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: problem of "Is there a factor less than k?"

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,

Hamiltonian cycles

A spanning cycle is called a Hamiltonian cycle. A graph is called Hamiltonian if it contains a Hamiltonian cycle.

Example $K_{m,n}$

Example. Petersen graph is not Hamiltonian

A spanning path is called a Hamiltonian path.

Recall: Matchings

A matching is a set of (non-loop) edges with no shared endpoints. The vertices incident to an edge of a matching M are saturated by M, the others are unsaturated. A perfect matching of G is matching which saturates all the vertices.

Examples. $K_{n,m}$, K_n, Petersen graph, Q_k; graphs without perfect matching

A maximal matching cannot be enlarged by adding another edge.

A maximum matching of G is one of maximum size.

Example. Maximum $\neq$ Maximal

Recall: Characterization of maximum matchings

Let M be a matching. A path that alternates between edges in M and edges not in M is called an M-alternating path.

An M-alternating path whose endpoints are unsaturated by M is called an M-augmenting path.

Theorem(Berge, 1957) A matching M is a maximum matching of graph G iff G has no M-augmenting path.

Proof. ($\Rightarrow$) Easy.

($\Leftarrow$) Suppose there is no M-augmenting path and let M^* be a matching of maximum size.

What is then $M \triangle M^*$???

> **Lemma** Let M_1 and M_2 be matchings of G. Then each connected component of $M_1 \triangle M_2$ is a path or an even cycle.
>
> For two sets A and B, the symmetric difference is $A \triangle B = (A \setminus B) \cup (B \setminus A)$.

Recall: Hall's Condition and consequences

Theorem (Marriage Theorem; Hall, 1935) Let G be a bipartite (multi)graph with partite sets X and Y. Then there is a matching in G saturating X iff $|N(S)| \geq |S|$ for every $S \subseteq X$.

Proof. ($\Rightarrow$) Easy.

($\Leftarrow$) Not *so* easy. Find an M-augmenting path for *any* matching M which does not saturate X.
(Let U be the M-unsaturated vertices in X. Define

$$\begin{aligned} T &:= \{y \in Y : \exists\ M\text{-alternating } U, y\text{-path}\}, \\ S &:= \{x \in X : \exists\ M\text{-alternating } U, x\text{-path}\}. \end{aligned}$$

Unless there is an M-augmenting path, $S \cup U$ **violates** Hall's condition.)

Corollary. (Frobenius (1917)) For $k > 0$, every k-regular bipartite (multi)graph has a perfect matching.

Recall: Application: 2-Factors

A factor of a graph is a spanning subgraph. A k-factor is a spanning k-regular subgraph.

Every regular bipartite graph has a 1-factor.

Not every regular graph has a 1-factor.

But...

Theorem. (Petersen, 1891) Every $2k$-regular graph has a 2-factor.

Proof. Use Eulerian cycle of G to create an auxiliary k-regular bipartite graph H, such that a perfect matching in H corresponds to a 2-factor in G.

Recall: Graph parameters

The size of the largest matching (*independent set of edges*) in G is denoted by $\alpha'(G)$.

A vertex cover of G is a set $Q \subseteq V(G)$ that contains at least one endpoint of every edge. (The vertices in Q *cover* $E(G)$).
The size of the smallest vertex cover in G is denoted by $\beta(G)$.

Claim. $\beta(G) \geq \alpha'(G)$.

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 particicluar 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 $\beta(G) = \alpha'(G)$.

Remarks

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

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

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

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 $S \subseteq X$.

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

Proof. Homework.

Problem: Certificate makes sense for bipartite graphs only.

How to find a maximum matching in bipartite graphs?

Augmenting Path Algorithm

Input. A bipartite graph G with partite sets X and Y, a matching M in G.

Output. EITHER an M-augmenting path OR a certificate (a cover of the same size) that M is maximum.

Idea. Let U be set of unsaturated vertices in X.
Explore M-alternating paths from U, letting $S \subseteq X$ and $T \subseteq Y$ be the sets of vertices reached.
As a vertex is reached, record the previous vertex on the M-alternating path from which it was reached.
Mark vertices of S that have been fully explored for path extensions (say, put them into a set Q).

Initialization. $S = U$, $Q = \emptyset$, and $T = \emptyset$.

Iteration.

IF $Q = S$ THEN

stop and report that M is a maximum matching

and $T \cup (X \setminus S)$, is a cover of the same size.

ELSE

select $x \in S \setminus Q$ and

FORALL $y \in N(x)$ with $xy \notin M$ DO

IF y is unsaturated, THEN

stop and report an M-augmenting path

from U to y.

