Matchings

Data Structures and Algorithms
Andrei Bulatov

Matchings

A matching M of a graph G = (V,E) is a set of edges such that every vertex is incident to at most one edge from M

Bipartite graphs: bipartition X, Y

The Bipartite Matching Problem

Instance:

A bipartite graph G

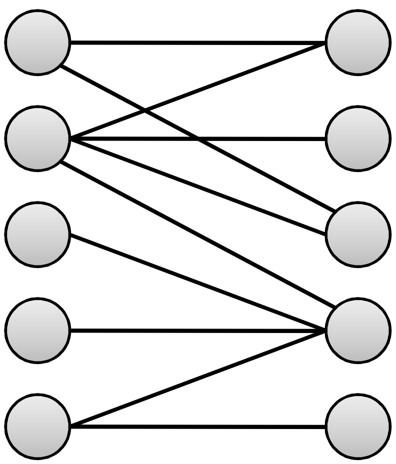
Objective:

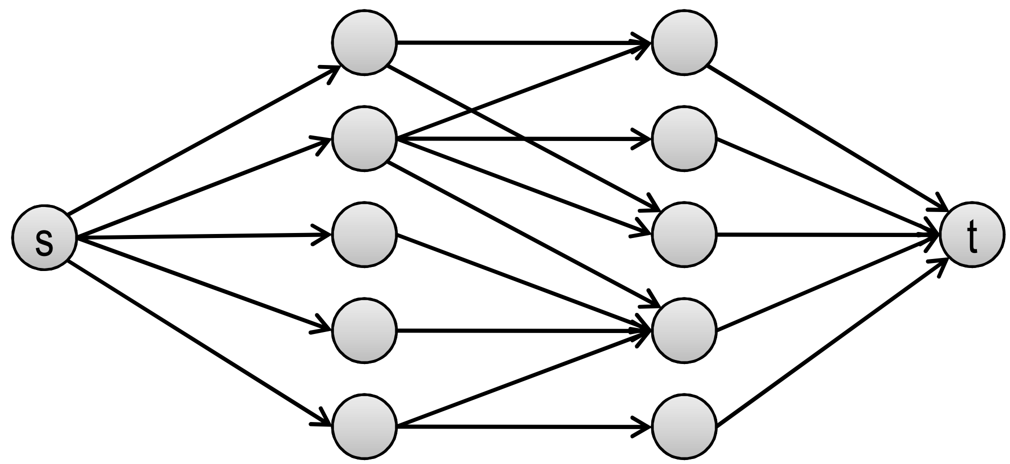
Find a matching in G of maximal size

Algorithm

We show how to reduce the Bipartite Matching problem to Network Flow

Let G be a bipartite graph with bipartition X, Y



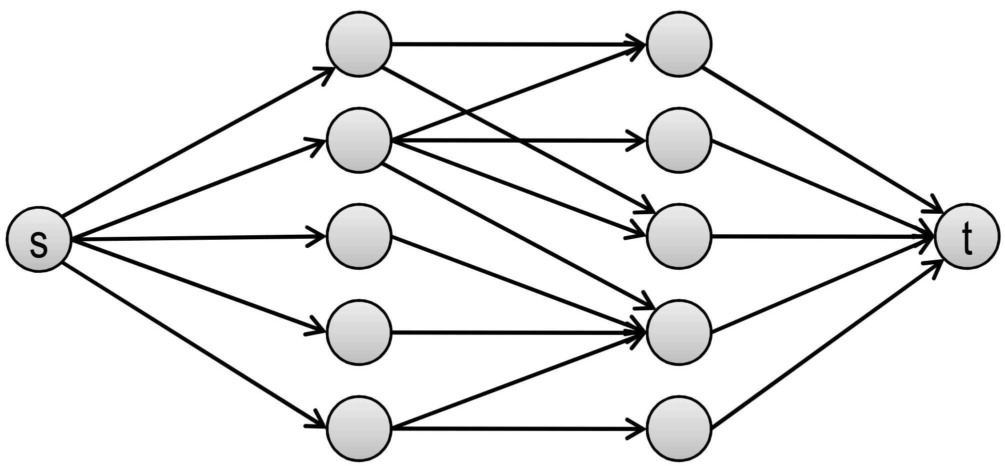


- orient all edges from X to Y
- add source s and sink t
- add arcs from s to all nodes in X, and from all nodes in Y to t
- set the weight of all arcs to be 1

Analysis

Lemma

Suppose there is a matching of G $(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)$ containing k edges. Then there is a flow in G' of value k



Proof

Straightforward

Analysis (cntd)

Lemma

Suppose there is a flow in G' of value k, then there is a matching of G containing k edges.

Proof

Let f be a flow in G' of value k.

Since all capacities in G' are integer, there is an integer flow of value at least k. So we can assume f is integer.

