P is called an M-alternating path if its edges alternate between edges in M and $E-M$. An M-alternating path is an M-augmenting path if P starts and ends with vertices that are not matches in M.

Theorem 3.3.2 (Berge) *M is maximum iff there is no augmenting path.*

In a bipartite graph, if all the vertices in V_1 are matched then these vertices are *saturated.*

Theorem 3.3.3 (Hall) *In a bipartite graph $G = (V_1 \cup V_2, E)$, there exists a matching that saturates all vertices in V_1 iff for all $S \subset V_1$, $|N(S)| \geq |S|$ where $N(S)$ is the set of all vertices in V_2 that are connected to S by edges in E.*

A graph is *planar* if it can be drawn on a plane without the edges crossing. (Strictly speaking, if the graph can be embedded on the sphere without edges crossing). It is known that every planar graph has a straight line embedding (i.e. all edges are straight line segments).

Lemma 3.3.4 (Euler's formula) *If G is a connected planar graph, then any plane graph embedding of G that has v vertices, e edges, and r regions satisfies $v+r-e=2$.*

A very elegant theorem due to Kuratowsky, gives a necessary and sufficient condition for a graph to be planar.

Theorem 3.3.5 (Kuratowski) *A graph is planar iff it doesn't contain any subgraph* homeomorphic *to K_5 (the complete graph on five vertices) or $K_{3,3}$ (complete bipartite graph on 6 vertices).*

Definition 3.3.6 A *colouring* of a graph assigns colours to vertices such that no two adjacent vertices have the same colour. The minimal number of colours required for a graph G is called the *chromatic number* and is usually denoted by $\chi(G)$. An *edge-colouring* of a graph is a colouring of the edges such that no two edges that are incident on the same vertex get the same colour.

Clearly bipartite graphs are two colourable. One of the classic colouring theorems concern planar graphs.

Theorem 3.3.7 (four-colour theorem) *Every planar graph is 4-colourable.*

There are many natural problems that can be modelled as graph coloring.
Example 3.3.8 : In a school each teacher has to teach a certain number of classes and each class must be taught by a certain number of teachers. The obvious constraints about scheduling the classes is that a teacher cannot teach two classes simultaneously and a class cannot be taught by two teachers. We are interested in scheduling the classes in a way that takes minimum number of hours (the duration of a lecture). It is not difficult to see that a valid scheduling corresponds to colouring the edges. So the answer to this problem is the minimum number of colours required. The following is an important result on edge-colouring.

Theorem 3.3.9 (Vizing's Theorem) *If the maximum degree of a graph is d, then we need d or $d+1$ colours to colour the edges.*

3.4 Problem Set

1. In a graph that has exactly two vertices of odd degree, there is a path connecting these vertices.

2. Prove or disprove
   The union of any two distinct paths (not necessarily simple) joining two vertices contains a cycle.

3. A graph is connected if and only if for any partition V into two subsets V_1 and V_2, there is an edge joining a vertex in V_1 with a vertex in V_2.

4. In a connected graph, any two longest paths have a point in common.

5. If a graph G is not connected, then the complement of G, $\bar{G}$ is connected. ($\bar{G} = (V, \bar{E})$, where $(v, w) \in \bar{E}$ iff $(v, w) \notin \bar{E}$)

6. If δ is the minimum degree of a vertex and κ and λ are the vertex and edge connectivity, show that $\kappa \leq \lambda \leq \delta$.

7. Show that any graph has two vertices of equal degree.
8. Show that $d_1 \leq d_2 \leq \ldots d_n$ is the degree sequence of a tree iff $d_1 \geq 1$ and $\sum_i d_i = 2n-2$.
9. Show that a tree is 2-colourable.
10. Let $G = (V, E)$ be a directed graph A. A *covering* is a partition of the arcs and in paths and cycles such that $E = \cup_i E_i$ where E_i is a path or a cycle and $E_i \cap E_j = \phi$ for $i \neq j$. A covering minimum k is called a minimal covering. Prove that if the graph is a directed connected Euler graph then it has a unique minimal cover, namely the Euler cycle.
    Hint: First show that the cover can contain only cycles and then show that it has exactly one cycle (by merging cycles).
11. A connected graph has an Euler circuit if and only if it can be partitioned into simple cycles.
12. There are n teams in a round-robin tournament. Show that they can be ordered according to their winning records such that each team immediately precedes a team that it has beaten. (This ordering is not unique).
13. Eleven students plan to have dinner together for several days. They will be seated in a round table and the plan calls for each student to have different neighbors each day. How many days are needed ?
14. If a graph has maximum degree d then show that it can be coloured using $d+1$ colours. Also show that if a graph has $O(|V|)$ edges then it can be coloured using $O(\sqrt{V})$ colours.
15. Show that the vertices of any graph can be partitioned into two sets such that for every vertex, the set of neighbours is equally distibuted into the two groups.
16. If every vertex has degree at least $|V|/2$ then there is a simple cycle consisting of all vertices.

Chapter 4

Counting techniques

The basic methods of counting using permutations and combinations are sometimes not adequate or are too complex to apply in many situations. There are many techniques that have been developed for specific problems which have grown from *ad hoc* to fairly general principles. We discuss three such techniques in this chapter - namely **pigeon-hole**, **principle of inclusion and exclusion** and one of the most successful in recent years called the **probabilistic method**.

4.1 The pigeon hole principle

It is based on a very simple observation that if more than n items are distributed in n pigeon-holes then at least one of them will have more than one item.
Example 4.1.1 : At least two vertices in a graph have the same degree. (Exercise problem in previous chapter)
Since there are n vertices and all the degrees must be in the range $[1, 2 \ldots n-1]$, at least two vertices must fall in the same value of the range.

A lot of geometric packing problems fall under this category.
Example 4.1.2 : Show that five points cannot be placed in an unit square such that every pair is at least unit distance apart.

A very common usage of this principle is that if the weighted sum of n items is w then no more than a $\frac{1}{k}$ of them can exceed k times the average weight (k is a positive integer). This is often known as Markov's inequality for expectation.

A classic application of the pigeon-hole is to the following problem also known as the Erdos-Szekeres theorem.

Theorem 4.1.3 *In any sequence of more than $(r-1) \cdot (s-1)$ different numbers there is an increasing subsequence of r terms or a decreasing subsequence of s terms* *or both. Roughly speaking, in a sequence of length n there is an increasing or a decreasing subsequence of length $\lceil\sqrt{n}\rceil$.*

Proof For each number n_i of the sequence, let us label with (x_i, y_i) which are the lengths of the largest increasing/decreasing subsequence beginning/ending at n_i. If there is no increasing/decreasing subsequence of length r/s, $1 \leq x_i \leq r-1$ and $1 \leq y_i \leq s-1$. Since there are more than $(r-1)\cdot(s-1)$ numbers, some pair must be repeated - say $x_i = x_j$ and $y_i = y_j$ for $j > i$. If $n_i < n_j$ then $x_i > x_j$, else $y_j > y_i$.

4.2 Principle of Inclusion and Exclusion

This is easier to understand in terms of sets of objects. It is very easy to show that for sets X and Y

$$|X \cup Y| = |X| + |Y| - |X \cap Y|$$

In general suppose there are N objects that have various properties numbered $\{1, 2, \ldots k\}$ (for convenience). Each object has none or many of these properties. Let N_i be the number of objects with property i and N_S be the number of objects that have properties $S \subset \{1, 2, \ldots k\}$. If we use N_0 to denote the number of objects that have none of the properties then

$$N_0 = N - (\sum_i N_i) + (\sum_{i,j} N_{i,j}) - (\sum_{i,j,k} N_{i,j,k}) \ldots + (-1)^k N_{1,2,3..k}$$

The proof of this can be worked out along the following lines. If an object does not satisfy any of the properties, then it contributes exactly 1 to both sides. Consider an object that satisfies exactly $r \geq 1$ properties. Then it contributes $-r$ to the first summation, $C(r, 2)$ to the second summation, $(-1)^i C(r, i)$ to the i-th summation. Therefore it is

$$1 - C(n, 0) + C(n, 2) \ldots (-1)^k = 0$$

which is exactly what it contributes to the left hand side.

Example 4.2.1 : **Euler's totient function** Let m be a positive integer whose distinct prime factors are $p_1, p_2 \ldots p_n$. Then the number of integers that are relatively prime to m (i.e. no common factors other than 1) is

$$\phi(m) = m\left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right)\ldots\left(1 - \frac{1}{p_n}\right)$$

Example 4.2.2 : Show that the number of permutations of $\{1, 2, \ldots n\}$ such that for all i, i does not map to the position i (also called derangement) is

$$n!\left(\frac{1}{2!} - \frac{1}{3!}\ldots(-1)^n\frac{1}{n!}\right)$$

4.3 The probabilistic method

Consider an experiment where there are a number of possible outcomes that we call the *sample space*. An *event* corresponds to a subset of the sample-space S; that is it corresponds to some outcomes of the sample-space. Historically Laplace defined probability of an event E as

$$\text{Probability of E } = \frac{|E|}{|S|}$$

where $|.|$ denotes the number of outcomes. This definition is applicable where all outcomes are equally likely. The above formula can be used to count the number of outcomes of E if we know the probability. This method is particularly useful when we are interested in a bound on $|E|$ rather than the exact count which is harder to obtain.

The notions of *independent* events and *conditional probability* are very useful in this regard. We give a very brief account of the basics of probability theory at the end of this chapter.

We motivate the use of the probabilistic method with the following problem. Given six people, where every pair of persons either know eachother or they are strangers, show that there always exists a set of three people who are mutually known or mutually strangers.

We do a case analysis from the perspective of any of the six persons (say person 1), he knows at least three others or doesn't know at least three persons among the remaining five. Consider the case that he knows three (the other case is symmetric), say X, Y, Z, we can easily argue that either X, Y, Z are mutually strangers or at least two among them know eachother (and of course know person 1).

The above problem can be posed as an equivalent problem in edge colouring, where in K_6, if we two colour the edges, then there is a monochromatic (all edges with same colour) triangle. The drawback with the previous solution is that it is very difficult to argue similar properties with somewhat larger number of vertices, even 10, using case analysis. You can try to convince yourself by trying to showw that in K_{18} there is a monochromatic K_4.

We now show the application of a new method based on probabilistic arguments.

Example 4.3.1 : Let $R(k, t)$ be the minimal n such that when complete graph K_n is edge coloured using blue and red colours, either there is a red K_t or a blue K_k. Using case analysis, one can show that K_5 contains either a red triangle or a blue triangle.

