MATHS 2P : Graphs & Networks

Professor Tara Brendle

Joao Almeida-Domingues[*]

University of Glasgow

September 25th, 2019 – December 4th, 2019

Contents

These lecture notes were collated by me from a mixture of sources , the two main sources being the lecture notes provided by the lecturer and the content presented in-lecture. All other referenced material (if used) can be found in the *Bibliography* and *References* sections.

The primary goal of these notes is to function as a succinct but comprehensive revision aid, hence if you came by them via a search engine , please note that they're not intended to be a reflection of the quality of the materials referenced or the content lectured.

Lastly, with regards to formatting, the pdf doc was typeset in LaTeX, using a modified version of Stefano Maggiolo's class

[*]2334590D@student.gla.ac.uk

1 Fundamentals

Lecture 1
September 25th, 2019

1.1 *Graphs*

A graph G is a pair (V, E), where V is any finite set, and E is a set whose elements are pairs of elements of V. We call the elements of V the *vertices** of G and those of E its *edges*. e.g. $G = \{\{a, b, c\}, \{ab, ac\}\}$

** often also referred as nodes*

1.1 definition. Adjacent Vertices are vertices connected directly through an edge. Formally, if $e = \{u, v\} \in E$, then u, v are adjacent

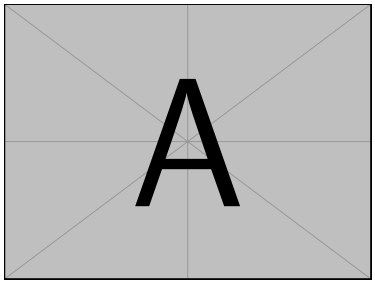
1.2 definition. Incident Edges are edges which share a vertex. We say that they are *"incident to v"*

Lecture 2
September 27th, 19

Representing Graphs

Note that the representation need not be unique

Pictorially



Note that this definition only holds for simple graphs, i.e without loops. But , it is easily generalised if the binary requirement is dropped

Adjacency Matrix

1.3 definition. Adjacency Matrix is the $n \times n$ binary matrix , where $n = |V|$ and $a_{ij} = 1 \iff e = \{u, v\} \in E$; i.e iff u, v are adjacent

$$A = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

1.4 remark. In this course, we'll only deal with *simple*, *undirected* graphs. Note that AMs of this type have the nice property of being symmetric (see 2B notes for properties)

Subgraphs

1.5 definition. Subgraphs are graphs obtained by deleting edges and/or edges of another graph

1.6 definition. Induced Subgraph is a graph formed by deleting only nodes and their incident edges. Formally: Let $W \subset V$, then the induced subgraph of G is given by $G[W] = \{W, \{\{xy\} \,|xy \in G\}\}$. We say that *"G is induced by W"*

1.7 example.

$G(V) = \{\{a,b,c\},\{ab,ac\}\}$ and $U = \{c\}$ then $G[U] = \{\{a,b,\not{c}\},\{ab,\not{ac}\}\}$

1.8 definition. Spanning Subgraph similar to the induced, but edges are deleted instead

Lecture 3
October 2nd, 19

1.2 *Graph Properties*

1.9 definition. Walk from u to v is a sequence of vertices $w1, \ldots, wp$ (for some natural number $p \geq 2$), with $w_1 = u$ and $w_p = v$, such that $w_i w_{i+1}$ is an edge for every $1 \leq i \leq p-1$

Informally, a walk is just a sequence of vertices, where each subsequent vertex added to the sequence forms an edge with the preceding one

1.10 definition. Trail a walk with distinct edges

1.11 definition. Path a walk with distinct vertices

1.12 remark. In general, every walk between two vertices contains a path

1.13 example. For a graph $P(\{a,b,c,d,e,f\},\{ab,ac,ad,bc,bd,cd,de,ef,\})$,

Walk: $W = \{abcdeacd\}$

Trail : $T = \{abcdea\} = W \setminus \{cd\}_2$

Path: $P = \{abcde\} = W \setminus \{acd\}_2$

where $\{x\}_2$ is improper notation for the repeated instances of x in a set

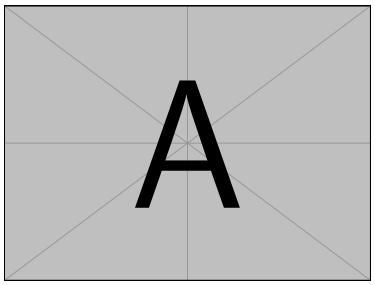
1.14 proposition. ***Number of Paths*** *For a graph with n vertices, there are $(n-1)^n$ paths (see proof below)*

1.15 definition. Connected A graph $G = (V, E)$ is connected if, for every two distinct vertices $u, v \in V$, there is a path in G from u to v

1.16 remark. A single vertex graph is connected. Since it has not distinct vertices, we say that the definition holds *vacuously*

1.17 definition. Connected Component H is a connected component of G if H is a connected induced subgraph of G and, for any subgraph H' of G such that $V(H) \subset V(H')$, H' is not connected

1.18 remark. The vertex sets of distinct connected components are necessarily disjoint



1.19 definition. Vertex Degree $d(v) = |E(v)|$, i.e. it is the size of the set of all edges connected to v

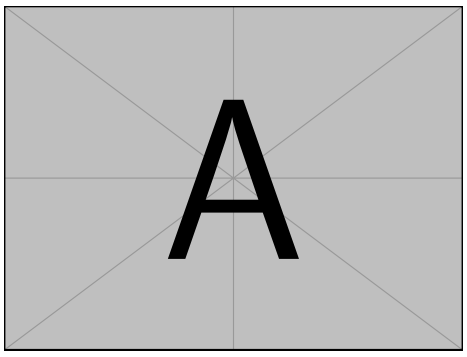
1.20 definition. Minimum Degree $\min_{v \in V(G)} d(v)$, i.e. a graph's minimum degree is equal to the lowest degree of its vertices

