# Lecture 1
# 進階圖論 | Advanced Graph Theory

## 1.1 圖的種類

### 1.1.1 平面圖

**平面圖**就是可以畫在平面上的圖，使得任兩個邊只會在點上相交而已。

而平面圖有一個很重要的定理。

**定理 (歐拉定理)：** 對於一個**連通**的平面圖，有

$$E = V + F - 2 \tag{1.1}$$

其中 $E, V, F$ 分別代表圖的邊數、點數還有面數。一個**面**定義爲被邊所切出的一個區域(包含最外面的無限區域)。

歐拉公式給出了平面圖邊和點的關係式。而從公式中也可以看出平面圖邊的個數不會太多! 事實上對於一個簡單的平面圖，有以下定理:

**定理：** 對於一個**簡單連通**的平面圖，如果 $|V| \geq 3$ 則有

$$|E| \leq 3|V| - 6 \tag{1.2}$$

*Proof.* 由 (1.1)，$|E| = |V| + |F| - 2$。但簡單圖沒有自環及重邊，一個面一定至少有三條邊，$|F| \geq 3|E|$。代入 (1.1) 得 $|E| \leq 3|V| - 6$。 □

邊很少的這個條件往往是解題的關鍵!

例題 **1-1** Defense Your Country NTUJ 2126

給你一個平面圖，請找一個最大團，也就是最大的一個完全子圖。($|V| \leq 2 \cdot 10^5$)

一看這個題目，不得了，連 $|E|$ 都沒有給，但不要忘了由 (1.2) 邊的數量不會超過 3 倍的 $|V|$。

但最大團是個 NPC 問題，如果我們不善用平面圖的性質，肯定解不出來。注意到**平面圖的子圖仍是平面圖**，而完全圖的邊的數量 $\mathcal{O}(E) = \mathcal{O}(V^2)$，平面圖的是 $\mathcal{O}(E) = \mathcal{O}(V)$。肯定**有個上界 $m$，$K_m$ 絕對不會出現在平面圖上**。事實上算一下會發現對於 $K_5$，邊有 10 條，點有 5 個，而 $10 > 3 \cdot 5 - 6 = 9$，所以根據 (1.2) $K_5$ 肯定不是平面圖! 因此平面圖的最大團一定不超過 4。

不過這題還沒有結束，雖然我們只要檢查 4 個點以下的完全圖，但還是要有一個有效率的方法，否則 $\mathcal{O}(V^4)$ 枚舉肯定要超時。再注意到 (1.2) 其實告訴了

$$\sum \deg(v) = 2E \leq 6V - 12.$$

因此 $\deg(v)$ 的平均值小於 6 ，也就是一定有一個點的度數小於 5，所以我們可以先找一個 $v_1$ 使得 $\deg(v_1) \leq 5$ ，然後 $2^6$ 枚舉他和他的鄰居的最大團就可以了。枚舉完了這個點，將這個點從圖上移除，新的圖仍是一個平面圖，因此我們又可以再找一個 $v_2$ 使得 $\deg(v_2) \leq 5$。一路做下去我們就得到了一個 $\mathcal{O}(V)$ 的做法!

這裡再總結一下平面圖的一些結論。

**定理：** 定義一個圖的 degeneracy 為 $\min \deg(v)$，則平面圖的 degeneracy $\leq 5$。

**定理：** 一個圖是平面圖的條件若且唯若 $K_5, K_{3,3}$ 不是他的 minor。$G$ 是 $H$ 的 minor 表示 $G$ 可以由

- 刪掉一條邊
- 刪掉一個點
- 收縮一條邊，也就是把兩個點合併。

得到。

首先我們定義什麼是字串。

**定義：** 給定一個字元集 $\Sigma$，一個字串是一些字元的有序序列，我們寫作

$$A = a_0 a_1 \cdots a_{n-1}$$

其中 $a_i \in \Sigma$。我們把 $n$ 叫作字串的長度。

舉個例子，如果 $\Sigma = \{$"A","C","G","T"$\}$，$A$ 可能表示你的 DNA 序列，或者如果 $\Sigma = \{$"0","1",$\cdots$"9"$\}$，那麼 $A$ 可能代表客戶的電話號碼。這時候就會有許多有趣的問題了，比如:

- 假設另一段 DNA 字串 $B$ 代表一個可能的致癌基因，如何快速的判斷一個人是否帶有這種基因?
- 現在有幾種不同的生物的 DNA 序列，如何判斷這些字串間的相似成度，以了解演化順序?

對於這些問題，數據的大小可以是非常大的，我們勢必會需要一些好的演算法來解決它們! 不過在這之前我們必需先說清楚一些定義。

**定義：** 給定一個字串 $A = a_0a_1\cdots a_{n-1}$

- 一個子字串是其連續的一段 $a_ia_{i+1}a_{i+2}\cdots a_j$ 記作 $A[i,j]$。
- 一個子序列是一個字串 $B = a_{q_1}a_{q_2}a_{q_3}\cdots a_{q_m}$，其中 $0 \le q_1 < q_2 < q_3 < \cdots < q_{m-1} < q_m < n$ 。
- 一個 $A$ 的前綴是 $A$ 的一個子字串 $a_0a_1a_2\cdots a_h$，其中 $0 \le h < n$，記作 $P_A(h)$。
- 一個 $A$ 的後綴是 $A$ 的一個子字串 $a_ka_{k+1}a_{k+2}\cdots a_n$，其中 $0 \le k < n$，我們特別記作 $S_A(k)$。並讓所有後綴的集合稱作 $\sigma(A)$。

舉例來說，如果 $A =$ "abcbbab"，那"bcb" 是他的子字串，"acb" 是他的子序列，而"bbab" 是他的一個後綴。

## 1.2 字串的儲存

通常最基本的儲存方式就是用一個陣列依序將字串的每一個字元存下來。不過當我們要同時儲存許多字串時，可能就要花點巧思了。而這邊要介紹一個可以同時儲存多個字串的資料結構—**字典樹 Trie**。

Trie 的道理非常簡單，其實就是用一棵樹來儲存字串。在這棵樹上，每個點 (除了根節點之外) 上都有一個字元，而從根節點一路走到某個節點，依序經過的字元串起來就是那個點代表的字串。最後我們再記錄哪些點是一個字串的最後一個字元即可! 如 Figure **??**就是一個儲存 $\{A_1 =$ "abc", $A_2 =$ "abde", $A_3 =$ "bc", $A_4 =$ "bcd"$\}$ 的 trie