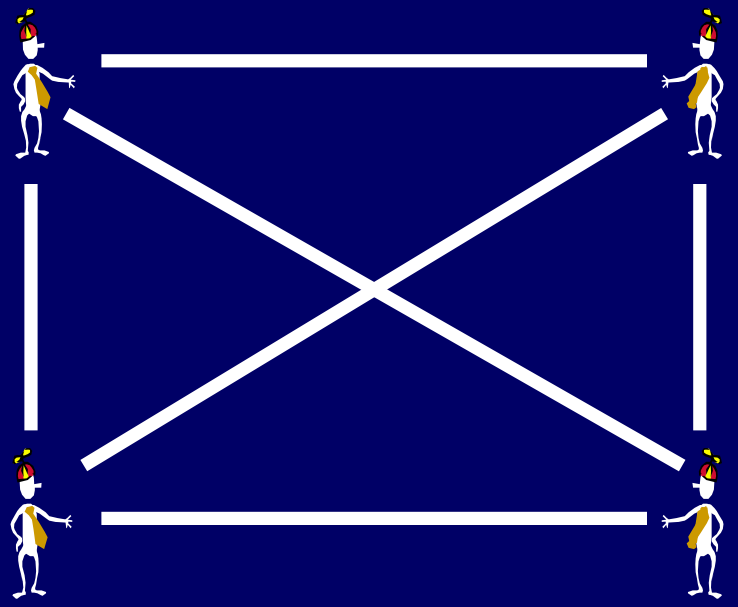
Great Theoretical Ideas In Computer Science

John Lafferty

CS 15-251 Fall 2006

Lecture 21 November 7, 2006 Carnegie Mellon University

Graphs II



Recap

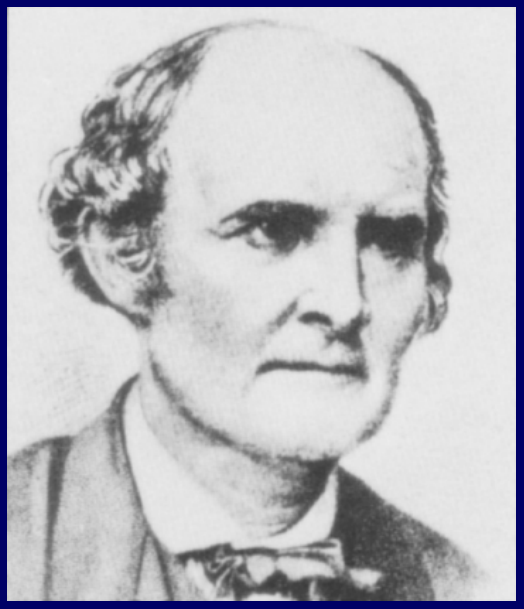
Theorem: Let G be a graph with n nodes and e edges.

The following are equivalent:

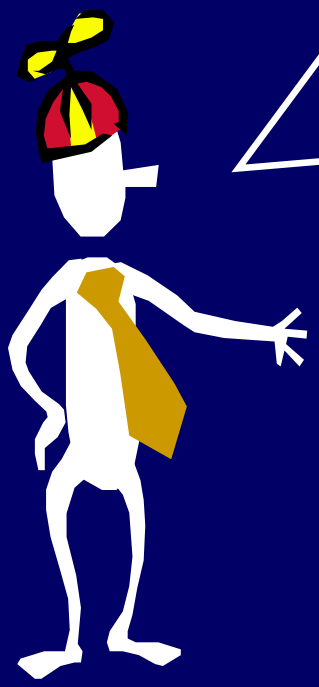
1. G is a tree (connected, acyclic)
2. Every two nodes of G are joined by a unique path
3. G is connected and $n = e + 1$
4. G is acyclic and $n = e + 1$
5. G is acyclic and if any two nonadjacent points are joined by a line, the resulting graph has exactly one cycle.

Cayley's formula

The number of labeled trees on n nodes is



$$n^{n-2}$$



A graph is *planar* if it can be drawn in the plane without crossing edges. A *plane graph* is any such drawing, which breaks up the plane into a number f of *faces* or regions

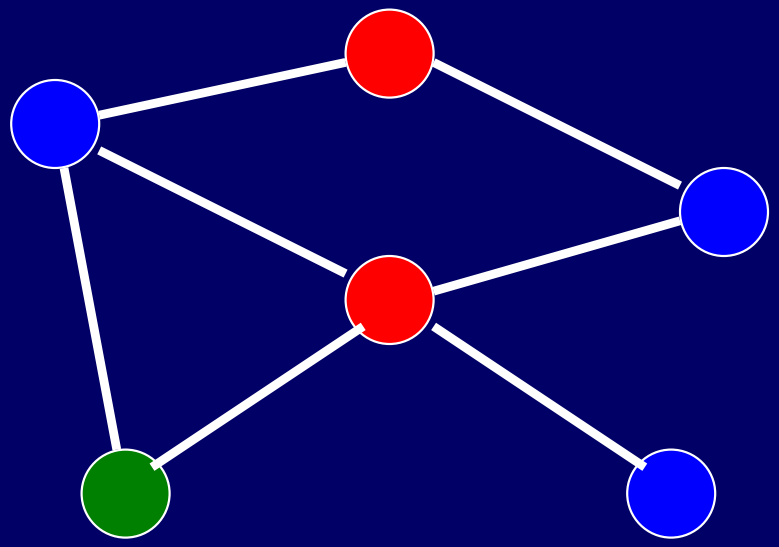
Euler's Formula

If G is a connected plane graph with n vertices, e edges and f faces, then $n - e + f = 2$

Euler's Formula: If G is a connected plane graph with n vertices, e edges and f faces, then $n - e + f = 2$

Graph Coloring

A coloring of a graph is an assignment of a color to each vertex such that no neighboring vertices have the same color.