For larger values of n, case analysis becomes intractable. Here is an alternate argument. Suppose we colour the edges of K_n red and blue choosing each colour with equal probability. So the sample space consists of all possible colourings of K_n. The probability that a set S of k vertices is monochromatic is $p_k = \frac{1}{2^{1+C(k,2)}}$ corresponding to the two colours. The probability that any of the $C(n,k)$ K_k are monochromatic is less than $\sum p_k \cdot C(n,k)\cdot$. (Note that the probability of the union of events is no more that than the sum of the probabilities of the individual events). If this probability is less than 1, it implies that among the sample space of all possible colourings of K_n, there exists some colouring where all the $C(n,k)$ cliques are not **monochromatic**, i.e. $R(k,k) > n$.

Our next example is an important problem in graph theory. A dominating set U of an undirected graph $G = (V,E)$ is a subset $U \subset V$ such that every vertex has a neighbour in U. The problem of computing a minimum cardinality dominating set is very hard (algorithmically intractable. But we can prove some interesting bounds using the probabilistic method.

Example4.3.2 : If the minimum degree of a graph is δ, then there is a dominating set of size at most $n \cdot (1 + \log(\delta+1))/(\delta+1)$.

Pick every vertex independently with probability $p = \log(\delta+1)/(\delta+1)$ and let X denote this sample. Let Y be the set of vertices in V that do not have a neighbour in X. The probability that a vertex v does not have a neighbour in X is the probability q that neither v nor any of the δ neighbours were picked in the sample which is

$$(1 - \log(\delta+1)/(\delta+1))^{\delta+1}$$

So the expected size of Y is nq expected size of X is np and using the linearity of expectation $E[X+Y] = n(p+q)$ which works out to be $\leq n \cdot (1+\log(\delta+1))/(\delta+1)$. This means that there is some choice of X for which there is a dominaing set $(X \cup Y)$ of the required size. In fact we can claim something stronger that by choosing the vertices randomly the probability that the dominating set exceeds twice the stated bound is less than half (Markov's inequality).

4.4 Problem Set

1. There are n letters which have corresponding n envelopes. If the letters are put blindly in the envelopes, show that the probability that none of the letters goes into the right envelope tends to $\frac{1}{e}$ as n tends to infinity.

2. How many 1-1 functions exist between $\{1, 2, \ldots m\} to \{$ 1,2, ... n } (for $n \geq m$) ?

For $n \leq m$, show that the number of onto functions is given by

$$n^m - C(n,1)(n-1)^m + C(n,2)(n-2)^m \ldots (-1)^{n-1} C(n,n-1) 1^m$$

3. There are 10 pairs of shoes in a closet. In how many ways can eight shoes be chosen such that no pair is chosen ? Exactly one pair is chosen ?

4. Given $n+1$ different positive integers $\leq 2n$, show that there exists a pair that adds upto $2n+1$.

5. Prove that in *any* $n+1$ integers there will be a pair which differs by a multiple of n. Using this or otherwise show that there exists some subset of n arbitrary positive integers that whose summation is a multiple of n.

6. Given an equilateral triangle T, show that it is not possible to cover T with three circles each of diameter less than $\frac{1}{\sqrt{3}}$.

7. Show that in a planar graph $G = (V, E)$, there is a constant $\alpha < 1$ (independent of the number of vertices or edges) such that there are at least $\alpha|V|$ vertices of degree less than 12.

8. Show that among 23 people, the probability that all their birthdays are distinct is less than 0.5. Assume that for each person all birthdays are equally likely. **Remark** You can think of this as a probabilistic analogue of the pigeon-hole for which there had to be 367 persons to guarantee (with probability 1) that there was some common birthday. In literature this is known as the birthday paradox.

9. What is probability that when 50 balls are thrown into 100 bins that these fall into 10 or less bins ?

10. What is the probability that when you throw m balls in n bins, that (at least) one of the bins is unoccupied ?

11. Consider the experiment of tossing a fair coin till two heads or two tails appear in succession.
(i) Describe the sample space.
(ii) What is the probability that the experiment ends with an even number of tosses ?
(iii) What is the expected number of tosses ?

12. A chocolate company is offering a prize for anyone who can collect pictures of n different cricketers, where each wrap has one picture. Assuming that each chocolate can have any of the pictures with equal probability, what is the expected number of chocolates one must buy to get all the n different pictures ?

13. In a temple, thirty persons give their shoes to the caretaker who hands back the shoes at random. What is the expected number of persons who get back their own shoes.

14. Imagine that you are lost in a new city where you come across a crossroad. Only one of them leads you to your destination in 1 hour. The others bring you back to the same point after 2,3 and 4 hours respectively. Assuming that you choose each of the roads with equal probability, what is the expected time to arrive at your destination ?

15. **Gabbar Singh problem** Given that there are 3 consecutive blanks and three consecutive loaded chambers in a pistol, and you start firing the pistol from a random chamber, calculate the following probabilities. (i) The first shot is a blank (ii) The second shot is also a blank given that the first shot was a blank (iii) The third shot is a blank given that the first two were blanks.

16. A gambler uses the following strategy. The first time he bets Rs. 100 - if he wins, he quits. Otherwise. he bets Rs. 200 and quits regardless of the result. What is the probability that he goes back a winner assuming that he has probability 1/2 of winning each of the bets.
    What is the generalization of the above strategy ?

17. Three prisoners are informed by the jailor that one of them will be acquited without divulging the identity. One of the prisoners requests the jailor to divulge the identity of one of the other prisoner who won't be acquited. The jailor reasons that since at least one of the remaining two will not be acquited, reveals the identity. However this makes this prisoner very happy. Can you explain this ?

18. Show that $R(s, g) \geq (s-1) \cdot (g-1) + 1$ using explicit construction, i.e. describe a colouring on $K_{(s-1)\cdot(g-1)}$.

19. Verify that $R(k, k) > 2^{k/2}$ using the probabilistic method. Note that this is a much superior bound compared to the previous problem.

20. Let $W(k)$ be the least n such that if the set $\{1, 2, \dots n\}$ is two-coloured, there exists a monochromatic arithmetic progression of k terms. Show that $W(k) > 2^{k/2}$ using the probabilistic method.

4.5 Some basics of probability theory

The sample space Ω may be infinite with infinite elements that are called *elementary events*. For example consider the experiment where we must toss a coin until a head comes up for the first time. A *probability space* consists of a sample space with a *probability measure* associated with the elementary events. The probability measure Pr is a real valued function on events of the sample space and satisfies the following

1. For all $A \subset \Omega$, $0 \leq \Pr[A] \leq 1$
2. $\Pr[\Omega] = 1$
3. For mutually disjoint events $E_1, E_2 \dots$, $\Pr[\cup_i E_i] = \sum_i \Pr[E_i]$

Sometimes we are only interested in a certain collection of events (rather the entire sample space)a, say F. If F is closed under union and complementation, then the above properties can be modified in a way as if $F = \Omega$.

The principle of Inclusion-Exclusion has its counterpart in the probabilistic world, namely

Lemma 4.5.1

$$\Pr[\cup_i E_i] = \sum_i \Pr[E_i] - \sum_{i<j} \Pr[E_i \cap E_j] + \sum_{i<j<k} \Pr[E_i \cap E_j \cap E_k] \dots$$

Definition 4.5.2 A *random variable* (r.v.) X is a real-valued function over the sample space, $X : \Omega \to \mathbb{R}$. A *discrete random variable* is a random variable whose range is finite or a countable finite subset of $\mathbb{R}$.
The *distribution function* $F_X : \mathbb{R} \to (0, 1]$ for a random variable X is defined as $F_X(x) \leq \Pr[X = x]$. The *probability density function* of a discrete r.v. X, f_X is given by $f_X(x) = \Pr[X = x]$.
The *expectation* of a r.v. X, denoted by $E[X] = \sum_x x \cdot \Pr[X = x]$.

A very useful property of expectation, called the *linearity property* can be stated as follows

Lemma 4.5.3 *If X and Y are random variables, then*

$$E[X + Y] = E[X] + E[Y]$$

Remark Note that X and Y do not have to be independent !

Definition 4.5.4 The *conditional probability* of E_1 *given* E_2 is denoted by $\Pr[E_1|E_2]$ and is given by

$$\frac{\Pr[E_1 \cap E_2]}{\Pr[E_2]}$$

assuming $\Pr[E_2] > 0$.

Definition 4.5.5 A collection of events $\{E_i | i \in I\}$ is *independent* if for all subsets $S \subset I$

$$\Pr[\cap_{i \in S} E_i] = \Pi_{i \in S} \Pr[E_i]$$

Remark E_1 and E_2 are independent if $\Pr[E_1|E_2] = \Pr[E_2]$.

The *conditional probability* of a random variable X with respect to another random variable Y is denoted by $\Pr[X = x|Y = y]$ is similar to the previous definition with events E_1, E_2 as $X = x$ and $Y = y$ respectively. The *conditional expectation* is defined as

$$E[X|Y = y] = \sum_x \Pr x \cdot [X = x|Y = y]$$

The **theorem of total expectation** that can be proved easily states that

$$E[X] = \sum_y E[X|Y = y]$$

Chapter 5

Recurrences and generating functions

Given a sequence $a_1, a_2 \ldots a_n$ (i.e. a function with the domain as integers), a compact way of representing it is an equation in terms of itself, a recurrence relation. One of the most common examples is the Fibonacci sequence specified as $a_n = a_{n-1} + a_{n-2}$ for $n \geq 2$ and $a_0 = 0,\ a_1 = 1$. The values a_0, a_1 are known as the *boundary conditions*. Given this and the recurrence, we can compute the sequence step by step, or better still we can write a computer program. Sometimes, we would like to find the general term of the sequence. Very often, the running time of an algorithm is expressed as a recurrence and we would like to know the explicit function for the running time to make any predictions and comparisons. A typical recurrence arising from a *divide-and-conquer* algorithm is

$$a_{2n} = 2a_n + cn$$

which has a solution $a_n \leq 2cn\lceil \log_2 n \rceil$. In the context of algorithm analysis, we are often satisfied with an upper-bound. However, to the extent possible, it is desirable to obtain an exact expression.

Unfortunately, there is no general method for solving all recurrence relations. In this chapter, we discuss solutions to some important classes of recurrence equations. In the second part we discuss an important technique based on *generating functions* which are also important in their own right.

5.1 An iterative method - summation

As starters, some of the recurrence relations can be solved by summation or *guessing* and verifying by induction.

