Lecture 2

Representation, Sameness and Parts

Reading: Some of the material in today's lecture comes from the beginning of Chapter 1 in

> Dieter Jungnickel (2013), *Graphs, Networks and Algorithms*, 4th edition, which is available online via `SpringerLink`.

If you are at the university, either physically or via the VPN, you can download the chapters of this book as PDFs.

2.1 Ways to represent a graph

The first part of this lecture is concerned with various ways of specifying a graph. It may seem unnecessary to have so many different descriptions for a mathematical object that is, fundamentally, just a pair of finite sets, but each of the representations below will prove convenient when we are developing algorithms (step-by-step computational recipes) to solve problems involving graphs.

2.1.1 Edge lists

From the first lecture, we already how to represent a graph $G(V, E)$ by specifying its vertex set V and its *edge list* E as, for example,

Example 2.1 (Edge list representation)**.** *The undirected graph $G(V, E)$ with*

$$V = \{1, 2, 3\} \qquad \textit{and} \qquad E = \{(1, 2),\ (2, 3),\ (1, 3)\}$$

is K_3, the complete graph on three vertices. But if we regard the edges as directed then G is the graph pictured at the right of Figure 2.1

Of course, if every vertex in $G(V, E)$ appears in some edge (equivalently, if every vertex has nonzero degree), then we can dispense with the vertex set and specify the graph by its edge list alone.

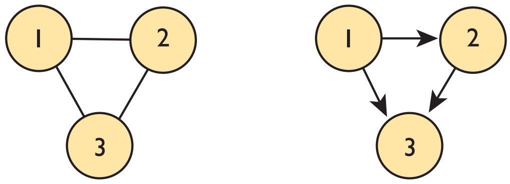


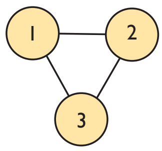
Figure 2.1: *If graph from Example 2.1 is regarded as undirected (our default assumption) then it is K_3, the complete graph on three vertices, but if it's directed then it's the digraph at right above.*

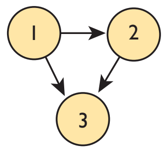
2.1.2 Adjacency matrices

A second approach is to give an *adjacency matrix*, often written A. One builds an adjacency matrix by first numbering the vertices, so that the vertex set becomes $V = \{v_1, v_2, \dots, v_n\}$ for a graph on n vertices. The adjacency matrix A is then an $n \times n$ matrix whose entries are given by the following rule:

$$A_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \in E \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

Once again, the directed and undirected cases are different. For the graphs from Example 2.1 we have:

if G is  then $A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$,

but if G is  then $A = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$.

Remark 2.2. *The following properties of the adjacency matrix follow readily from the definition in Eqn. (2.1).*

- *The adjacency matrix is not unique because it depends on a numbering scheme for the vertices. If one renumbers the vertices, the rows and columns of A will be permuted accordingly.*
- *If $G(V, E)$ is undirected then its adjacency matrix A is symmetric. That's because we think of the edges as unordered pairs, so, for example, $(1, 2) \in E$ is the same thing as $(2, 1) \in E$.*
- *If the graph has no loops then $A_{jj} = 0$ for $1 \le j \le n$. That is, there are zeroes down the main diagonal of A.*

- *One can compute the degree of a vertex by adding up entries in the adjacency matrix. I leave it as an exercise for the reader to establish that in an undirected graph,*

$$\deg(v_j) = \sum_{k=1}^{n} A_{jk} = \sum_{k=1}^{n} A_{kj}, \tag{2.2}$$

  where the first sum runs across the j-th row, while the second runs down the j-th column. Similarly, in a directed graph we have

$$\deg_{out}(v_j) = \sum_{k=1}^{n} A_{jk} \qquad \textit{and} \qquad \deg_{in}(v_j) = \sum_{k=1}^{n} A_{kj}. \tag{2.3}$$

