Introduction to Graph Theory

George Voutsadakis[1]

[1]Mathematics and Computer Science
Lake Superior State University

LSSU Math 351

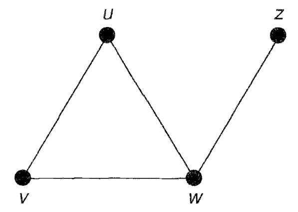
Subsection 1

Definition

Simple Graphs

- A **simple graph** G consists of:
  - A non-empty finite set $V(G)$ of elements called **vertices** (or **nodes**);
  - A finite set $E(G)$ of distinct unordered pairs of distinct elements of $V(G)$ called **edges**.
- We call $V(G)$ the **vertex set** and $E(G)$ the **edge set** of G.
- An edge $\{v, w\}$ is said to **join** the vertices v and w, and is usually abbreviated to vw.

Example: The simple graph G has:

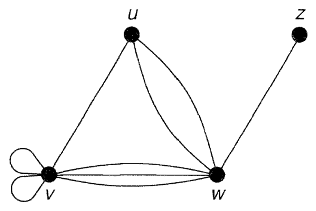


- Vertex set $V(G) = \{w, v, w, z\}$;
- Edge set $E(G)$ consisting of the edges uv, uw, vw and wz.

General Graphs

- In any simple graph there is at most one edge joining a given pair of vertices.
- Many results valid for simple graphs can be extended to more general objects in which two vertices may have several edges joining them.
- In addition, we may remove the restriction that an edge joins two distinct vertices, and allow **loops** - edges joining a vertex to itself.
- The resulting object, in which loops and multiple edges are allowed, is called a **general graph** or, simply, a **graph**.

Example:

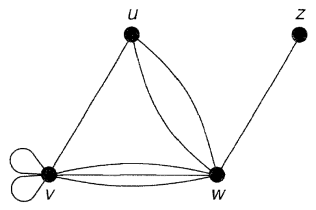


Every simple graph is a graph, but not every graph is a simple graph.

General Graphs: Terminology

- A **graph** G consists of a non-empty finite set $V(G)$ of elements called **vertices**, and a finite family (multiset) $E(G)$ of unordered pairs of (not necessarily distinct) elements of $V(G)$ called **edges**;
- The use of the word "family" allows multiple edges.
- We call $V(G)$ the **vertex set** and $E(G)$ the **edge family** of G.
- An edge $\{v, w\}$ is said to **join** the vertices v and w, and is again abbreviated to vw.

  Example:

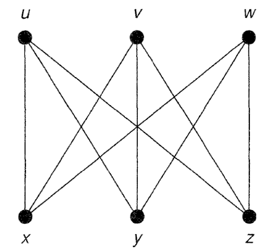


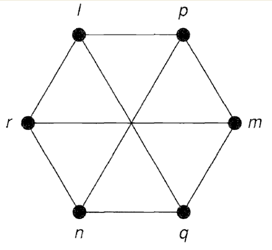
  $V(G)$ is the set $\{u, v, w, z\}$. $E(G)$ consists of the edges uv, vv (twice), vw (three times), uw (twice), and wz. Note that each loop vv joins the vertex v to itself.
- We sometimes have to restrict our attention to simple graphs, but we prove our results for general graphs whenever possible.

Isomorphism

- Two graphs G_1 and G_2 are **isomorphic** if there is a one-one correspondence between the vertices of G_1 and those of G_2, such that the number of edges joining any two vertices of G_1 is equal to the number of edges joining the corresponding vertices of G_2.