Example 5.1.1 : The number of moves required to solve the *Tower of Hanoi* problem with n discs can be written as

$$a_n = 2a_{n-1} + 1$$

By substituting for a_{n-1} this becomes

$$a_n = 2^2 a_{n-2} + 2 + 1$$

By expanding this till a_1, we obtain

$$a_n = 2^{n-1}a_1 + 2^{n-2} + \ldots.. + 1$$

This gives $a_n = 2^n - 1$ by using the formula for geometric series and $a_1 = 1$.
Example 5.1.2 : For the recurrence

$$a_{2n} = 2a_n + cn$$

we can use the same technique to show that $a_{2n} = \sum_{i=0} \log_2 n 2^i n/2^i \cdot c + 2na_1$.
Remark We made an assumption that n is a power of 2. In the general case, this may present some technical complication but the nature of the answer remains unchanged. Consider the recurrence

$$T(n) = 2T(\lfloor n/2 \rfloor) + n$$

Suppose $T(x) = cx \log_2 x$ for some constant $c > 0$ for all $x < n$. Then $T(n) = 2c\lfloor n/2 \rfloor \log_2 \lfloor n/2 \rfloor + n$. Then $T(n) \leq cn \log_2(n/2) + n \leq cn \log_2 n - (cn) + n \leq cn \log_2 n$ for $c \geq 1$.

A very frequent recurrence equation that comes up in the context of divide-and-conquer algorithms (like mergesort) has the form

$$T(n) = aT(n/b) + f(n) \quad a, b \ \text{ are constants and } f(n) \text{ a positive monotonic function}$$

Theorem 5.1.3 *For the following different cases, the above recurrence has the following solutions*

- *If* $f(n) = O(n^{\log_b a - \epsilon})$ *for some constant* ϵ, *then* $T(n)$ *is* $\Theta(n^{\log_b a})$.
- *If* $f(n) = O(n^{\log_b a})$ *then* $T(n)$ *is* $\Theta(n^{\log_b a} \log n)$.
- *If* $f(n) = O(n^{\log_b a + \epsilon})$ *for some constant* ϵ, *and if* $af(n/b)$ *is* $O(f(n))$ *then* $T(n)$ *is* $\Theta(f(n))$.

Example 5.1.4 : What is the maximum number of regions induced by n lines in the plane ? If we let L_n represent the number of regions, then we can write the following recurrence

$$L_n \leq L_{n-1} + n \quad L_0 = 1$$

Again by the method of summation, we can arrive at the answer $L_n = \frac{n(n+1)}{2} + 1$.
Example 5.1.5 : Let us try to solve the recurrence for Fibonacci, namely

$$F_n = F_{n-1} + F_{n-2} \qquad F_0 = 0, \quad F_1 = 1$$

If we try to expand this in the way that we have done previously, it becomes unwieldy very quickly. Instead we "guess" the following solution

$$F_n = \frac{1}{\sqrt{5}} \left(\phi^n - \bar{\phi}^n\right)$$

where $\phi = \frac{(1+\sqrt{5})}{2}$ and $\bar{\phi} = \frac{(1-\sqrt{5})}{2}$. The above solution can be verified by induction. Of course it is far from clear how one can magically guess the right solution. We shall address this later in the chapter.

5.2 Linear recurrence equations

A recurrence of the form

$$c_0 a_r + c_1 a_{r-1} + c_2 a_{r-2} \ldots + c_k a_{r-k} = f(r)$$

where c_i are constants is called a *linear recurrence equation* of order k. Most of the above examples fall under this class. If $f(r) = 0$ then it is *homogeneous linear recurrence*.

5.2.1 Homogeneous equations

We will first outline the solution for the homogeneous class and then extend it to the general linear recurrence. Let us first determine the number of solutions. It appears that we must know the values of $a_1, a_2 \ldots a_k$ to compute the values of the sequence according to the recurrence. In absence of this there can be different solutions based on different boundary conditions. Given the k boundary conditions, we can *uniquely* determine the values of the sequence. Note that this is not true for a *non-linear* recurrence like

$${a_r}^2 + a_{r-1} = 5 \text{ with } a_0 = 1$$

This observation (of unique solution) makes it somewhat easier for us to guess some solution and verify.

Let us guess a solution of the form $a_r = A\alpha^r$ where A is some constant. This may be justified from the solution of Example 5.1. By substituting this in the homogeneous linear recurrence and simplification, we obtain the following equation

$$c_0\alpha^k + c_1\alpha^{k-1} \ldots + c_k = 0$$

This is called the *characteristic equation* of the recurrence relation and this degree k equation has k roots, say $\alpha_1, \alpha_2 \ldots \alpha_k$. If these are all distinct then the following is a solution to the recurrence

$$a_r{}^{(h)} = A_1\alpha_1^r + A_2\alpha_2^r + \ldots A_k\alpha_k^r$$

which is also called the *homogeneous solution to linear recurrence*. The values of $A_1, A_2 \ldots A_k$ can be determined from the k boundary conditions (by solving k simultaneous equations).

When the roots are not unique, i.e. some roots have multiplicity then for multiplicity m, $\alpha^n, n\alpha^n, n^2\alpha^n \ldots n^{m-1}\alpha^n$ are the associated solutions. This follows from the fact that if α is a multiple root of the characteristic equation, then it is also the root of the derivative of the equation.

5.2.2 Inhomogeneous equations

If $f(n) \neq 0$, then there is no general methodology. Solutions are known for some particular cases, known as *particular* solutions. Let $a_n^{(h)}$ be the solution by ignoring $f(n)$ and let $a_n^{(p)}$ be a particular solution then it can be verified that $a_n = a_n^{(h)} + a_n^{(p)}$ is a solution to the non-homogeneous recurrence.

The following is a table of some particular solutions

d a constant	B
dn	$B_1n + B_0$
dn^2	$B_2n^2 + B_1n + B_0$
ed^n, e, d are constants	Bd^n

Here B, B_0, B_1, B_2 are constants to be determined from initial conditions. When $f(n) = f_1(n) + f_2(n)$ is a sum of the above functions then we solve the equation for $f_1(n)$ and $f_2(n)$ separately and then add them in the end to obtain a particular solution for the f(n).

5.3 Generating functions

An alternative representation for a sequence $a_1, a_2 \dots a_i$ is a polynomial function $a_1 x + a_2 x^2 + \dots a_i x^i$. Polynomials are very useful objects in mathematics, in particular as "placeholders." For example if we know that two polynomials are equal (i.e. they evaluate to the same value for all x), then all the corresponding coefficients must be equal. This follows from the well known property that a degree d polynomial has no more than d distinct roots (unless it is the zero polynomial). The issue of convergence is not important at this stage but will be relevant when we use the method of differentiation.

Example 5.3.1 : Consider the problem of changing a Rs 100 note using notes of the following denomination - 50, 20, 10, 5 and 1. Suppose we have an infinite supply of each denomination then we can represent each of these using the following polynomials where the coefficient corresponding to x^i is non-zero if we can obtain a certain sum using the given denomination.

$$P_1(x) = x^0 + x^1 + x^2 + \dots$$

$$P_5(x) = x^0 + x^5 + x^{10} + x^{15} + \dots$$
$$P_{10}(x) = x^0 + x^{10} + x^{20} + x^{30} + \dots$$
$$P_{20}(x) = x^0 + x^{20} + x^{40} + x^{60} + \dots$$
$$P_{50}(x) = x^0 + x^{50} + x^{100} + x^{150} + \dots$$

For example, we cannot have 51 to 99 using Rs 50,so all those coefficients are zero.

By multiplying these polynomials we obtain

$$P(x) = E_0 + E_1 x + E_2 x^2 + \dots E_{100} x^{100} + \dots E_i x^i$$

where E_i is the number of ways the terms of the polynomials can combine such that the sum of the exponents is 100. Convince yourself that this is precisely what we are looking for. However we must still obtain a formula for E_{100} or more generally E_i, which the number of ways of changing a sum of i.

Note that for the polynomials $P_1, P_5 \dots P_{50}$, the following holds

$$P_k(1 - x^k) = 1 \quad \text{for } k = 1, 5, ..50 \text{ giving}$$
$$P(x) = \frac{1}{(1-x)(1-x^5)(1-x^{10})(1-x^{20})(1-x^{50})}$$

We can now use the observations that $\frac{1}{1-x} = 1 + x^2 + x^3 \dots$ and $\frac{1-x^5}{(1-x)(1-x^5)} = 1 + x^2 + x^3 \dots$. So the corresponding coefficients are related by $B_n = A_n + B_{n-5}$ where

A and B are the coefficients of the polynomials $\frac{1}{1-x}$ and $\frac{1}{(1-x)(1-x^5)}$. Since $A_n = 1$, this is a linear recurrence. Find the final answer by extending these observations. Let us try the method of generating function on the Fibonacci sequence.

Example 5.3.2 : Let the generating function be $G(z) = F_0 + F_1 x + F_2 x^2 \ldots F_n x^n$ where F_i is the i-th Fibonacci number. Then $G(z) - zG(z) - z^2 G(z)$ can be written as the infinite sequence

$$F_0 + (F_1 - F_2)z + (F_2 - F_1 - F_0)z^2 + \ldots (F_{i+2} - F_{i+1} - F_i)z^{i+2} + \ldots = z$$

for $F_0 = 0, F_1 = 1$. Therefore $G(z) = \frac{z}{1-z-z^2}$. This can be worked out to be

$$G(z) = \frac{1}{\sqrt{5}} \left(\frac{1}{1 - \phi z} - \frac{1}{1 - \bar{\phi} z} \right)$$

where $\bar{\phi} = 1 - \phi = \frac{1}{2}\left(1 - \sqrt{5}\right)$.

5.3.1 Binomial theorem

The use of generating functions necessitates computation of the coefficients of power series of the form $(1+x)^\alpha$ for $|x| < 1$ and *any* α. For that the following result is very useful - the coefficient of x^k is given by

$$C(\alpha, k) = \frac{\alpha \cdot (\alpha - 1) \ldots (\alpha - k + 1)}{k \cdot (k-1) \ldots 1}$$

This can be seen from an application of Taylor's series. Let $f(x) = (1+x)^\alpha$. Then from Taylor's theorem, expanding around 0 for some z,

$$f(z) = f(0) + zf'(0) + \alpha \cdot z + z^2 \frac{f''(0)}{2!} + \ldots z^k \frac{f^{(k)}(0)}{k!} \ldots$$

$$= f(0) + 1 + z^2 \frac{\alpha(\alpha-1)}{2!} + \ldots C(\alpha, k) + \ldots$$

Therefore $(1+z)^\alpha = \sum_{i=0}^{\infty} C(\alpha, i) z^i$ which is known as the binomial theorem.