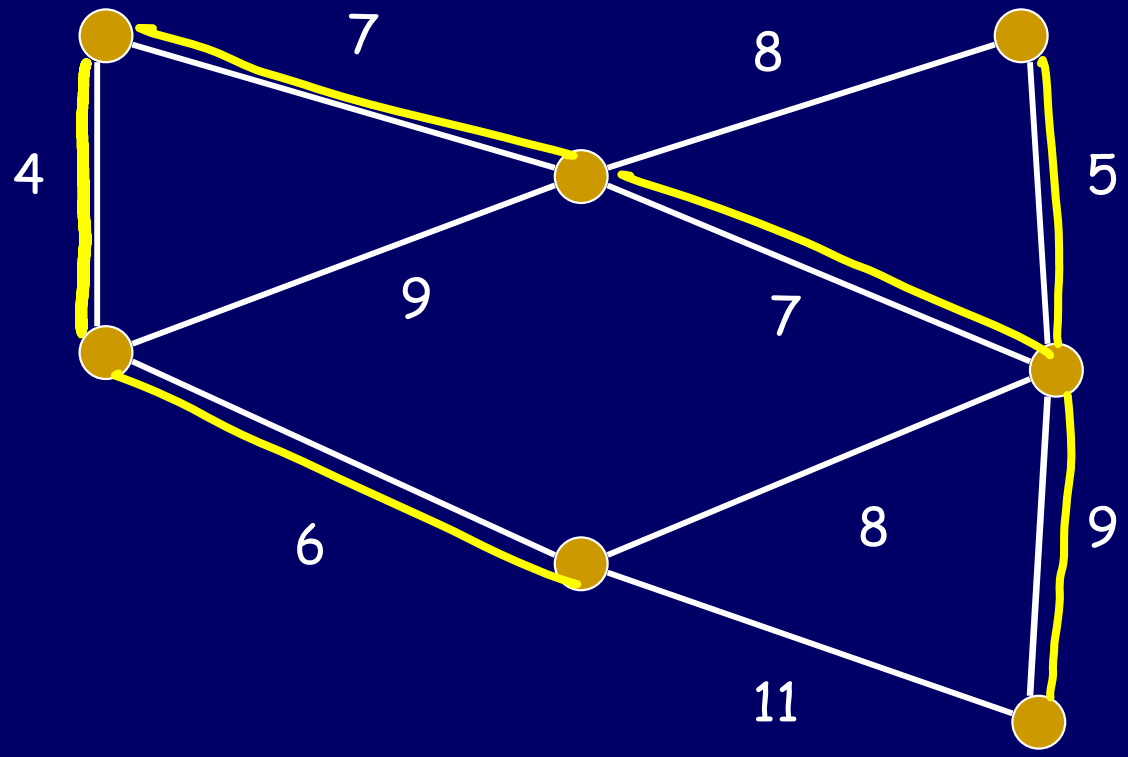
Finding Optimal Trees

- Trees have many nice properties (uniqueness of paths, no cycles, etc.)
- May want to compute the "best" tree approximation to a graph
- If all we care about is *communication*, then a tree may be enough. Want tree with smallest communication link costs.

Finding Optimal Trees

Problem: Find a *minimum spanning tree*, that is, a tree that has a node for every node in the graph, such that the sum of the edge weights is minimum.

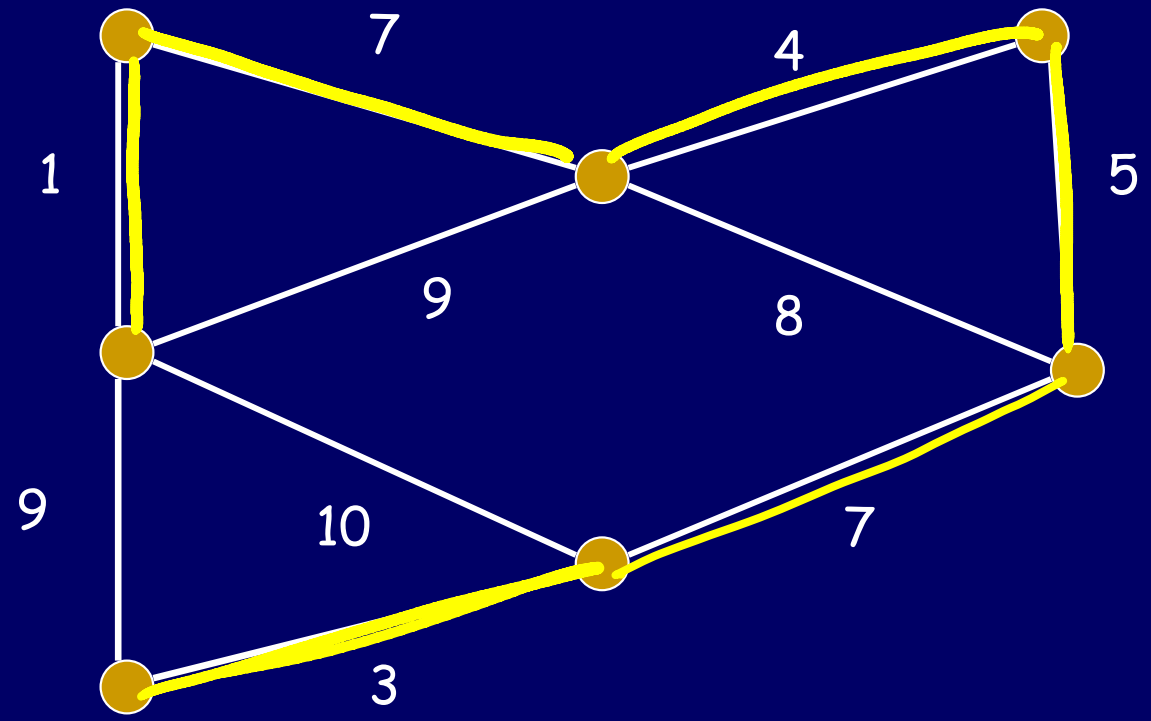
Tree Approximations



Finding an MST: Kruskal's Algorithm

- Create a forest where each node is a separate tree
- Make a sorted list of edges S
- While S is non-empty:
  - Remove an edge with minimal weight
  - If it connects two different trees, add the edge. Otherwise discard it.

