CS 341 – Algorithms

Lecture 16 – Bipartite Vertex Cover

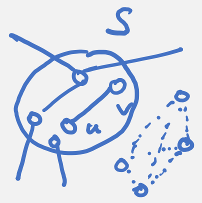
16 July 2021

Today’s Plan

1. Min-Max Theorem
2. Good Characterizations
3. Applications and Looking Forward

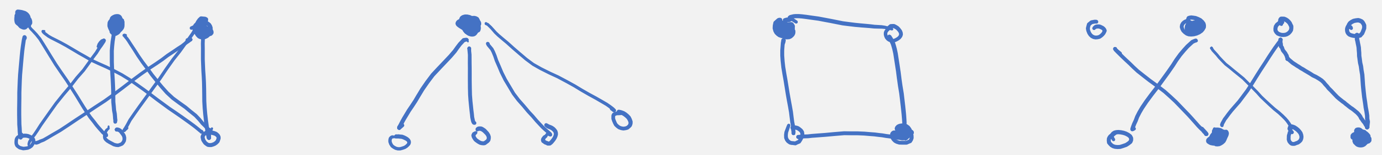
Bipartite Vertex Cover

Given a graph $G = (V, E)$, a subset of vertices $S \subseteq V$ is a vertex cover if $\{u, v\} \cap S \neq \emptyset$ for all $uv \in E$.



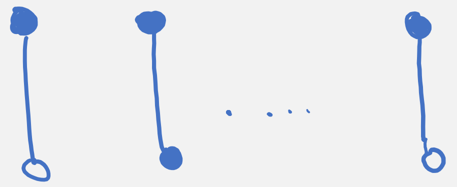
Input: A bipartite graph $G = (X, Y; E)$.

Output: A vertex cover of minimum cardinality.



Dual Problems

The bipartite vertex cover problem doesn't seem to be related to the bipartite matching problem, but they are actually "dual" of each other in a precise and meaningful way.



matching of size k

$\Rightarrow$ vertex cover of size $\geq k$

matching M, a vertex cover S

$|M| \leq |S|$. true $\forall M, \forall S$.

$\Rightarrow$ size of maximum matching $\leq$ size of a minimum vertex cover

Min-Max Theorem

<u>König's Theorem</u>: In a bipartite graph, the max size of a matching is equal to the min size of a vertex cover.

<u>Proof plan</u>: We will provide an algorithmic proof of Konig's theorem.

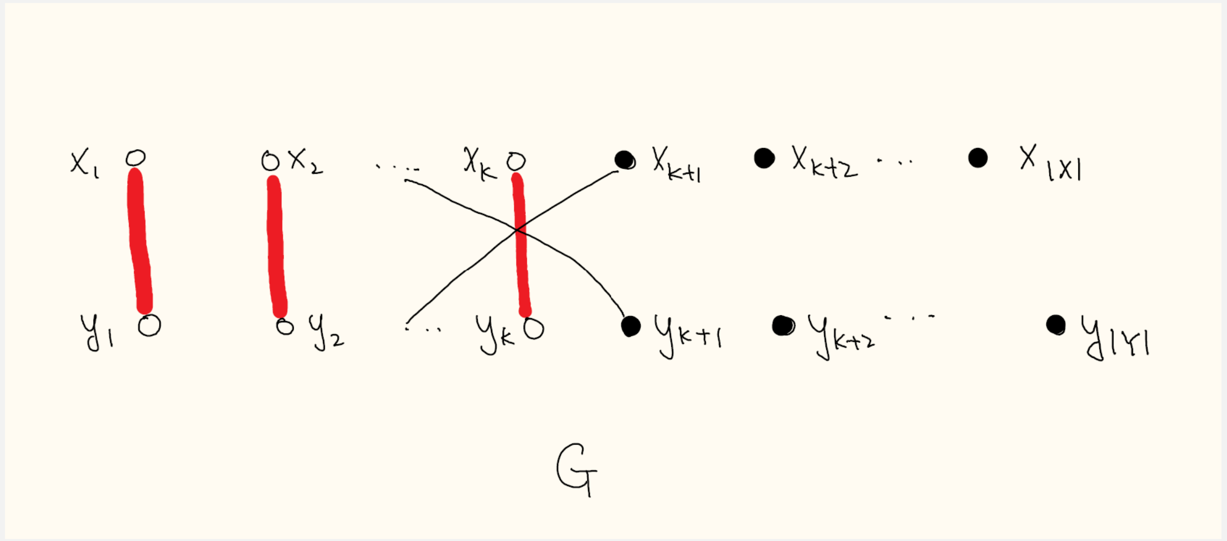
We will prove that given the current matching M of size k, if we couldn't find an augmenting path of M, then we can find a vertex cover $S \subseteq V$ of size k.