$f(e)$ equals 0 or 1 for every edge e

Let M be the set of arcs with the flow value 1

Analysis (cntd)

M contains k edges

Indeed, consider the cut (A,B) with $A = X \cup \{s\}$

The value of the flow through the cut equals the number of arcs from X to Y where the flow is non-zero

The set of such arcs is exactly the set M

Every node from X is the beginning of at most one arc from M

It follows straightforwardly from the conservation property

Every node from Y is the end of at most one arc from M

Same argument

Therefore M is a matching

Running Time

Theorem

The Ford-Falkerson algorithm can be used to find a maximal matching in a bipartite graph in $O(mn)$ time

Proof

We can assume that G has no isolated vertices, and so $m \geq n/2$

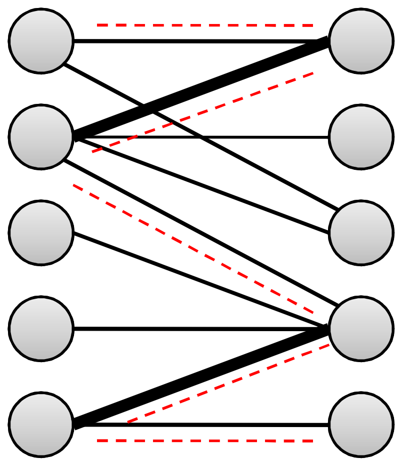
The maximal value of a flow in G' does not exceed $C = c(s) = |X| \leq n$

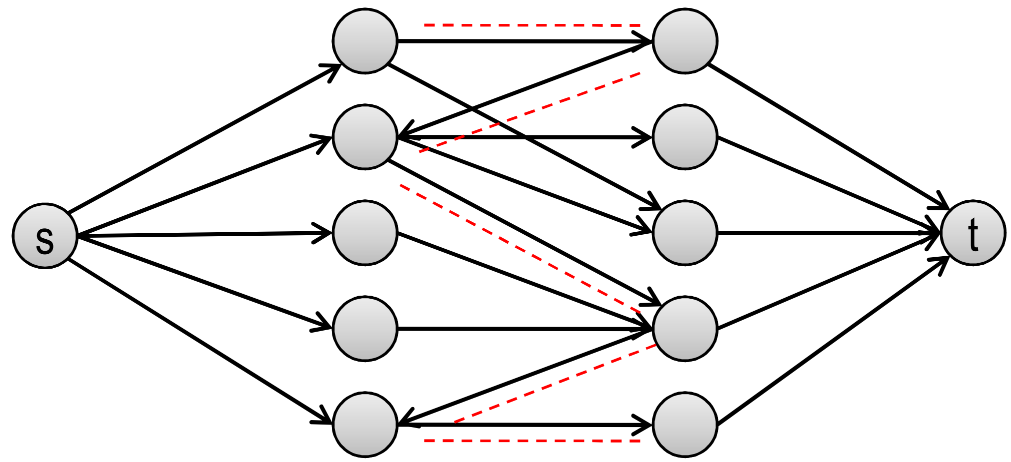
By the theorem on the running time of the F.-F. algorithm, it runs in $O(mC) = O(mn)$ time

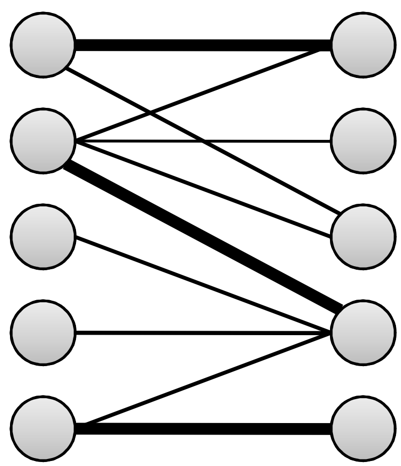
QED

Augmenting Paths in Bipartite Graphs

There is another algorithm for Bipartite Matching. It finds alternating paths







Perfect Matching and Hall's Theorem

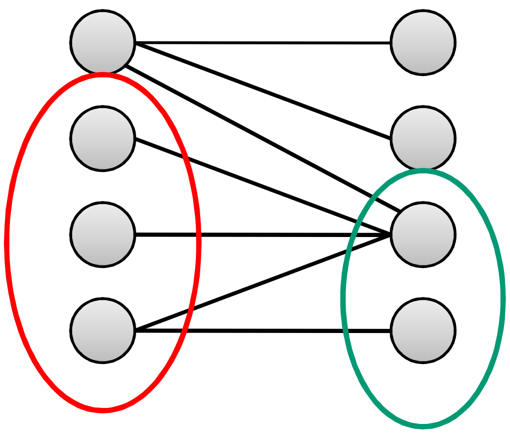
If both parts of a bipartite graph have the same number of elements , a perfect matching can exist, that is a matching that includes all vertices of the graph

How is it possible that a bipartite graph does not have a perfect matching

If there is $A \subseteq X$ such that for the set of neighbors $N(A)$

$$|N(A)| \le |A|$$

(or same for Y)



Theorem (Hall)

If G is a bipartite graph, and for any $A \subseteq X$ and any $B \subseteq Y$, we have $|A| \le |N(A)|$, $|B| \le |N(B)|$, then there is a perfect matching of G.