- *Sometimes one sees a modified form of the adjacency matrix used to describe multigraphs (graphs that permit two or more edges between a given pair of vertices). In this case one takes*

$$A_{ij} \;=\; \textit{number of times the edge } (i,j) \textit{ appears in } E \tag{2.4}$$

2.1.3 Adjacency lists

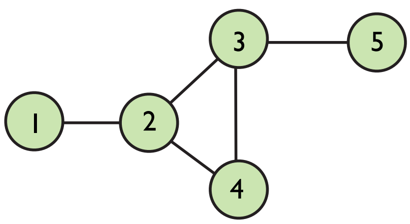
One can also specify an undirected graph by giving the *adjacency lists* of all its vertices.

Definition 2.3. *In an undirected graph $G(V, E)$ the **adjacency list** associated with a vertex v is the set $A_v \subseteq V$ defined by*

$$A_v = \{u \in V \mid (u, v) \in E\}.$$

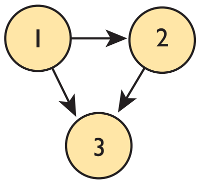
An example appears in Figure 2.2. It follows readily from the definition of degree that

$$\deg(v) = |A_v|. \tag{2.5}$$



$$\begin{aligned} A_1 &= \{2\} \\ A_2 &= \{1,\ 3,\ 4\} \\ A_3 &= \{2,\ 4,\ 5\} \\ A_4 &= \{2,\ 3\} \\ A_5 &= \{3\} \end{aligned}$$

Figure 2.2: *The graph at left has adjacency lists as shown at right.*



v	Predecessors	Successors
1	$\emptyset$	$\{2, 3\}$
2	$\{1\}$	$\{3\}$
3	$\{1, 2\}$	$\emptyset$

Figure 2.3: *The directed graph at left has the predecessor and successor lists shown at right.*

Similarly, one can specify a directed graph by providing separate lists of *successors* or *predecessors* (these terms were defined in Lecture 1) for each vertex.

Definition 2.4. *In an directed graph $G(V, E)$ the* ***predecessor list*** *of a vertex v is the set $P_v \subseteq V$ defined by*

$$P_v = \{u \in V \mid (u, v) \in E\}$$

while the ***successor list*** *of v is the set $S_v \subseteq V$ defined by*

$$S_v = \{u \in V \mid (v, u) \in E\}.$$

Figure 2.3 gives some examples. The analogues of Eqn. (2.5) for a directed graph are

$$\deg_{in}(v) = |P_v| \qquad \text{and} \qquad \deg_{out}(v) = |S_v|. \tag{2.6}$$

2.2 When are two graphs the same?

For the small graphs that appear in these notes, it's usually fairly obvious when two of them are the same. But in general it's nontrivial to be rigorous about what we mean when we say two graphs are "the same". The point is that if we stick to the abstract definition of a graph-as-two-sets, we need to formulate our definition of sameness in a similar style. Informally we'd like to say that two graphs are the same (we'll use the term *isomorphic* for this) if it is possible to relabel the vertex sets in such a way that their edge sets match up. More precisely:

Definition 2.5. *Two graphs $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$ are said to be* ***isomorphic*** *if there exists a bijection*[1] *$\alpha : V_1 \to V_2$ such that the edge $(\alpha(a), \alpha(b)) \in E_2$ if and only if $(a, b) \in E_1$.*

Generally it's difficult to decide whether two graphs are isomorphic. In particular, there are no known fast algorithms[2] (we'll learn to speak more precisely about what it means for an algorithm to be "fast" later in the term) to decide. One can,

[1] Recall that a bijection is a mapping that's one-to-one and onto.

[2] Algorithms for graph isomorphism are the subject of intense current research: see Erica Klarreich's Jan. 2017 article in *Quanta Magazine*, `Complexity Theory Problem Strikes Back`, for a popular account of some recent results.