  Example:





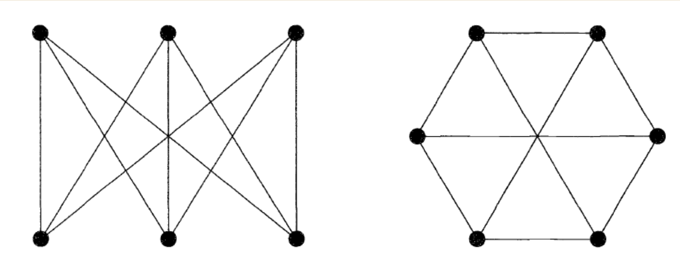
  The following correspondence is an isomorphism:
  $u \leftrightarrow l, v \leftrightarrow m, w \leftrightarrow n, x \leftrightarrow p, y \leftrightarrow q, z \leftrightarrow r.$

Isomorphism Between Unlabeled Graphs

- For many problems, the labels on the vertices are unnecessary and we drop them.

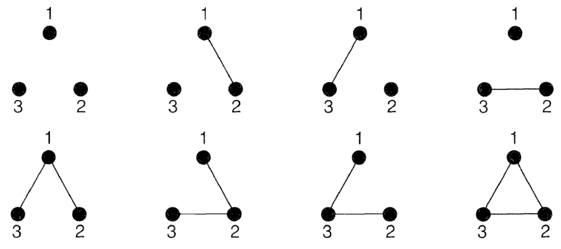
  We then say that two "unlabeled graphs" are isomorphic if we can assign labels so that the resulting "labeled graphs" are isomorphic.

  Example:

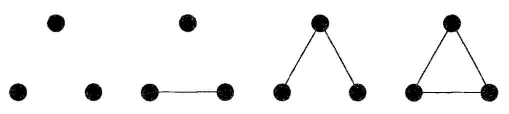


Counting Labeled and Unlabeled Graphs

- The difference between labeled and unlabeled graphs becomes more apparent when we try to count them.
  - If we restrict ourselves to graphs with three vertices, then there are, up to isomorphism, eight different labeled graphs:



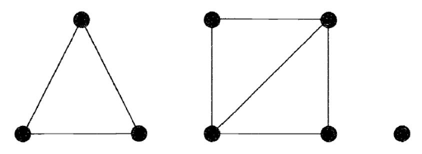
but only four unlabeled ones:



Union and Connectedness

- If the two graphs are $G_1 = (V(G_1), E(G_1))$ and $G_2 = (V(G_2), E(G_2))$, where $V(G_1)$ and $V(G_2)$ are disjoint, then their **union** $G_1 \cup G_2$ is the graph with vertex set $V(G_1) \cup V(G_2)$ and edge family $E(G_1) \cup E(G_2)$.
- A graph is **connected** if it cannot be expressed as the union of two graphs, and **disconnected**, otherwise.
- Clearly any disconnected graph G can be expressed as the union of connected graphs, each of which is a **component** of G.

  Example: A graph with three components is shown below:

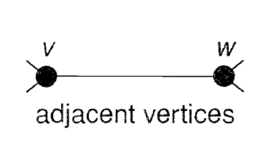


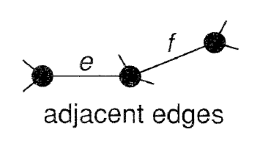
- When proving results about graphs in general, we can often obtain the corresponding results for connected graphs and then apply them to each component separately.

Adjacency and Incidence

- We say that two vertices v and w of a graph G are **adjacent** if there is an edge vw joining them.

  The vertices v and w are then **incident** with such an edge.





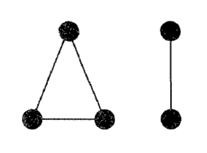
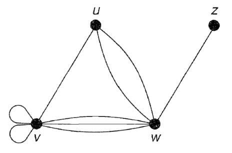
- Similarly, two distinct edges e and f are **adjacent** if they have a vertex in common.

Degrees and Degree Sequence

- The **degree** of a vertex v of G is the number of edges incident with v, and is written $\deg(v)$;

  In calculating the degree of v, we usually make the convention that a loop at v contributes 2 (rather than 1) to the degree of v.
- A vertex of degree 0 is an **isolated vertex** and a vertex of degree 1 is an **end-vertex**.