ELSE

$\exists w \in X$ with $yw \in M$. Update

$T := T \cup \{y\}$ (y is reached from x),

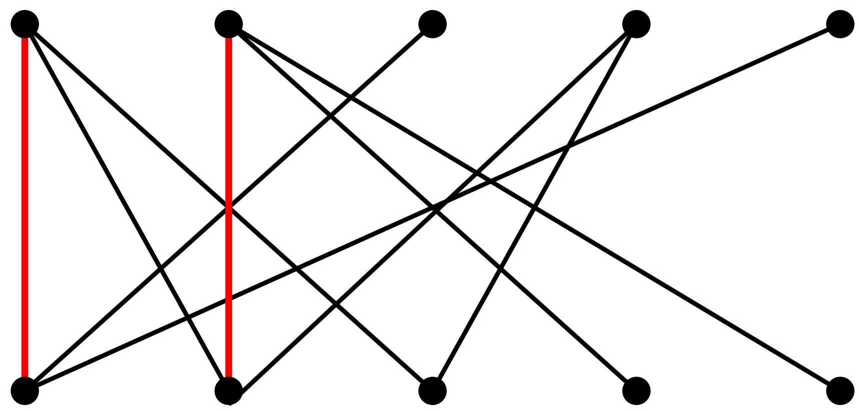
$S := S \cup \{w\}$ (w is reached from y).

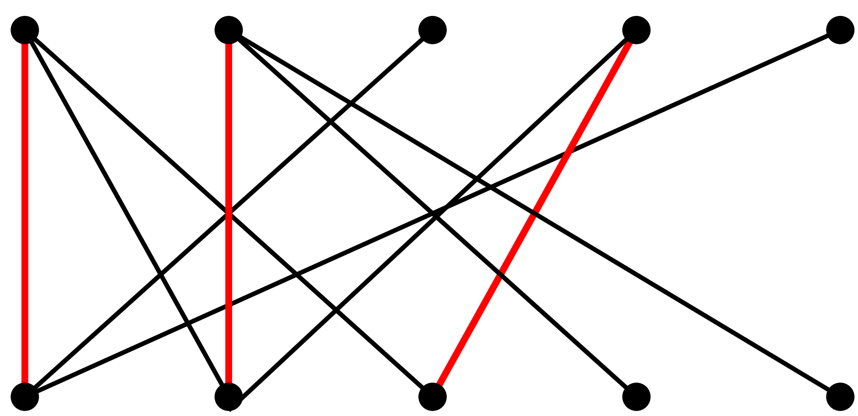
update $Q := Q \cup \{x\}$

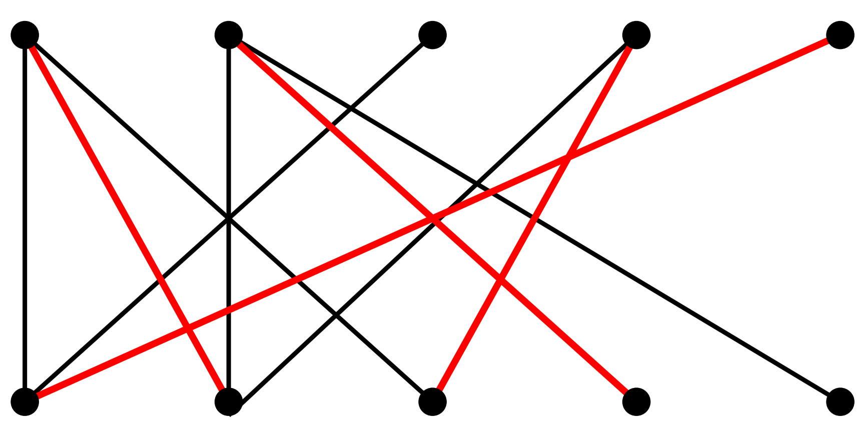
iterate.

Theorem. Repeatadly applying the Augmenting Path Algorithm to a bipartite graph produces a maximum matching and a minimum vertex cover.

If G has n vertices and m edges, then this algorithm finds a maximum matching in $O(nm)$ time.







Proof of correctness

If Augmenting Path Algorithm does what it supposed to, then after at most $n/2$ application we can produce a maximum matching.
Why does the APA terminate? It touches each edge at most once. Hence running time is $O(nm)$.

What if an M-augmenting path is returned? It is OK, since y is an unsaturated neighbor of $x \in S$, and x can be reached from U on an M-alternating path.

What if the APA returns M as maximum matching and $T \cup (X \setminus S)$ as minimum cover?

Then all edges leaving S were explored, so there is **no edge between S and $Y \setminus T$**.

- Hence $T \cup (X \setminus S)$ is indeed a cover.
- $|M| = |T| + |X \setminus S|$ (By selection of S and T.)