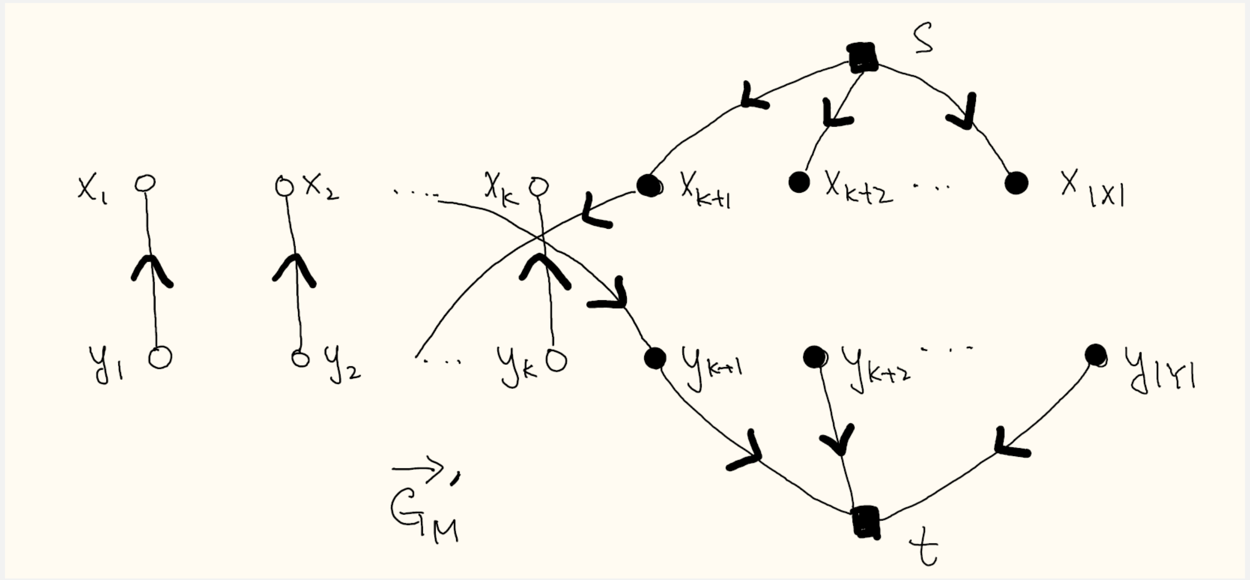
This will prove that M is a maximum matching and S is a minimum vertex cover.