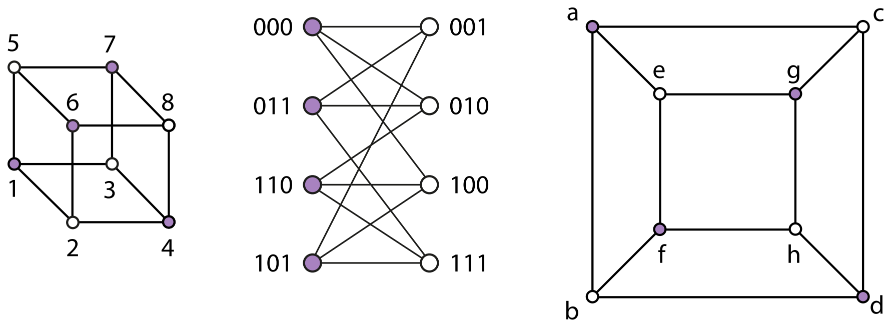


Figure 2.4: *Here are three different graphs that are all isomorphic to the cube graph I_3, which is the middle one. The bijections that establish the isomorphisms are listed in Table 2.1.*

v	000	001	010	011	100	101	110	111
$\alpha_L(v)$	1	2	3	4	5	6	7	8
$\alpha_R(v)$	a	b	c	d	e	f	g	h

Table 2.1: *If we number the graphs in Figure 2.4 so that the leftmost is $G_1(V_1, E_1)$ and the rightmost is $G_3(V_3, E_3)$, then the bijections $\alpha_L : V_2 \to V_1$ and $\alpha_R : V_2 \to V_3$ listed above establish that G_2 is isomorphic, respectively, to G_1 and G_3.*

of course, simply try all possible bijections between the two vertex sets, but there are $n!$ of these for graphs on n vertices and so this brute force approach rapidly becomes impractical. On the other hand, it's often possible to detect quickly that two graphs *aren't* isomorphic. The simplest such tests are based on the following propositions, whose proofs are left to the reader.

Proposition 2.6. *If $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$ are isomorphic then $|V_1| = |V_2|$ and $|E_1| = |E_2|$.*

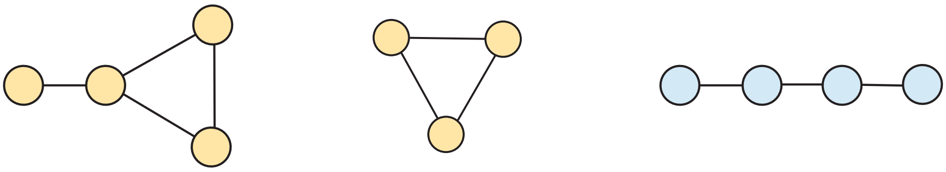
Proposition 2.7. *If $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$ are isomorphic and $\alpha : V_1 \to V_2$ is the bijection that establishes the isomorphism, then $\deg(v) = \deg(\alpha(v))$ for every $v \in V_1$ and $\deg(u) = \deg(\alpha^{-1}(u))$ for every $u \in V_2$.*

Another simple test depends on the following quantity, examples of which appear in Figure 2.5.

Definition 2.8. *The **degree sequence** of an undirected graph $G(V, E)$ is a list of the degrees of the vertices, arranged in ascending order.*

The corresponding test for non-isomorphism depends on the following proposition, whose proof is left as an exercise.