Key Lemma If a cover and a matching have the same size in any graph, then they are both optimal.

$$|M| \leq \alpha'(G) \leq \beta(G) \leq |T \cup (X \setminus S)| = |M|.$$

How to find a maximum weight matching in a bipartite graph?

In the maximum weighted matching problem a non-negative weight $w_{i,j}$ is assigned to each edge $x_i y_j$ of $K_{n,n}$ and we seek a perfect matching M to maximize the total weight $w(M) = \sum_{e \in M} w(e)$.

With these weights, a (weighted) cover is a choice of labels $u_1, \dots, u_n$ and $v_1, \dots, v_n$, such that $u_i + v_j \geq w_{i,j}$ for all i, j. The cost $c(u, v)$ of a cover (u, v) is $\sum u_i + \sum v_j$. The minimum weighted cover problem is that of finding a cover of minimum cost.

Duality Lemma For a perfect matching M and a weighted cover (u, v) in a bipartite graph G, $c(u, v) \geq w(M)$. Also, $c(u, v) = w(M)$ iff M consists of edges $x_i y_j$ such that $u_i + v_j = w_{i,j}$. In this case, M and (u, v) are both optimal.

The algorithm

The equality subgraph $G_{u,v}$ for a weighted cover (u, v) is the spanning subgraph of $K_{n,n}$ whose edges are the pairs $x_i y_j$ such that $u_i + v_j = w_{i,j}$. In the cover, the excess for i, j is $u_i + v_j - w_{i,j}$.

Hungarian Algorithm

Input. A matrix $(w_{i,j})$ of weights on the edges of $K_{n.n}$ with partite sets X and Y.

Idea. Iteratively adjusting a cover (u, v) until the equality subgraph $G_{u,v}$ has a perfect matching.

Initialization. Let $u_i = \max\{w_{i,j} : j = 1, \ldots, n\}$ and $v_j = 0$.

Iteration.

Form $G_{u,v}$ and find a maximum matching M in it.

IF M is a perfect matching, THEN

  stop and **report** M as a maximum weight matching and (u, v) as a minimum cost cover

ELSE

  let Q be a vertex cover of size $|M|$ in $G_{u,v}$.

  $R := X \cap Q$

  $T := Y \cap Q$

  $\epsilon := \min\{u_i + v_j - w_{i,j} : x_i \in X \setminus R, y_j \in Y \setminus T\}$

  Update u and v:

    $u_i := u_i - \epsilon$ if $x_i \in X \setminus R$

    $v_j := v_j + \epsilon$ if $y_j \in T$

Iterate

Theorem The Hungarian Algorithm finds a maximum weight matching and a minimum cost cover.

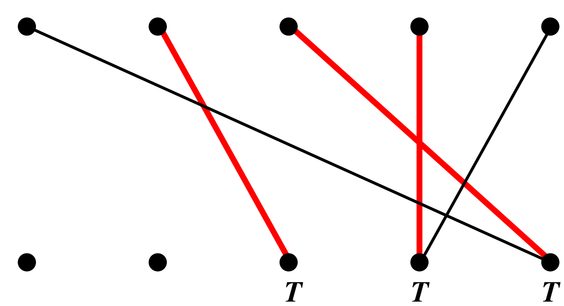
The Assignment Problem — An example

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 7 & 2 \\ 1 & 3 & 4 & 4 & 5 \\ 3 & 6 & 2 & 8 & 7 \\ 4 & 1 & 3 & 5 & 4 \end{pmatrix}$$

Excess Matrix

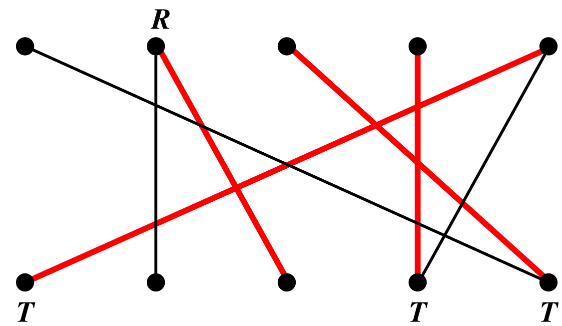
$$\begin{array}{cc} & \begin{array}{ccccc} 0 & 0 & 0 & 0 & 0 \end{array} \\ \begin{array}{c} 5 \\ 8 \\ 5 \\ 8 \\ 5 \end{array} & \begin{pmatrix} \color{blue}{4} & \color{blue}{3} & 2 & 1 & 0 \\ \color{blue}{2} & \color{green}{1} & 0 & 1 & 6 \\ \color{blue}{4} & \color{blue}{2} & 1 & 1 & 0 \\ \color{blue}{5} & \color{blue}{2} & 6 & 0 & 1 \\ \color{green}{1} & \color{blue}{4} & 2 & 0 & 1 \end{pmatrix} \\ & \begin{array}{ccccc} & & T & T & T \end{array} \end{array}$$