5.4 Exponential generating functions

If the terms of a sequence is growing too rapidly, i.e. the n-th term exceeds x^n for any $0 < x < 1$, then it may not converge. It is known that a sequence converges iff the sequence $|a_n|^{1/n}$ is bounded. Then it makes sense to divide the coefficients by a rapidly growing function like $n!$. For example, if we consider the generating function for the number of permutations of n *identical* objects

$$G(z) = 1 + \frac{p_1}{1!}z + \frac{p_2}{2!}z^2 \ldots \frac{p_i}{i!}z^i$$

where $p_i = P(i, i)$. Then $G(z) = e^z$. The number of permutations of r objects when selected out of (an infinite supply of) n kinds of objects is given by the exponential generating function (EGF)

$$\left(1 + \frac{p_1}{1!}z + \frac{p_2}{2!}z^2 \ldots\right)^n = e^{nx} = \sum_{r=0}^{\infty} \frac{n^r}{r!}x^r$$

Example5.4.1 : Let D_n denote the number of derangements of n objects. Then it can be shown that $D_n = (n-1)(D_{n-1}+D_{n-2})$. This can be rewritten as $D_n - nD_{n-1} = -(D_{n-1} - (n-2)D_{n-2}$. Iterating this, we obtain $D_n - nD_{n-1} = (-1)^{n-2}(D_2 - 2D_1)$. Using $D_2 = 1, D_1 = 0$, we obtain

$$D_n - nD_{n-1} = (-1)^{n-2} = (-1)^n.$$

Multiplying both sides by $\frac{x^n}{n!}$, and summing from $n = 2$ to ∞, we obtain

$$\sum_{n=2}^{\infty} \frac{D_n}{n!}x^n - \sum_{n=2}^{\infty} \frac{nD_{n-1}}{n!}x^n = \sum_{n=2}^{\infty} \frac{(-1)^n}{n!}x^n$$

If we let $D(x)$ represent the exponential generating function for derangements, after simplification, we get

$$D(x) - D_1 x - D_0 - x(D(x) - D_0) = e^{-x} - (1 - x)$$

or $D(x) = \frac{e^{-x}}{1-x}$.

5.5 Recurrences with two variables

For selecting r out of n distinct objects, we can write the familiar recurrence

$$C(n, r) = C(n-1, r-1) + C(n-1, r)$$

with boundary conditions $C(n, 0) = 1$ and $C(n, 1) = n$.

The general form of a linear recurrence with constant coefficients that has two indices is

$$C_{n,r}a_{n,r} + C_{n,r-1}a_{n,r-1} + \ldots C_{n-k,r}a_{n-k,r} \ldots C_{0,r}a_{0,r} + \ldots = f(n, r)$$

where $C_{i,j}$ are constants. We will use the technique of generating functions to extend the one variable method. Let

$$A_0(x) = a_{0,0} + a_{0,1}x + \ldots a_{0,r}x^r$$

$$A_1(x) = a_{1,0} + a_{1,1}x + \ldots a_{1,r}x^r$$

$$A_n(x) = a_{n,0} + a_{n,1}x + \ldots a_{n,r}x^r$$

Then we can define a generating function with $A_0(x), A_1(x)A_3(x)\ldots$ as the sequence - the new indeterminate can be chosen as y.

$$A_y(x) = A_0(x) + A_1(x)y + A_2(x)y^2 \ldots A_n(x)y^n$$

For the above example, we have

$$F_n(x) = C(n,0) + C(n,1)x + C(n,2)x^2 + \ldots C(n,r)x^r + \ldots$$

$$\sum_{r=0}^{\infty} C(n,r)x^r = \sum_{r=1}^{\infty} C(n-1,r-1)x^r + \sum_{r=0}^{\infty} C(n-1,r)x^r$$

$$F_n(x) - C(n,0) = xF_{n-1}(x) + F_{n-1}(x) - C(n-1,0)$$

$$F_n(x) = (1+x)F_{n-1}(x)$$

or $F_n(x) = (1+x)^n C(0,0) = (1+x)^n$ as expected.

5.6 Probability generating functions

The notion of generating functions have useful applications in the context of probability calculations also. Given a non-negative integer-valued discrete random variable X with $\Pr[X = k] = p_k$, the probability generating function (PGF) of X is given by

$$G_X(z) = \sum_{i=0}^{\infty} p_i z^i = p_0 + p_1 z + \ldots p_i z^i \ldots$$

This is also known as the z-transform of X and it is easily seen that $G_X(1) = 1 = \sum_i p_i$. The convergence of the PGF is an important issue for some calculations involving differentiation of the PGF. For example,

$$E[X] = \frac{dG_X(z)}{dz}|z = 1$$

The notion of *expectation of random variable* can be extended to function $f(X)$ of random variable X in the following way

$$E[f(X)] = \sum_i p_i f(X = i)$$

Therefore, PGF of X is the same as $E[z^X]$. A particularly useful quantity for a number of probabilistic calculations is the **Moment Generating Function** (MGF) defined as

$$M_X(\theta) = E[e^{X\theta}]$$

Since

$$e^{X\theta} = 1 + X\theta + \frac{X^2\theta^2}{2!} + \dots \frac{X^k\theta^k}{k!} \dots$$
$$M_X(\theta) = 1 + E[X]\theta + \dots \frac{E[X^k]\theta^k}{k!} \dots$$

from which $E[X^k]$ also known as *higher moments* can be calculated. There is also a very useful theorem known for *independent* random variables $Y_1, Y_2 \dots Y_t$. If $Y = Y_1 + Y_2 + \dots Y_t$, then

$$M_Y(\theta) = M_{Y_1}(\theta) \cdot M_{Y_2}(\theta) \cdot \dots M_{Y_t}(\theta)$$

i.e., the MGF of the sum of independent random variables is the product of the individual MGF's.

5.6.1 Probabilistic inequalities

In many applications, especially in the analysis of randomized algorithms, we want to guarantee correctness or running time. Suppose we have a bound on the expectation. Then the following inequality known as Markov's inequality can be used.

Markov's inequality

$$\Pr[X \geq kE[X]] \leq \frac{1}{k} \tag{5.6.1}$$

Unfortunately there is no symmetric result.

If we have knowledge of the second moment, then the following gives a stronger result

Chebychev's inequality

$$\Pr[(X - E[X])^2 \geq t] \leq \frac{\sigma^2}{t} \tag{5.6.2}$$

where σ is the variance, i.e. $E^2[X] - E[X^2]$.

With knowledge of higher moments, then we have the following inequality. If $X = \sum_i^n x_i$ is the sum of n mutually independent random variables where x_i is uniformly distributed in {-1 , +1 }, then for any $\delta > 0$,

Chernoff bounds

$$\Pr[X \geq \Delta] \leq e^{-\lambda\Delta} E[e^{\lambda X}] \tag{5.6.3}$$

If we choose $\lambda = \Delta/n$, the RHS becomes $e^{\Delta^2/2n}$.

5.7 Problem Set

1. Find a recurrence for the number of ways a frog can jump n stairs if each step covers either 1 or 2 or 3 stairs.

2. Find a recurrence for the number of n-digit binary sequences with no consecutive 1's. Repeat the same for ternary sequences.

3. Find a recurrence for the number of n digit ternary sequences in which no 2 appears anywhere to right of any 1.

4. Find a recurrence for te number of ways to pick k objects with repetition from n types.

5. Find a recurrence relation for the number of permutatins of the first n integers such taht each integer differs by one (except for teh first) from some integer to the left of it in the permuation.

6. Find a recurrence for computing the number of spanning trees in the "ladder" graph with n rungs ($2n$ vertices).

7. Gossip is spread among r people via telephone. Specifically, in a conversation between A and B, A tells B all the gossip he has heard and B does the same. Let a_r denote the number of calls among r people such taht the gossips will be known to everyone and write a recurrence for a_r.

8. Let a_r denote the number of partitions of a set of r elements. Show that

$$a_{r+1} = \sum_{i=0}^{r} C(r,i) a_i$$

where $a_0 = 1$.

9. Let a_r denote the number of subsets of the set $\{1, 2, \ldots r\}$ that do not contain two consecutive numbers. Determine a_r.

10. In predicting future discoveries of oil, the assumptin is that the teh amount discovered next year will be average of te amount discovered this year and the last year. Write a recurrence for a_n and solve it.

11. In a singles tournament, $2n$ players are paired off in n matches and $f(n)$ is the number of ways in which this is done. Write a recurrence for $f(n)$ and solve it.

12. If F_n is the n-th Fibonacci number, find a simple expression for $F_1 + F_2 + \ldots F_n$ which invoves F_p for only one p.

13. Consider a variation of the Tower of Hanoi problem where we have to move disks from A to B such that no disk can be moved directly from A to B. What is the minimum number of moves.

14. What is the number of distict spanning trees of complete graph ? (Two distinct spanning trees will have different edge sets).
Solution We will prove that it is n^{n-2}, $n \geq 2$, which was first proved by Cayley. We will first prove the following claim The number of spanning trees with degree sequence $(d_1, d_2 \ldots d_n) =$

$$\frac{(n-2)!}{d_1 - 1!d_2 - 1! \ldots d_n - 1!}$$

where the degree sequence corresponds to i-th vertex having degree d_i.