matching of size k $\Rightarrow$ vertex cover cannot be smaller than k

vertex cover of size k $\Rightarrow$ matching cannot be larger than k

Algorithmic Proof

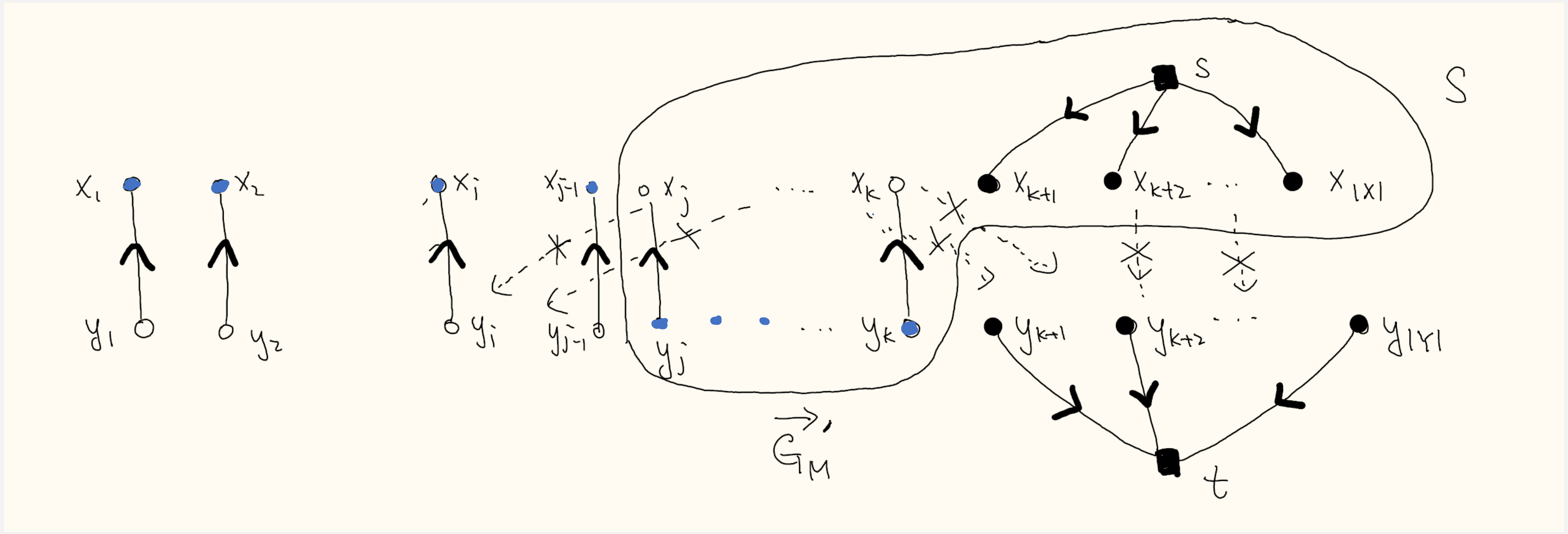




There is no augmenting path of M in G if and only if there is no directed path from s to t in $\overrightarrow{G_M}'$.

Algorithmic Proof

Let S be the set of vertices reachable from s. Note that there are no edges with tail in S and head in $V - S$.



Claim: $\{x_1, \ldots, x_{j-1}\} \cup \{y_j, \ldots, y_k\}$ is a vertex cover of G.

2 types: with an endpoint in $\{x, \ldots x_{j-1}\}$, with an endpoint in $\{x_j, \ldots, x_{|X|}\}$

$\Downarrow$ covered by $\{x, \ldots x_{j-1}\}$ $\qquad$ $\Downarrow$ covered by $\{y_j, \ldots y_k\}$

Algorithm and Complexity

1. Use an efficient algorithm to find a maximum matching M — doesn't have to be the augmenting path algorithm

$O(m+n)$ 2. Construct the directed graph $\vec{G}'_M$ as described above and in the last lecture.

$O(m+n)$ 3. Do a BFS/DFS on $\vec{G}'_M$ to identify all vertices $S \subseteq V$ reachable from s.

$O(m+n)$ 4. Return $(Y \cap S) \cup (X - S)$ as the vertex cover $\{x_1 \dots x_{j-1}\} \cup \{y_j \dots y_k\}$

time complexity : $O(\text{ALG}_{\text{matching}})$

$O(mn)$ LIS $\quad O(m\sqrt{n})$ Edmonds-karp

Today’s Plan

1. Min-Max Theorem
2. Good Characterizations
3. Applications and Looking Forward

Good Characterization

Imagine that we work for a company and our boss asks us to find a maximum bipartite matching.

If $\exists$ PM, then boss happy.

If $\nexists$ PM, how do we explain?

L15 no augmenting path

L16 vertex cover of size $< \frac{n}{2}$

Min-Max Theorems

These min-max theorems are some of the most beautiful results in combinatorial optimization, providing both succinct “proofs” from both the YES-instances and NO-instances.

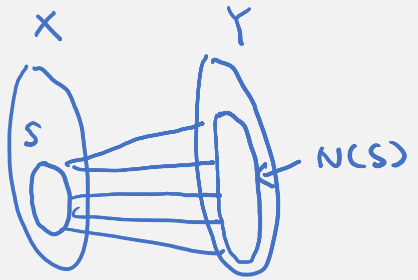
This shows the ***non-existence*** of a solution by the ***existence*** of a simple obstruction.

Remark: Contrast this with dynamic programming algorithms.

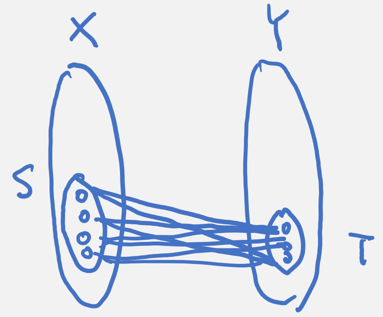
Hall’s Theorem

Hall’s theorem characterizes when a bipartite graph has a perfect matching or not.