  Example: Each of the two following graphs

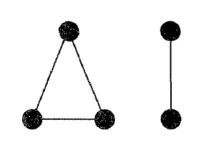
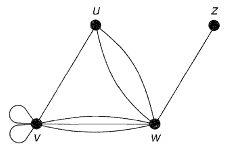
  has two end-vertices and three vertices of degree 2.

  The third graph has one end-vertex, one vertex of degree 3, one of degree 6 and one of degree 8.

Degree Sequence and Handshaking Lemma

- The **degree sequence** of a graph consists of the degrees written in increasing order, with repeats where necessary.

  Example: The degree sequence of the first two graphs below is $(1, 1, 2, 2, 2)$ and of the third graph $(1, 3, 6, 8)$.

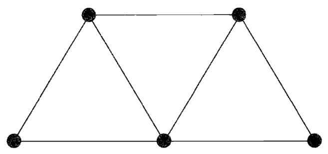
- **The Handshaking Lemma**: In any graph the sum of all the vertex-degrees is an even number - in fact, twice the number of edges.

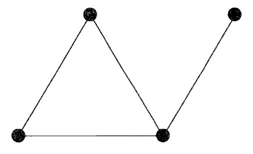
  This happens since each edge contributes exactly 2 to the sum.
- As a corollary, in any graph the number of vertices of odd degree is even.

Subgraphs

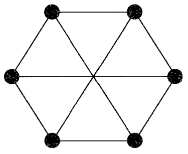
- A **subgraph** of a graph G is a graph, each of whose vertices belongs to $V(G)$ and each of whose edges belongs to $E(G)$.

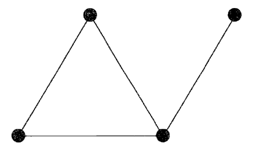
  Example: The graph on the right is a subgraph of the graph on the left.





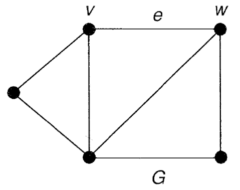
  The graph on the right is not a subgraph of the graph on the left, since the latter graph contains no "triangle".

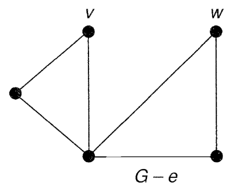


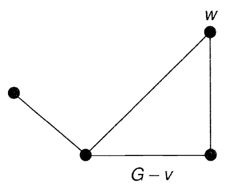


Edge and Vertex Deletions

- We can obtain subgraphs of a graph by deleting edges and vertices:
  - If e is an edge of a graph G, we denote by $G - e$ the graph obtained from G by deleting the edge e.
  - More generally, if F is any set of edges in G, we denote by $G - F$ the graph obtained by deleting the edges in F.
  - If v is a vertex of G, we denote by $G - v$ the graph obtained from G by deleting the vertex v together with the edges incident with v.
  - More generally, if S is any set of vertices in G, we denote by $G - S$ the graph obtained by deleting the vertices in S and all edges incident with any of them.



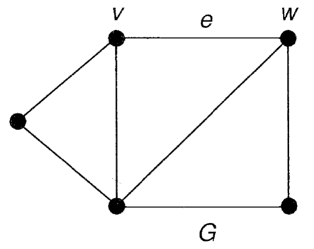


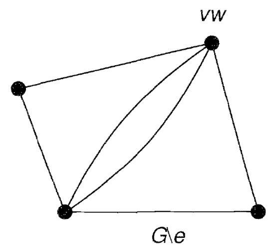


Edge Contractions

- We denote by $G\backslash e$ the graph obtained by taking an edge e and **contracting it**, i.e., removing it and identifying its ends v and w so that the resulting vertex is incident with those edges (other than e) that were originally incident with v or w.