Proposition 2.9. *If $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$ are isomorphic then they have the same degree sequence.*



(1, 2, 2, 3) (2, 2, 2) (1, 1, 2, 2)

Figure 2.5: *Three small graphs and their degree sequences.*

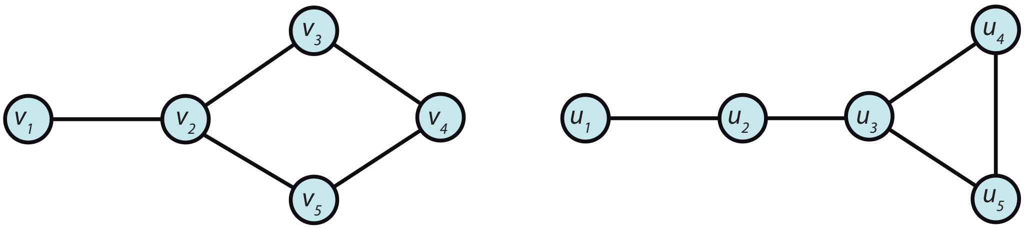


Figure 2.6: *These two graphs both have degree sequence* $(1, 2, 2, 2, 3)$*, but they're not isomorphic: see Example 2.10 for a proof.*

Unfortunately although it's a necessary condition for two isomorphic graphs to have the same degree sequence, a shared degree sequence isn't sufficient to establish isomorphism. That is, it's possible for two graphs to have the same degree sequence, but not be isomorphic: Figure 2.6 shows one such pair, but it's easy to make up more.

Example 2.10 (Proof that the graphs in Figure 2.6 aren't isomorphic)**.** *Both graphs in Figure 2.6 have the same degree sequence,* $(1, 2, 2, 2, 3)$*, so both contain a single vertex of degree 1 and a single vertex of degree 3. These vertices are adjacent in the graph at left, but not in the one at right and this observation forms the basis for a proof by contradiction that the graphs aren't isomorphic.*

Assume, for contradiction, that they are isomorphic and that

$$\alpha : \{v_1, v_2, v_3, v_4, v_5\} \to \{u_1, u_2, u_3, u_4, u_5\}$$

is the bijection that establishes the isomorphism. Then Prop. 2.7 implies that it must be true that $\alpha(v_1) = u_1$ *(as these are the sole vertices of degree one) and* $\alpha(v_2) = u_3$*. But then the presence of the edge* (v_1, v_2) *on the left would imply the existence of an edge* $(\alpha(v_1), \alpha(v_2)) = (u_1, u_3)$ *on the right, and no such edge exists. This contradicts our assumption that* α *establishes an isomorphism, so no such* α *can exist and the graphs aren't isomorphic.*

2.3 Terms for parts of graphs

Finally, we'll often want to speak of parts of graphs and the two most useful definitions here are:

Definition 2.11. *A* ***subgraph*** *of a graph* $G(V, E)$ *is a graph* $G'(V', E')$ *where* $V' \subseteq V$ *and* $E' \subseteq E$.

and

Definition 2.12. *Given a graph* $G(V, E)$ *and a subset of its vertices* $V' \subseteq V$*, the* ***subgraph induced by*** V' *is the subgraph* $G'(V', E')$ *where*

$$E' = \{(u, v) \mid u, v \in V' \text{ and } (u, v) \in E\}.$$

That is, the subgraph induced by the vertices V' consists of V' itself and *all* those edges in the original graph that involve only vertices from V'. Both these definitions are illustrated in Figure 2.7.

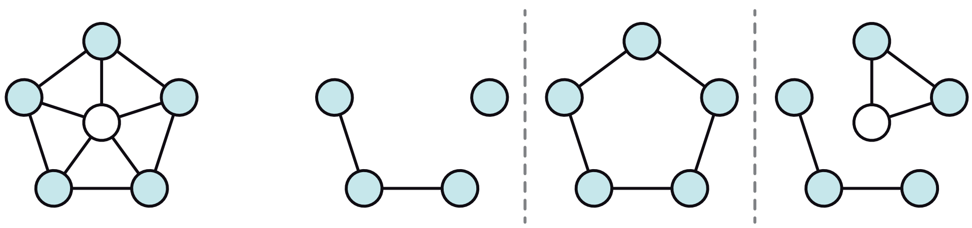


Figure 2.7: *The three graphs at right are subgraphs of the one at left. The middle one is the subgraph induced by the blue shaded vertices.*