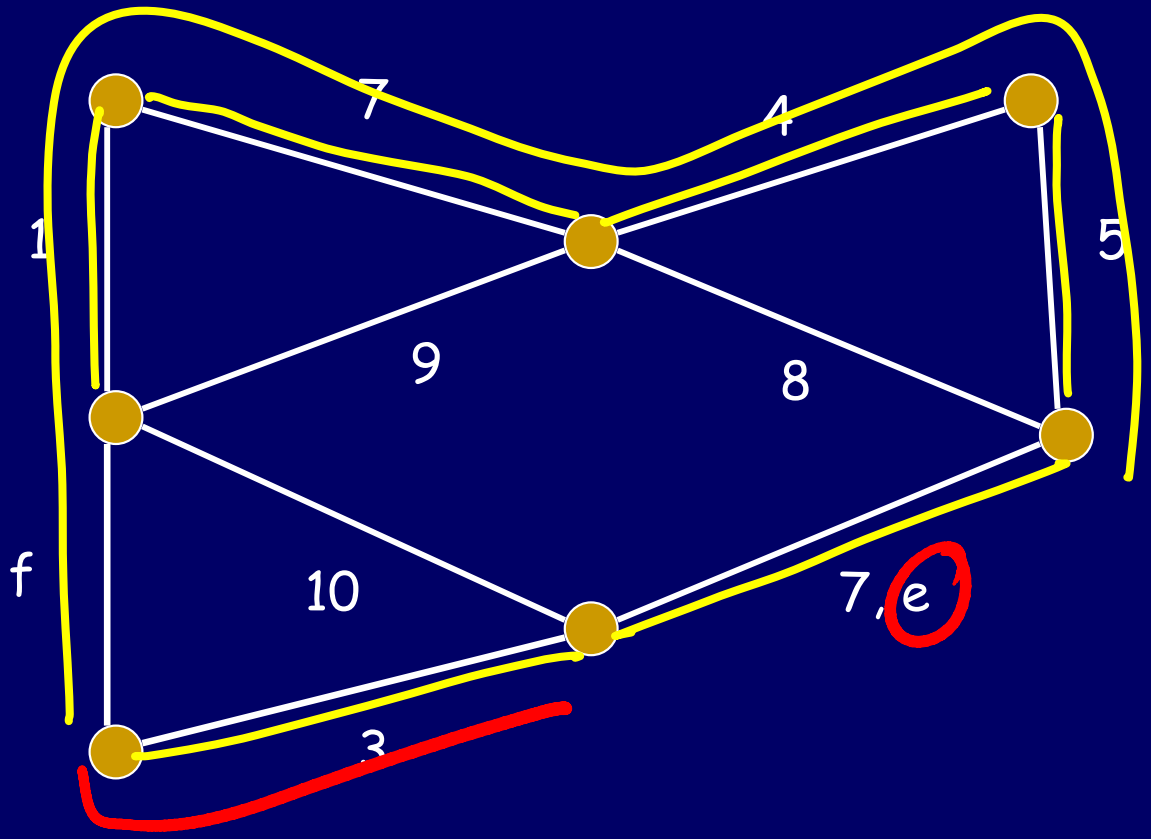
Applying the Algorithm



Analyzing the Algorithm

- The algorithm outputs a spanning tree T. Suppose that it's not minimal. (For simplicity, assume all edge weights in graph are distinct).
- Let M be a minimum spanning tree.
- Let e be the first edge chosen by the algorithm that is not in M. If we add e to M, it creates a cycle. Since this cycle isn't fully contained in T, it has an edge f not in T.
- N = M+e-f is another spanning tree.

Analyzing the Algorithm



Analyzing the Algorithm

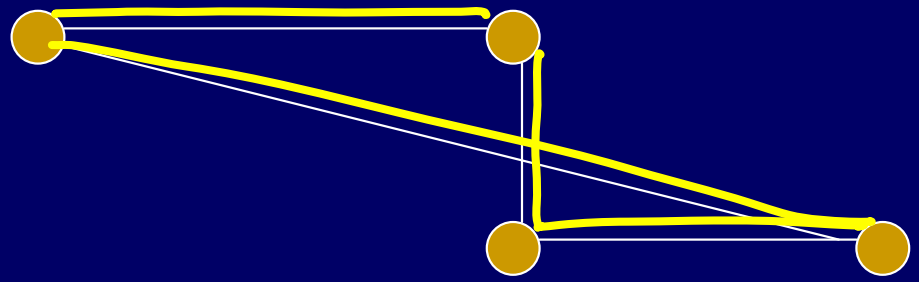
- N = M+e-f is another spanning tree.
- Claim: e < f, and therefore N < M
- Suppose not: e > f
- Then f would have been visited before e by the algorithm, but not added, because adding it would have formed a cycle.
- But all of these cycle edges are also edges of M, since e was the first edge not in M. This contradicts the assumption M is a tree.

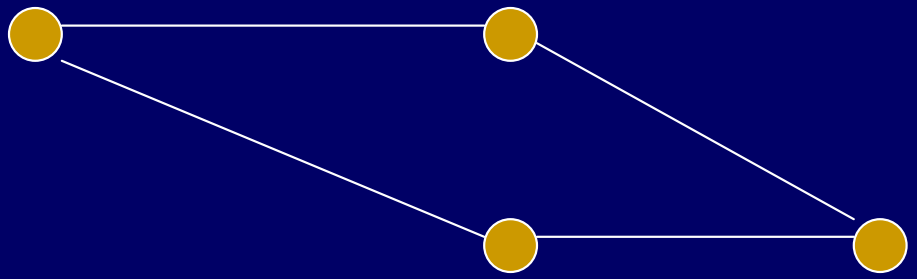
Greed is Good (In this case…)

The greedy algorithm, by adding the least costly edges in each stage, succeeds in finding an MST.

But--in math and life--if pushed too far, the greedy approach can lead to bad results.

