Revision notes - MA3269

Ma Hongqiang

October 6, 2017

Contents

1 Counting

1.1 Geometric and Arithmetic Series

Definition 1.1 (Geometric series).
A **geometric series** is a sum of the form

$$a + ar + ar^2 + \cdots + ar^n$$

Theorem 1.1 (Sum of Geometric series).
The value of sum of a geometric series up to nth term, G_n, for $r \neq 1$, is given by the formula

$$G_n = a\frac{r^{n+1} - 1}{r - 1}$$

The following is an example of solving generating functions of an infinite sequence.
Suppose $a_0 = 1, a_1 = 1, a_2 = 3$. Also, the recurrence relation of the sequence is $a_n = 2a_{n-1}+1$.
Define the **generating function** $U(x)$ as follows,

$$\begin{aligned}
U(t) &:= \sum_{n=0}^{\infty} a_n t^n \quad (|t| < 1) \\
&= t + \sum_{n=2}^{\infty} a_n t^n \\
&= t + \sum_{n=2}^{\infty} (2a_{n-1} + 1)t^n \\
&= t + \sum_{n=2}^{\infty} t^n + 2t \sum_{n=2}^{\infty} a_{n-1} t^{n-1} \\
&= \sum_{n=1}^{\infty} t^n + 2t \sum_{n=1}^{\infty} a_n t^n \\
&= \sum_{n=0}^{\infty} t^n - 1 + 2t \sum_{n=0}^{\infty} a_n t^n \\
&= \sum_{n=0}^{\infty} t^n - 1 + 2tU(t) \\
&= \frac{1}{1-t} - 1 + 2tU(t)
\end{aligned}$$

Solving $U(t)$ and partial fractioning, we have

$$
\begin{aligned}
U(t) &= \frac{1}{1-2t} - \frac{1}{1-t} \\
&= \sum_{n=0}^{\infty}(2t)^n - \sum_{n=0}^{\infty} t^n \\
&= \sum_{n=0}^{\infty}(2^n - 1)t^n
\end{aligned}
$$

By definition,

$$U(t) = \sum_{n=0}^{\infty} a_n t^n = \sum_{n=0}^{\infty}(2^n - 1)t^n$$

Hence,

$$a_n = 2^n - 1$$

Theorem 1.2 (Trianglar Number)**.**

$$T_n = 1 + 2 + 3 + \cdots + n = \frac{1}{2}n(n+1)$$

Definition 1.2 (Arithmetic series)**.**
More generally, an **arithmetic series** is a sum of the form

$$a + (a+d) + (a+2d) + \cdots + (a+nd)$$

The number a is called the **first term** and the number d is called the **common difference** of the arithmetic series.

Theorem 1.3 (Sum of arithmetic series)**.**
The sum of an arithmetic series, up to nth term, A_n is given by

$$A_n = \frac{1}{2}(n+1)(2a+dn)$$

1.2 Sets

Similar objects are often gathered together for easy reference. Such a collection is called a **set**.
The item in a set are often referred to as **elements** or **members** of the set.
We exhibit members of a set within parentheses:

$$S = \{a, e, i, o, u\} \quad \text{or} \quad S = \{x : x \text{ is a vowel of the English alphabet}\}$$

We use the notation $x \in S$ to mean "x *is* a member of S".
The notation $x \notin S$means "x *is not* a member of S".

Definition 1.3 (Empty Set)**.**
The **empty set** is the set containing *no* members. This is denoted by $\varnothing$. That is

$$\varnothing = \{\}$$

Definition 1.4 (Union).
The **union** of sets A and B is the set whose elements are precisely those belong to A or B. Symbolically, we denote the union by $A \cup B$:

$$A \cup B = \{x : x \in A \text{ or } x \in B\}$$

Definition 1.5 (Intersection).
The **intersection** of sets A and B is the set whose elements are precisely those belong to A and B. Symbolically, we denote the intersection by $A \cap B$:

$$A \cap B = \{x : x \in A \text{ and } x \in B\}$$

Definition 1.6 (Disjoint Set).
Sets A and B are **disjoint** if

$$A \cap B = \varnothing$$

Theorem 1.4 (Properties of union and intersection).