Proof: *basis* for n=2, it is 1.
Suppose it is true upto $m-1 \geq 2$. Given a tree on m nodes, we know that there is at least one vertex of degree one, say v_j and it is connected to vertex v_i. If we pluck out v_j, then we are left with a $m-1$ vertices and degree of v_i is $d_i - 1$. We can now apply the inductive hypothesis to the graph with $m-1$ nodes, i.e., $\frac{(m-3)!}{d_1-1!d_2-1!\ldots d_i-1!\ldots}$. Since the degree 1 node can be attached to any of the vertices, we have the same number of trees for each of the $m-1$ edges. We are in essence, looking at the degree 1 vertices attached to all possible nodes - $v_1 to v_{m-1}$ and by addition principle we can add then up. Notice that the degree sequnces are different in each case. If more than one vertex has degree 1 connected to v_j, note that it suffices to consider any one of them, since there is only one way that can be connected. Summing over all instances of d_i we obtain

$$\sum_{d_i \geq 1} \frac{(m-3)!}{(d_1-1)!(d_2-1!)\ldots(d_i-2!)..(d_j-1)\ldots}$$

Multiplying by numerator and denominator $d_i - 1$, we obtain

$$\frac{(m-2)! \cdot (d_i - 1)}{d_1 - 1!d_2 - 1! \ldots d_i - 1!..}.$$

Summing over all degree sequences

$$\sum_{d_i \geq 1} \frac{(n-3)!d_i - 1}{d_1 - 1!d_2 - 1! \dots d_i - 2!..(d_j - 1) \dots}$$

Since $\sum_i (d_i - 1) = 2(m-2) + 1 - (m-1) = m-2$, the induction proof is complete. $\square$

Now we add up the previous bounds over all degree sequences to obtain

$$\sum_{d_1, d_2 \geq 1, d_1 + d_2 \dots d_n = 2(n-2)} \frac{(n-3)!d_i - 1}{d_1 - 1!d_2 - 1! \dots d_i - 2!..(d_j - 1) \dots}$$

This is a multinomial which gives us n^{n-2}.

15. What is the average root-leaf distance of an oriented rooted tree with n nodes ?

Proof: The path length can be viewed as lengths of internal path I (concerning internal nodes) and external path E (pertaining to the leaf nodes). We can write $E = I + 2n$ where n is the number of internal nodes including the root node since there are two external nodes for every terminal internal node. We will make use of a two dimensional recurrence

$$B(w, z) = \sum_{n,p \geq 0} = \sum b_{n,p} w^p z^n$$

where $b_{n,p}$ is the number of binary trees with n nodes and internal path length p.
For example (by brute force calculation)

$$B(w, z) = 1 + z + 2wz^2 + (w^2 + 4w^3)z^3 + \dots$$

Clearly $B(1, z)$ = generating function for number of (oriented) trees with n nodes.

$$b_{n,p} = \sum_{k+l=n-1; r+s+n-1=p} b_{k,r} b_{l,s}$$

It follows that $zB^2(w, wz) = B_(wz) - 1$.
By taking the partial deriuvative wrt z we obtain

$$2zB(w, wz)(B_w(w, wz) + zB_z(w, wz)) = B_w(w, z)$$

Let H(z) is the generating function for the total internal path length with n nodes, then

$$H(z) = B_w(1, z) = \sum_i h_i z^n.$$

Moreover $H(z) = 2zB(z) = (H(z) + zB'(z).)$ Using the formula for $B(z)$ (Catalan numbers),

$$H(z) = \frac{1}{1-4z} - \frac{1}{z}\left(\frac{1-z}{sqrt 1 - 4z} - 1\right)$$

giving

$$h_n = 4^n - \frac{3n+1}{n+1}C(2n, n)$$

The average value of total internal path length is h_n/b_n and average value of path length of a node is h_n/nb_n. The asymptotic value of this is $\sqrt{\pi n} - 3 + O(1)$.
$\square$

Chapter 6

Modular Arithmetic

In this chatper, we will discuss some useful properties of numbers when calculations are done modulo n, where $n > 0$. In the context of computer science, n is usually a power of 2 since representation is binary.

6.1 Divisibility

Definition 6.1.1 An integer b is divisible by an integer a $(a \neq 0)$, if there is an integer x such that $b = ax$. This will be denoted by $a|b$.

We begin by formalising some elementary observations about integer division.

Theorem 6.1.2
1. *$a|b$ implies $a|bc$ for any integer c.*
2. *$a|b$ and $b|c$ implies $a|c$.*
3. *$a|b$ and $a|c$ implies $a|bx + cy$.*
4. *if $m \neq 0$ then $a|b \equiv ma|mb$.*

Theorem 6.1.3 ***(Divison Algorithm)*** *Given integers a and b with $a > 0$, there exist* **unique** *integers q and r such that $b = qa + r$, $0 \leq r < a$.*

Definition 6.1.4 The gcd of two numbers a and b is the largest among the common divisors of a and b. If this is 1 then a, b are *relatively prime*.

The following properties of $gcd(x, y)$ are known

Theorem 6.1.5
1. *If c is a common divisor of a, b, then $c|gcd(a, b)$.*
2. *$gcd(x, y) = \min\{ax + by\}$ where x, y are integers, such that $ax + by > 0$.*

3. $m \cdot gcd(a,b) = gcd(ma, mb)$.

4. *If* $gcd(a,m) = gcd(b,m) = 1$ *then* $gcd(ab,m) = 1$.

5. *If* $c|ab$ *and* $gcd(b,c) = 1$ *then* $c|a$.

6. $gcd(a,b) = gcd(a, b - qa)$*for any* q

The beginning of number theory goes back to Euclid's algorithm that exploited some of the properties of divisibility to compute the gcd of two integers.From property 6, it follows that to fing gcd of a and b we can find gcd of a and b-qa(repeatedly). If $a|b$,then clearly a is the gcd, so that can be used as a terminating case. Computing q can be done using the division algorithm which is how Euclid's algorithm works. In addition, it also computes numbers x and y such that gcd $= ax + by$. For this, we maintain an invariant that $ax_i + by_i = r_i$ where r_i is the remainder in the i-th iteration with initial values $x_0 = 1$ and $y_0 = 0$. And this is what is known as Extended Euclid's algorithm. The correctness of the algorithm follows from induction.

Prime numbers (with no divisors other than 1 and the number itself) are extremely important in number theory.

Theorem 6.1.6 ***(Fundamental Theorem of Arithmetic)***
Every positive integer can be expressed as product of primes and this factorization is unique except for the order of the prime factors.

Proof: We know that if $p|q_1 q_2$ where p is prime then either $p|a$ or $p|b$ or both. □

The fact that number of primes is infinite was given in an elegant proof of Euclid. Extending his argument it can be shown that there are arbitrary gaps between two primes. The *prime number theorem* says that among the first n integers there are very nearly $\frac{n}{\ln n}$ prime numbers.

6.2 Congruences

Definition 6.2.1 If an integer m, not zero, divides the difference $a - b$, we say that a is **congruent** to b **modulo** m and is denoted by $a \equiv b(modm)$.

(Since $m|(a - b)$ is equivalent to $-m|(a - b)$, we will always assume that $m > 0$.) The following properties follow from the definition.

Theorem 6.2.2 1. $a \equiv b(modm)$ *is the same as* $a - b \equiv 0(modm)$.

2. $a \equiv b(modm)$ *and* $b \equiv c(modm)$ *implies* $a \equiv c(modm)$. *(transitive - infact* $\equiv (modm)$ *is an equivalence relation).*

3. *If* $a \equiv b(modm)$ *and* $c \equiv d(modm)$ *then* $ax + cy \equiv bx + dy(modm)$

4. *If* $a \equiv b(modm)$ *and* $c \equiv d(modm)$*, then* $ac \equiv bd(modm)$

5. *If* $a \equiv b(modm)$ *and* $d|m$*,* $d > 0$*, then* $a \equiv b(modd)$*.*

The degree of a polynomial (with integral coefficients) modulo m is the highest power of x for which the coeffient is non-zero modulo m. For $f(x) = a_0x^n + a_1x^{n-1} + \ldots a_n$, if $f(u) \equiv 0(modm)$ then we say that u is a solution of the *congruence* $f(x) \equiv 0(modm)$. It is known that

Theorem 6.2.3 *If* $a \equiv b(modm)$*, then* $f(a) \equiv f(b)(modm)$

An important problem is the solution of congruences and in particular linear (degree 1) congruence. Any such congruence has the form

$$ax \equiv b(modm)$$

For the special case that $gcd(a, m) = 1$, we have a solution $x_1 = a^{\phi(m)-1}b$, where $\phi(m)$ is the *totient* function (defined by Euler). It is the number of integers less than m that are relatively prime to m (if m is prime then $\phi(m) = m - 1$). This follows from the following theorem of Euler.

Theorem 6.2.4 *If* $gcd(a, m) = 1$*, then* $a^{\phi(m)} \equiv 1(modm)$*.*

Another way of viewing the solution is to multiply both sides by a number a^{-1} such that $a \cdot a^{-1} \equiv 1(modm)$. We have the following equivalent of cancellation laws

Theorem 6.2.5

1. *If* $ax \equiv ay(modm)$ *and* $gcd(a, m) \equiv 1(modm)$ *then* $x \equiv y(modm)$*.*

2. $ax \equiv ay(modm)$ *iff* $x \equiv y(mod\frac{m}{gcd(a,m)})$*. (generalization)*

The remaining solutions (when $gcd(a, m) = 1$) are of the form $x_1 + jm$ for any integer j. In other words there is a unique solution modulo m. For the other case (when a and m are not relatively prime), the solutions are described by the following theorem.

Theorem 6.2.6 *Let* $g = gcd(a, m)$*. Then* $ax \equiv b(modm)$ *has no solutions if* g *does not divide* b*. If* $g|b$*, it has* g *solutions* $x \equiv (b/g)x_0 + t(m/g),\ t = 0, 1 \ldots g - 1$*, where* x_0 *is any solution of* $(a/g)x \equiv 1(mod(m/g))$*.*

Algorithmically, in both cases, we can use the (extended) Euclid's algorithm to compute x_1 or x_0.

An alternate method is to solve a set of simultaneous congruences by factorising $m = \prod_{i=1}^{k} p_i^{e_i} = \prod_{i=1}^{k} m_i$ where $m_i = p_i^{e_i}$. Since m_i are relatively prime in pairs, it can be shown that solving the congruence $ax \equiv b(mod m)$ is the same as solving the congruences $ax \equiv b(mod m_i)$ simultaneously for all i. Suppose the individual congruences have solutions

$$ax_i \equiv b(mod m_i)$$

Then these can be combined using a result called *Chinese Remaindering Theorem.*

Theorem 6.2.7 *The common solution is given by*

$$x_0 = \sum_{j=1}^{k} \frac{m}{m_j} b_j x_j$$

where b_j is given by solutions to $(m/m_j)b_j \equiv 1(mod m_j)$.