The Greedy Traveling Salesman





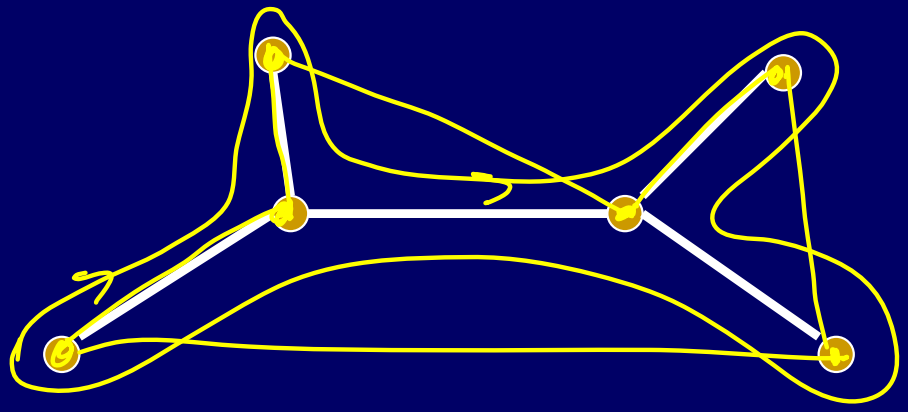
Tours from Trees

We can use an MST to derive a tour that is no more expensive than *twice* the optimal tour.

Idea: walk "around" the MST and take shortcuts if a node has already been visited.

We assume that all pairs of nodes are connected, and edge weights satisfy the *triangle inequality* d(x,y) <= d(x,z) + d(z,y)

Tours from Trees



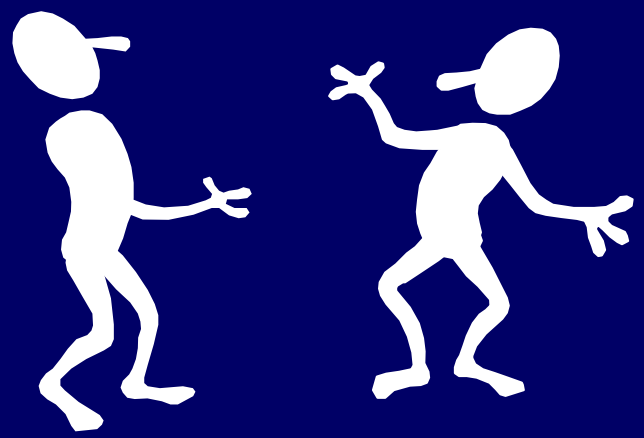
Shortcuts only decrease the cost, so

$$\text{Cost(Greedy Tour)} \leq 2\,\text{Cost(MST)}$$
$$\leq 2\,\text{Cost(Optimal Tour)}$$

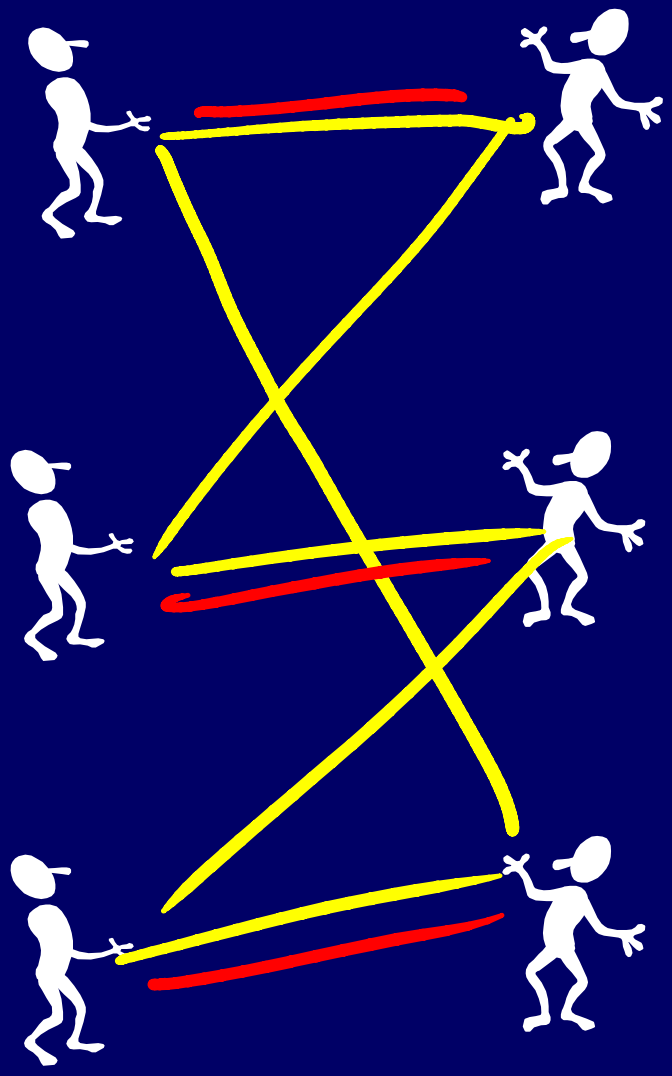
This is a *2-competitive* algorithm

Dancing Partners

A group of 100 boys and girls attend a dance. Every boy knows 5 girls, and every girl knows 5 boys. Can they be matched into dance partners so that each pair knows each other?



Dancing Partners



Perfect Matchings

Theorem: If every node in a bipartite graph has the same degree d >= 1, then the graph has a perfect matching.

