November 14, 2017

21 - Flows with Unit Capacities

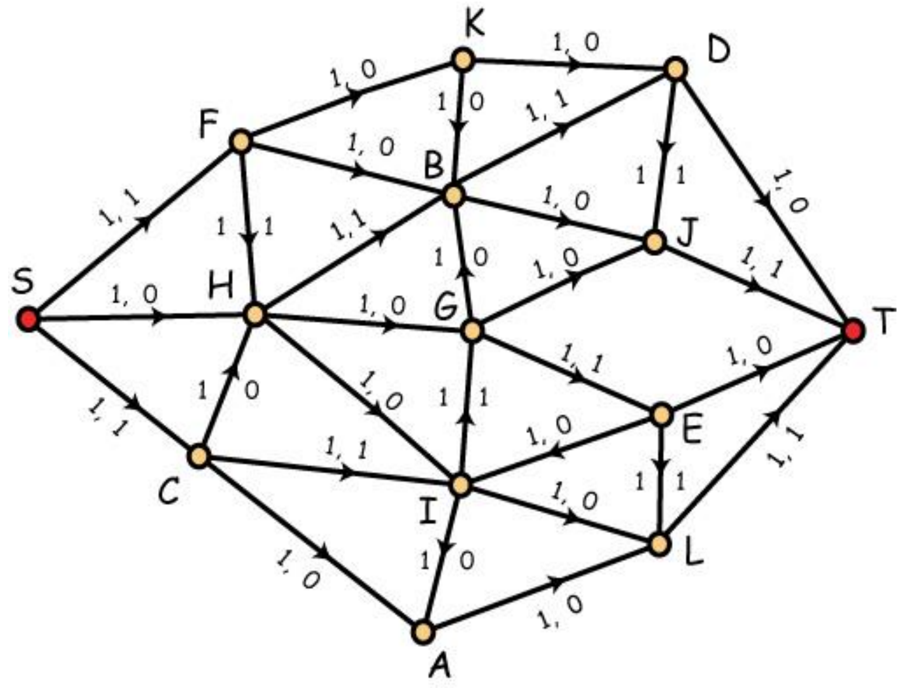
William T. Trotter
trotter@math.gatech.edu

A Key Detail on Network Flow Problems

Fact A network flow problem posed with integer capacities on edges always has a maximum flow in which the flow on every edge is an integer. The proof of this fact is an immediate consequence of the fact that the Ford-Fulkerson labelling algorithm uses only addition, subtraction and minimum as its three operations.

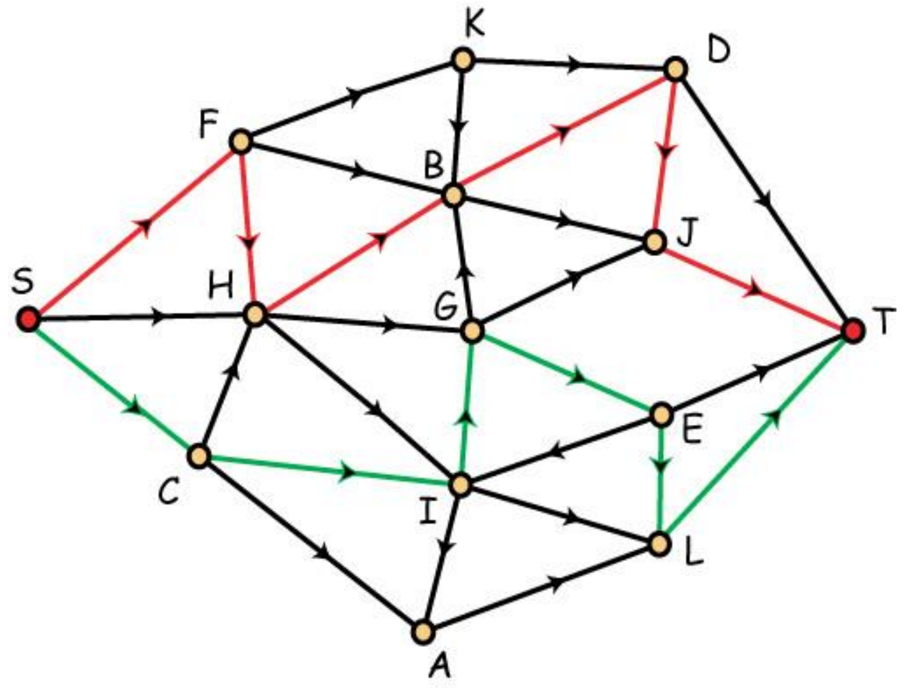
Remark It is an important general problem to determine when optimization posed with integer constraints have integer valued solutions. Network flows are just one example.

