Victoria University of Wellington

Math 214

Course Notes | Discrete Mathematics 2 — Graph Theory | 2004

Lectures 9+10+11+12: Matching

Matching is pairing: dividing a collection of objects into pairs in an optimal way, where the pairs have to be chosen from a prescribed list.

Here is an example. On a certain day a company has to make a number of deliveries. It has drivers available; each can make two deliveries that day. It is clearly advantageous to let as many drivers as possible make two deliveries. However not each two deliveries can be combined (because they are to far apart or they have to be delivered at overlapping times, or what ever); it is known which pairs can be combined and which not. Combining as much deliveries as possible is a matching problem.

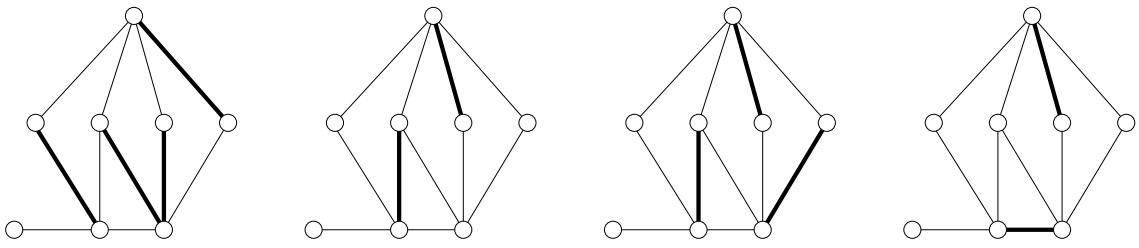
The matching problem

For a formal definition of matching problems we turn to graphs (in these matching notes graphs are undirected unless specified otherwise). A set M of edges in a graph is a *matching* if no vertex of the graph is end of more than one edge in M. A matching M is a *maximum matching* if no other matching in the graph has more edges than M has. The size of a maximum matching in a graph G is called the *matching number* of G; it is denoted by $\nu(G)$. The *matching problem* is to find a maximum matching in a given graph.

To justify that the delivery problem above is indeed a matching problem, I have tell you what the graph is; I leave that to you for now.

One may think naively that one can find a maximum matching by starting with the matching with no edges and making larger and larger matchings by greedily adding edges one by one. This may not work however as one could get stuck with a "maximal" matching that is not maximum; a *maximal* matching is a matching that cannot be extended to a larger matching by adding an edge. So finding maximum matching needs some intelligence.

The pictures below illustrate some of the just defined notions. The third one from the left claims that the matching number of that graph is 3. How I justify that? ...



Not a matching | A matching | A maximum matching | A maximal matching that is not maximum

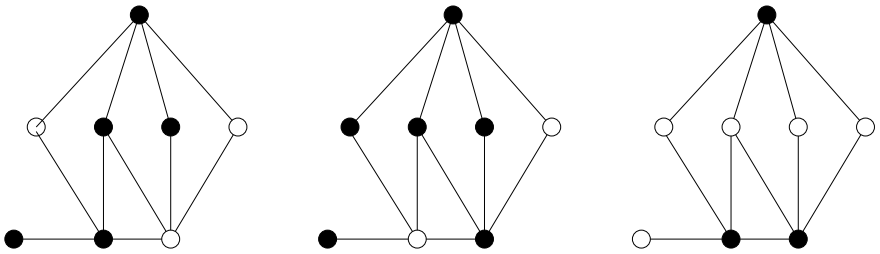
Upper bounds—vertex covers—König's matching theorem

In solving a optimization problem it is very important to have an upper bound to the optimal value (in this case $\nu(G)$). For the matching problem we can use so called vertex covers to get upper bounds. A set U of vertices of a graph is a *vertex cover* if every edge of the graph has at least one of its ends in U.

(1) LEMMA. *If M is a matching and U a vertex cover in the same graph, then $|M| \leq |U|$.*

Proof. Indeed, each edge in M has at least one of its ends in U (as U is a vertex cover). As on the other hand no vertex of U is on two or more edges in M (as M is a matching), we see that indeed $|M| \leq |U|$. □

So the size of each vertex cover in a graph G is an upper bound for $\nu(G)$. So if we have a matching and a vertex cover and they happen to have the same cardinality, we know that the matching is maximum. For instance the vertex cover in the picture below on the right shows that the matching on the previous page in the second picture from the right is maximum.

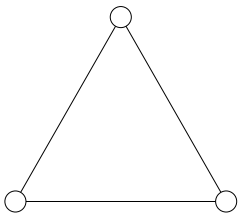


Not a vertex cover A vertex cover A minimum vertex cover

The smallest size of a vertex cover in a graph G is called the *vertex cover number* of G and denoted by $\tau(G)$. Lemma (1) implies that for any graph G we have that

(2) $\nu(G) \leq \tau(G)$.

