Review VIII(Slides 418 - 486)

Graph Theory

Construct your graph in a nice way!

HamHam

University of Michigan-Shanghai Jiao Tong University Joint Institute

February 15, 2022

VE203 - Discrete Mathmatics

Asymptotic Notation

We define:

$$
\begin{aligned}
O(g(n)) &= \{f(n) \mid \exists c, n_0 \text{ s.t. } 0 \le f(n) \le c \cdot g(n),\ \text{for } n \ge n_0\} \\
\Omega(g(n)) &= \{f(n) \mid \exists c, n_0 \text{ s.t. } 0 \le c \cdot g(n) \le f(n),\ \text{for } n \ge n_0\} \\
\Theta(g(n)) &= O(g(n)) \cap \Omega(g(n)) \\
&= \{f(n) \mid \exists c_1, c_2, n_0 \text{ s.t. } c_1 g(n) \le f(n) \le c_2 g(n),\ \text{for } n \ge n_0\}
\end{aligned}
$$

Yep, that's end of the story. I guess $\omega(n)$ and $o(n)$ probably won't occur in the exam.

Exercise

1. Which of these symbols

$$\Theta \; O \; \Omega \; o \; \omega$$

can go in these boxes? (List all that apply.)

$$
\begin{aligned}
2n + \log n &= \quad (n) \\
\log n &= \quad (n) \\
\sqrt{n} &= \quad (\log_{300} n) \\
n2^n &= \quad (n) \\
n^7 &= \quad (1.01^n)
\end{aligned}
$$

(Taken from CCP 8.4)

Master Theorem

If $T(n) = aT(n/b) + f(n)$ (for constants $a \geq 1, b > 1, d \geq 0$), then

1. $T(n) = \Theta(n^{\log_b a})$ if $f(n) = O(n^{\log_b a-\varepsilon})$ for some constant $\varepsilon > 0$
2. $T(n) = \Theta(n^{\log_b a} \lg n)$ if $f(n) = \Theta\left(n^{\log_b a}\right)$
3. $T(n) = \Theta\left(f(n)\right)$, if $f(n) = \Omega\left(n^{\log_b a+\varepsilon}\right)$ for some constant $\varepsilon > 0$ and regular condition (why?)

Exercise: Solve

$$T(n) = 4T\left(\sqrt{n}\right) + \log^5 n$$

Recipe

- Compare $f(n)$ with $n^{\log_b a}$
- Do substitution if necessary

Comment. This would be provided in the exam paper.

Terminology

Some definitions...

- vertex set V
- edge set E
- adjacent
- loop
- parallel
- simple graph
- isomorphism $G \cong H$
- complment $\overline{G}$
- degree $\deg(v)$
- distance $\operatorname{dist}(u, v)$

Standard Graphs

You should remember both the names and the notations. Let's see them in Mathematica!

- Complete Graph K_n
- Clique
- Path P_n
- Cycle Graph C_n
- Bipartite Graphs $K_{m,n}$
- *Wheel Graph W_n
- *Qubic Graph Q_n

Attention: null graph

$G = (V, \varnothing)$ or $G = (\varnothing, \varnothing)$?

Exercise

3. The complement of a simple graph $G = (V, E)$ is given by $G^c = (V, E^c)$, where $E^c = V \times V \setminus E$, i.e., the complement has the same vertex set and an edge is in E^c if and only if it is not in E. A graph G is said to be *self-complementary* if G is isomorphic to G^c .

i) Show that a self-complementary graph must have either $4m$ or $4m + 1$ vertices, $m \in \mathbb{N}$.

ii) Find all self-complementary graphs with 8 or fewer vertices.

(Taken from Ve203 FA2020 Assignment10)

The Handshaking Theorem

Undirected graph:

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

Directed graph:

$$|E| = \sum_{v \in V} \deg^+(v) = \sum_{v \in V} \deg^-(v)$$

Remark:

- A vertex is said to be isolated if it has degree zero.
- A vertex is said to be pendant if it has degree one.
- $\deg^+(v)$: in-degree of a vertex v
- $\deg^-(v)$: out-degree of a vertex v

Walks and Connectivity

Definition

A walk W in G is a sequence of vertices $\{v_i\}_{i=0}^{n}$ and edges $\{e_i\}_{i=1}^{n}$ so that e_i is incident with v_{i-1} and v_i.