Flow Problems with Unit Capacities



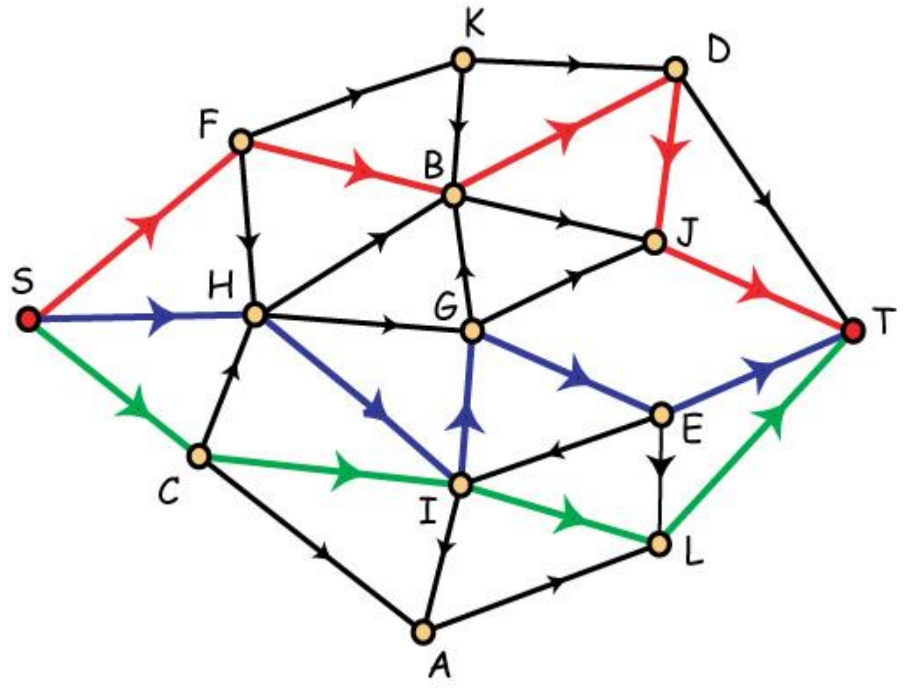
Exercise Carry out the Ford-Fulkerson labelling algorithm on the network flow. Remember that at each step, the flow on an edge will always be either 0 or 1, i.e., edges will always be in one of two states: empty or full.

Flow Problems with Unit Capacities (2)



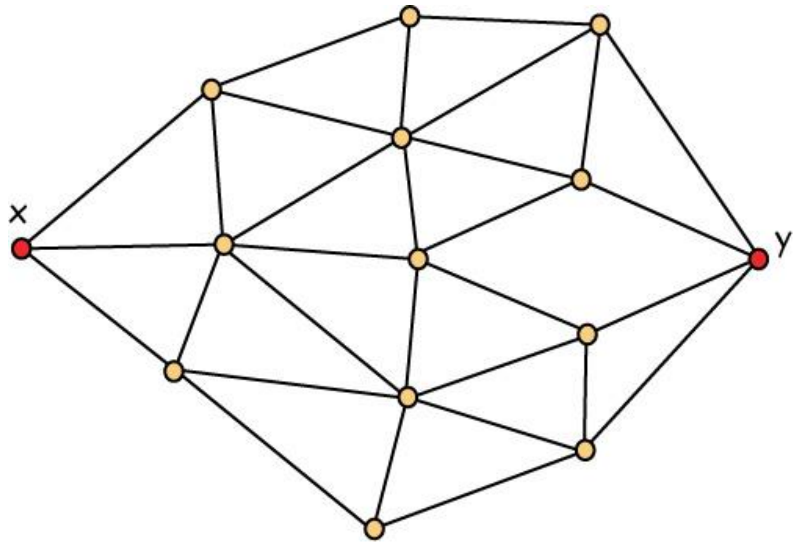
Remark Here the presence of an edge signals capacity 1, and the flow is indicated with two colors. This flow has value 2.