- $A \cup B = B \cup A$, $A \cap B = B \cap A$.
- $(A \cup B) \cup C = A \cup (B \cup C)$
- $(A \cap B) \cap C = A \cap (B \cap C)$
- $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$
- $A \cap (B \cup C) = (A \cap C) \cup (A \cap C)$
- $A \cup A = A$, $A \cap A = A$
- $A \cup \varnothing = A$, $A \cap \varnothing = \varnothing$

Definition 1.7 (Subset).
If every element of A is also an element of B, then A is a **subset** of B.

$$A \subseteq B$$

Some immediate consequence of the definition of a subset:

- For any set A, $A \subseteq A$, $\varnothing \subseteq A$.
- For sets A, B,
$$A \subseteq A \cup B, \quad A \cap B \subset A$$
- If $A \subseteq B$, then
$$A \cup B = B, \quad A \cap B = A$$

We are often interested in subsets of a fixed reference set called the **unviersal set**, usually denoted by U.

Definition 1.8 (Complement).
Suppose U is the given universal set, and $A \subseteq U$. Then the **complement** of A, denoted by A^c, is the set consisting of all the elements of U which are not in A. That is

$$A^c = \{x \in U : x \notin A\}$$

By definition,

$$A \cup A^c = U, \quad A \cap A^c = \varnothing$$

Theorem 1.5 (de Morgan's laws).
The following properties hold for set A,B.

- $(A \cup B)^c = A^c \cap B^c$
- $(A \cap B)^c = A^c \cup B^c$

Theorem 1.6 (Principle of Inclusion and Exclusion).

$$\begin{aligned} P(A_1 \cup A_2 \cup \cdots \cup A_n) = &\sum_{i=1}^{n} P(A_i) - \sum_{1 \leq i_1 \leq i_2 \leq n} P(A_{i_1} A_{i_2}) + \cdots \\ &+ (-1)^{r+1} \sum_{1 \leq i_1 \leq \cdots \leq i_r \leq n} P(A_{i_1} \cdots A_{i_r}) \\ &+ \cdots + (-1)^{n+1} P(A_1 \cdots A_n) \end{aligned}$$

Definition 1.9 (Floor function).
For any real number x, the **floor** of x, denoted by

$$\lfloor x \rfloor$$

is the largest integer $\leq x$.

Definition 1.10 (Ceiling function).
For any real number x, the **ceiling** of x, denoted by

$$\lceil x \rceil$$

is the smallest integer $\geq x$.

Theorem 1.7 (Number of multiples).
Let i and n be positive integers. The number of multiples of i among the integers $1, 2, \ldots, n$ is

$$\left\lfloor \frac{n}{i} \right\rfloor$$

1.3 Counting Principles

Theorem 1.8 (Addition Principle)**.**
If a choice from set A_i can be made in n_i ways for $i = 1, \ldots, m$, then the number of choices from $A_1 \cup \cdots \cup A_m$ is

$$n_1 + \cdots + n_m$$

Necessary condition: The sets $A_1, \ldots, A_m$ are **pairwise/mutually disjoint**, i.e. $A_i \cap A_j = \varnothing$ for all $i \neq j$.

Theorem 1.9 (Multiplication Principle)**.**
If a task involves a sequence of m steps, where the ith step can be completed in n_i ways, then there are

$$n_1 \times \cdots n_m$$

ways to complete the task.
Necessary condition: The ways each step can be completed are **indepedent** of each other.

1.4 Arrangements and Combinations

Theorem 1.10.
Let ${}^n\mathrm{P}_k$ be the number of ways of arranging, in a row, k *different* objects taken from n *different* objects. Then

$${}^n\mathrm{P}_k = \frac{n!}{(n-k)!}$$

Theorem 1.11.
The number of ways of arranging in a row n_1 *identical* objects of Type 1, n_2 identical objects of Type 2, …, and n_k identical objects of Type k, is equal to

$$\frac{(\sum_{i=1}^{k} n_i)!}{\prod_{i=1}^{k} n_i!}$$

Theorem 1.12 (Circular Arrangements)**.**
The number of ways of arranging n different objects in a circle is

$$(n-1)!$$

Theorem 1.13.
Let ${}^n\mathrm{C}_k$ denote the number of ways of choosing k objects from a set of n different objects. Then

$${}^n\mathrm{C}_k = \frac{n!}{k!(n-k)!}$$

Another notation for ${}^n\mathrm{C}_k$ is $\binom{n}{k}$.