6.3 Problem Set

1. Prove that an integer is divisible by 9 iff the sum of its digits is divisible by 9.
2. Prove that an integer is divisible by 11 iff the difference between the sum of the digits in the odd places and the sum of the digits in the even places is divisible by 11.
3. Give an easy test for divisibility by 7.
4. If p is a prime > 5, then prove that it divides infinitely many of the integers 9,99,999,9999 ...
5. For what integer values of n, $2^n + 1$ is divisible by 3 ?
6. Is any prime of the form 3k+1 is of the form 6k+1 ?Justify.
7. Prove that if $2^n + 1$ is a prime then n is power of 2 and if $2^n - 1$ is a prime then n is a prime.
8. Prove that there can be arbitrary gaps between two consecutive primes.
9. Given any positive integer k,prove that there are k-consecutive integers divisible by a square> 1.
10. Given $n > 2$,prove that there exists a prime p such that $n < p < n!$.
11. A positive integer is said to be a square-free if it is product of distinct primes.What is the largest number of consecutive square-free positive integers ?

12. Find the gcd of 2613 and 2171 by Euclidean algorithm.

13. Find two integers x and y such that gcd(841,160)=841x+160y.Are these x and y unique?

14. Find x and y such that

    (a) $243x + 198y = 9$

    (b) $71x - 50y = 1$

    (c) $6x + 10y + 15z = 1$

15. For what integer values of d,exist two integers x and y such that $21x+35y = d$?

16. Find all the solutions to the congruences

    (a) $13x \equiv 4(mod25)$

    (b) $5x \equiv 2(mod26)$

    (c) $9x \equiv 12(mod15)$

    (d) $6x \equiv 3(mod210$

17. Solve $17x \equiv 9mod276$,by using Chinese Remainder Theorem.

18. Find an integer x such that$19x \equiv 103mod900$ and$10x \equiv 511$ mod 841.

19. Find all integers that give the remainder 1,2,4 when divided by 3,5,4 respectively.

20. When eggs in a basket are taken out 2,5,9,23 at a time,there remain respectively 1,3,7,19 eggs.Find the smallest number of eggs in the basket.

Part II
Applications

Chapter 7

Application of probability: Information and Coding theory

7.1 Quantifying information

Information is central to solving problems and designing algorithms. The efficiency of algorithms is basically defined by how quickly it can gather information about the input. Although the term *information* is mostly used in an abstract descriptive sense, there exists a formal characterization of the term which was first defined in the context of designing and transmitting codes.

The amount of *information* is identified with the element of uncertainity. A statement doesn't convey any information if we already knew (or deduced) the contents. A generalization of this is associating probability $\Pr(E)$ of an event E with the information $I(E)$ it conveys.

1. If $\Pr(E)$ is 1, then (it is certain) $I(E) = 0$.
2. If $\Pr(E) = 0$ then $I(E) = \infty$.
3. If $\Pr(E_1) \leq \Pr(E_2)$, then $I(E_1) \geq I(E_2)$.
4. If E_1 and E_2 are independent, i.e., $\Pr(E_1 \cap E_2) = \Pr(E_1) \cdot \Pr(E_2)$, then knowing about one doesn't divulge any information about the other. So $I(E_1 \cap E_2)$ should be $I(E_1) + I(E_2)$.

Therefore $I(E_1) = \log_b(\frac{1}{\Pr(E_1)})$ is a natural choice that satisfies all the above properties and the base b is chosen to be 2 because of technical convenience. A very useful quantity is the expected amount of uncertainity called **entropy**. For example, a coin that has a probability p of heads and $q(= 1 - p)$of tails has entropy

$$p \log(1/p) + q \log(1/q)$$

and it can be verified that this is maximised when $p = q = 1/2$ and equals 0 when the outcome is certain ($p = 0$ or $p = 1$).

For a general probability space $\Omega = (E_1, E_2 \dots E_n)$, the entropy is defined as

$$\sum_{i=1}^{n} = p_i \log(1/p_i), \quad p_i = \Pr(E_i)$$

The entropy of a discrete random variable X is similarly defined

$$H(X) = \sum_i \Pr(X = i) \cdot \log(1/\Pr(X = i)$$

One can verify easily that entropy is maximised when all events are equally likely (maximum uncertainity).

7.2 Codes

Given a finite set of symbols X, a *coding* of this is a function from X to strings finite set of strings over a (usually) small set of alphabet Σ. For example $X = \{1, 2, 3, 4\}$ can be mapped to $\{0, 10, 110, 111\}$ over alphabet $\Sigma = \{0, 1\}$. The length of a code is the number of symbols in the string. A sequence of symbols from X is encoded as the concatenation of the codes for the sequence of the symbols. For example 132 will be encoded as 011010. The set of all strings over a finite alphabet S will be denoted by S^+

A code is *uniquely decipherable* if the mapping $X^+ \to \Sigma^+$ is a 1-1 function, i.e. every string from X^+ is mapped to a unique string in Σ^+. In other words, these strings can be decoded unambiguously.

A code is called a **prefix code**, if no codeword is a prefix of another codeword. Clearly a prefix code is uniquely decipherable. The above example is a prefix code. The following result characterizes the uniquely decipherable codes very elegantly.

Theorem 7.2.1 (Kraft-McMillan) *For any uniquely decipherable code over* $\{0, 1\}$*, the lengths of the codewords must satisfy*

$$\sum_i 2^{-l_i} \leq 1$$

where l_i is the length of the i-th codeword. Moreover, if a given set of codewords satisfy the above inequality, then there exists a prefix code with these codeword lengths.

We will only prove the first part for prefix codes. We can represent the prefix code using a trie where each leaf node corresponds to a codeword. If the maximum length of a codeword is h, then it follows that the number of leaves of a complete binary tree of depth h is greater than the number of leaf nodes of the subtrees rooted at each of the nodes corresponding to the codewords. In particular,

$$\sum_i 2^{h-l_i} \leq 2^h$$

which when divided by 2^h gives us the required inequality.

If we are given the probability of occurence of each symbol in X, then the expected length of a code is defined as

$$\sum_{x \in X} \Pr(x) \cdot l(x) \tag{7.2.1}$$

where $l(x)$ is the length of the codeword of x. An important problem in designing codes is to minimize the expected codelength. Intuitively, we should choose a smaller codeword for a symbol that has a larger probability. Also it should satisfy the previous theorem, since we are interested in U.D. code.

Let $J(l_1, l_2 \ldots l_n) = \sum_i p_i \cdot l_i + \lambda \sum_i 2^{-l_i}$ where λ denotes Langrange's multiplier. Differentiating with respect to each l_i, we get the following equations for each i

$$p_i - \lambda 2^{-l_i} \ln 2 = 0$$

This yields $2^{-l_i} = p_i/(\lambda ln2)$ and using $\sum_i 2^{-l_i} = 1$, we obtain $\lambda = 1/ln2$, given $\sum_i p_i = 1$. Substituting this value in the previous equation, we have an interesting result, namely $2^{-l_i} = p_i$ or $l_i = \lceil -\log_2 p_i \rceil$. Without the integrality condition it satisfies KraftMcMillan's inequality and more importantly the expected code length of a code is lower bounded by the the entropy. Therefore, the entropy or the quantity of information is an important determinant for minimizing average codelength or more generally determines if data can be compressed.

There is a fairly simple and efficient algorithm for designing optimal code, namely **Huffman code**.

7.2.1 Huffman code

If we have prior knowledge of the probability of the frequency of all symbols, then we can design codes that are optimal in the sense of equation 7.2.1. If p_i is the probability of occurence of a_i, then choose the smallest two symblos p_is,say p_j and p_k. Also associate a probability $p_k + p_j$ to it. Make them leaf nodes of a new node b_n and assign a probability $p_i + p_k$ to b_n . Repeat his procedure with the symbols $\{a_1, a_2 \ldots a_n\}$ and we termnate when there is only node. By assigning labels 0 abd 1 to the left and right branch, we obtain codes for all the symbols a_i. We are leaving the proof of correctness to the reader as an exercise.

Note Although Huffman coding is near optimal, it does not adapt to local changes, and consequenly the Ziv-Lempel scheme is more compact which uses a sliding window method to encode, Consider a scenario where the text consists of a large number of a's followed by b's etc. Clearly the adaptive scheme will be superior to Huffman although Huffman's encoding is globally optimal.

7.3 Relative information

If X and Y are two random variables with pdf $\Pr(X = x_i) = p_i$ and $\Pr(Y = y_j) = q_j$, then the **joint entropy** of X, Y

$$H(X,Y) = \sum_{i,j} \Pr(X = x_i, Y = y_j) \log \frac{1}{\Pr(X = x_i, Y = y_j)}$$

It follows from the above definition that if X, Y are independent, i.e., $\Pr(X = x_i, Y = y_j) = \Pr(X = x_i)\Pr(Y = y_j)$ then the reader can verify easily that

$$H(X,Y) = H(X) + H(Y)$$

The **marginal entropy** of X is the entropy of the distribution $\Pr(X|Y = y_j)$, i.e.,

$$H(X|Y = y_j) = \sum_{i} \Pr(X = x_i|Y = y_j) \log \frac{1}{\Pr(X = x_i|Y = y_j)}$$

Can it happen that $H(X|Y = y_j) > H(X)$?

The **conditional entropy** of X given Y is the expectation of the marginal entropy over all values of Y

$$H(X|Y) = E[H(X|Y = y_j] = \sum_{j} \Pr(Y = y_j) \left[\sum_{i} \Pr(X = x_i|Y = y_j) \log \frac{1}{\Pr(X = x_i|Y = y_j)} \right]$$

Theorem 7.3.1 (Chain rule of entropy)

$$H(X,Y) = H(X) + H(Y|X) = H(Y) + H(X|Y)$$

The **relative entropy** or Kullback-Leibler distance between two distributions p, q of a random variable X

$$D_{KL}(p||q) = \sum_{i} p_i \log \frac{(p_i)}{(q_i)}$$

so it is not symmetric.

Theorem 7.3.2 (Gibb's inequality)

$$D_{KL}(p||q) \geq 0$$

Proof We will apply Jensen's inequality which is given by $E[f(X)] \leq f(E(X)$ for any random variable X and *convex* function f[1] The inequality is reversed for concave functions like log.

Let $X = p_i/q_i$. Then from Jensen's inequality and the fact that log is a concave function we obtain

$$\sum_i p_i \log(q_i/p_i) \leq \log(\sum_i (p_i \cdot q_i/p_i))$$

By negating both sides and noting that the RHS is 0, we obtain

$$\sum_i p_i \log \frac{(p_i)}{(q_i)} \geq= 0.$$

An useful application of Gibb's ineuality is in the proof of the following useful inequality

Corollary 7.3.3

$$H(X|Y) \leq H(X)$$

i.e. uncertainity can only decrease with more information. Therefore

$$H(X,Y) \leq H(X) + H(Y)$$

where equality holds if X and Y independent.