The converse is true of the maximum

1.3 *Isomorphisms*

1.21 definition. Isomorphism from $G_1 = (V_1, E_1)$ to $G_2 = (V_2, E_2)$ is a bijection $f : V_1 \to V_2$ such that, for every $u, v \in V_1, f(u)f(v) \in E_2 \equiv uv \in E_1$

1.22 remark. Specifically, we can consider f to be a process whereby one *relabels* the vertices



To do (4)

2 Special Graphs

2.1 definition. Complete Graphs Every pair of distinct vertices forms an edge

2.2 notation. K_n , for a graph with n vertices

2.3 proposition. ***Number of Edges*** *K_n has $\frac{1}{2}n(n-1)$ edges (see proof below)*
To do (5)

2.4 definition. Paths a path on n vertices is a graph that is isomorphic to the graph (V, E) where $V = \{v_1, \ldots, v_n\}$ and $E = \{v_i v_{i+1}\} : 1 \leq i \leq n-1$

2.5 notation. P_n

2.6 remark. P_n has $n-1$ edges

2.7 definition. Cycle a cycle on n vertices is a graph that is isomorphic to the graph (V, E) where $V = \{v_1, \ldots, v_n\}$ and $E = \{v_i v_{i+1}\} : 1 \leq i \leq n-1 \cup \{v_n v_1\}$. i.e, it's a path with the end vertices connected

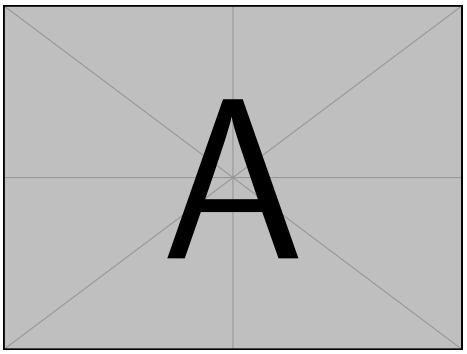
2.8 notation. C_n

3 Trees

3.1 definition. Forest an *acyclic* graph, i.e. without cycles

3.2 definition. Tree connected acyclic graph

3.3 definition. Leaf vertex of degree 1



To do (6)

3.1 *Basic Properties*

3.4 proposition. ***Leafs*** *Every tree as at least one leaf (see proof below)*

3.5 proposition. ***Number of Edges*** $T_n \implies |E(T)| = n-1$ *(see proof below)*

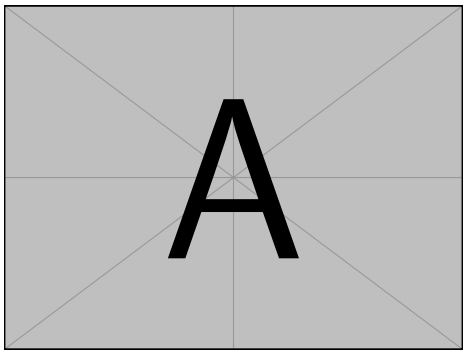
3.6 proposition. ***Connected Graph*** *Every connected graph with n vertices and* $n-1$ *edges is a Tree (see proof below)*

3.7 proposition. ***Forests*** *For a forest* F_n *with c connected components ,* $|E(T)| = n-c$ *(see proof below)*

3.8 remark. Hence, note that a tree is just a special case of a forest, where $c = 1$

3.2 *Spanning Trees*

3.9 definition. Spanning Tree Spanning subgraph which is not a tree
To do (7)



3.10 proposition. ***Necessity*** *Every connected graph contains a spanning tree (see proof below)*

as

3.11 theorem. *Cayley's Formula : A complete graph with n vertices has n^{n-2} (labelled) spanning trees*

It follows from 3.11 that , if we're interested in finding a minimum spanning tree (a weighted sp.tree of minimum weight), an exhaustive search through all possible trees becomes a gruelling task very quickly. There are however two greedy algorithms which help

3.3 Kruskal's Algorithm

For every edge not in the tree, add the one which has minimum weight and does not form a cycle. Stop when connected

Data: *this text*
Result: *how to write algorithm with LaTeX2e*
initialization;
while *not at end of this document* **do**
  read current;
  if *understand* **then**
    go to next section;
    current section becomes this one;
  else
    go back to the beginning of current section;
  end
end

Algorithm 1: How to write algorithms

3.12 theorem. *Kruskal's will always output a M.S.T*

Reproduction not examinable, merely analysis

Proof. QED

3.4 Prim's

Similar to Kruskal's but instead of looking for $min(E)$, we look for the smallest which adjacent to a node in the last iteration

4 Proof Techniques

To do...

- ☐ *1 (p. 3): Remove commented out cits*
- ☐ *2 (p. 3): Add pictorial representation*
- ☐ *3 (p. 3): proof paths*
- ☐ *4 (p. 4): methods to determine isomorphism*
- ☐ *5 (p. 4): proof #edges*
- ☐ *6 (p. 5): Take fig.1 from lectures and connect the adjacent endpoints of all trees do contrast with forest*
- ☐ *7 (p. 5): Add example of sp.sub is vs is not*
- ☐ *8 (p. 6): Add Kruskal's typesetting. Change label to header*
- ☐ *9 (p. 6): Fix broken layout, where theorem environment after caley's is spilling over the rest of the text*
- ☐ *10 (p. 8): BibTex : Diestel,Reinhard ; Graph Theory*
- ☐ *11 (p. 8): Adjust class to remove italics from prop*