Unfortunately there are graphs for which $\nu(G)$ is strictly smaller than $\tau(G)$. For instance the graph below has matching number 1 and vertex cover number 2.



So in a graph like that we have no hope that we can prove the optimality of a matching by a vertex cover. However for bipartite graphs things are nice.

(3) KÖNIG'S MATCHING THEOREM. *If G is a bipartite graph, then $\nu(G) = \tau(G)$.*

This tells us that in a bipartite graph we can certify optimality of a matching by a vertex cover. It does however not tell us how to find a maximum matching and how to find a minimum vertex cover. I will now give an algorithm that does this in a bipartite graph. Of course I also need to prove König's theorem; I will use the algorithm for that.

Augmenting paths ... larger matchings

Let M be a matching. We call an edge in M a *matching edge* and each other edge a *nonmatching edge*. We call a vertex u *free* it is **not** an end of any matching edge. We call a path $P = u_0, e_1, u_1, \ldots, e_k, u_k$ an *augmenting path* (with respect to M) if:

- u_0 and u_k are free,
- the even numbered edges $e_2, e_4, \ldots, e_{k-1}$ are matching edges.

Note that it follows from this definition that k is odd and that the odd numbered edges are nonmatching edges. The importance of augmenting paths lies in the following easy fact:

(4) *The set of edges $M \setminus \{e_2, e_4, \ldots, e_{k-1}\} \cup \{e_1, e_3, \ldots, e_k\}$ is also a matching; it has one edge more than M has.*

So if we find an augmenting path, we can construct a larger matching.

How to find augmenting paths in bipartite graphs

As of now our graph G is bipartite. So we have a partition of the vertex set in two parts A and B and each edge of G has one end in A and one end in B. The following easy observation is crucial in understanding how to find augmenting paths in bipartite graphs.

(5) *Each augmenting path in bipartite graph has one end in A and one end in B. Following such augmenting path starting from its end in A, we traverse nonmatching edges from A to B and matching edges from B to A.*

Motivated by this we construct the following auxiliary graph. It is a directed graph obtained from G and M by orienting all nonmatching edges from A to B and all matching edges from B to A. We denote this directed graph by $\overrightarrow{G_M}$. Now (5) immediately implies the following:

(6) *G contains an augmenting path with respect to M if and only if $\overrightarrow{G_M}$ contains a directed path from a free vertex in A to a free vertex in B.*

Finding such a directed path in B just amounts to a simple graph search, so we can find augmenting paths in a bipartite graph easily.

What if no augmenting path exists?

Assume that there is no directed path as in (6). Let X denote the vertices that are reachable by a directed path that starts at a free vertex in A and let $U := (A \setminus X) \cup (B \cap X)$.

(7) *U is a vertex cover.*

Proof. Suppose not; then G has an edge ab with $a \in A$, $b \in B$ and $a, b \notin U$. Hence, by definition of U, we have that $a \in X$ and $b \notin X$. So there exists a directed path in $\overrightarrow{G_M}$ that starts in a free node in A and ends in a; call it P. As b is not in X, b does not lie on P.

If a is free, then $ab \notin M$, hence $\overrightarrow{ab} \in \overrightarrow{G_M}$, so $b \in X$. As that is not the case, a is not free. So P has more vertices than just a. Let $\overrightarrow{xa}$ be the last edge on P, so $ax \in M$. Then as b is not on P, x is not b. Hence as M is a matching $ab \notin M$, so $\overrightarrow{ab} \in \overrightarrow{G_M}$. Thus $P + \overrightarrow{ab}$ is directed path from a free node in A to b. This contradicts that $b \notin X$. So (7) follows. □

(8) $|U| = |M|$.

Proof. We first proof that U contains no free vertices. Indeed if $u \in U \cap A$ then $u \notin X$, so u is not reachable by a directed path from a free vertex in A, so as $u \in A$ it is not free by itself. If $u \in U \cap B$, then $u \in X$. Hence u is reachable by a directed path from a free vertex in A, so as there is no augmenting path, u is not free it self. So U has no free vertices indeed. Hence each element of U is end of a matching edge.

Next we prove that no matching edge has both ends in U. Indeed, otherwise there exist a matching edge ab with $a \in U \cap A$ and $b \in U \cap B$. Then $b \in X$ and $\overrightarrow{ba} \in \overrightarrow{G_M}$, so $a \in X$. However this contradicts that $a \notin U$. So indeed no matching edge has both ends in U.

So $|U| \leq |M|$. As by (7) and (1) we already know that $|M| \leq |U|$, (8) follows. □

The maximum matching algorithm for bipartite graphs

Summarizing we get the following algorithm for finding a maximum matching in a bipartite graph.

Initialize: $M := \emptyset$.

Search an augmenting path: Let X be the set of vertices reachable by a directed path starting at a free vertex in A. (We can find X by a search in $\overrightarrow{G_M}$.)

