The algorithm proceeds by growing a tree, starting at s, in the underlying graph, using only usable arcs as frontier edges. If the tree eventually includes t we have found a flow augmenting path, otherwise the tree vertices form one set of a minimum cut.

**Algorithm:**

Initialize the tree $S = \{s\}$, a counter $i = 1$, and label $s$ with 0.

While $S$ does not contain t do

a. Locate a usable arc as a frontier edge with the smallest labeled endpoint.

b. If $w$ is the unlabeled endpoint, and $v$ the labeled endpoint

Set $\text{back}(w) = v$.

Label $w$ with $i$.

Let $i := i + 1$;

Put $w$ in $T$.

If there is no usable arc, return the cut $S, V - S$.

If $t$ is in $S$, the flow augmenting path is obtained from reading the vector $\text{back}()$ starting at $t$.

For a given $(s,t)$-network, we can find the maximum flow by starting with all arcs having 0 flow, then finding a flow augmenting path and adjusting the flows accordingly. Repeat this until there are no further flow augmenting paths.

**Matchings, Transversals and Vertex Covers**

A set $M$ of edges in a graph $G$ is called a **matching** of $G$ if no two edges in set $M$ have an endpoint in common. A **maximum matching** of graph $G$ is a matching of $G$ with the greatest number of edges while a **maximal matching** is a matching which is not contained in any larger matching. While any maximum matching is certainly maximal, the reverse is not generally true.

In the above graph, $\{a\}, \{b,d\}$, and $\{c,e\}$ are all maximal matchings, but only the last two are maximum matchings.

**Job Assignment Problem revisited**

Given a bipartite graph $(X, Y)$ we can form an $(s,t)$-network as follows: Add a vertex $s$, and join it with arcs going from $s$ to each vertex in $X$. Orient all the edges in the bipartite graph from $X$ to $Y$. Finally, add a vertex $t$, and join it with arcs from each vertex in $Y$ to $t$. Give all of the arcs leaving $s$ or entering $t$ a capacity of 1, but the arcs between $X$ and $Y$ will get infinite capacity. Call this network $N$.

Notice that any feasible flow of $N$ where all the flow values are integers (necessarily either 0 or 1), corresponds to a matching of the original graph (the matching consists of the edges which correspond to arcs.
between X and Y which have positive flow), and vice versa, any matching of the graph gives rise to a feasible flow. The value of the flow is the number of edges in the matching.

**Vertex Covers**

Let G be a graph and C a subset of the vertices of G. The set C is a *vertex cover* of graph G if every edge of G is incident with at least one vertex in C. A minimum vertex cover is a vertex cover with the least number of vertices.

**Proposition 12.4.5**: Let M be a matching in a graph G, and let C be a vertex cover of G. Then |M| ≤ |C|.

**Corollary 12.4.6**: Let M be a matching and let C be a vertex cover of a graph G such that |M| = |C|. Then M is a maximum matching and C is a minimum vertex cover.

The converse of this corollary is not necessarily true, but it is true for bipartite graphs as we shall see below.

Since we have a relationship between matchings in a bipartite graph and flows in the corresponding network, we should consider what a vertex cover looks like in the network.

Let A be a subset of X and B a subset of Y. We will consider the set K = A ∪ B. Let S = {s} ∪ (X-A) ∪ B and T = {t} ∪ (Y-B) ∪ A, which is clearly a cut in the network.
Theorem: K is a vertex covering of the graph if and only if (S,T) is a cut of the network of finite capacity.

Proof: Suppose (S,T) is a cut of finite capacity. Then no arcs from X to Y are in this cut, in particular there are none from X-A to Y-B in the cut. So, all arcs from A go to Y-B, and all arcs from X-A go to B. Therefore, every edge of the graph has one vertex in A or one vertex in B, and so, K is a covering. Note that the arcs in the cut that start at s all go to A, and those that end at t all start in B, each having capacity 1, so the capacity of the cut is |A∪B|.

Now, suppose that K is a vertex covering of the graph. Let A be the intersection of K with X and B the intersection of K with Y. Form S and T as before. Consider an arc from a vertex in S to one in T. If this arc starts at s, it must go to A (and so, has capacity 1). If it starts in X-A, it must go to B since K is a vertex covering, so such an arc does not go to T. If it starts in B, then it must go to t, and so, has capacity 1. Thus, all arcs from S to T have finite capacity, so the cut (S, T) has finite capacity.

Theorem 12.4.7: (König, 1931) Let G be a bipartite graph. Then the size of a maximum matching equals the size of a minimum vertex cover.

Proof: Consider the associated network of the bipartite graph. Change the infinite capacities to 2's, this will not change any flows. All capacities are integers, so the maximum flow exists and will have integer values. The value of this flow is the number of edges in a maximum matching, call it n. By the Ford-Fulkerson theorem, n is the minimum capacity of any (S, T) cut. The trivial cut, S = {s}, has finite capacity, so the minimum will have to be finite. By the previous theorem, any cut with the minimum capacity corresponds to a covering with |A∪B| = n. But now, there exists a cover with n vertices and a matching with n edges, so the cover must be a minimum cover.

An interesting application of this result is:
**Theorem 12.4.8:** (König-Egerváry, 1931) Let A be a 0-1 matrix. The maximum number of 1's in the matrix, no two of which lie in the same row or column, is equal to the minimum number of rows and columns that together contain all the 1's in A.