  Example:

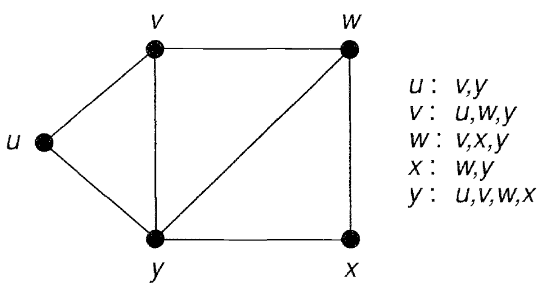




Adjacency List Representation

- One way of storing a simple graph in a computer is by listing the vertices adjacent to each vertex of the graph.

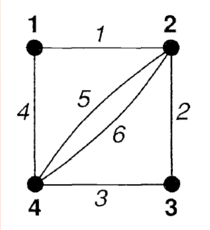
  Example:



Matrix Representation

- If G is a graph with vertices labeled $\{\mathbf{1}, \mathbf{2}, \ldots, \boldsymbol{n}\}$, its **adjacency matrix A** is the $n \times n$ matrix whose ij-th entry is the number of edges joining vertex $\boldsymbol{i}$ and vertex $\boldsymbol{j}$.
- If, in addition, the edges are labeled $\{1, 2, \ldots, m\}$, its **incidence matrix M** is the $n \times m$ matrix whose ij-th entry is 1 if vertex $\boldsymbol{i}$ is incident to edge j, and 0 otherwise.

  Example:



$$\mathbf{A} = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 2 \\ 0 & 1 & 0 & 1 \\ 1 & 2 & 1 & 0 \end{pmatrix}$$

$$\mathbf{M} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

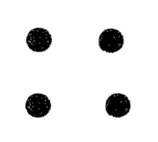
Subsection 2

Examples

Null Graphs

- A graph whose edge-set is empty is a **null graph**.
- We denote the null graph on n vertices by N_n.

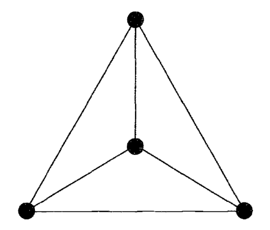
  Example: Here is N_4:

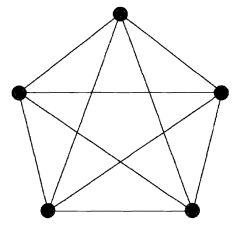


- Note that each vertex of a null graph is isolated.

Complete Graphs

- A simple graph in which each pair of distinct vertices are adjacent is a **complete graph**.
- We denote the complete graph on n vertices by K_n;
- Example: K_4 and K_5 are shown:



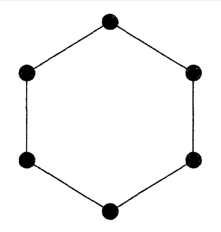
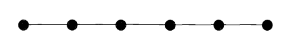
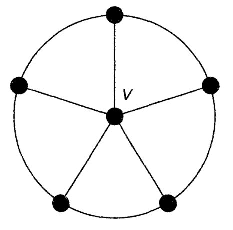


- The complete graph K_n on n vertices has $\dfrac{n(n-1)}{2}$ edges.

Cycle Graphs, Path Graphs and Wheels

- A connected graph that is regular of degree 2 is a **cycle graph**.
- We denote the cycle graph on n vertices by C_n.
- The graph obtained from C_n by removing an edge is the **path graph** on n vertices, denoted by P_n.
- The graph obtained from C_{n-1} by joining each vertex to a new vertex v is the **wheel** on n vertices, denoted by W_n.

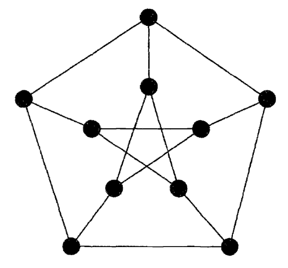
  Example: The graphs C_6, P_6 and W_6 are shown below:

Regular Graphs

- A graph in which each vertex has the same degree is a **regular graph**.
- If each vertex has degree r, the graph is **regular of degree** r or r-**regular**.
- The **cubic graphs** are the 3-regular graphs.