Note: if degrees are the same then |A| = |B|, where A is the set of nodes "on the left" and B is the set of nodes "on the right"

A Matter of Degree

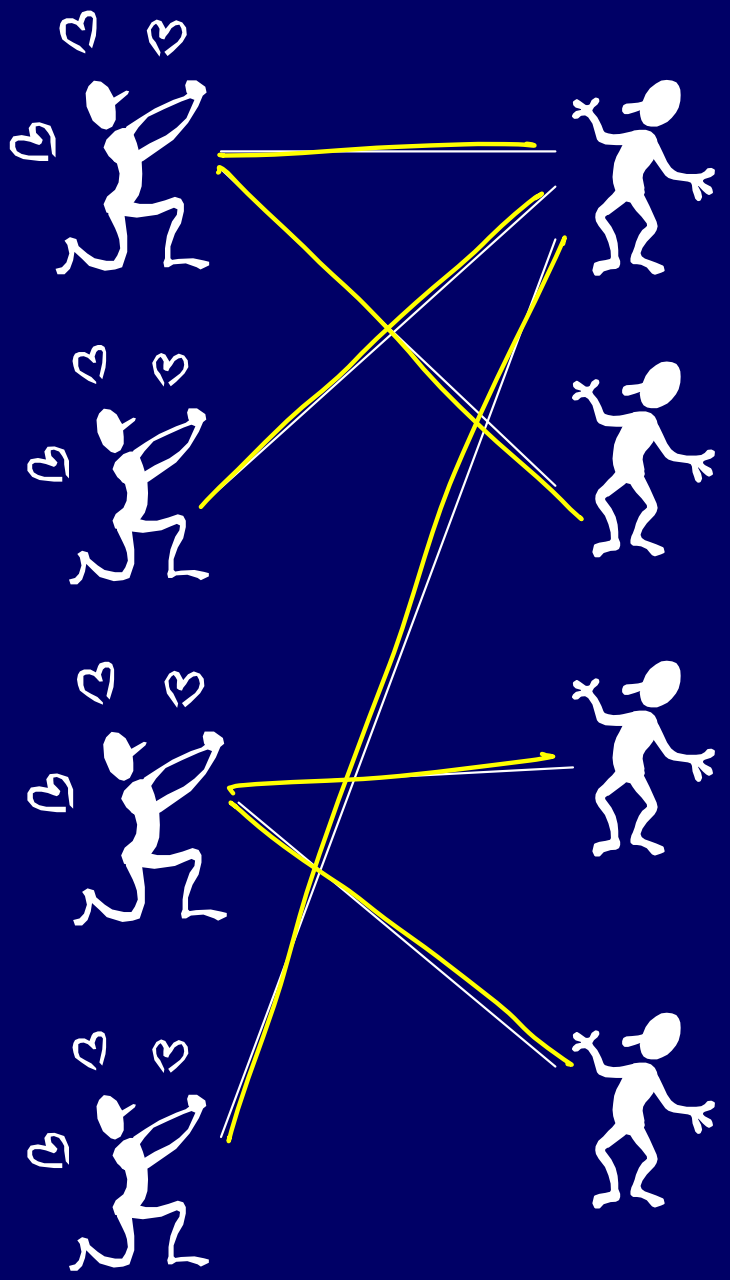
Claim: If degrees are the same then |A| = |B|.

If there are m boys, there are md edges.

n girls, " " nd edges.

The Marriage Theorem

(or, "What's Love Got to Do With it?")



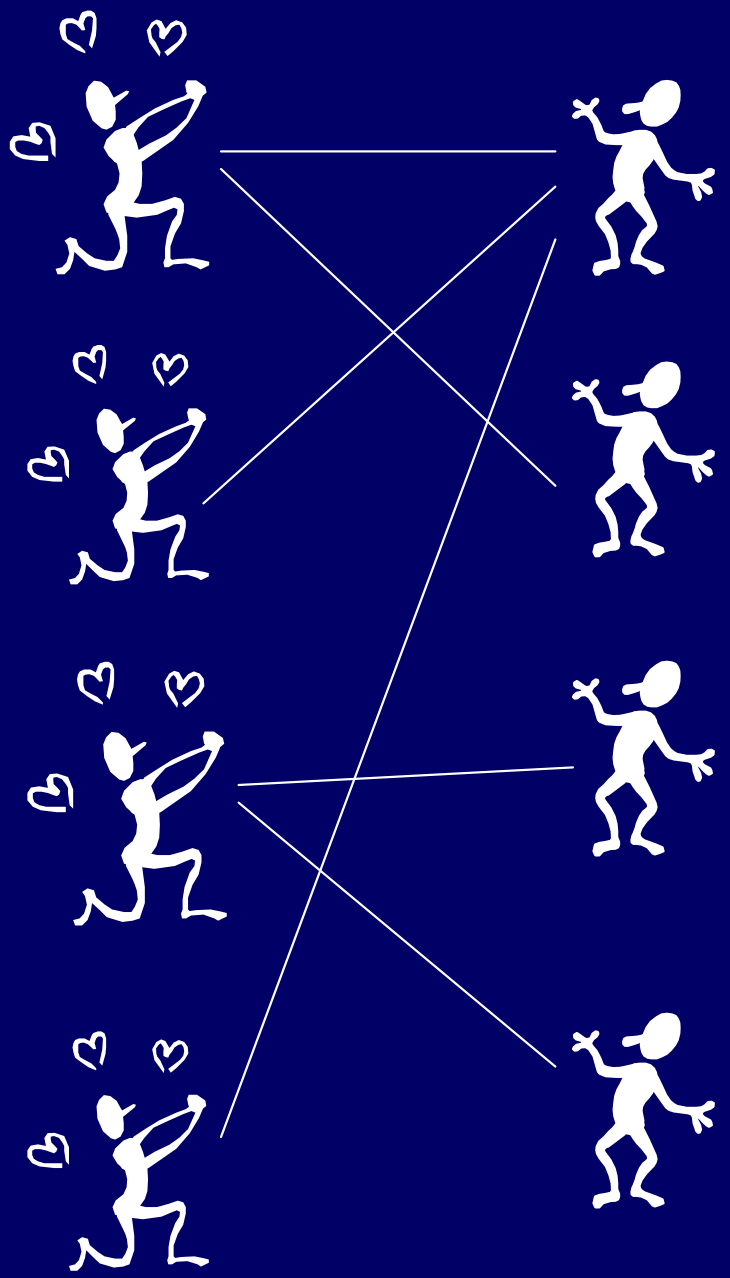
Each woman would happily marry some subset of the men, and any man would be happy to marry any woman who would be happy with him.

Is it possible to match the men and women into pairs of happy couples?

The Marriage Theorem

Theorem: A bipartite graph has a perfect matching if and only if $|A| = |B|$ and for any subset of (say) k nodes of A there are at least k nodes of B that are connected to at least one of them.

