Max Matchings in General Graphs

The Tutte-Berge formula uses that odd components to determine deficiency. Oddness is important also for:

Thm: A graph G is bipartite iff G contains no odd cycles.

So, if there are no odd cycles, then we can find a maximum matching! We now need to ~~deal~~ control the odd cycles.

Let G be a graph, ~~and~~ M a matching in G, and x an M-unsaturated vertex. We build an

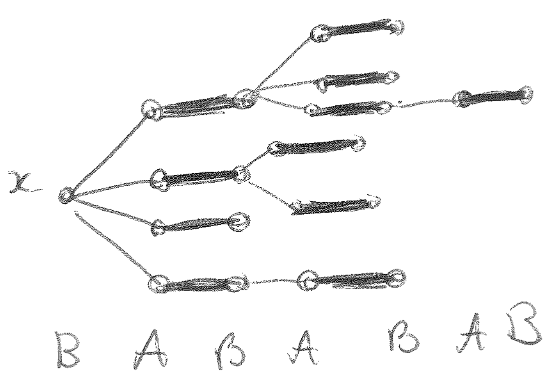
M-alternating tree as follows:

Init: $B = \{x\}$, $A = \emptyset$, $E(T) = \emptyset$

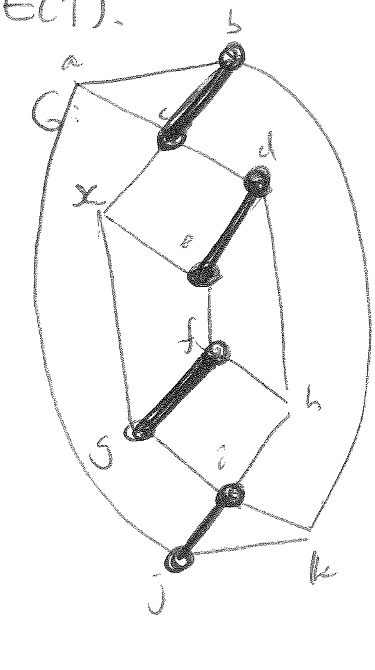
Iteration: If $vw \in E$, $v \in B$, $w \notin A \cup B$, and $wz \in M$,

then add w to A, z to B,

add vw & wz to $E(T)$.



Ex: G:



M-alternating trees have the following properties:

(a) every node $y \in T$ other than x is covered by an edge in $M \cap E(T)$

(b) every node $y \in T$, the xy path in T is M-alternating.

(c) T is a tree rooted at x.

Recall A & B the odd and even vertices.

$|B| = |A|+1$, since all but x are matched to A.

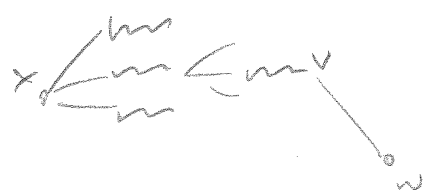
Obs: If T is an M-alternating tree rooted at x

and vw is an edge w/ $v \in B$, $w \notin A \cup B$,

and $vw \notin M$, and w is M-unsaturated,

then attaching vw to the path in T from x to v

is an M-augmenting path. Augment!



So, if there is an edge from $B(T)$ to $\overline{A \cup B}$, then

we can either **extend** **T**, **or** find an M-augmenting

path!!

If every edge ~~leaving~~ w/ one endpt in B

has the other in A, then T is **frustrated**

Prop 5.6: Suppose that G has a matching M and an M-alternating tree T that is frustrated. Then G has no perfect matching.