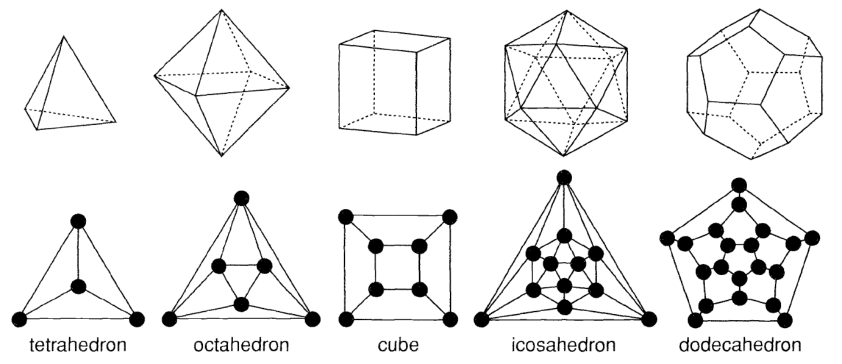
  Example: The **Petersen graph** is an important example of a cubic graph:



The null graph N_n is regular of degree 0, the cycle graph C_n is regular of degree 2, and the complete graph K_n is regular of degree $n-1$.

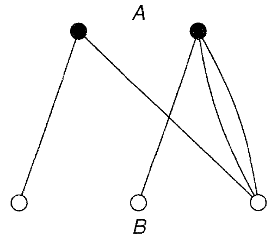
Platonic Graphs

- The **Platonic graphs** are the regular graphs formed from the vertices and edges of the five regular (Platonic) solids - the tetrahedron, octahedron, cube, icosahedron and dodecahedron:



Bipartite Graphs

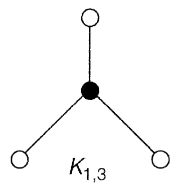
- If the vertex set of a graph G can be split into two disjoint sets A and B so that each edge of G joins a vertex of A and a vertex of B, then G is a **bipartite graph**.

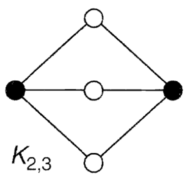


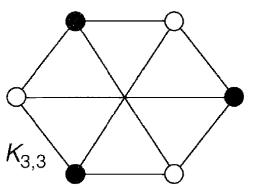
- Alternatively, a bipartite graph is one whose vertices can be colored black and white in such a way that each edge joins a black vertex (in A) and a white vertex (in B).

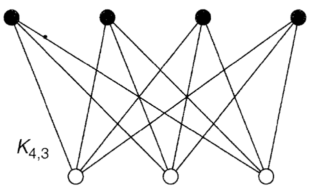
Complete Bipartite Graphs

- A **complete bipartite graph** is a bipartite graph in which each vertex in A is joined to each vertex in B by just one edge.
- We denote the bipartite graph with r black vertices and s white vertices by $K_{r,s}$;
  Example:







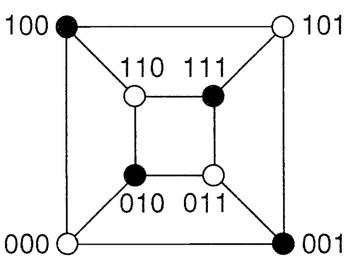


- $K_{r,s}$ has $r + s$ vertices and rs edges.

Cubes

- The k-**cube** Q_k is the graph whose vertices correspond to the sequences $(a_1, a_2, \ldots, a_k)$, where each $a_i = 0$ or 1, and whose edges join those sequences that differ in just one place.

  Example: The graph of the cube is the graph Q_3:

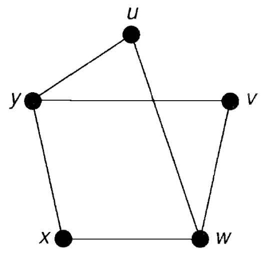


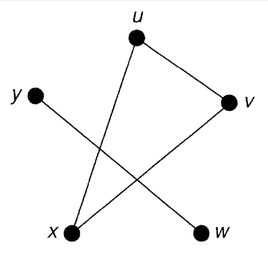
- Q_k has 2^k vertices and $k \cdot 2^{k-1}$ edges, and is regular of degree k.