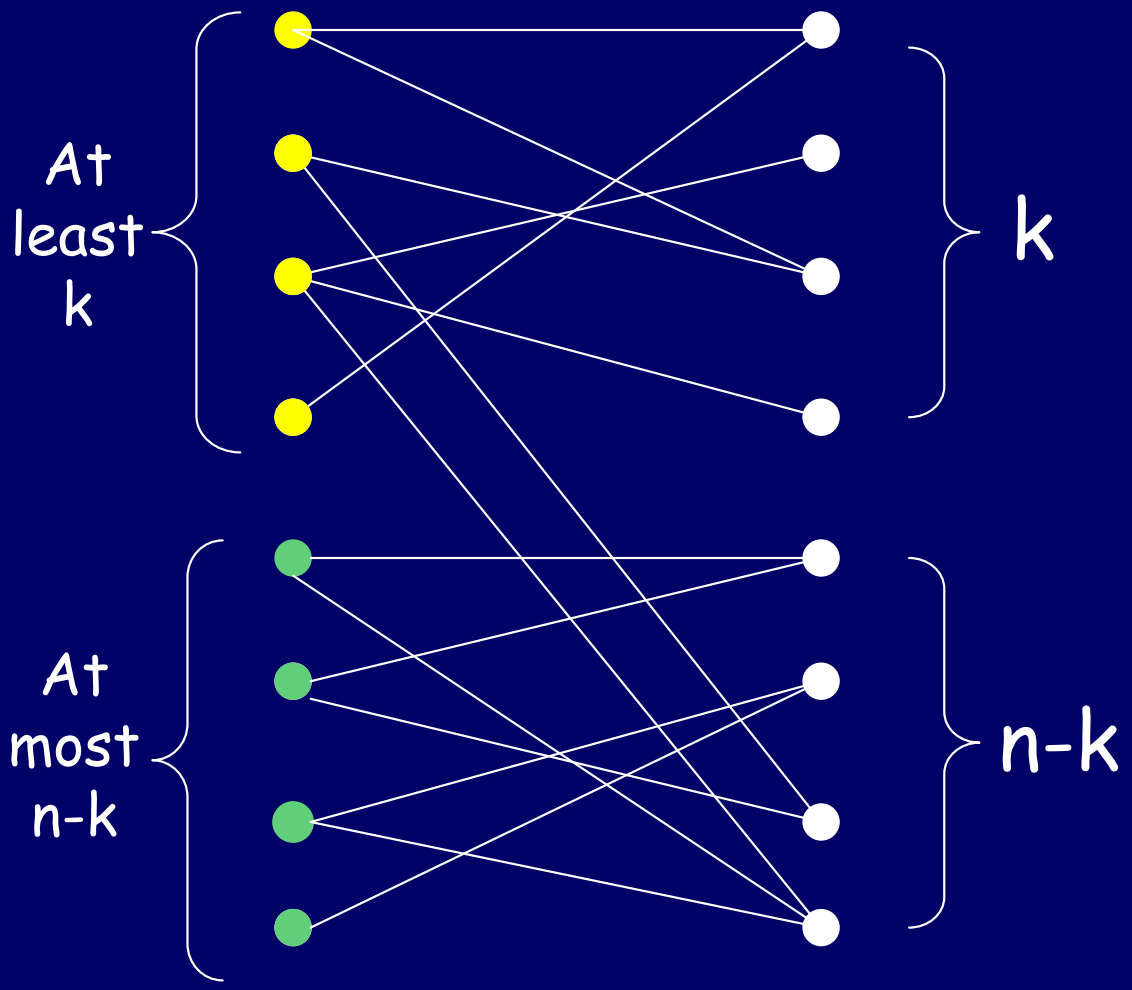
The Marriage Theorem



The condition fails for this graph.

The Feeling is Mutual

The condition of the theorem still holds if we swap the roles of A and B: If we pick any k nodes in B, they are connected to at least k nodes in A.



Proof of Marriage Theorem

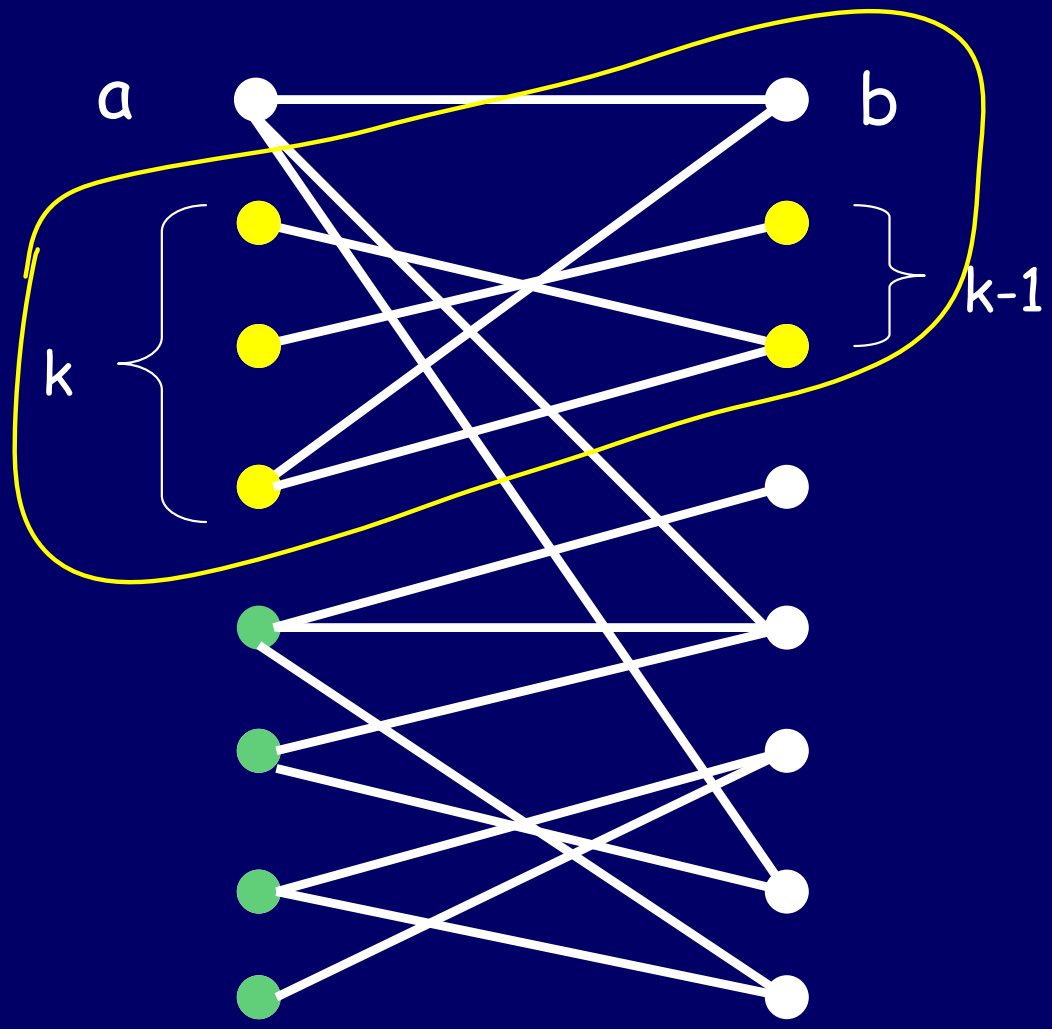
Call a bipartite graph "matchable" if it has the same number of nodes on left and right, and any k nodes on the left are connected to at least k on the right.