Theorem 1.14 (Simple properties of binomial coefficients)**.**

- $\binom{n}{0} = \binom{n}{n} = 1$
- $\binom{n}{k} = \binom{n}{n-k}$
- $\sum_{i=0}^{n} \binom{n}{i} = 2^n$

1.5 Number of Routes on Rectangular Grid

Theorem 1.15.
On a rectangular grid, the number of routes from (i, j) to (k, l) moving easterly or notherly without back-tracking is

$$\frac{((k-i)+(l-j))!}{(k-i)!(l-j)!} = \binom{k+l-i-j}{k-i} = \binom{k+l-i-j}{l-j}$$

1.6 Pigeonhole Principle

Theorem 1.16 (Pigeonhole Principle)**.**
Suppose m objects are distributed among n pigeonholes. If $m > n$, then there is at least **one** pigeonhole with at least **two** of the distributed objects.

Theorem 1.17 (Extended Pigeonhole Principle)**.**
If m objects are distributed among n pigeonholes and $m > n$, then there will be **one** pigeonhole which contains **at least** $\lceil \frac{m}{n} \rceil$ objects.

2 Graphing

2.1 Introduction

Definition 2.1 (Graph)**.**
A **graph** is a collection of points and lines connecting *some pairs* of the points.
The points are called the **vertices**.
The lines joining any the vertices are called **edges**.

Two vertices that are joined by an edge are called **adjacent** vertices.

Definition 2.2 (Simple Graph)**.**
A graph without loops and multiple edges is called a **simple graph**.

2.2 Basic Technology

Definition 2.3 (Walk)**.**
A **walk** is a sequence of vertices and edges in a graph such that

- the sequence alternates between vertices and edges, starting and ending with vertices; and

- each edge in the sequence joins the vertices that occur immediately before and after it in the sequence

A walk that starts and ends at *different* vertices is called an **open walk**.
A walk that starts and ends at *the same* vertex is called a **closed walk**.
A walk that contains *no repeated* vertices *and* edges is called a **path**.

Definition 2.4 (Cycle)**.**
A **cycle** in a graph is a **closed walk** in which the only repitition is the **first and last** vertex.

Definition 2.5 (Length)**.**
The **length** of a walk is defined as the number of edges in the walk, including repetitions.

Definition 2.6 (Degree)**.**
The **degree** of a vertex in a graph is the number of edges that occur at that vertex, with every *loop counted as two.*

Theorem 2.1 (Degree Theorem)**.**
In any graph, the sum of all the degrees is equal to **twice** the number of edges.
In particular, the sum of all the degrees must be even.

Definition 2.7 (Odd and Even Vertex)**.**
An **odd** vertex is a vertex whose degree is an odd number.
An **even** vertex is a vertex whose degree is an even number.

Definition 2.8 (Minimum and Maximum Degree)**.**
In any graph G, the symbol $\delta(G)$ represents the **minimum** degree in G; the symbol $\Delta(G)$ represents the maximum degree.

2.3 Trees

Definition 2.9 (Trees).
A **tree** is a simple graph that is connected and contains no cycle.

Definition 2.10 (Leaf).
A **leaf** is a vertex of degree 1.

Theorem 2.2 (Leaf Lemma).
Every tree with two or more vertices has at least two vertices of degree 1.

Theorem 2.3 (Tree theorem).
Every tree with n vertices has exactly $n-1$ edges.

2.4 Minimal Spanning Trees

Definition 2.11 (Weighted Graphs).
A **weighted graph** is a graph in which each edge has a number associated with it, which we refer to as the **weight** of that edge.

Definition 2.12 (Subgraph).
A **subgraph** of a graph G is a graph H whose vertices and edges are taken from those of G.

Definition 2.13 (Weight of a graph).
The **weight** of a graph G is the sum of weights of all its edges. Symbolically,

$$w(G) = \sum_{e \text{ edge of } G} w(e)$$

where $w(e)$ denotes the weight of the edge e.

Definition 2.14 (Spanning Tree).
A **spanning tree** of a given graph G is a subgraph T of G which is a tree and it contains **all** the vertices of G.

Definition 2.15 (Minimal Spanning Tree).
A **minimal spanning tree** of G is a spanning tree which has the *minimum weight* among all the spanning trees of G.