Flow Problems with Unit Capacities (3)



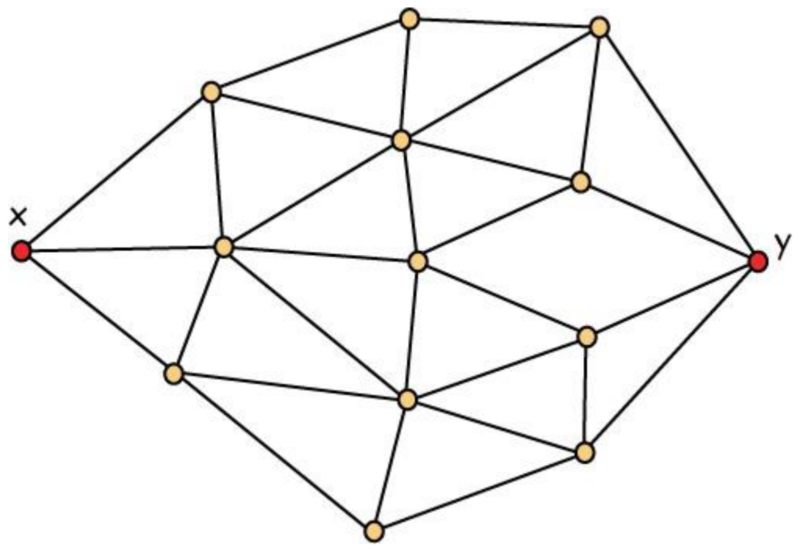
Remark Here the presence of an edge signals capacity 1, and the flow, which has value 3 is indicated with three colors.

Disjoint Paths



Problem Find the maximum number of pairwise disjoint paths from x to y.

Disjoint Paths (2)



Remark There are two different notions of “disjoint”. We could simply require that two different paths share no edges. Or we could make the stronger requirement that they have no vertices in common other than x and y. Network flows will find the maximum number of disjoint paths in either case.

Disjoint Paths From x to y

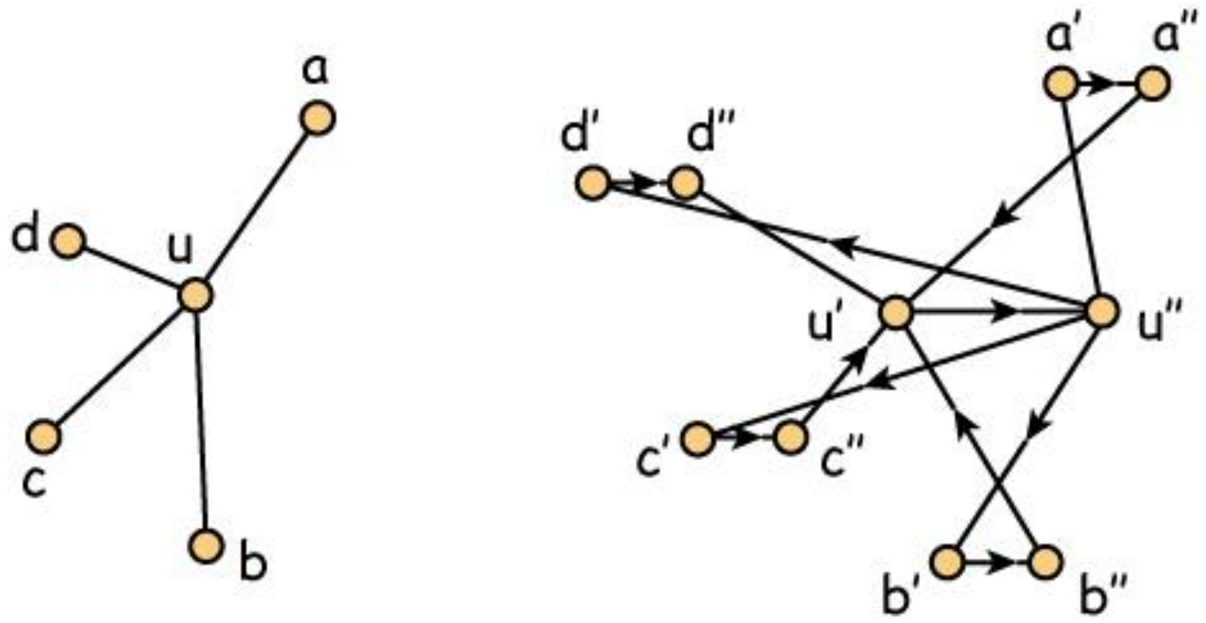
Gadgets We modify the original graph by first replacing each edge ab by two directed edges, one from a to b and the other back from a to b. Second, we split each vertex a into two vertices a′ and a″ with an edge from a′ to a″. An edge which used to go from a to b is moved to become an edge from a″ to b′. Now all edges are of two forms:

Interior edges from a′ to a″.