Proof We can show that $H(X) - H(X|Y) \geq 0$ using the previous definitions. The following is a very important result that we will exploit repeatedly.

Theorem 7.3.4 *It follows that if $Z = f(X_1, X_2 \ldots X_n)$ then*

$$H(Z) \leq H(X_1) + H(X_2) \ldots H(X_n)$$

Proof:

$$H(f(X)) = -\sum_a \Pr[f(X) = a] \log \Pr[f(X) = a]$$

$$H(X) = -\sum_x \Pr[X = x] \log \Pr[X = x] = -\sum_a \sum_{x: f(x)=a} \Pr[X = x] \log \Pr[X = x]$$

[1] A function $f(x)$ is convex if for all x_1, x_2 and $0 \leq \lambda \leq 1$, $f(\lambda x_1 + (1-\lambda x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$.

Now compare term by term:

$-\Pr[f(X)=a]\log\Pr[f(X)=a]$ versus $-\sum_{x:f(x)=a}\Pr[X=x]\log\Pr[X=x]$.

Using the concavity of the function $\log x$:

$$\sum_{x:f(x)=a}(\Pr[X=x]/\Pr[f(X)=a])\log\Pr[X=x])$$

$$\leq \sum_a \log(\sum_{x:f(x)=a}(\Pr[X=x]/\Pr[f(X)=a])\Pr[X=x])$$

$$\leq \log(\sum_{x:f(x)=a}\Pr[X=x]) = \log\Pr[f(X)=a] \quad \text{since } \Pr[X=x]/\Pr[f(X)=a]\leq 1.$$

By negating both sides of each term and then summing over all terms, we obtain the desired result.

□

Example 7.3.5 : [**fake coin problem**] Suppose we are given n coins and one of them is counterfeit. We are told that counterfeit has a different weight and we are allowed to use a simple balance. How many weighings are necessary ?
For instance if it is 3 coins, then we can identify using at most 2 weighings. In the beginning, the entropy is $H(1/n.1/n\ldots 1/n)\geq \log n$. Each weighing can have 3 outcomes - right $>$ / $<$ left, or right equals left. Hence we can get atmost $\log 3$ bits of information. This implies that at least $\frac{\log n}{\log 3}$ weighings are necessary.
We leave it as an exercise about how close we can come to this figure to actually identify the fake coin.

Our next example is the well known result of sorting.

Example 7.3.6 : If we want to sort n elements, i.e., permute them into a non-decresing order. If all permutations are equally likely, then the information content is $\log n!$. In a comparison, the number of outcomes is two implying that the entropy is $\log 2 \; = 1$. If the minimum number of comparisons is m, then $H(f(X_1, X_2 \ldots X_m)) \geq \log(n!)$ where X_i are random variables corresponding to each comparison. From theorem 7.3.4 $H(X_1) + H(X_1 \ldots H(X_m) \geq \log(n!)$ which implies $m = \Omega(n \log n)$.

Chapter 8

Sorting and Searching

8.1 Skip Lists - an alternative to balaced BST

Skip-list is a data structure introduced by Pugh [?] as an alternative to balanced binary search trees for handling dictionary operations on ordered lists. The underlying idea is to substitute complex book-keeping information used for maintaining balance conditions for binary trees by random sampling techniques. It has been shown by Pugh [?] that, given access to random bits, the *expected* search time in a skip-list of n elements is $O(\log n)$ [1] which compares very favourably with balanced binary trees. Moreover, the procedures for insertion and deletion are very simple which makes this data-structure a very attractive alternative to the balanced binary trees.

Since the search time is a stochastic variable (because of the use of randomization), it is of considerable interest to determine the bounds on the tails of its distribution. Often, it is crucial to know the behavior for any individual access rather than a chain of operations since it is more closely related to the real-time response.

8.1.1 Review of Skip-lists

We briefly review the basic data-structure proposed by Pugh. This data-structure is maintained as a hierarchy of sorted linked-lists. The bottom-most level is the entire set of keys S. We denote the linked list at level i from the bottom as L_i and let $|L_i| = N_i$. By definition $L_0 = S$ and $|L_0| = n$. For all $0 \leq i$, $L_i \subset L_{i-1}$ and the topmost level, say level k has constant number of elements. Moreover, correspondences are maintained between common elements of lists L_i and L_{i-1}. For a key with value E, for each level i, we denote by T_i a tuple (l_i, r_i) such that $l_i \leq E \leq r_i$ and $l_i, r_i \in L_i$. We call this tuple *straddling pair* (of E) in level i.

[1] Note that all logarithms are to base 2 unless otherwise mentioned.

The search begins from the topmost level L_k where T_k can be determined in constant time. If $l_k = E$ or $r_k = E$ then the search is successful else we recursively search among the elements $[l_k, r_k] \bigcap L_0$. Here $[l_k, r_k]$ denotes the closed interval bound by l_k and r_k. This is done by searching the elements of L_{k-1} which are bounded by l_k and r_k. Since both $l_k, r_k \in L_{k-1}$, the *descendence* from level k to $k-1$ is easily achieved in $O(1)$ time. In general, at any level i we determine the tuple T_i by walking through a portion of the list L_i. If l_i or r_i equals E then we are done else we repeat this procedure by *descending* to level $i-1$.

In other words, we refine the search progressively until we find an element in S equal to E or we terminate when we have determined (l_0, r_0). This procedure can also be viewed as searching in a tree that has variable degree (not necessarily two as in binary tree).

Of course, to be able to analyze this algorithm, one has to specify how the lists L_i are constructed and how they are dynamically maintained under deletions and additions. Very roughly, the idea is to have elements in i-th level point to approximately 2^i nodes ahead (in S) so that the number of levels is approximately $O(\log n)$. The time spent at each level i depends on $[l_{i+1}, r_{i+1}] \bigcap L_i$ and hence the objective is to keep this small. To achieve these conditions on-line, Pugh [**?**] uses the following elegant method. The nodes from the bottom-most layer (level 0) are chosen with probability p (for the purpose of our discussion we shall assume $p = 0.5$) to be in the first level. Subsequently at any level i, the nodes of level i are chosen to be in level $i+1$ independently with probability p and at any level we maintain a simple linked list where the elements are in sorted order. If $p = 0.5$, then it is not difficult to verify that for a list of size n, the *expected* number of elements in level i is approximately $n/2^i$ and are spaced about 2^i elements apart. The expected number of levels is clearly $O(\log n)$, (when we have just a trivial length list) and the expected space requirement is $O(n)$.

To insert an element, we first locate its position using the search strategy described previously. Note that a byproduct of the search algorithm are all the T_i's. At level 0, we choose it with probability p to be in level L_1. If it is selected, we insert it in the proper position (which can be trivially done from the knowledge of T_1), update the pointers and repeat this process from the present level. Deletion is very similar and it can be readily verified that deletion and insertion have the same asymptotic run time as the search operation. So we shall focus on this operation.

8.1.2 Analysis

To analyze the run-time of the search procedure, we look at it backwards, i.e., retrace the path from level 0. The search time is clearly the length of the path (number of links) traversed over all the levels. So one can count the number of links one traverses before climbing up a level. In other words the expected search time can be expressed in the following recurrence (from [**?**])

$$C(k) = (1-p)(1 + C(k)) + p(1 + C(k-1))$$

where C(k) is the expected cost for climbing k levels. From the boundary condition C(0) $= 0$, one readily obtains $C(k) = k/p$. For $k = O(\log n)$, this is $O(\log n)$. The recurrence captures the crux of the method in the following manner. At any node of a given level, we climb up if this node has been chosen to be in the next level or else we add one to the cost of the present level. The probability of this event (climbing up a level) is p which we consider to be a success event. Now the entire search procedure can be viewed in the following alternate manner. We are tossing a coin which turns up heads with probability p - how many times should we toss to come up with $O(\log n)$ heads ? Each head corresponds to the event of climbing up one level in the data structure and the total number of tosses is the cost of the search algorithm. We are done when we have climbed up $O(\log n)$ levels (there is some technicality about the number of levels being $O(\log n)$ but that will be addressed later). The number of heads obtained by tossing a coin N times is given by a Binomial random variable X with parameters N and p. Using Chernoff bounds from Theorem **??**, for $N = 15 \log n$ and $p = 0.5$, $\Pr[X \leq 1.5 \log n] \leq 1/n^2$ (using $\epsilon = 9/10$ in equation 1). Using appropriate constants, we can get rapidly decreasing probabilities of the form $\Pr[X \leq c \log n] \leq 1/n^{\alpha}$ for $c, \alpha > 0$ and α increases with c. These constants can be fine tuned although we shall not bother with such an exercise here.

We thus state the following lemma.

Lemma 8.1.1 *The probability that access time for a fixed element in a skip-list data structure of length n exceeds $c \log n$ steps is less than $O(1/n^2)$ for an appropriate constant $c > 1$.*

Proof We compute the probability of obtaining fewer than k (the number of levels in the data-structure) heads when we toss a fair coin ($p = 1/2$) $c \log n$ times for some fixed constant $c > 1$. That is, we compute the probability that our search procedure exceeds $c \log n$ steps. Recall that each head is equivalent to climbing up one level and we are done when we have climbed k levels. To bound the number of levels, it is easy to see that the probability that any element of S appears in level i is at most $1/2^i$, i.e. it has turned up i consecutive heads. So the probability that any fixed element appears in level $3 \log n$ is at most $1/n^3$. The probability that $k > 3 \log n$ is the probability that at least one element of S appears in $L_{3 \log n}$. This is clearly at most n times the probability that any fixed element survives and hence probability of k exceeding $3 \log n$ is less than $1/n^2$.

Given that $k \leq 3 \log n$ we choose a value of c, say c_0 (to be plugged into equation 1 of Chernoff bounds) such that the probability of obtaining fewer than $3 \log n$ heads in

$c_0 \log n$ tosses is less than $1/n^2$. The search algorithm for a fixed key exceeds $c_0 \log n$ steps if one of the above events fail; either the number of levels exceeds $3 \log n$ or we get fewer than $3 \log n$ heads from $c_0 \log n$ tosses. This is clearly the summation of the failure probabilities of the individual events which is $O(1/n^2)$. □.

