# Discrete Maths

Nguyen Kim Tuyen 20205196

February 27, 2023

## 1 Counting Problems

### 1.1 Euler Candies Dividing Problem

$$x_1 + x_2 + ... + x_k = n \tag{1}$$

The equation has:

- $\binom{n-1}{k-1}$ positive solutions.
- $\binom{n+k-1}{k-1}$ non-negative solutions.

**Variance:**

$$x_1 + x_2 + ... + x_k \leq n \tag{2}$$

The equation has:

- $\binom{k+n}{k}$ positive solutions.

### 1.2 Letter Shuffling Problem

How many $n$-permutation of sequence $1, 2, 3, ..., n$ such that $i$ is not at index $i$ of the sequence $\forall i \in (1, n)$

**Solution:** $n!(1 - \frac{1}{1!} + \frac{1}{2!} - ... + \frac{(-1)^n}{n!})$

### 1.3 Adjacent Selecting Problem

**Problem 1:** How many ways of selecting k elements in a sequence of n elements such that no 2 adjacent elements are selected?

Solution: $\binom{n-k+1}{k}$

**Problem 2:** How many ways of selecting k elements in a circle of n elements such that no 2 adjacent elements are selected?

Solution: $\frac{n}{n-k}\binom{n-k}{k}$

# 2 Recurrence relations

## 2.1 A linear homogeneous recurrence relation of degree k

**Definition:** A linear homogeneous recurrence relation of degree k with constant coefficients is a recurrence relation of the form:

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \ldots + c_k a_{n-k} \tag{3}$$

where $c_1, c_2, \ldots, c_k$ are real number constants, and $c_k \neq 0$.
**Characteristic equation:** $r^k - c_1 r^{k-1} - \ldots - c_k = 0$

- **Theorem 1**: Let $c_1$ and $c_2$ be real numbers. Suppose that $r^2 - c_1 r - c_2 = 0$ has 2 distinct roots $r_1$ and $r_2$. Then the sequence $\{a_n\}$ is a solution of the recurrence relation $a_n = c_1 a_{n-1} + c_2 a_{n-2}$ if and only if

  $$a_n = \alpha_1 (r_1)^n + \alpha_2 (r_2)^n \tag{4}$$

  $n = 0, 1, 2...$, where $\alpha_1, \alpha_2$ are constants.

- **Theorem 2**: Let $c_1$ and $c_2$ be real numbers with $c_2 \neq 0$. Suppose that $r^2 - c_1 r - c_2 = 0$ has only 1 roots $r_0$. Then the sequence $\{a_n\}$ is a solution of the recurrence relation $a_n = c_1 a_{n-1} + c_2 a_{n-2}$ if and only if

  $$a_n = \alpha_1 (r_0)^n + \alpha_2 n (r_0)^n \tag{5}$$

  $n = 0, 1, 2...$, where $\alpha_1, \alpha_2$ are constants.

## 2.2 Linear nonhomogeneous recurrence relation

**Definition:** Linear nonhomogeneous recurrence relation with constant coefficients) has the term $F(n)$ dependent on $n$ (and independent on any value of $a_i$):

$$a_n = c_1 a_{n-1} + c_2 a_{n-2} + \ldots + c_k a_{n-k} + F(n) \tag{6}$$

The solution of linear nonhomogeneous recurrence relation is of the form:

$$a_n = p(n) + h(n) \tag{7}$$

where $a_n = h(n)$ is the solution of the linear homogeneous part: $a_n = \sum_{i=1}^{k} c_i a_{n-i}$
**Special case:**

$$F(n) = G(n) \times s^n = (b_t n^t + b_{t-1} n^{t-1} + \ldots + b_0) s^n \tag{8}$$

where $s$ is a constant; $G(n)$ is polynomial in $n$.

- If $s$ is not equal to characteristic root, then particular solution $p(n)$ is of the form $F(n)$.
- If $s$ is equal to characteristic root with multiplicities $m$, then particular solution $p(n)$ is of the form $n^m \times Q(n) \times s^n$ where $Q(n)$ is of the form $G(n)$.

### 2.3 Generating Functions

**Definition:** Let $a_0, a_1, a_2, \dots$ be a sequence of real numbers. The function:

$$g(x) = a_0 + a_1x + ...a_kx^k + ... = \sum_{i=0}^{\infty} a_i x^i \tag{9}$$

is called the generating function for the given sequence.