Exterior edges from a″ to b′ where a and b are distinct.

Applications For the specified vertices x and y, delete x′ and y″. Give all exterior edges capacity 1. If we want vertex disjoint paths, give interior edges capacity 1. If we want edge disjoint paths, give interior edges capacity n where n is the number of vertices in the graph.

Computational Gadget



Computational Detail

Observation Consider any pair zw of vertices for which there is a unit of flow on both (z'', w') and (w'', z'). For all such pairs, we reduce the flow on these edges as well as on (z', z'') and (w', w'') by 1. This changes do not decrease the value of the flow.

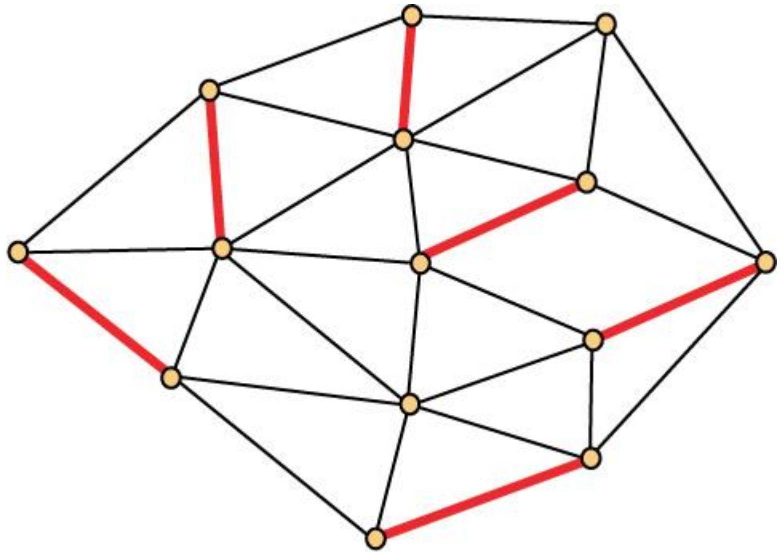
Fact This step eliminates the apparent use of the edge zw in both directions.

Graph Theory Consequences

Theorem (Menger's Theorem – Edge Version) Let x and y be distinct vertices in a connected graph G. Then the maximum number of edge disjoint paths from x to y is equal to the minimum number of edges whose removal from G leaves x and y in different components.

Theorem (Menger's Theorem – Vertex Version) Let x and y be distinct non-adjacent vertices in a connected graph G. Then the maximum number of vertex disjoint paths from x to y is equal to the minimum number of vertices whose removal from G leaves x and y in different components.

Matchings in Graphs



Definition A matching in a graph is a set of edges no two of which share an end point. Typically the problem is to find a maximum size matching. The matching shown is maximal. Is it maximum? The same kind of algorithm used to solve network flows will find a maximum matching in a graph.

Augmenting Paths in Graphs

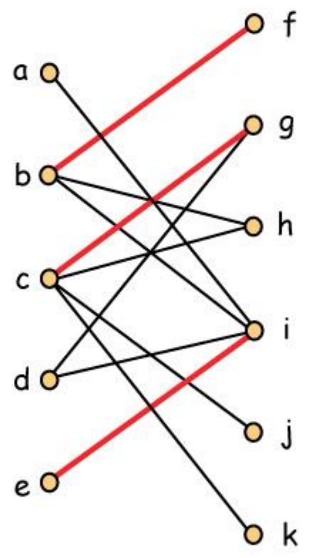
Definition Consider a maximal matching M. A sequence $(y_0, x_1, y_1, x_2, y_2, \ldots, x_k y_k, x_{k+1})$ is called an augmenting path for M when y_1 and x_{k+1} are not endpoints of any of the edges in M while x_i is matched with y_i for each $i = 1, 2, \ldots, k$.

Fact 1 Augmenting paths, if they exist, are easy to find.

Fact 2 If there is an augmenting path, then the matching M can be replaced by a matching have one more edge than M.