Theorem 8.1.2 *The probability that the access time for any arbitrary element in skip-list exceeds $O(\log n)$ is less than $1/n^\alpha$ for any fixed $\alpha > 0$.*

Proof: A list of n elements induces $n + 1$ intervals. From the previous lemma, the probability P that the search time for a fixed element exceeding $c \log n$ is less than $1/n^2$. Note that all elements in a fixed interval $[l_0, r_0]$ follow the same path in the data-structure. It follows that for any interval the probability of the access time exceeding $O(\log n)$ is n times P. As mentioned before, the constants can be chosen appropriately to achieve this. □

It is possible to obtain even tighter bounds on the space requirement for a skip list of n elements. From Pugh [**?**] it is known that the expected space is $O(n)$. Moreover it is clear that it does not exceed $O(n \log n)$ with probability $1 - 1/n^2$ (no element survives more than $O(\log n)$ levels with this probability from the previous lemma). One can obtain a much stronger bound by viewing the entire skip list structure as a stochastic experiment each node corresponds to a Bernoulli trial that turns up heads (similar to obtaining the obtaining the query bound). Each element is replicated till the the trial turns up tails. Since there are n elements, the number of nodes corresponds to the number of Bernoulli trials required to obtain n tails. This is a negative binomial distribution and one can use Chernoff bounds (Theorem **??**) directly to obtain the following result.

Theorem 8.1.3 *For any constant $\alpha > 0$, the probability of the space exceeding $2n + \alpha \cdot n$, is less than* $\exp^{\Omega(-\alpha^2 n)}$.

8.2 Trieps : Randomized Search Trees

The class of binary (dynamic) search trees is perhaps the first introduction to non-trivial data-structure in computer science. However, the update operations, although asymptotically very fast are not the easiest to remember. The rules for *rotations* and the *double-rotations* of the AVL trees, the splitting/joining in B-trees and the color-changes of red-black trees are often complex, as well as their correctness proofs. The *Randomized Search Trees* (also known as randomized trieps) provide a practical alternative to the Balanced BST. We still rely on rotations, but no explicit balancing rules are used. Instead we rely on the magical properties of random numbers.

The Randomized Search Tree (RST) is a binary tree that has the keys in an in-order ordering. In addition, each element is assigned a priority (Wlog, the priorities are unique) and the nodes of the tree are heap-ordered based on the priorities. Simultaneously, the key values follow in-order numbering. It is not difficult to see that for a given assignment of priorities, there is exactly one tree. If the priorities are assigned randomly in the range $[1, N]$ for N nodes, the expected height of the tree is *small*. This is the crux of the following analysis of the performance of the RSTs.

Let us first look at the way search time using a technique known as *backward analysis*. For that we (hypothetically) insert the N elements in a decreasing order of their *priorities* and then count the number of elements that Q can *see* during the course of their insertions. This method (of assigning the random numbers on-line) makes arguments easier and the reader must convince himself that it doesn't affect the final results. Q can *see an element* N_i if there are no previously inserted elements in between.

Lemma 8.2.1 *The tree constructed by inserting the nodes in order of their priorities (highest priority is the root) is the same as the tree constructed on-line.*

Lemma 8.2.2 *The number of nodes Q sees is exactly the number of comparisons performed for searching Q. In fact, the order in which it sees corresponds to the search path of Q.*

Theorem 8.2.3 *The expected length of search path in RST is $O(H_N)$ where H_N is the N-th harmonic number.*

In the spirit of backward analysis, we pretend that the tree-construction is being reversed, i.e. nodes are being deleted starting from the last node. In the forward direction, we would count the expected number of nodes that Q sees. In the reverse direction, it is the number of times Q's visibility changes (convince yourself that these notions are identical). Let X_i be a Bernoulli rv that is 1 if Q sees N_i (in the forward direction) or conversely Q's visibility changes when N_i is deleted in the reverse sequence. Let X be the length of the search path.

$$X = \sum X_i$$

$$E[X] = E[\sum X_i] = \sum E[X_i]$$

We claim that $E[X_i] = \frac{2}{i}$. Note that the expectation of a Bernoulli variable is the probability that it is 1. We are computing this probability over all permutations of N being equally likely. In other words, if we consider a prefix of length i, all subsets of size i are equally likely. Let us find the probability that $X_i = 1$, *conditioned* on a *fixed* subset $N^i \subset N$. Unconditioning is easy if probability that $X_i = 1$ does not depend on N^i itself. So, given a fixed N^i, all N^{i-1} are equally likely, so the probability that $X_i = 1$ is the probability that one of the (maximum two) neighboring elements was removed in the reverse direction. The probability of that is less than $\frac{2}{i}$ which is independent of any specific N^i. So, the unconditional probability is the same as conditional probability - hence $E[X_i] = \frac{2}{i}$. The theorem follows as $\sum_i E[X_i] = 2\sum_i \frac{1}{i} = O(H_N)$.

The X_i's defined in the previous proof are independent but not identical. So, we can apply Chernoff-Hoeffding bounds to to obtain strong tail-estimates for deviation from the expected bound. From Theorem **??** , it follows that

Theorem 8.2.4 *The probability that the search time exceeds* $2\log n$ *comparisons in a randomized triep is less than* $O(1/n)$.

A similar technique can be used for counting the number of rotations required for RST during insertion and deletions. Backward analysis is a very elegant technique for analyzing randomized algorithms, in particular in a paradigm called randomized incremental construction.

8.3 Lower bounds for searching and sorting

We often take for granted that the best time to search in a set of N elements is $O(\log N)$, namely. by using a binary search kind of procedure. Of course we have seen that hashing allows us to search in $O(1)$ expected time for any subset of the universe. We will now give a formal argument that when the uiverse is *very* large, $\Omega(\log n)$ probes are indeed necessary.

For this we will make use of a Ramsey-kind of theorem that generalises it to more than two colours.

Theorem 8.3.1 *For any* k, t, r, *tere exists a finite number* $R(k, r, t)$ *such that for a set* S, $|S| \geq R(k, r, t)$, *if we colour the family of* r *element subsets of* S, *then there exists a set* $k \geq r$ *such that all its* r*-element subsets have the same colour.*

By choosing $k = 2n - 1, r = n\ .t = n!$, we conclude that all n-element subsets of 2n elements with the same order type have the same colour (which denotes an order type). We shall now show that given a fixed ordered type we will need at least $\log n$ comparisons for searching in a table of n elements. Let $f(n.m)$ denote the minimum number of probes (usually comarisons) reqired to search for an element in a **sorted** table of size n and an universe of size m. In particular we will show it for $m = 2n-1$. *Basis* n = 2, m= 3 . we can show by case analysis that at least two probes are required. Let it be true for all $n < n_o$ (and $m = 2n - 1$. For $n = n_o$. Suppose the first probe is at position p, $p \leq \lceil n_o/2 \rceil$. Let the element be p. So the key can be anywhere between $\lceil n_o/2 \rceil$ and n_o. Therefore after one probe, the table can contain any of the $p+1, \ldots 2 \cdot n_o - 1$ elements in the remaining $n_o - p$ positions. Therefore the inductive hypothesis can be invoked giving at least $1 + \log(\lfloor n_o/2 \rfloor + 1)$ probes.

Lemma 8.3.2 $f(2,3) \geq 2$

Chapter 9

Universal Hashing

9.1 Notations

- Universe : $\mathcal{U}$
- Set of elements : S also $|S| = n$
- Hash locations : $\{0, 1, ..., m-1\}$ usually, $n \geq m$

9.2 Collision

If $x, y \in \mathcal{U}$ are mapped to the same location by a hash function h.

$$\begin{aligned}\delta_h(x,y) &= \begin{cases} 1 & : \quad h(x) = h(y), x \neq y \\ 0 & : \quad \text{otherwise} \end{cases} \\ \delta_h(x,S) &= \sum_{y \in S} \delta_h(x,y)\end{aligned}$$

Hash by chaining: The more the collision the worse the performance. Look at a sequnce $O_1(x_2), O_2(x_2), ..., O_n(x_n)$ where $O_i \in \{\text{Insert, Delete, Search}\}$ and $x_i \in \mathcal{U}$

Let us make the following assumptions

1. $|h^{-1}(i)| = |h^{-1}(i')|$ where $i, i' \in \{0, 1, ..., m-1\}$
2. In the sequence, x_i can be any element of $\mathcal{U}$ with equal probability.

Claim: Total expected cost $= O((1+\beta)n)$ where $\beta = \frac{n}{m}$ (load factor).

Proof: Expected cost of $(k+1)$th operation = expected number of elements in location $\leq 1 + k(\frac{1}{m})$ assuming all the previous operations were Insert.

So total expected cost $\leq \sum_{k=1}^{n} 1 + \frac{k}{m} = n + \frac{n(n+1)}{2m} = (1+\frac{\beta}{2})n$. This is *worst* case over *operations* but not over *elements*. □

9.3 Universal Hash Functions

Definition 9.3.1 A collection $H \subset \{h | h : [0...N-1] \to [0...m-1]\}$ is c-universal if for all $x, y \in [0...N-1]$,

$$|\{h | h \in H \text{ and } h(x) = h(y)\}| \leq c\frac{|H|}{m}$$

for some small constant c. Roughly $\sum_h \delta_h(x,y) \leq c\frac{|H|}{m}$

Claim:

$$\frac{1}{|H|}\sum_{h\in H} 1 + \delta_h(x,S) \leq 1 + c\frac{n}{m}$$

where $|S| = n$.

Proof: LHS

$$\begin{aligned}
&= \frac{1}{|H|}\sum_{h\in H} 1 + \frac{1}{|H|}\sum_{h}\sum_{y\in S} \\
&= 1 + \frac{1}{|H|}\sum_{y}\sum_{h}\delta_h(x,y) \\
&\leq 1 + \frac{1}{|H|}\sum_{y} c\frac{|H|}{m} \\
&= 1 + \frac{c}{m}n
\end{aligned}$$

So expected cost of n operation $= \sum(1 + \frac{ci}{m}) \leq (1 + c\beta)n$ □

9.4 Example of a Universal Hash function

$H' : h_{a,b}; h_{ab}(x) \to ((ax+b) \bmod N) \bmod m$ where $a, b \in 0...N-1$ (N is prime).

If $h_{ab}(x) = h_{ab}(y)$ then for some $q \in [0...m-1]$and $n, s \in [0...\frac{N-1}{m}]$

$$\begin{aligned} ax + b &= (9 + rm) \bmod N \\ ay + b &= (9 + sm) \bmod N \end{aligned}$$

This is a unique solution for a, b once q, r, s are fixed. So there are a total of $m(\frac{N^2}{m})$ solutions $= \frac{N^2}{m}$. Also, since $|H'| = N^2$, therefore H' is "1" universal.