Theorem 2.4 (Prim's algorithm).
Prim's Algorithm is a procedure for constructing a minimal spanning tree in a given weighted graph:
Input: a weighted graph G
Output: a minimal spanning tree T of G
Algorithm:

- Start with any vertex and select an edge having the minimum weight among all the edges at that vertex
- Consider *all* edges that go from each vertex already reached to a new vertex.
  Select one that has *minimum weight*.
- Continue until all vertices have been reached.

2.5 Euler Walks

In this section, we allow graphs to contain loops and multiple edges.

Definition 2.16 (Directed graph)**.**
A **directed graph** is a graph in which *every* edge is assigned a direction indicated by an arrow(called a **directed edge**).

Definition 2.17 (Walk in directed graph)**.**
A **walk** in a directed graph is a sequence of vertices and directed edges such that

- the sequence alternates between vertices and directed edges, starting and ending with vertices; and

- each directed edge in the sequence joins the vertices that occur immediately before and after it in the sequence **in the direction** indicated by the arrow of the directed edge.

Definition 2.18 (Euler walk)**.**
An **Euler walk** in a graph is a walk that uses every edge in the graph exactly once.
A **closed** Euler walk (also known as **Euler circuit**) is an Euler walk that starts and ends at the same vertex.
An **open** Euler walk is an Euler walk that starts and ends at different vertices.

Note: When tracing an Euler walk,

- a vertex may be visited more than once

- every edge is visited exactly once

- the entire graph is traced without lifting the pen

Theorem 2.5 (Euler Walk Theorem I)**.**
A connected *undirected* graph contains a *closed* Euler walk if and only if *every* vertex has **even** degree.

Theorem 2.6 (Euler Walk Theorem II)**.**
A connected *undirected* graph contains an *open* Euler walk starting from vertex A and ending at vertex B if and only if

- vertices A and B have odd degree; and

- all the other vertices have even degree.

Theorem 2.7 (Euler Walk Theorem I – directed version)**.**
A connected directed graph contains a *closed* Euler walk if and only if for *every* vertex the number of arrows pointing **in** is *equal* to the number of arrows pointing **out**.

Theorem 2.8 (Euler Walk Theorem II – directed version)**.**
A connected directed graph contains an *open* Euler walk starting from vertex A and ending at vertex B if and only if

- for vertex A, the number of arrows pointing out is exactly one more than the number of arrows pointing in;
- for vertex B, the number of arrows pointing in is exactly one more than the number of arrows pointing out;
- for all other vertices, the number of arrows pointing in is equal to the number of arrows pointing out.

An algorithm to construct an Euler circuit:

(1) Make sure the graph is connected and all vertices are even.

(2) Start anywhere. Construct a closed walk without repeated edges.

(3) If the closed walk covers all edges, DONE.

(4) If not, construct another closed walk without repeated edges and combine the two to get a bigger closed walk.

(5) Repeat (4) and stop when all edges are used

Chinese Postman Problem:
Given a connected weighted graph or directed graph G, find the shortest circuit that uses each edge in G **at least once**.

The simplest case: This occurs when every vertex in the graph has even degree, for in this case an Euler circuit solves the problem. **General case: Vertices of odd degree present**

1. List all odd vertices
2. List all possible pairing of odd vertices
3. For each pairing, find paths that connect the vertices with the minimum weight. Find the pairings such that the sum of the weights is minimised.
4. On the original graph, add the edges that have been found in Step 3
5. The length of an optimal Chinese postman route is the sum of all the edges added to the total found in Step 4
6. A route corresponding to this minimum weight is an Euler circuit in the graph obtained in Step 5.

2.6 Vertex Coloring

Definition 2.19 (Proper Vertex Coloring).
A **proper vertex coloring** of a graph is an assignment of a color to each vertex of the graph in such a way that any two vertices that are adjacent have *different* colours.

Definition 2.20 (Minimal Proper Vertex Coloring).
We are interested in a proper vertex coloring of a given graph G using the smallest possible number of colors. Such a coloring is called a **minimal proper vertex coloring** of G.
The number of colors that occurs in a minimal proper vertex coloring is called the **chromatic number** of G, denoted by

$$\chi(G)$$

Definition 2.21 (Complete Graph).
The **complete graph** on n vertices is a graph on n vertices such that any two vertices are joined by an edge. It is denoted by K_n.

