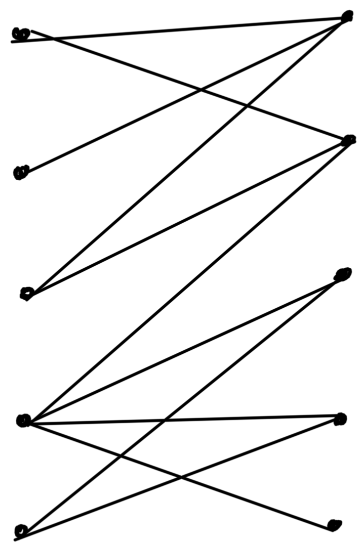
COL 351 Lecture 32 2023/04/05

Topic : König's Theorem

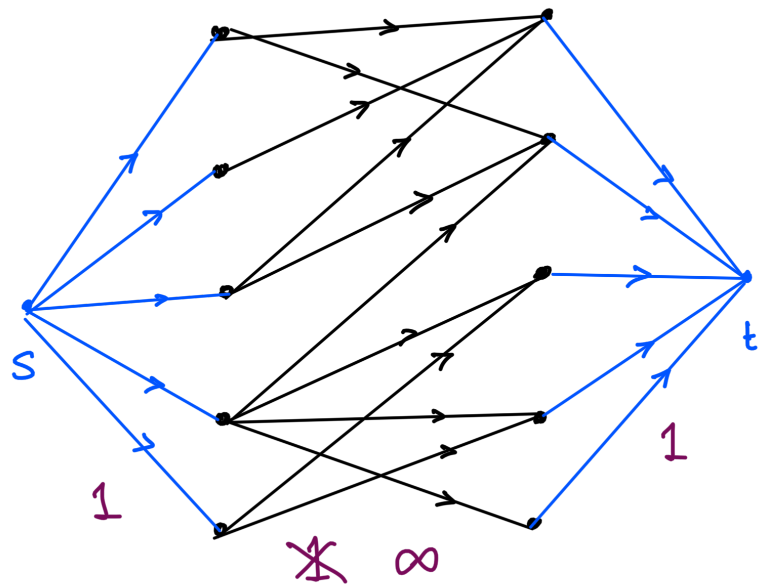
Cook Reduction

Recap:

G



$\mathcal{N}^{\infty}(G)$



Claim: A mincut in $\mathcal{N}^{\infty}(G)$ is of the form $\{s\} \cup L' \cup N(L')$ for $L' \subseteq L$

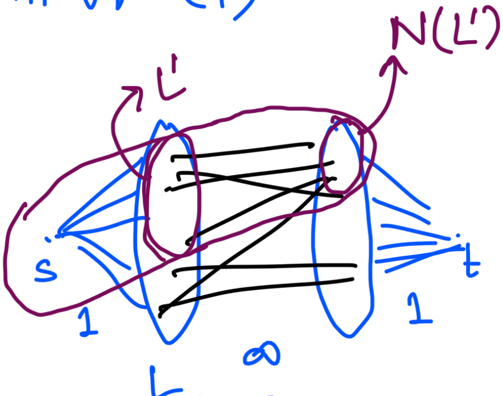
König's Theorem: In every bipartite graph, the size of max matching equals the size of min vertex cover.

Proof: size of max matching $\leq$ size of min vertex cover $\rightarrow$ last class.

Let G be a bipartite graph, and let n' be the size of max matching in G.

$n' =$ max flow in $\mathcal{N}^{\infty}(G)$ = min cut in $\mathcal{N}^{\infty}(G)$

Let $S^* = \{s\} \cup L' \cup N(L')$ be a mincut, where $L' = S^* \cap L$



$c(S^*) = |L \setminus L'| + |N(L')|$

$\therefore\ n' = |L \setminus L'| + |N(L')|.$

But $(L \setminus L') \cup (N(L'))$ is a vertex cover of G.

$\therefore$ size of min vertex cover $\leq |(L \setminus L') \cup N(L')| = n'.$

Definition: Let P, Q be computational problems. A <u>Turing-reduction</u> from P to Q is an algorithm A for P in an alternate universe where Q can be computed in $O(1)$ time. If, additionally, A runs in polynomial time (in the alternate universe), then A is called a <u>Cook-reduction</u>. P is said to be <u>Cook-reducible</u> (resp. Turing-reducible) to Q if such a Cook-reduction (resp. Turing-reduction) exists.

Claim: Let P, Q be computational problems such that P is Cook-reducible to Q.
If Q has a polynomial time algorithm, then P has a polynomial time algorithm.

Proof: Let A be a Cook-reduction from P to Q and B be a polynomial-time algorithm for Q. Let C be the algorithm obtained by replacing the ($O(1)$-time) Q-computation operations in P by calls to B. Then C solves P.

Suppose A runs in time $O(n^p)$ and B runs in time $O(n^q)$, (where p, q are constants independent of input size n).

On input of size n: Time spent in A is $O(n^p)$.

A makes $O(n^p)$ calls to B. Every call is given an input of size $O(n^p)$

$\therefore$ Total time spent in calls to B is

$$O\Big(\underbrace{n^p}_{\#\text{calls}} \times \underbrace{(n^p)^q}_{\text{size of input to } B}\Big) = O(n^{p+pq})$$

$\therefore$ Running time of C is $O(n^{p+pq})$

$\therefore$ C is a polynomial-time algorithm.