$\rightarrow$ We find $a_n$ by determining the coefficient of $x_n$ in $g(x)$

**Some useful generating function:**

| Sequence | Function |
|---|---|
| $1, 1, 1, \dots, 1, 0, 0, 0, \dots$ (the first n+1 terms are 1) | $\frac{1-x^{n+1}}{1-x}$ |
| $1, 1, 1, \dots, 1, \dots$ | $\frac{1}{1-x}$ |
| $1, 2, 3, 4, \dots (\sum_{k=0}^{\infty}(k+1)x^k)$ | $\frac{1}{(1-x)^2}$ |
| $C(n,0), C(n,1), \dots, C(n,k), \dots, C(n,n), 0, 0, 0 \dots$ | $(x+1)^n$ |

## 3 Enumeration Problem: Successive generation algorithms

### 3.1 Enumerate binary sequences

Enumerate all binary strings of length $n$:
$b_1 b_2 \cdots b_n$ where $b_i \in \{0, 1\}$.

1. Assume $b_1 b_2 ... b_n$ be the current string:
   - If this string consists of all 1 then finish.
   - Otherwise, the next string is obtained by adding 1 (module 2, memorize) to the current string.
2. Thus, we have following rule to generate the next string:
   - Find the first $i$ (in the order $i = n, n-1, ..., 1$) such that $b_i = 0$.
   - Reassign $b_i = 1$ and $b_j = 0$ for all $j > i$. The newly obtained sequence will be the string to find.

### 3.2 Enumerate k-combinations of 1, 2, ..., n

Assume $a = (a_1, a_2, ..., a_k)$ is the current configuration but not the final yet, then its next configuration in the lexicographic order could be built by using the following rules:

- Scan from the right to the left of sequence $a_1, a_2, ..., a_k$ : find the first element $a_i \neq n - k + i$.
- Replace $a_i$ by $a_i + 1$
- Replace $a_j$ by $a_i + j - i$, where $j = i + 1, i + 2, ..., k$

### 3.3 Enumerate permutations of 1, 2, . . . , n

Assume $a = (a_1, a_2, ..., a_n)$ is not the last permutation, then its next permutation could be built using the following rules:

- Find from the right to left: first index $j$ satisfying $a_j < a_{j+1}$ ( $j$ is the max index satisfying $a_j < a_{j+1}$
- Find $a_k$ be the smallest number among those on the right of $a_j$ in the sequence and greater than $a_j$
- Swap $a_j$ and $a_k$
- Revese the sequence from $a_{j+1}$ to $a_n$

## 4 Optimization Problems

### 4.1 Traveling Salesman Problem

Use The Nearest Neighbor Method (Approximate Solution).
For Optimal Solution use Branch and Bound Method

### 4.2 Knapsack Problem

Use Greedy algorithm.

### 4.3 Scheduling on 2 machines A and B

1. Divide the jobs into two groups:
    - $N_1$: contains jobs $i$ such that $a_i \leq b_i$
    - $N_2$: contains jobs $i$ such that $a_i \geq b_i$
2. Sort jobs of $N_1$ in a increasing order of $a_i$ (list $L_1$)
3. Sort jobs of $N_2$ in a decreasing order of $a_i$ (list $L_2$)
4. Concatenation $L_1 :: L_2$ is an optimal solution

# 5 Graph

## 5.1 Undirected Graph

**Theorem 1 (The Handshaking Theorem):**
The undirected Graph G = (V, E):

$$\sum_{v \in V} deg(v) = 2|E| \tag{10}$$

**Theorem 2:**
An undirected graph has even number of vertices with odd degree.

## 5.2 Directed Graph

**Theorem 3:**
The directed Graph (either simple or multiple) G = (V, E):

$$\sum_{v \in V} deg^{-}(v) = \sum_{v \in V} deg^{+}(v) = \frac{1}{2} \sum_{v \in V} deg(v) = |E| \tag{11}$$

## 5.3 Isomorphism of Graphs

The simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are isomorphic if there is an one-to-one and onto function $f$ from $V_1$ to $V_2$ with the property that $a$ and $b$ is adjacent in $G_1$ iff $f(a)$ and $f(b)$ is adjacent in $G_2$, $\forall a, b \in V_1$

## 5.4 Path and Cycle

- A path is called elementary if all the **edges** are distinct.
- A path is called simple if all the **vertices** are distinct.
- A path is closed if $v_0 = v_n$.
- closed + elementary = cycle.

## 5.5 Connectedness

- An undirected graph is connected if there exists is a simple path between every pair of vertices.
- If a graph is not connected then it splits up into a number of connected subgraphs, called its connected components.
- A directed graph is strongly connected if there is a path from u to v and from v to u whenever u and v are vertices in the graph.
- A directed graph is weakly connected if its corresponding undirected graph is connected.