Perfect Matching and Hall’s Theorem (cntd)

Proof

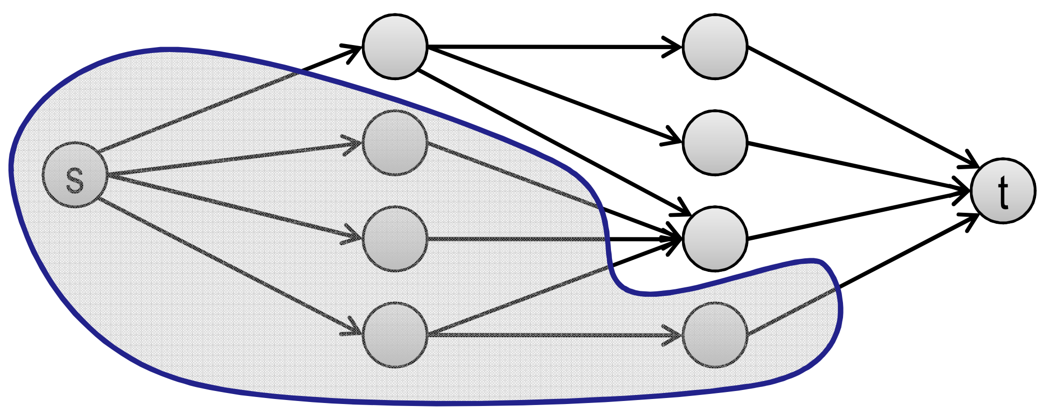
We use graph G’. Assume $|X| = |Y| = n$

If there is know perfect matching of G, a maximal flow in G’ has value less than n

We use this fact to find a set A (a subset of X or Y) such that $|A| < |N(A)|$

Since the value of maximal flow equals the capacity of a minimal cut, there is a cut (A’, B’) with capacity $< n$

Set A’ contains s, but can contain vertices from both sides



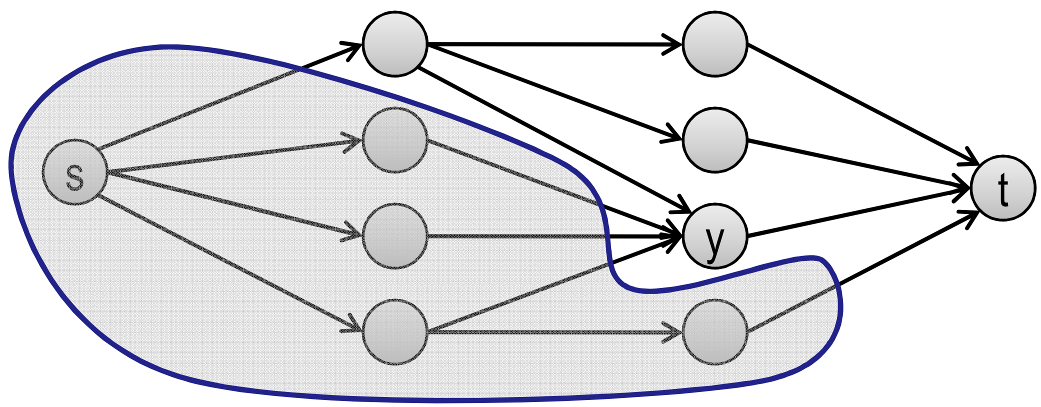
Set $A = X \cap A'$

Perfect Matching and Hall's Theorem (cntd)

We show that (A', B') can be chosen such that $N(A) \subseteq A'$

Take a node $y \in B' \cap N(A)$

Prove that $(A' \cup \{y\}, B - \{y\})$ is a cut of capacity not exceeding that of (A',B')



Indeed, the new cut crosses the arc (y,t),

but since $y \in N(A)$, there is at least one arc arriving to y from A, and so now is not crossed

Consider the capacity of (A',B') assuming $N(A) \subseteq A'$

The only arcs out of A' are those leaving s, or arriving to t

Perfect Matching and Hall's Theorem (cntd)

Thus

$$c(A',B') = |X \cap B'| + |Y \cap A'|.$$

Observe that $|X \cap B'| = n - |A|$, and $|Y \cap A'| \geq |N(A)|$

Then the assumption $c(A',B') < n$ implies

$$n - |A| + |N(A)| \leq |X \cap B'| + |Y \cap A'| = c(A',B') < n$$

We get

$$|A| > |N(A)|$$

QED