- W is called **closed** if $v_n = v_0$
- The **length** of W is its number of edges n
- G is connected if $\forall u, v \in V(G)$, there is a walk from u to v

Exercise

4. Show that a simple graph $G := (V, E)$ with $|V| = n$ is connected if $|E| > (n-1)(n-2)/2$.

Exercise

4. Show that a simple graph $G := (V, E)$ with $|V| = n$ is connected if $|E| > (n-1)(n-2)/2$.

Solution:

Solution:

K_{n-1} could only have at most $\binom{n-1}{2}$ edges.

Exercise

5. Judge whether the following statements are true or false.
   - A walk must be a path or cycle.
   - If there is a walk from u to v, there is also such a path.
   - G is disconnected iff there is a partition $\{X, Y\}$ of $V(G)$ such that no edge has an end in X and an end in Y.
   - For two connected subgraphs $H_1, H_2 \subset G$ that $V(H_1) \cap V(H_1) \neq \varnothing$, $H_1 \cup H_2 := (V(H_1) \cup V(H_1), E(H_1) \cup E(H_1))$ is connected.

Components

Definition

A component of a graph G is a **maximal connected subgraph** in G. In other words, it is not contained in any other connected subgraphs.

The number of components of G is denoted as $\text{comp}\,(G)$.

Theorem

Every vertex is a **unique** component.

Note

If a graph G isn't connected, it may be useful to consider its components.

Cuts

Definition(substraction)

Given $G = (V, E)$, $S \subset E$, $X \subset V$, then $G - S := (V, E \setminus S)$ and $G - X := (V \setminus X, \{e \in E : e \text{ not incident with } x \in X\})$.

Definition

- $e \in E$ is a **cut-edge** or **bridge** if no cycle contains e
- $v \in V$ is a **cut-vertex** if $\text{comp}\,(G - v) > \text{comp}\,(G)$

What happens when we delete an edge or vertex?

- If e is a cut-edge, $\text{comp}\,(G - e) = \text{comp}\,(G) + 1$
- If e is not, $\text{comp}\,(G - e) = \text{comp}\,(G)$
- Further, $\text{comp}\,(G - v) \leq \text{comp}\,(G) + \deg\,(v) - 1$

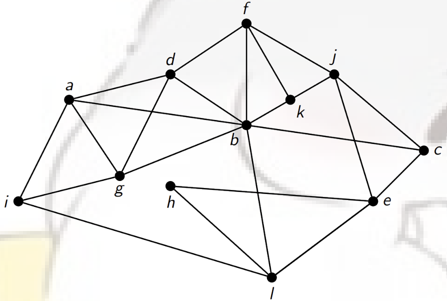
Induced subgraph

Please remember to delete both the vertexes and the edges!

Exercise

The followings are really likely to appear in the exam 😅.

6. Given a graph G shown as follows:



List the corresponding vertices or edges of G for the following:

- Find a clique of size 4/ size 5 if possible
- Find a induced cycle of size 4 / size 5 if possible
- Find a maximal matching that is **NOT** maximum.

Bipartation & Matching

Theorem

For every graph G, the following are equivalent:

- G is bipartite
- G has no cyle of odd length
- G has no closed walk of odd length
- G has no induced cycle of odd length.

Compare and Contrast

- Maximal chain/ maximum chain
- Maximal matching/maximum matching

Group Tran ,Transversals?

That's difficult QAQ, and let's look at what xrz left us.

Hall's Theorem

Let G be a bipartite graph with bipartation (A, B). There exists a matching covering A iff there does not exist $X \subset A$ with $|N(X)| < |X|$.