Strategy: Break up the graph into two matchable parts, and recursively partition each of these into two matchable parts, etc., until each part has only two nodes.

Proof of Marriage Theorem

- Select two nodes $a \in A$ and $b \in B$ connected by an edge.
- Idea: Take $G_1 = (a,b)$ and G_2 = everything else
- Problem: G_2 need not be matchable. There could be a set of k nodes that has only k-1 neighbors.

Proof of Marriage Theorem



The only way this could fail is if one of the missing nodes is b. Add this in to form G_1, and take G_2 to be everything else.

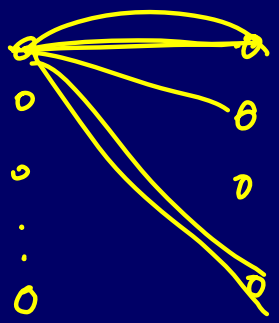
This is a matchable partition!

Generalized Marriage: Hall's Theorem

Let $S = \{S_1, S_2, \ldots\}$ be a set of finite subsets that satisfies: For any subset $T = \{T_i\}$ of S, $| \cup T_i | >= |T|$. Thus, any k subsets contain at least k elements.

Then we can choose an element $x_i \in S_i$ from each S_i so that $\{x_1, x_2, \ldots\}$ are all distinct.

Example



Suppose that a standard deck of cards is dealt into 13 piles of 4 cards each.

Then it is possible to select a card from each pile so that the 13 chosen cards contain exactly one card of each rank.

Graph Spectra

Finally, we'll discuss a different *representation* of graphs that is extremely powerful, and useful in many areas of computer science: AI, information retrieval, computer vision, machine learning, CS theory,…

Adjacency matrix

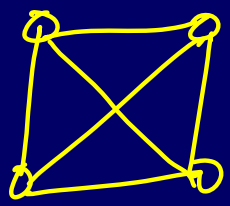
Suppose we have a graph G with n vertices and edge set E. The *adjacency matrix* is the nxn matrix $A=[a_{ij}]$ with

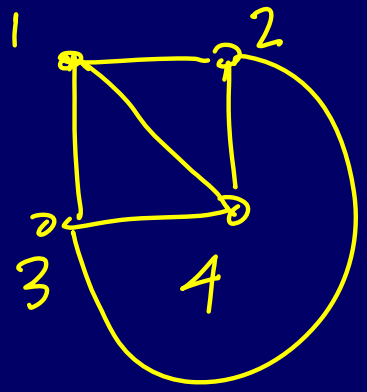
$a_{ij} = 1$ if (i,j) is an edge

$a_{ij} = 0$ if (i,j) is not an edge

Example

K_4



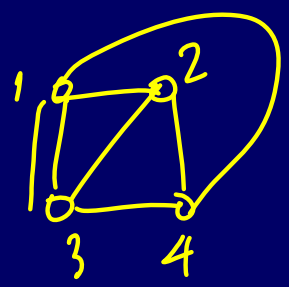


$$A = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

Counting Paths

The number of paths of length k from node i to node j is the entry in position (i,j) in the matrix A^k

K_4 $A = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$ $A^2 = \begin{pmatrix} 3 & 2 & 2 & 2 \\ 2 & 3 & 2 & 2 \\ 2 & 2 & 3 & 2 \\ 2 & 2 & 2 & 3 \end{pmatrix}$



$$A^K_{ij} = \left(A^{K-1} A\right)_{ij} = \sum_{\ell} A^{K-1}_{i\ell} A_{\ell j}$$

By I.H., # walks of length K-1 from i to ℓ

Eigenvalues

An nxn matrix A is a linear transformation from n-vectors to n-vectors

An *eigenvector* is a vector *fixed* (up to length) by the transformation. The associated *eigenvalue* is the scaling of the vector.