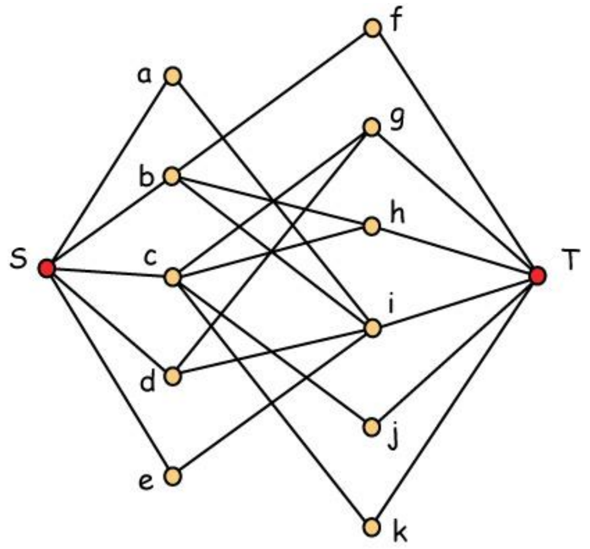
Fact 3 If there are no augmenting paths, the matching M is maximum.

Matchings in Bipartite Graphs



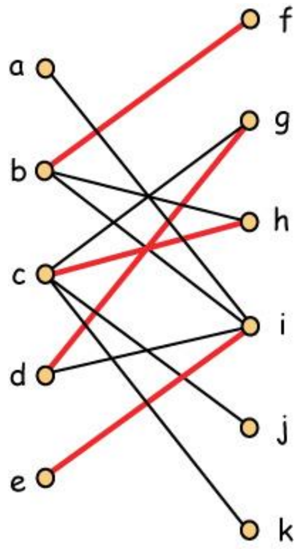
Remark We are particularly interested in finding a maximum matching in a bipartite graph. Again, the matching shown is maximal. Is it maximum?

Maximum Matchings in Bipartite Graphs



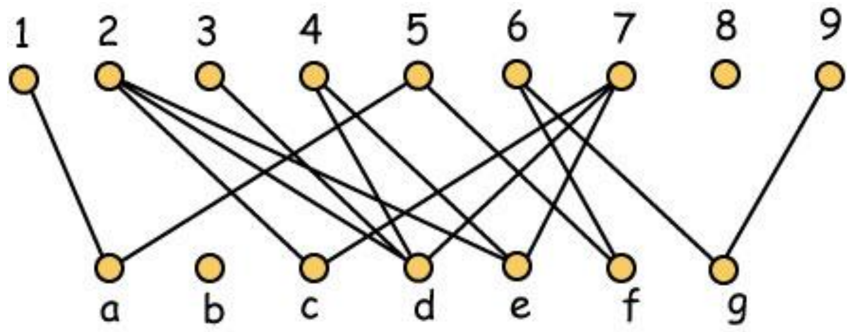
Observation There is a natural way to form a network flow problem from a bipartite graph. Simply add a source and a sink as shown, orient all edges left to right and give them capacity 1. Turn on Ford-Fulkerson and go get a cup of coffee.

Maximum Matchings in Bipartite Graphs (2)



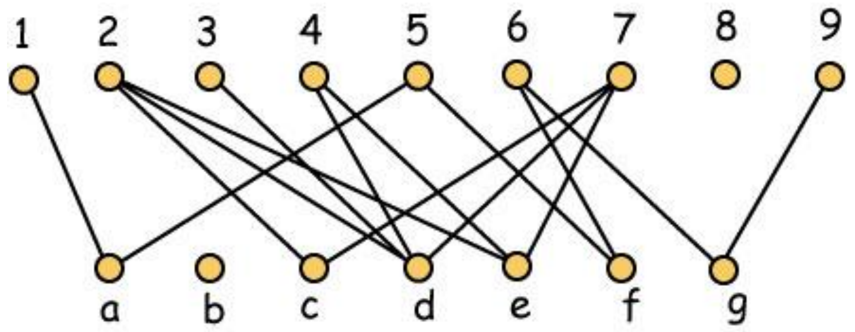
Observation It isn't really necessary to draw the source and sink as their configuration is understood. Now the matching shown is maximum. To see this, turn on Ford-Fulkerson and enjoy a donut with your coffee.

More on Matchings in Bipartite Graphs



Setup A company has 9 open positions and 7 applicants. The graph has an edge from applicant x to position i when x is capable of performing i. A matching is then an employment plan, and it is natural to try to fill as many open positions as possible. Note that some applicants may not be capable of doing any job and there may be some jobs that no applicant can do.