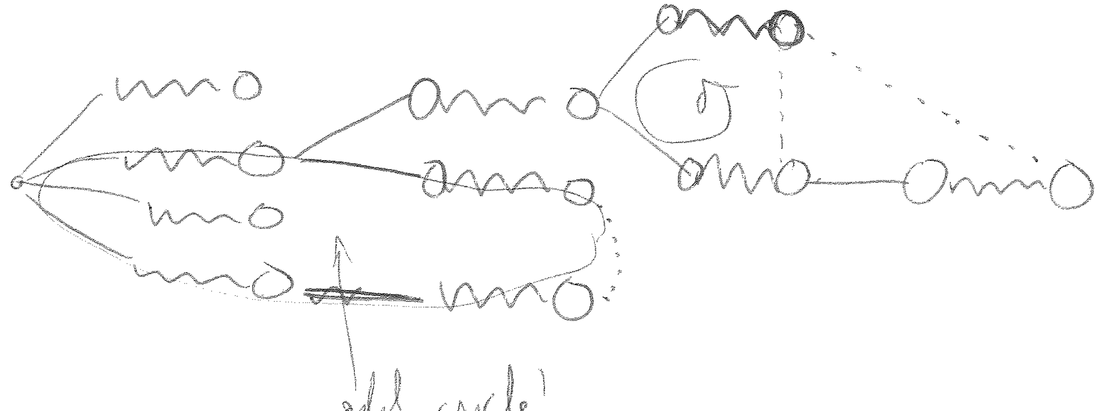
Pf: Consider $A(T)$.

In $G - A(T)$, every node $v \in B(T)$ is an isolated vertex, so,

$$oc(G - A(T)) \geq |B(T)| = |A(T)| + 1 > |A|.$$

Tutte's Thm $\Rightarrow$ No perfect matching!! $\square$

Obs: If G is bipartite, then all edges w/ one end in $B(T)$ have the other outside $B(T)$.
(Otherwise, an odd cycle exists!)

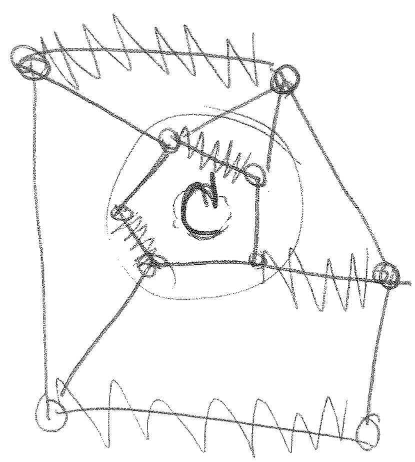


odd cycle!

So, in the **bipartite** case, we can build an M-alternating tree and see if it can ~~extend~~ augment M or if it will be **frustrated**.

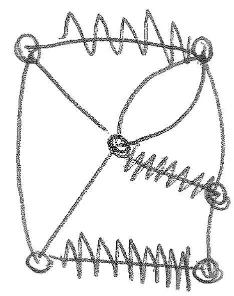
What about odd cycles?

G:



$G \times C$:

Shrinking by odd cycles.

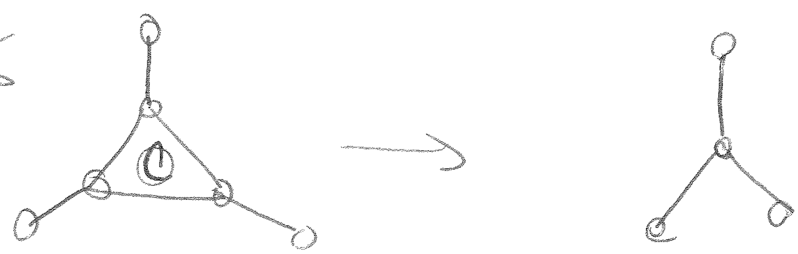


has p.m.

has p.m.

Careful!

G



has p.m.

does **not** have p.m.!

Really:

$$\mathrm{def}(G) \leq \mathrm{def}(G \times C).$$

C is **tight** if this is **equality**!

Build an M-alternating tree from $B=\{x\}$, $A=\emptyset$, where x is M-unmatched.

Extend: If $vw \in E$, $v \in B$, $w \notin A \cup B$, $wz \in M$, then add
$B \leftarrow B \cup \{z\}$, $A \leftarrow A \cup \{w\}$, $E \leftarrow E \cup \{vw, wz\}$.