The Complement of a Simple Graph

- If G is a simple graph with vertex set $V(G)$, its **complement** $\overline{G}$ is the simple graph with vertex set $V(G)$ in which two vertices are adjacent if and only if they are not adjacent in G.

  Example:





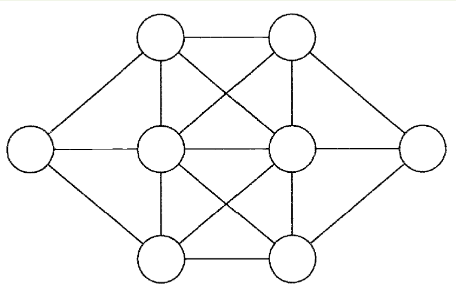
- Note that:
  - The complement of a complete graph is a null graph;
  - The complement of a complete bipartite graph is the union of two complete graphs.

Subsection 3

Three Puzzles

The Eight Circles Problem

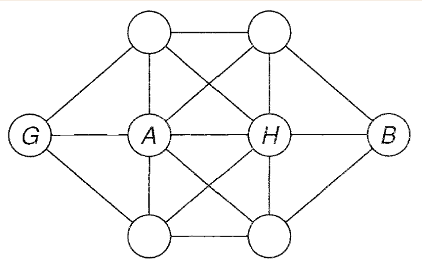
- Place the letters A, B, C, D, E, F, G, H into the eight circles in the figure,



  in such a way that no letter is adjacent to a letter that is next to it in the alphabet.
- Trying all the possibilities is not a practical proposition, as there are $8! = 40320$ ways of placing eight letters into eight circles.

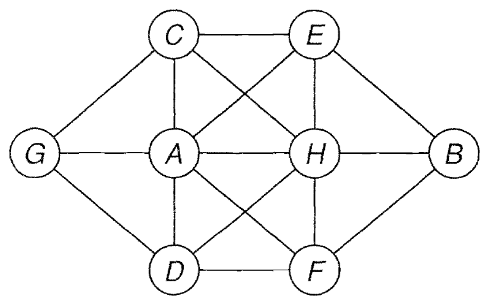
Placing A,H and B,G

- Note that:
  - (i) the easiest letters to place are A and H, because each has only one letter to which it cannot be adjacent (namely, B and G, respectively);
  - (ii) the hardest circles to fill are those in the middle, as each is adjacent to six others.
- This suggests that we place A and H in the middle circles.
- If we place A to the left of H, then the only possible positions for B and G are as shown below:



Completing the Puzzle

- The letter C must now be placed on the left-hand side of the diagram, and F must be placed on the right-hand side.
- It is then a simple matter to place the remaining letters:



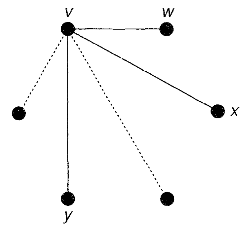
Six People at a Party

Six People at a Party: Show that, in any gathering of six people, there are either three people who all know each other or three people none of whom knows either of the other two.

- To solve this, we draw a graph in which:
  - Each person is represented by a vertex;
  - Two vertices are joined by:
    - a solid edge, if the corresponding people know each other;
    - by a dotted edge if not.

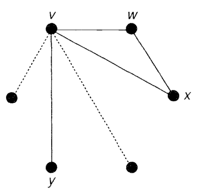
We show that there is always a solid triangle or a dotted triangle.

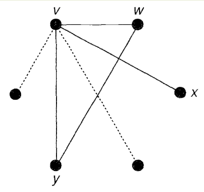
- Let v be any vertex.
  Then there must be exactly five edges incident with v, either solid or dashed.
  So at least three of these edges must be of the same type, say solid (dashed works analogously).