Hall’s Theorem: A bipartite graph $G = (X, Y; E)$ with $|X| = |Y|$ has a perfect matching if and only if for all $S \subseteq X$, it holds that $|N(S)| \geq |S|$ where $N(S)$ is the neighbor set of S in Y.



$|N(S)| \geq |S|$



$|T| < |S|$

Exercise : Derive Hall's theorem from König's theorem.

Perfect Matching in Regular Bipartite Graphs

Corollary: Every d-regular bipartite graph $G = (X, Y; E)$ has a perfect matching.

proof :

$|X| = |Y|$

$d|X|$ edges

no vertex cover of size $\leq |X| - 1$ $\Rightarrow$ min vertex cover of size $|X|$

$\overset{\text{König}}{\Rightarrow}$ max bipartite matching of size $|X|$. $\square$

expected (randomized)

Remark: There is a very interesting $O(n \log n)$ time algorithm to find such a perfect matching using random walks! (See CS 466 notes if interested.)

Today’s Plan

1. Min-Max Theorem
2. Good Characterizations
3. Applications and Looking Forward

Applications

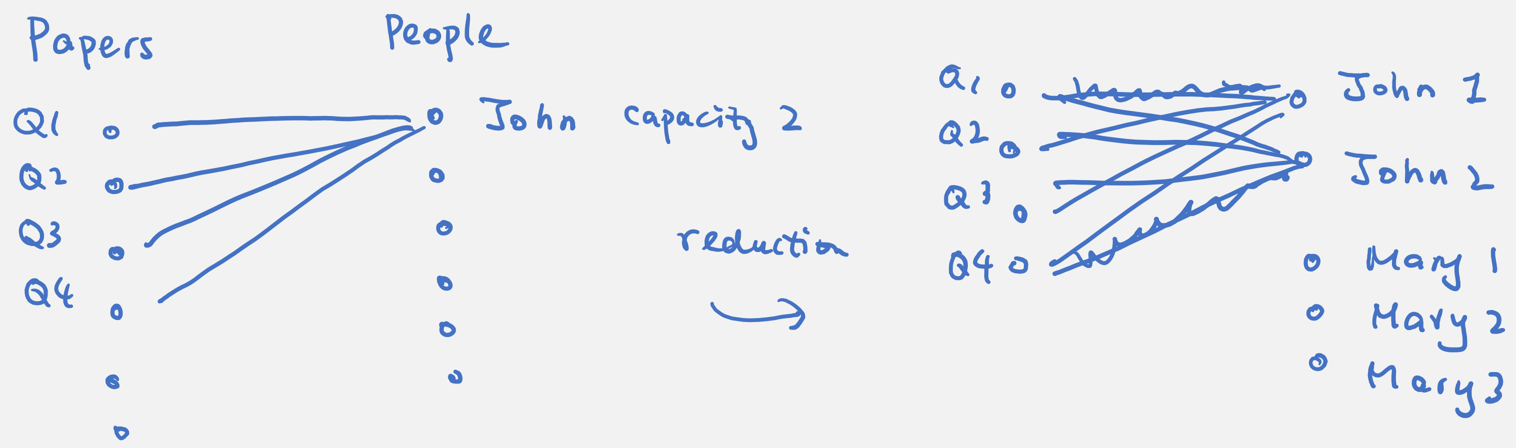
There are many interesting and non-trivial applications of bipartite matching and network flows.

We don't have time, so we will just do one example, and we break the problem into two parts.

Capacitated job assignment. There is a small generalization of the job assignment problem.

Each person has a capacity c_i to handle at most c_i jobs.

The task is to assign all jobs to people, so that no person is overloaded.



Basketball League Winner

Input: The current standing, and the remaining schedule.