Equality Subgraph



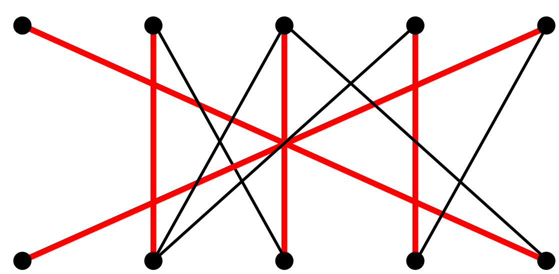
$\epsilon = 1$

$$
\begin{array}{c c c}
 & \begin{array}{ccccc} 0 & 0 & 1 & 1 & 1 \end{array} & \\
\begin{array}{c} 4 \\ 7 \\ 4 \\ 7 \\ 4 \end{array} &
\left(\begin{array}{ccccc}
3 & {\color{blue}2} & {\color{blue}2} & 1 & 0 \\
1 & 0 & 0 & 1 & 6 \\
3 & {\color{green}1} & {\color{green}1} & 1 & 0 \\
4 & {\color{green}1} & {\color{blue}6} & 0 & 1 \\
0 & {\color{blue}3} & {\color{blue}2} & 0 & 1
\end{array}\right) &
\begin{array}{c} \\ R \\ \\ \\ \\ \end{array} \\
 & \begin{array}{ccccc} T & & & T & T \end{array} &
\end{array}
$$



$$\epsilon = 1$$

$$
\begin{array}{c c}
 & \begin{array}{ccccc} 1 & 0 & 1 & 2 & 2 \end{array} \\
\begin{array}{c} 3 \\ 7 \\ 3 \\ 6 \\ 3 \end{array} &
\left(\begin{array}{ccccc}
3 & 1 & 1 & 1 & 0 \\
2 & 0 & 0 & 2 & 7 \\
3 & 0 & 0 & 1 & 0 \\
4 & 0 & 5 & 0 & 1 \\
0 & 2 & 1 & 0 & 1
\end{array}\right)
\end{array}
$$



DONE!!

The Duality Lemma states that if $w(M) = c(u,v)$ for some cover (u,v), then M is maximum weight.

We found a maximum weight matching (transversal). The fact that it is maximum is certified by the indicated cover, which has the same cost:

$$
\begin{array}{cc}
 & \begin{array}{ccccc} 1 & 0 & 1 & 2 & 2 \end{array} \\
\begin{array}{c} 3 \\ 7 \\ 3 \\ 6 \\ 3 \end{array} &
\begin{pmatrix}
1 & 2 & 3 & 4 & \mathbf{5} \\
6 & \mathbf{7} & 8 & 7 & 2 \\
1 & 3 & \mathbf{4} & 4 & 5 \\
3 & 6 & 2 & \mathbf{8} & 7 \\
\mathbf{4} & 1 & 3 & 5 & 4
\end{pmatrix}
\end{array}
$$

$$
\begin{aligned}
w(M) &= 5+7+4+8+4 = 28 = \\
&= 1+0+1+2+2+ \\
&\quad\; 3+7+3+6+3 = c(u,v)
\end{aligned}
$$

Hungarian Algorithm — Proof of correctness

Proof. If the algorithm ever terminates and $G_{u,v}$ is the equality subgraph of a (u, v), which is indeed a cover, then M is a m.w.m. and (u, v) is a m.c.c. by Duality Lemma.

Why is (u, v), created by the iteration, a cover?

Let $x_i y_j \in E(K_{n,n})$. Check the four cases.

$x_i \in R,$	$y_j \in Y \setminus T$	$\Rightarrow$	u_i and v_j do not change.
$x_i \in R,$	$y_j \in T$	$\Rightarrow$	u_i does not change v_j increases.
$x_i \in X \setminus R,$	$y_j \in T$	$\Rightarrow$	u_i decreases by ϵ, v_j increases by ϵ.
$x_i \in X \setminus R,$	$y_j \in Y \setminus T$	$\Rightarrow$	$u_i + v_j \geq w_{i,j}$ by definition of ϵ.

Why does the algorithm terminate?

M is a matching in the new $G_{u,v}$ as well. So either
(i) max matching gets larger or
(ii) # of vertices reached from U by M-alternating paths grows. (U is the set of unsaturated vertices of M in X.)