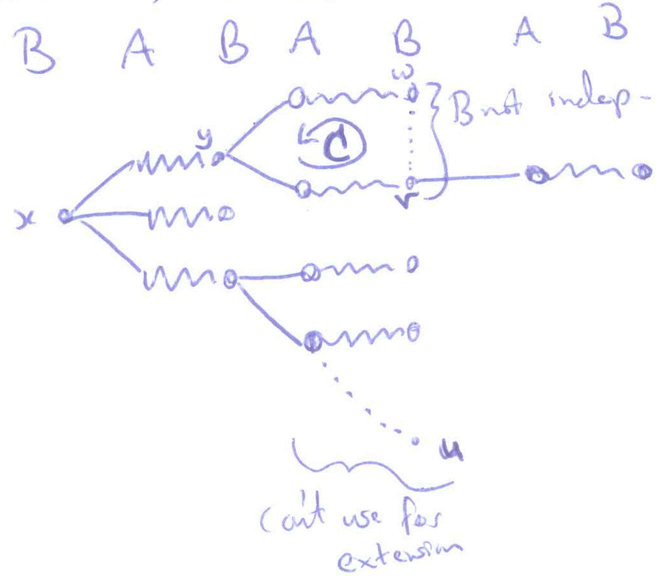
Augment: If $vu \in E$, $v \in B$, u unmatched, then path
P from x to v in T, then vu is M-augmenting.

If neither step above occurs, then T is stable $\Rightarrow$ all edges w/ an endpt in B have other endpt in $A \cup B$.

If T is stable and B is indep set, then T is frustrated.

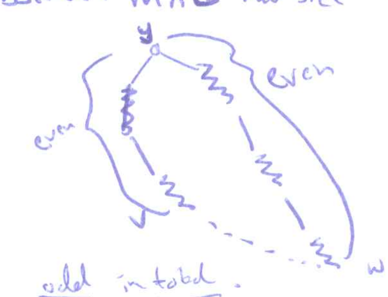
If a frustrated tree exists, then G has no perfect matching!

Otherwise, a stable tree looks like:



Let C be an odd circuit given by an edge vw w/ $v, w \in B(T)$.

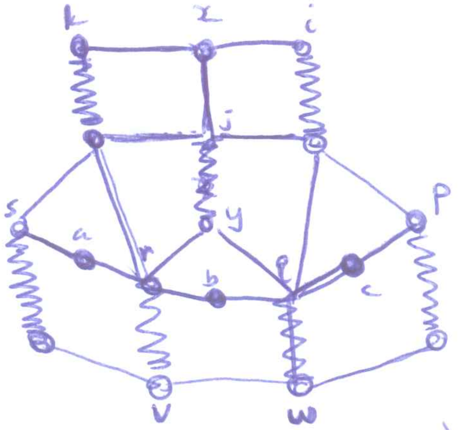
Observe: $M \cap C$ has size $\frac{|C|-1}{2}$.



If we shrink G on C, we get a new graph G', and new matching M' that are easy to predict!

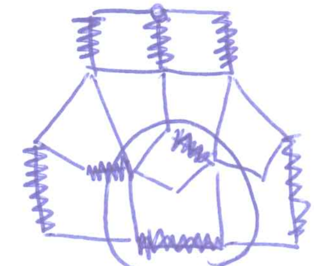
Ex:

G:



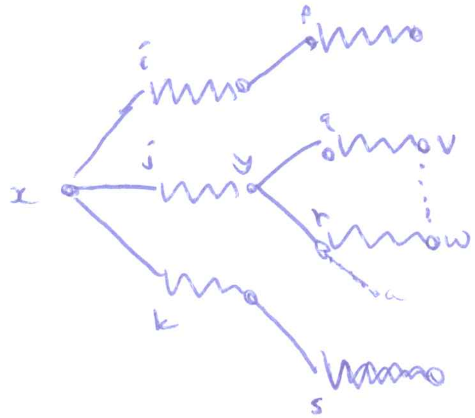
Observe: Other unsaturated vtxs are only adjacent to things in A(T)

total of 3 edges changed!



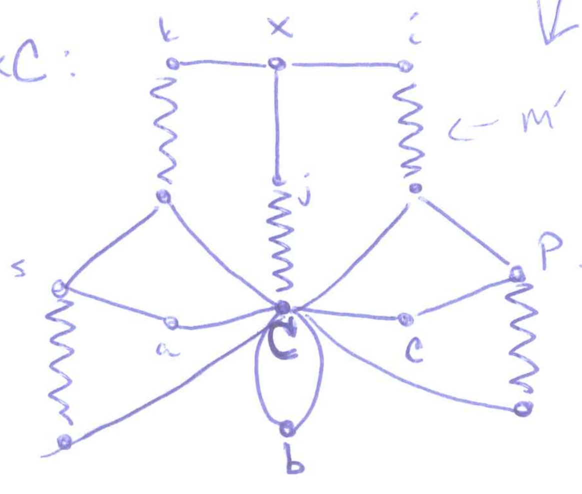
Corresponds to M-augmenting path xjyqwvra (requires backtracking on T!!)

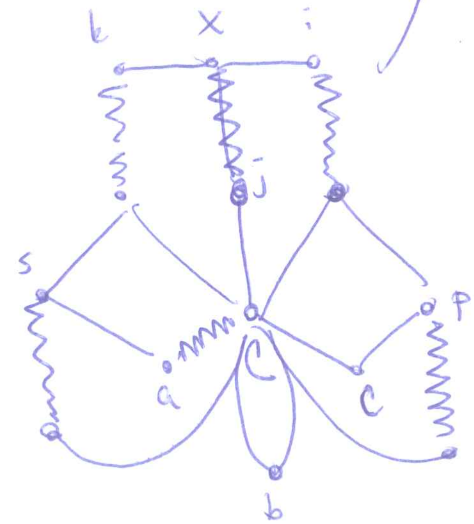
T:



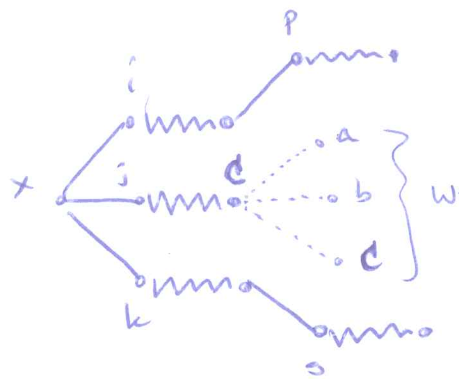
$C = yqvwr$.

$G' = G \times C$:



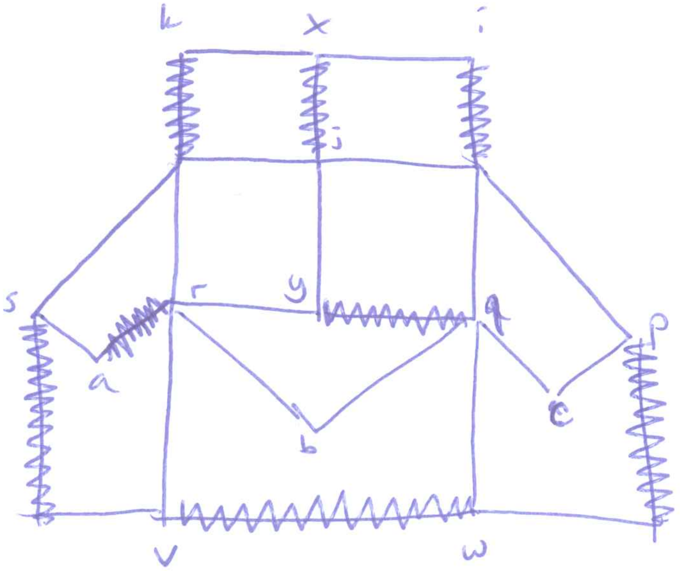


T':

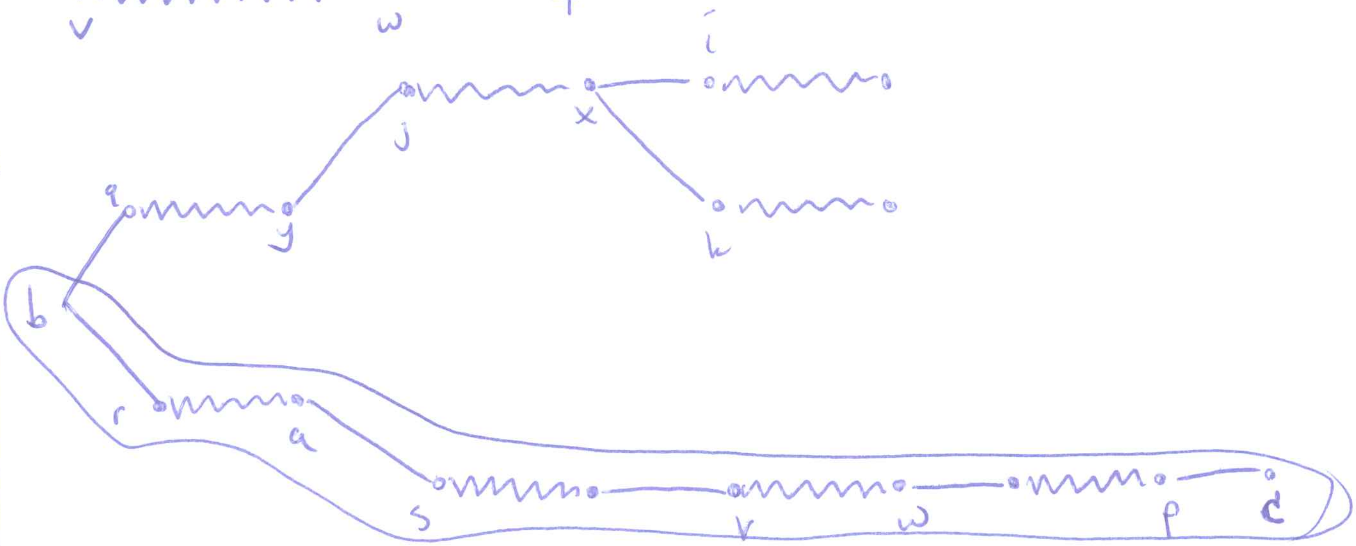


we can augment!

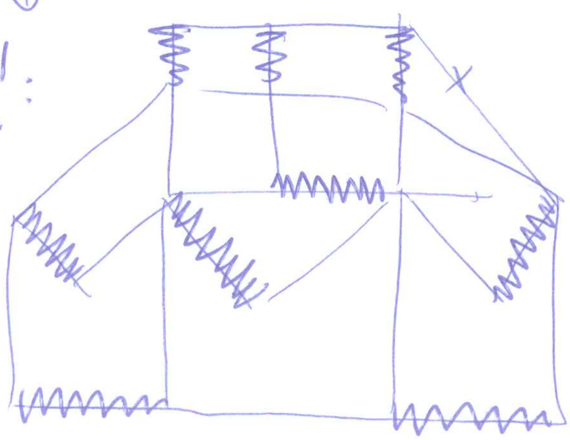
$P = x\, j\, C\, a$



T:



M':



Blossom Algorithm for Perfect Matching

Input graph G and matching M (not perfect).

Output: Either a ~~M-augmenting path~~ larger matching or a set A s/t $oc(G-A) > |A|$.

Init: $M' \leftarrow M$, $G' \leftarrow G$,

Let x be M'-unsaturated. Let $T = \{x\}$.

Iteration: ~~Build~~ Extend an ~~stable~~ M'-alternating tree T rooted at x. If an M'-augmenting path P' is found, then output P, w/ prop

~~Iteration~~: If T is frustrated, then output $A(T)$.

Otherwise: $\exists\, vw \in E(G')$ w/ $v, w \in B(T)$.

Let C be the odd cycle given by vw-path in $T + vw$.

$G' \leftarrow G' \times C$

$M' \leftarrow M - E(C)$

$T \leftarrow T \times (C)$ w/sh

Repeat

If P is M'-augmenting, then augment M' by P to find M''. Unshrink G' to G, iteratively extending M'' to a matching of G.

Crucial: Shrinking odd cycles places a weight on the resulting vtx.

This weight will always be odd!

Use vw to shrink and update M' and T.

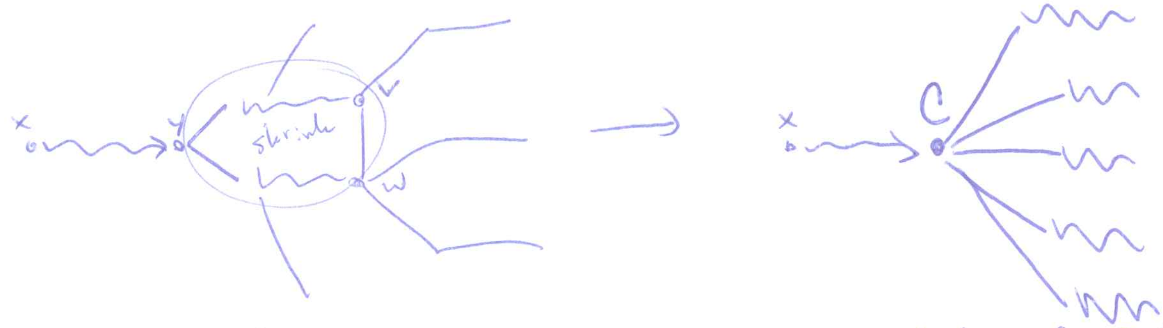
Given: M' a matching in G', T an M'-alternating tree, and edge vw in G' w/ $v, w \in B(T)$