Output: Whether it is possible that our favorite team can still win the league.

n teams. schedule teams play against each other

	wins	losses
Boston	41	17
New York	40	18
Phily	38	19
Miami	38	20
⋮		
Toronto	33	25

remaining Schedule

Miami – Boston x3

New York – ~~Bo~~Boston x2

⋮

Toronto – New York x2

⋮

① can assume that Toronto wins all the remaining games W^*

Reduction to Capacitated Assignment

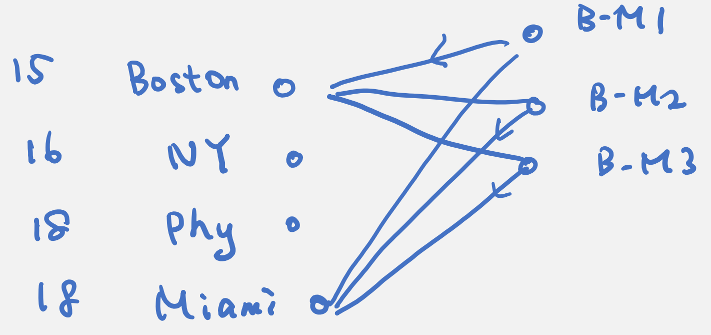
idea: want all other teams win $< w^*$ games

say team i has current win # w_i

want team i to win $\le w^* - w_i - 1$ games

idea: assign the winners of all the games so that no team is assigned more than $w^* - w_i - 1$ games

Toronto $w^* = 57$



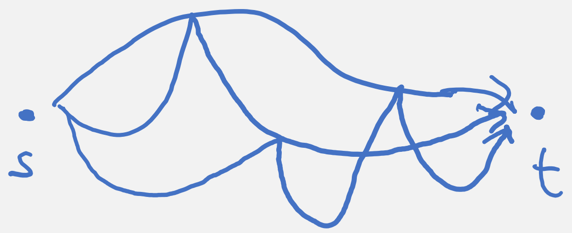
<u>Remark</u>: It is an NP-hard problem to determine if our favorite football team can still win the league.

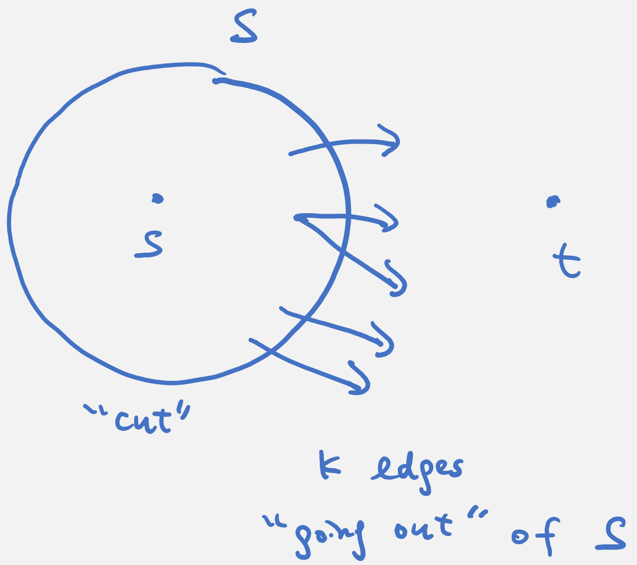
Max-Flow Min-Cut

The network flow problem can be solved using the augmenting path method for bipartite matching.

One useful application is to find the maximum number of edge-disjoint paths between two vertices s and t.

The famous "max-flow min-cut" theorem is a very important result with many applications.





General Matching

The maximum matching problem can be solved in polynomial time.

Even the maximum weighted version, where every edge has a weight, can be solved in polynomial time.

This was a ground-breaking result by Edmonds in the 70s, still considered difficult 50 years later.

The maximum weighted matching problem can be used to solve the Chinese postman problem in L01.

Duality

Why bipartite vertex cover is the dual problem of bipartite matching? min-max thms

How do we come up with it?

There is a systematic way to define the dual problem of an optimization problem,

through the use of linear programming.

Many beautiful min-max theorems can be derived systematically through linear programming duality!

$\max \sum_{e \in E} x_e$

$\forall v \in V \quad \sum_{e \in \delta(v)} x_e \leq 1$

~~$x_e \in \{0,1\}$~~

$\forall e \in E \quad x_e \geq 0$

bipartite matching

LP duality $\longrightarrow$

$=$

$\min \sum_{v} y_v$

$y_u + y_v \geq 1 \quad \forall uv \in E$

$y_v \geq 0 \quad \forall v \in V$

vertex cover

Linear Programming

Most combinatorial optimization problems can be solved in the general framework of linear programming.

This is one of the most, if not the most, powerful algorithmic framework for polynomial time computation.

The augmenting path method can be understood as the simplex method of solving linear programs.

LP   Convex programming

local optimal = global optimal