### 5.6 Special Graphs

- Complete graphs $K_n$
- Bipartite graphs
- Complete bipartite graphs $K_{m,n}$
- Planar graph
- Euler graph and Hamilton graph

### 5.7 Planar Graph

A graph is called planar if it can be drawn in the plane without any edges crossing.

**Euler's Formula:**

Let G be a connected planar simple graph with e edges and v vertices. Let r be the number of regions in a planar representation of G.

$$r = e - v + 2 \tag{12}$$

- Corollary 1: If $v \geq 3$ then $e \leq 3v - 6$.
- Corollary 2: $\exists v \in G$ such that $deg(v) \leq 5$.
- Corollary 3: If $v \geq 3$ and there's no cycles of length 3 then $e \leq 2v - 4$.

### 5.8 Euler Graphs

- Euler path is a path that traverses through every **edge** exactly once (Half-Euler Graphs).
- Euler cycle is a Euler path begins and ends at the same vertex (Euler Graphs).

**Theorem 1:**

1. If a graph has any vertices of odd degree, then it can not have any Euler cycle.
2. If a graph is connected and every vertex has an even degree, then it has at least one Euler cycle.

**Theorem 2:**

1. If a graph has more than two vertices of odd degree, then it cannot have an Euler path.
2. If a graph is connected and has exactly two vertices of odd degree, then is has at least one Euler path. Any such path must start at one of the odd degree vertices and must end at the other odd degree vertex.

### 5.9 Hamilton Graphs

- Hamilton path is a path that traverses through every **vertex** exactly once (Half-Hamilton Graphs).
- Hamilton cycle is a Hamilton path begins and ends at the same vertex (Hamilton Graphs).

**Theorem Dirac:**
If G is simple connected graph with $n \geq 3$ vertices, and $\forall v, deg(v) \geq \frac{n}{2}$, then G has Hamilton cycle.
**Theorem Ore:**
If G is simple connected graph with $n \geq 3$ vertices, and $deg(u) + deg(v) \geq n, \forall u, v$ not adjacent, then G has Hamilton cycle.

## 6 Graph Representation

- Incidence matrix
- Adjacency matrix
- Weight matrix
- Adjacency list

## 7 Graph Traversal

### 7.1 Breadth First Search – BFS

Always produces shortest path in term of number of edges traveled

### 7.2 Depth First Search – DFS

**Lemma of nested intervals**
Given directed graph $G = (V, E)$, and arbitrary DFS tree, 2 arbitrary vertices $u, v$ of $G$. Then:

- $u$ is a descendant of $v$ iff $[d[u], f[u]] \subseteq [d[v], f[v]]$
- $u$ is ancestor of $v$ iff $[d[u], f[u]] \supseteq [d[v], f[v]]$
- $u$ and $v$ are not related iff $[d[u], f[u]]$ and $[d[v], f[v]]$ are not intersecting.

**Distinction between edges in DFS Tree**

- Tree edge: The edge whereby from a vertex visits a new vertex.
- Back edge: Going from descendant to ancestor.
- Forward edge: Non-tree edges going from ancestor to descendant.
- Cross edge: Edge connecting 2 non-relative vertices.

### 7.3 Topological Sort

Topological sort of a DAG is a linear ordering of all vertices in graph $G$ such that vertex $u$ comes before vertex $v$ if edge $(u, v) \in G$

Topological-Sort():

- Run DFS
- When a vertex $u$ is finished (i.e. when $u.f$ is assigned a time value), output it

Vertices are output in reverse topological order.

## 8 Tree and Spanning Tree

### 8.1 Definition

- Tree is an undirected, connected graph, and contains no cycle.
- Forest is an undirected graph containing no cycle.
- Each connected component of a forest is a tree.

**Theorem:** Given an undirected graph $G = (V, E)$, the following conditions are equivalent:

1. $G$ is a connected graph with no cycles. (Thus $G$ is a tree by the above definition).
2. For every two vertices $u, v \in V$, there exists exactly one simple path from $u$ to $v$.
3. $G$ is connected, and removing any edge from $G$ disconnects it (each edge of $G$ is a bridge).
4. $G$ has no cycles, and adding any edge to $G$ gives rise to a cycle. (Thus $G$ is a maximal acyclic graph).
5. $G$ is connected and $|E| = |V| - 1$.

**Theorem:** Number of spanning trees of $K_n$ is $n^{n-2}$ .

### 8.2 Spanning Tree

**Count the number of spanning trees of a graph**

1. Create Adjacency Matrix for the given graph.
2. Replace all the diagonal elements with the degree of nodes.

3. Replace all non-diagonal 1's with -1.
4. Calculate co-factor for any element.
5. The cofactor that you get is the total number of spanning tree for that graph.

## 8.3 Minimum Spanning Tree Problem