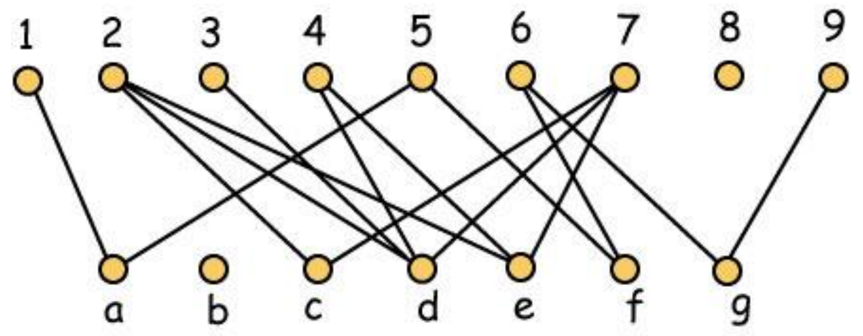
The Concept of Defect



Definitions Let $G = (X, Y, E)$ be a bipartite graph. For each subset S of X, let $N(S)$ denote the set of all elements y in Y for which there is some x in X adjacent to y. We call $N(S)$ the set of neighbors of S. The defect of G, denoted $d(G)$, is:

$$d(G) = \max \{ |S| - N(S): \ S \subseteq X \}$$

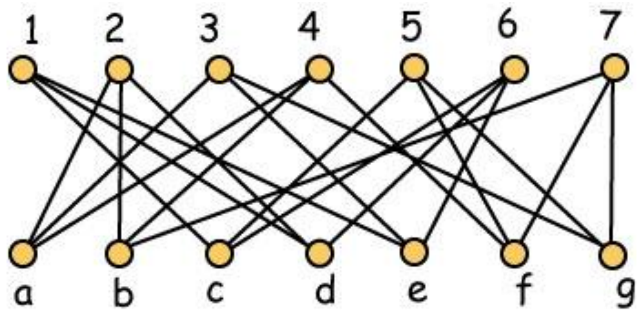
Hall's Theorem (Defect Form)

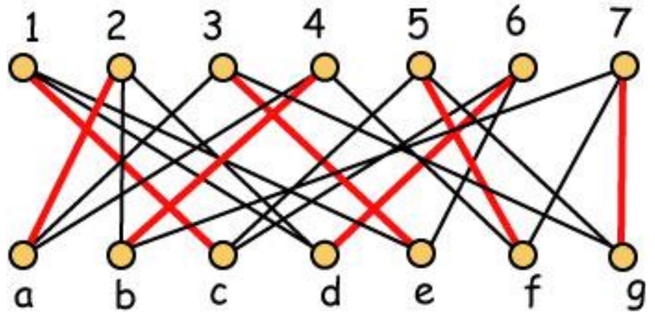


Theorem (Hall) Let $G = (X, Y, E)$ be a bipartite graph. Then the maximum size of a matching in G is $|X| - d(G)$.

Corollary There is a matching of size $|X|$ if and only if $d(G) = 0$, i.e., $|N(S)| \geq |S|$ for every subset S of X.

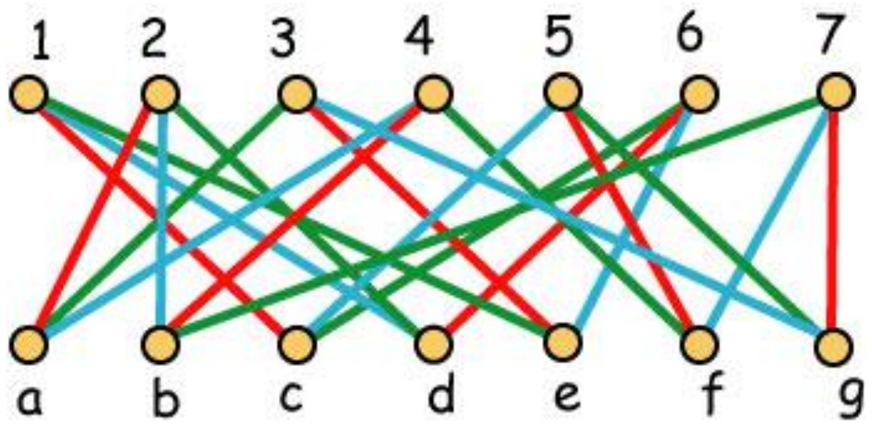
Regular Balanced Bipartite Graphs





Corollary Let $G = (X, Y, E)$ be a balanced regular bipartite graph. Then there is a matching of size $|X|$ in G.

Regular Balanced Bipartite Graphs



Corollary Let $G = (X, Y, E)$ be a balanced regular bipartite graph. If the degree is r, then the edge set of G can be partitioned into r matchings.

Computational Details

Observations There are special purpose algorithms for finding maximum matchings in graphs which run slightly faster than our network flow approach. This is particularly the case in the bipartite graph situation.