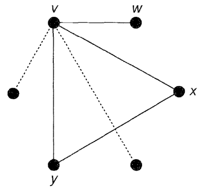


Existence of a Solid or a Dashed Triangle

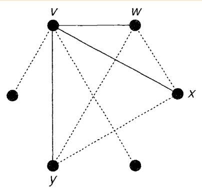
- If the people corresponding to the vertices w and x know each other, then v, w and x form a solid triangle, as required.





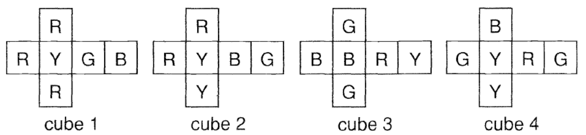


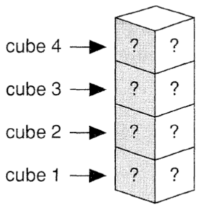
- Similarly, if the people corresponding to w and y, or to x and y, know each other, then we again obtain a solid triangle.
- Finally, if no two of the people corresponding to the vertices w, x and v know each other, then w, x and y form a dotted triangle.



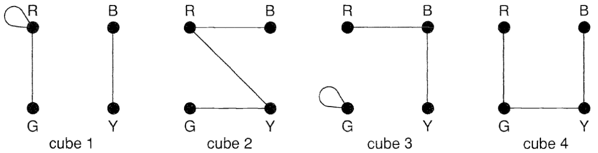
The Four Cubes Problem

Instant Insanity: Given four cubes whose faces are colored red, blue, green and yellow, can we pile them up so that all four colors appear on each side of the resulting 4×1 stack?



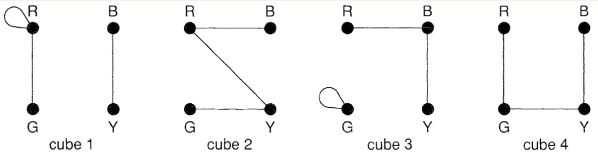


- We represent each cube by a graph with four vertices, R, B, G and Y, one for each color, in which two vertices are adjacent if and only if the cube in question has the corresponding colors on opposite faces:

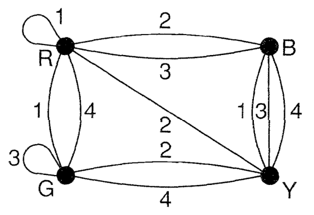


Superimposing the Graphs

- We next superimpose the graphs



to form a new graph G:



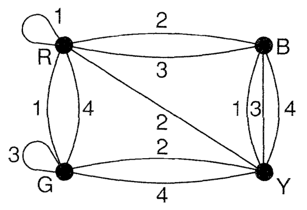
- A solution of the puzzle is obtained by finding two subgraphs H_1 and H_2 of G.
  - The subgraph H_1 tells us which pair of colors appears on the front and back of each cube;
  - The subgraph H_2 tells us which pair of colors appears on the left and right.

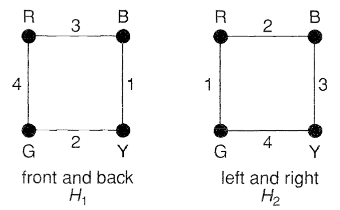
Properties of Required Subgraphs

- To fulfill their role the subgraphs H_1 and H_2 have the following properties:
  (a) each subgraph contains exactly one edge from each cube; this ensures that each cube has a front and back, and a left and right, and the subgraphs tell us which pairs of colors appear on these faces.
  (b) the subgraphs have no edges in common; this ensures that the faces on the front and back are different from those on the sides.
  (c) each subgraph is regular of degree 2; this tells us that each color appears:
    - exactly twice on the sides of the stack (once on each side);
    - exactly twice on the front and back (once on the front and once on the back).

A Choice of Subgraphs and the Corresponding Solution

- The subgraphs corresponding to our particular example:





The corresponding solution: