Recall: Matchings in bipartite graphs

Theorem (Marriage Theorem; Hall, 1935) There is a matching in G saturating X iff $|N(S)| \geq |S|$ for every $S \subseteq X$.

$\alpha'(G)$ = size of largest matching

$C \subseteq V(G)$ is a vertex cover if for every edge $e \in E(G)$, $e \cap C \neq \emptyset$.
$\beta(G)$ = cardinality of the smallest vertex cover

Theorem. (König (1931), Egerváry (1931))
If G is bipartite then $\beta(G) = \alpha'(G)$.

Proof. For any minimum vertex cover Q, apply Hall's Condition to match $Q \cap X$ into $Y \setminus Q$ and $Q \cap Y$ into $X \setminus Q$. $\square$

Remark: Not true for general graphs, say for C_3 (or for any odd cycle)

Recall: Certificates

Suppose we knew that in some graph G with 1121 edges on 200 vertices, a particular set of 87 edges is (one of) the largest matching one could find. How could we convince somebody about this?

Once the particluar 87 edges are shown, it is easy to check that they are a matching, indeed.

But why isn't there a matching of size 88? Verifying that none of the $\binom{1121}{88}$ edgesets of size 88 forms a matching could take some time...

If we happen to be so lucky, that we are able to exhibit a vertex cover of size 87, we are saved. It is then reasonable to check that all 1121 edges are covered by the particular set of 87 vertices.

Exhibiting a vertex cover of a certain size **proves** that no larger matching can be found.

Certificate for bipartite graphs — Take 1

1. Correctness of the certificate:

A vertex cover $Q \subseteq V(G)$ is a certificate proving that no matching of G has size larger than $|Q|$.
That is: $\beta(G) \geq \alpha'(G)$, valid for every graph.

2. Existence of optimal certificate for bipartite graphs:

Theorem. (König (1931), Egerváry (1931))
If G is bipartite then $\beta(G) = \alpha'(G)$.

König's Theorem $\Rightarrow$ For bipartite graphs there always exists a vertex cover proving that a particular matching of maximum size is really maximum.

Remark. This is NOT the case for general graphs: C_5.

Certificate for bipartite graphs — Take 2

Let G be a bipartite graph with partite sets X and Y.

1. Correctness of the certificate:

A subset $S \subseteq X$ is a certificate proving that the largest matching in G has size at most $|X| - |S| + |N(S)|$.

2. Existence of optimal certificate:

Theorem (Marriage Theorem; Hall, 1935) There is a matching in G saturating X iff $|N(S)| \geq |S|$ for every $S \subseteq X$.

Corollary There exists a subset $S \subseteq X$, such that $\alpha'(G) = |X| - |S| + |N(S)|$.

Proof. Homework.

Problem: Certificate makes sense for bipartite graphs only.

Goal: Find a certificate for general graphs.

Matchings in general graphs

An odd component is a connected component with an odd number of vertices. Denote by $o(G)$ the number of odd components of a graph G.

Theorem. (Tutte, 1947) A graph G has a perfect matching iff $o(G - S) \leq |S|$ for every subset $S \subseteq V(G)$.

Proof.

$\Rightarrow$ Easy.

$\Leftarrow$ (Lovász, 1975) Consider a counterexample G with the maximum number of edges.

Claim. $G + xy$ has a perfect matching for any $xy \notin E(G)$.

Proof of Tutte's Theorem — Continued

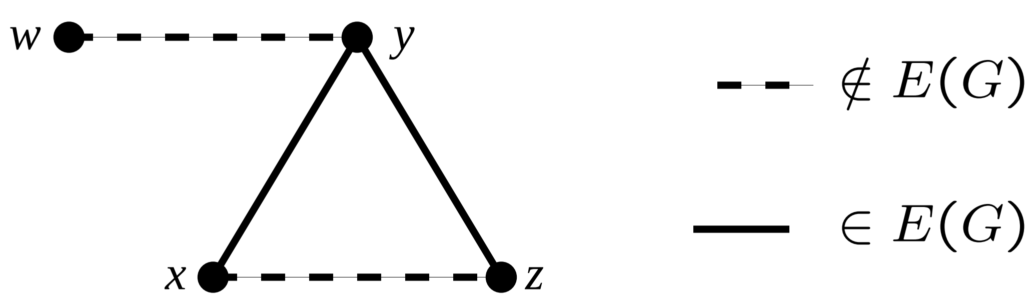
Define $U := \{v \in V(G) : d_G(v) = n(G) - 1\}$

Case 1. $G - U$ consists of disjoint cliques.

Proof: Straightforward to construct a perfect matching of G.

Case 2. $G - U$ is not the disjoint union of cliques.

Proof: Derive the existence of the following subgraph.



Obtain contradiction by constructing a perfect matching M of G using perfect matchings M_1 and M_2 of $G+xz$ and $G + yw$, respectively.

Corollaries

Yet another min/max theorem:

Corollary. (Berge,1958) In every graph G, he maximum number of vertices saturated by a matching is

$$2\alpha'(G) = \min\{n - o(G-S) + |S| : S \subseteq V(G)\}.$$

Proof. HW

Corollary. (Petersen, 1891) Every 3-regular graph with no cut-edge has a perfect matching.

Proof. Check Tutte's condition. Let $S \subseteq V(G)$.
Double-count the number of edges between an S and the odd components of $G - S$.
Observe that between any odd component and S there are at least three edges.

Arthur and Merlin – a touch of complexity

A: Show me a pairing, so my 150 knights can marry these 150 ladies!
M: Not possible!
A: Why?
M: Here are these 93 ladies and 58 knights, none of them are willing to marry each other.
A: Alright, alright ...

A: Seat my 150 knights around the round table, so that neighbors don't fight!
M: Not possible!
A: Why?
M: It will take me forever to explain you.
A: I don't believe you! Into the dungeon!

A YES/NO-problem problem is in the class *NP:* The answer **YES** can be checked “efficiently”
”efficiently”: within a time, which is polynomial in the size of the input

In other words:

- there is a ”certificate”, which a computer (i.e., Arthur, i.e., a polynomial time algorithm) can verify within a reasonable time

Note: the certificate can be provided by an all-powerful supercomputer (i.e., Merlin)

Examples:

“Does this bipartite graph have a perfect matching?” (provide perfect matching)

“Does this bipartite graph have **no** perfect matching?” (provide vertex cover of size **less** than $n/2$; certificate exists because of König’s Theorem)

“Does this graph have a Hamilton cycle?” (provide Hamilton cycle)

Merlin’s Pech: “Does this graph have **no** Hamilton cycle?” is not (known to be) in NP

A YES/NO-problem is in the class *co-NP:* The answer **NO** can be checked efficiently

Properties having a “good” characterization or a “min/max theorem” are both in NP and co-NP

Examples:

- ”Is this graph 2-colorable?” (NP: provide a 2-coloring; co-NP: provide an odd cycle)
- ”Is this graph Eulerian?” (NP: provide an ordered list of the edges for an Eulerian circuit; co-NP: provide a vertex with an odd degree; co-NP certificate **exists** because of Euler’s Theorem)
- ”Does this graph have a perfect matching?” (NP: provide a perfect matching; co-NP: provide a subset S whose deletion creates more than $|S|$ odd components; co-NP certificate **exists** because of Tutte’s Theorem)
- ”Is this graph k-connected?” (NP: for each two vertices $x, y \in V(G)$ provide a list of k internally disjoint x, y-path; co-NP: provide a cut-set of size less than k; NP-certificate **exists** because of Menger’s Theorem)

A YES/NO-problem is in the class *P*: The answer can be **found** efficiently (i.e., there is a polynomial time algorithm to actually obtain the certificate (i.e., no need for Merlin))

Of course: *P* $\subseteq$ *NP* $\cap$ *co-NP*

Often: Problems in *NP* $\cap$ *co-NP* are also in P

However: People think *P* $\neq$ *NP* $\cap$ *co-NP*

We don't know: status of problem "Is there a factor of n less than k?" (until 2002 the status of the problem "Is n a prime?" was also not known)

People also think: *P* $\neq$ *NP* (1,000,000 US dollars)

We don't know: Hamiltonicity, 3-colorability, $\Delta(G)$-edge-colorability, k-independence set, NP: “nondeterministic polynomial time”

NP-hard problem: every problem in NP can be reduced to it in polynomial time (consequently, giving a polynomial time algorithm for it would result in a polynomial time algorithm for **all** problems in NP, and hence P=NP)

NP-complete problem: NP-hard and contained in NP

Many problems are NP-complete: Hamiltonicity, 3-colorability, $\Delta(G)$-edge-colorability

Approximation algorithm for TSP

Traveling Salesman Problem (TSP)

Input: $w : E(K_n) \to \mathbb{R}$.

Output: Hamilton cycle H of smallest weight $w(H) = \sum_{e \in E(H)} w(e)$.

Special case: Is there a Hamilton cycle in G? (reduction via $0/1$-weights)

Hence it is NP-complete as well (The decision problem version: Is there Hamilton cycle with weight at most k?)

A practical approach: Let w_{OPT} be the weight of a traveling salesman tour of minimum weight. For a $c \geq 1$, a c-*approximation algorithm* is an algorithm which outputs a Hamilton cycle H with $w(H) \leq c \cdot w_{OPT}$

Algorithm `TSP-Approx`

Step 1. Find MST T

Step 2. Create walk W "around" T, traversing each edge twice

Step 3. Set $H = W$ and go around H and iteratively change it by "shortcuting" at any vertex which is used the second time. Output H when $e(H) = n$

Remark. Running time: fast (Kruskal + $O(n)$)

Theorem If w satisfies the triangle inequality, then `TSP-Approx` is a 2-approximation algorithm.

Proof. Let T_{min} a MST of G. Then $w(W) = 2w(T_{min})$

By triangle inequality, shortcut decreases the sum of the weights of H, so $w(H) \leq 2w(T_{min})$

Hamilton path within an optimal traveling salesman tour is a spanning tree, so $w(T_{min}) \leq w_{OPT}$ $\square$