Theorem 2.9 (Vertex Coloring Theorem).
If a graph G contains a complete graph on n vertices, then a proper vertex coloring of G must use at least n colors. Therefore, $\chi(G) \geq n$.

Definition 2.22 (Cycle Graph).
A **cycle graph** of length n is denoted by C_n.

Theorem 2.10 (Vertex Coloring Theorem II).
If a graph G contains a cycle graph C_n on n vertices where n is odd, then a proper vertex coloring of G must use at least 3 colours. Therefore, $\chi(G) \geq 3$.

Theorem 2.11 (Upper bound algorithm for χ).

1. Arrange the degrees of a graph G in decreasing order:

$$d_1 \geq d_2 \geq d_3 \geq \cdots$$

2. Place the integers $1, 2, 3, \ldots$ directly under these degrees until you reach an integer k such that $k + 1 > d_{k+1}$.

Then for this graph G, we have

$$\chi(G) \leq k + 1$$

3 Clocking

3.1 Parity of integers

Definition 3.1 (Parity).
Two integers are said to be of the same **parity** if they are either both odd or both even.

Theorem 3.1 (Difference of integers of same parity).
If two integers are of the same parity, then their difference is an even integer.

3.2 Congreunce Equations

Definition 3.2.
Suppose a and b are integers such that their difference is a multiple of a positive integer n. Then we write

$$a \equiv b \pmod{n}$$

where n is called the modulus.

Theorem 3.2 (Properties of modulo arithematic).
If the remainder of a when divided by n is r, then

$$a \equiv r \pmod{n}$$

$a \equiv bpmodn$ if and only if a and b have the same remainder when divided by n.
If $a \equiv b \pmod{n}$, then $a \equiv b \pm n \pmod{n}$.
Suppose $a \equiv b \pmod{n}$, then

$$ka \equiv kb \pmod{n}$$

where k is a integer; and

$$a^p \equiv b^p \pmod{n}$$

where p is a positive integer.
Congruences are transitive in a sense that, if $a \equiv b \pmod{n}$ and $b \equiv c \pmod{n}$ then

$$a \equiv c \pmod{n}$$

Two congruences with the same modulus can be added to each other or multiplied one by the other.
Suppose $a \equiv b \pmod{n}$ and $c \equiv d \pmod{n}$ then

$$a + c \equiv b + d \pmod{n}$$
$$ac \equiv bd \pmod{n}$$

Theorem 3.3 (Congruence mod 9).
Let S be the sum of digits of the decimal representation of the positive integer N. Then

$$N \equiv S \pmod{9}$$

Theorem 3.4 (Checking product).
Suppose $A \times B = C$, then

$$S \times T \equiv U \pmod{n}$$

where S, T, U are the sums of digits of A, B, C respectively.

4 Coding

4.1 Number representation System

Number representation systems are covered in `CS2100 Revision Notes`.

4.2 Error Detection Code

Definition 4.1 (Weighted Sum)**.**
Given the sequence (or word) $S_n S_{n-1} \cdots S_2 S_1$, its **weighted sum** is the sum

$$\sum_{i=1}^{n} i \times e_i$$

where e_i is the numerical value which corresponds to the symbol $S_i, 1 \leq i \leq n$.

Theorem 4.1 (Encoding Procedure Modulo 37)**.**
Input: A sequence $\mathbf{S} = S_n S_{n-1} \cdots S_2, n \leq 36$.

1. Find the *check digit* c such that

$$w(e_n e_{n-1} \cdots e_2 c) \equiv 0 \mod 37$$

   where w denotes the weighted sum.

2. Find the symbol S_1 that corresponds to c.

Output: The encoded sequence is

$$S_n S_{n-1} \cdots S_2 S_1$$

Definition 4.2 (Error)**.**
A word is said to contain k errors if k of its letters are erroneous.

Theorem 4.2 (Error Detection)**.**
If A is a correctly encoded word, and during transmission, some errors occur and the word A' is received. The errors in A' is said to be detected if its weighted sum

$$w(A') \not\equiv 0 \mod 37$$

Theorem 4.3. In weighted sum encoding modulo 37, if a single error occurs, then it can be detected.

Theorem 4.4. Weighted sum modulo 37 can detect a transposition error.