Eigenvalues

Vector x is an eigenvector of A with eigenvalue λ if

$$Ax = \lambda x$$

A symmetric nxn matrix has at most n distinct real eigenvalues

Characteristic Polynomial

The *determinant* of A is the product of its eigenvalues:

$$\det A = \lambda_1 \lambda_2 \dots \lambda_n$$

The *characteristic polynomial* of A is the polynomial

$$p_A(\lambda) = \det(\lambda I - A) = (\lambda - \lambda_1)(\lambda - \lambda_2) \dots (\lambda - \lambda_n)$$

Example: K_4

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

$$A\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 3 \\ 3 \\ 3 \\ 3 \end{pmatrix} = 3\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

$$A\begin{pmatrix} 1 \\ 1 \\ -1 \\ -1 \end{pmatrix} = \begin{pmatrix} -1 \\ -1 \\ 1 \\ 1 \end{pmatrix} = -1\begin{pmatrix} 1 \\ 1 \\ -1 \\ -1 \end{pmatrix}$$

$$P_A(\lambda) = (\lambda - 3)(\lambda + 1)^3$$

$$= \lambda^4 - 6\lambda^2 - 8\lambda - 3$$

If graph G has adjacency matrix A with characteristic polynomial

$$p_A(\bullet) = \bullet^n + c_1 \bullet^{n-1} + c_2 \bullet^{n-2} + \ldots + c_n$$

then

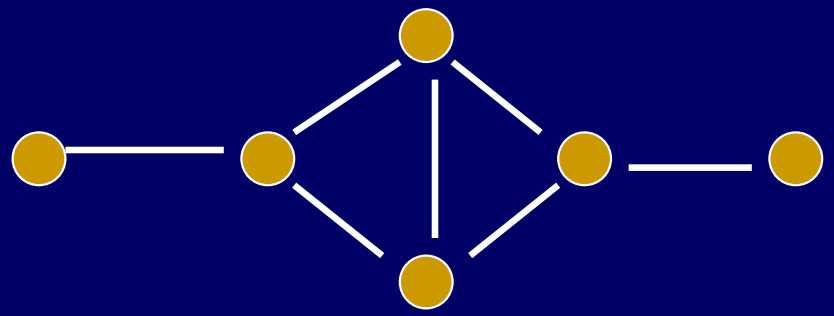
$c_1 = 0$

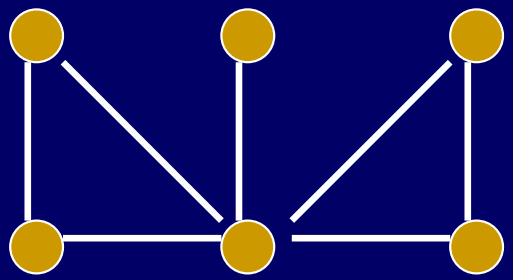
$-c_2 =$ # of edges in G

$-c_3 =$ twice # of triangles in G

Two different graphs with the same spectrum

$$p_A(\bullet) = \bullet^6 - 7\bullet^4 - 4\bullet^3 + 7\bullet^2 + 4\bullet - 1$$





Let your spectrum do the counting…

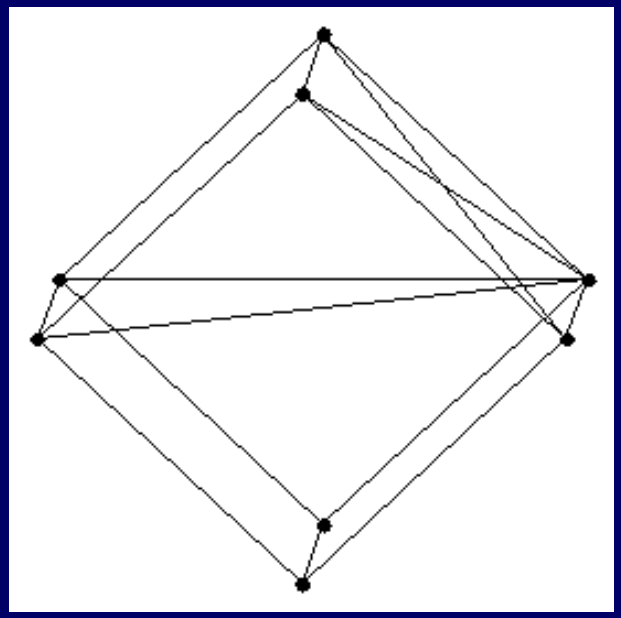
A *closed walk* or *loop* in a graph is a path whose initial and final vertices are the same. Then

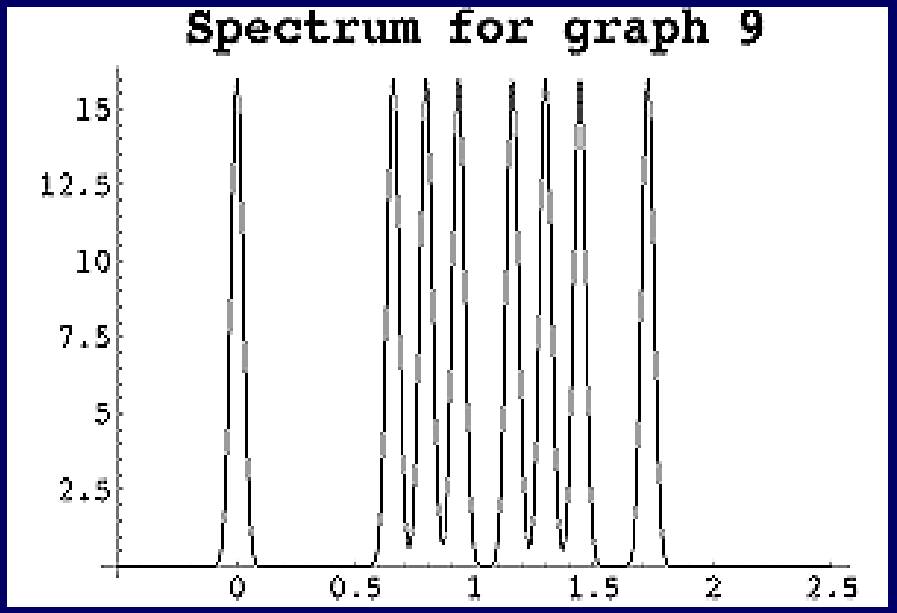
$\text{trace}(A) = \lambda_1 + \lambda_2 + \ldots + \lambda_n = 0$

$\text{trace}(A^2) = \lambda_1^2 + \lambda_2^2 + \ldots + \lambda_n^2$
= twice # of edges

$\text{trace}(A^3) = \lambda_1^3 + \lambda_2^3 + \ldots + \lambda_n^3$
= six times # of triangles

Graph Muzak





http://math.ucsd.edu/~fan/hear/

References

L. Lovász, J. Pelikán K. Vesztergombi, *Discrete Mathematics*, Springer Verlag, 2003.