Action: Let C be odd cycle formed by vw together w/ the path in T from v to w.

$$G' \leftarrow G' \times C$$

$$M \leftarrow M' - E(C)$$

$$T \leftarrow \underbrace{\cancel{T - E(C)}}_{\cancel{\text{as edges in } G'}} \quad \underbrace{T \times C}_{\text{shrink the vertices of } C}$$



Obs: After shrinking, we have M' a matching of G' and T is M'-alternating.

Blossom Alg for Max Matching

Extend M to largest "local" matching until finding a frustrated tree $= T$.

Let $G' = G - T$, then look in G' for an unsaturated vtx.
$M' = M - V(T)$ Start over, until frustrated.

We continue until we have $T_1, \dots, T_k$, disjoint frustrated trees ~~and~~ rooted at M-unsaturated vertices, and a perfect matching of $G_{k+1} = G - \bigcup_{i=1}^{k} T_i$.

So, $\underline{\mathrm{def}(G) \leq k.}$

But $|A(T_i)| \leq |B(T_i)| - 1$ for all $i \in \{1, \dots, k\}$,

AND, NO EDGES from $B(T_i)$ to $B(T_j)$, so

Stable Matchings

Let n universities and n recent Ph.D's consider each other. Each Ph.D wants exactly one job, and each university wants exactly one new hire.

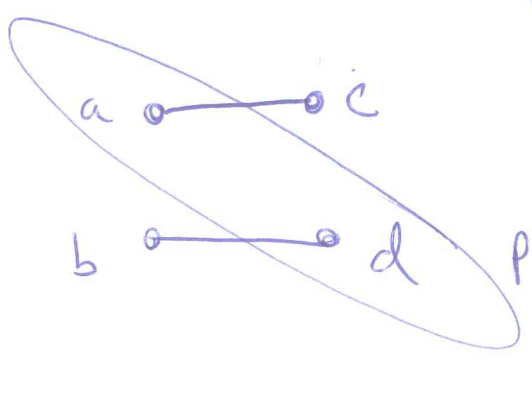
Each has some **preference** of the others.

Ex: $n = 4$. i, j, k, l graduates

+ w, x, y, z univ's.

i: $x > w > y > z$	w: $i > j > k > l$
j: $w > y > x > z$	x: $j > k > l > i$
k: $y > x > w > z$	y: $l > i > j > k$
l: $w > y > z > x$	z: $i > l > k > j$

A (perfect) matching M is stable if there is no pair of matched pairs ac, bd s/t a prefers d over c & d prefers a over b.



prefer each other over their matches, so they ignore the matching and go outside the "system".

Q: Do stable matchings always exist?

Gale-Shapely Proposal Algorithm

Init: Every university has a list of all applicants.

Iteration: Let x be a university that is not "engaged".

Let y be the top preference of x that has not yet rejected x.

x asks y "will you match with me?"

If y is not engaged, or y is engaged to x' and y prefers x to y then

(y rejects x')

y says "maybe" to x

Else:

y rejects x.

Termination: Let all "engagements" be permanent, output M.

EXAMPLE!!!!!

Thm: The Gale-Shapely Algorithm outputs a stable matching in $O(n^2)$ time.

Pf: There are at most n^2 pairs x, y, so at most n^2 iterations.

Let M be resulting matching. *

Suppose x & y form an unstable pair.

Then, when x proposed to y, x would have been accepted.

But since y is matched to x', y prefers x' to x.

Once a y is proposed to, it stays matched.

*: The matching exists: ~~Suppose x was rejected by all n y's~~

Prop: Among all stable matchings, the one produced from G-S alg is <u>best</u> for those doing the proposing.