- If $B \cap X$ contains a free vertex, then there exists a directed path from a free vertex in A to a free vertex in B (this path can be found via the parent labels created by the search); the path forms an augmenting path in G.
  Remove the matching edges on the path from M and add the nonmatching edges on the path to M.
  Repeat "*Search an augmenting path*" with the new matching.
- If $B \cap X$ contains no free vertices, **stop**: M is a maximum matching and $U := (A \setminus X) \cup (B \cap X)$ is a minimum vertex cover.

It is quite easy to see that this only takes polynomially many of steps, so I leave that to you.

Proof of König's theorem

The correctness of the algorithm also proves *König's theorem*. Indeed, let G be a bipartite graph. Run the algorithm on G, as the matching gets larger at every iteration of "*Search an augmenting path*" it cannot run forever. When it stops, it gives as output a matching and a vertex cover with the same size. So by (2), $\nu(G)$ and $\tau(G)$ have to be equal.

Intermezzo: what if the graph is nonbipartite?

Also if the graph is nonbipartite an augmenting path yields a larger matching. It turns out that the reverse is also true: if no augmenting path exists the matching is maximum. There is also an algorithm that finds such an augmenting path if it exists and finds an upper bound that certifies that we cannot do better otherwise (clearly the upper bound is more involved

then the vertex cover bound). The algorithm uses only polynomially many steps. So the matching problem can be solved in a polynomial number of steps in any graph. However this general algorithm is a lot more complicated as the one above that only works for bipartite graphs. It has been published by Jack Edmonds in a famous 1963 paper. The paper is also important because it was the first one that mentioned the notion of "polynomial time" as a crucial measure of the efficiency of an algorithm.

(Side comment: if you want to find minimum vertex covers in general graphs, you may run into trouble. No polynomial time algorithms are known for that.)

Hall's marriage theorem

Back to bipartite graphs. Again each edge has one end in A and one end in B. We now consider the question: *Can we match A into B?* By that we mean: Does there exists a matching M such that each vertex in A is end of an edge in M?

Clearly, the answer is no if there exists a set of vertices X such that the set $\Gamma(X)$ of neighbours of X is smaller than X. It turns out that this criterion characterizes if we can match A into B.

(9) HALL'S MARRIAGE THEOREM. *For each bipartite graph with parts A and B either we can match A into B or (exclusively)*[1] *A has a subset X such that $|\Gamma(X)| < |X|$.*

Proof. We already observed that if a set X as indicated exists we cannot match A into B. So it remains to prove that if we cannot match A into B we can find a set X as claimed.

So assume we cannot match A into B. This means that $\nu(A) < |A|$. Hence, by König's Theorem, $\tau(G) < |A|$. So the graph has a vertex cover W with $|W| < |A|$. Let $X := A \setminus W$.

We prove that $\Gamma(X) \subseteq W \cap B$. Indeed, let $b \in \Gamma(X)$. Then there exists a vertex $a \in X$ such that ab is an edge. As $X := A \setminus W$, vertex a does not lie in W. So, as W is a vertex cover, $b \in W$. Moreover, as $a \in X \subseteq A$, we have also that $b \in B$. Hence, $b \in W \cap B$. So we proved that $\Gamma(X) \subset W \cap B$, as required.

So

$$|\Gamma(X)| \leq |W \cap B| = |W| - |W \cap A| = |W| - |A| + |A \setminus W| = \tau(G) - |A| + |X| < |X|.$$

So X is indeed a subset of A with $\Gamma(X) < |X|$. □

Systems of distinct representatives

A recurring theme in discrete mathematics is that it may happen that different notions and theorems appear to be very different but in fact are the same thing in disguise. We give an example of this phenomenon.

Let S be a set and $S_1, S_2, \ldots, S_m$ a collection of subsets of S. We call a list $(a_1, a_2, \ldots, a_n)$ a *system of distinct representatives* for the collection $S_1, S_2, \ldots, S_m$ if the a_i's are all different and $a_i \in S_i$ for each $i = 1, \ldots, m$.

(10) THEOREM. *Let S be a finite set. A collection of subsets of S either has a system of distinct representatives or (exclusively) there exists an integer k and k sets in the collection such that the union of these k sets has less than k elements.*

[1] By "or (exclusively)" I mean that there is no bipartite graph for which both properties hold.

Proof. Let $S_1, S_2, \ldots, S_m$ be the collection of subsets of S. Construct a bipartite graph: one part of the vertex set is $S_1, S_2, \ldots, S_m$ and the other part is S; moreover, S_i (with $i = 1 \ldots, m$) and $s \in S$ are joined by an edge if and only if $s \in S_i$. It is clear that the family has a system of distinct representatives if and only if in this graph $S_1, S_2, \ldots, S_m$ matches into S. The theorem then follows by interpreting Hall's marriage theorem or this graph. □

The proof shows that we can find systems of distinct representatives by matching algorithms.