Exercise

Given a sequence of (not necessarily distinct) sets $S_1, S_2, \ldots, S_m$, there exists a sequence of distinct elements $x_1, x_2, \ldots, x_m$ such that $x_i S_i$ for all $i = 1, 2, \ldots, m$ if and only if **Hall's condition** holds. State **Hall's condition** in this context.

Hall's Theorem

Let G be a bipartite graph with bipartation (A, B). There exists a matching covering A iff there does not exist $X \subset A$ with $|N(X)| < |X|$.

Exercise

Given a sequence of (not necessarily distinct) sets $S_1, S_2, \ldots, S_m$, there exists a sequence of distinct elements $x_1, x_2, \ldots, x_m$ such that x_iS_i for all $i = 1, 2, \ldots, m$ if and only if **Hall's condition** holds. State **Hall's condition** in this context.

Solution:

For every $k = 1, 2, \ldots, m$, the union of any k sets has at least k elements, that is

$$|\bigcup_{i\in I} S_i| \geq |I| \text{ for all } I \subset \{1, \ldots, m\}$$

Trees

Recall them:

- forest
- tree
- leaf
- root
- spanning tree

Exercise

Let T be a tree, v be its leaf. Judge whether the following statements are correct or not :

1. $\text{comp}(G) = |V(G)| - |E(G)|$
2. $|V(T)| = |E(T)| + 1$
3. $T - v$ is a tree

Kruskal's Algorithm

Goal: Find the minimum spanning tree (path of length $n-1$).

Steps:

1. sort the edges according to their costs
2. choose the minimum one, that is not choosed yet and would **not form a cycle**

Comment:

This is a *Greedy Algorithm*. But why is it correct?

Dijkstra's Algorithm

Goal: Find the minimum distance from the root (one specified vertex).

Steps:

1. start from the nearest points you know.
2. update the distance of other vertexes according to this vertex that you choose.

Comment:

There're other shortest path algorithm, including Freud's Algorithm, Bellman-Ford's Algorithm. I guess this would be taught in VE477 by Mn.

Here is just an overview:

`https://www.cnblogs.com/fxzemmm/p/14847987.html`

A Question

Wait a moment, there is a question when you apply the Kruskal's Algorithm:

How to judge that, when you are adding one specified edge, would it form a cycle or not? How do computers do that?

We need a data structure, called **union and find set**.

Union and Find Set

MAKE_SET(x):

$$\text{Father}\ [\ x\ ] \leftarrow x$$

FIND(x): Is the father of x itself?

$$\text{IF Father}\ [\ x\ ] = \text{x}\ ,\ \text{RETURN}\ x.$$
$$\text{ELSE RETURN FIND(Father}[\ x\ ])$$

UNION(x,y):

$$\text{Father}[\ \text{FD}(y)\ ] \leftarrow \text{FD}(x)$$

A typical Question

Question link: `https://vijos.org/p/1034`

```
#include <iostream>
using namespace std;
int find_root(int* father,int son){
    while (son!=father[son]) son=father[son];
    return son;
}

int main(){
    int m,n,p;//n people; m relations; enquiry p pairs
    cin >> n >> m >> p;
    int* father=new int[n+1];
    for (int i = 0; i < n; i++){
        father[i+1]=i+1;
    }
    int temp1,temp2;
    int f1,f2;//temperary father
```

Sample Code(Cont.)

```
        for (int i = 0; i < m; i++){
            cin >> temp1 >> temp2;
            f1=find_root(father,temp1);f2=find_root(father,temp2);
            if (f1!=f2) father[f2]=f1;
        }
        for (int i = 0; i < p; i++){
            cin >> temp1 >> temp2;
            f1=find_root(father,temp1);f2=find_root(father,temp2);
            if (f1==f2) cout << "Yes" << endl;
            else cout << "No" << endl;
        }
        delete[] father;
        return 0;
    }
```

Reference

- Exercises from Ve203-2020-Fall Assignment.
- Exercises from Ve203-2021-Fall TA Xue Runze.
- Exericses from Ve203-2021-Summer Final Exam.
- Liu Dayou etc.*Data Structure*, third edtion, Beijing: Higher Education Press, 2019.5 print.