

Matching Problems

Suppose we have a bipartite graph G , and seek a set of disjoint pairing of vertices, one member of each pair in each part, under the condition that paired vertices are adjacent (they form an edge) in the graph G .

Among the questions that we can raise about such pairings, or “matchings” in a graph are: How large can such a pairing be? How can we find a maximum sized matching? How many steps does it take to do so?

Similar problems exist in non-bipartite graphs with interesting answers, and there are similar problems in which each edge in the graph has a “value” and we want a matching whose edge value sum is as large as possible. And there are somewhat more general “commodity flow” problems that are closely related to matching problems.

A bipartite graph is also called a two colorable graph, and we will call the two parts of G its Left and Right parts. A bipartite graph can be described by a matrix with rows corresponding to the vertices of the Left part, and columns to vertices in the Right part. This matrix, called G 's “adjacency matrix” has a 1 if its row and column form an edge and 0 otherwise. (If edges have values these values can replace the 1's in the matrix?) The vertices adjacent to any vertex v are often called v 's neighbors.

We define the “**deficiency**” of a set S of rows, by its size less the size of the set of columns that is the union of the neighbors of its elements, when it is positive.

$$D(S) = |S| - |\cup_{v \in S} N(v)|.$$

The answer to our first question is then:

The size of a maximum matching is the size of L less the maximum deficiency among the subsets of S .

At first glance this seems like a useless condition because L has a number of subsets that is 2 raised to the power of L 's cardinality, which can be huge. Fortunately there is an easy way to find the maximum deficiency of a subset of L , which also provides us with a way to prove this condition.

First notice that if $|S|$ vertices have only $|S|-D$ neighbors we cannot possibly supply more than $|S|-D$ matches to elements of S . We therefore need only prove that we can actually match all but the maximum deficiency of the vertices in L .

To prove this, we suppose that we have a maximum sized matching, and it leaves k unmatched vertices in L . We will use a “labeling” procedure to prove our claim here. (This same labeling procedure can be used to find a maximum sized matching.)

The labeling procedure consists of starting with the unmatched vertices in L , and putting a label or marker on each of their neighbors. If any of these neighbors is not matched in

our matching, we can add the edge consisting of it and its unmatched L neighbor, to our matching. Doing so is called “Augmenting” the matching, and augmenting increases the size of the matching. Since we assume that our matching has maximum size, no augmentation can exist which means all the neighbors we find in R must be matched with vertices of L.

We then label the neighbors of the L vertices paired with of our first set of neighbors and it is still true that if any of these are unmatched we can find an augmentation. Thus if unmatched v is a neighbor of w which is matched to x which has an unmatched neighbor y , we can unpair w and x , and instead pair v with w , and x with y , which is an augmentation. Such augmentations are possible whenever we can find a path from an unpaired L vertex to an unpaired R vertex that alternates between edges out of and in our matching.

We can continue labeling neighbors of matches of neighbors of matches of unmatched vertices of L and continue in this fashion until all vertices of R that can be reached in this manner are reached. Notice that there will be an augmentation unless every one of the reachable vertices in R is matched to a vertex in L. (Notice how similar this idea is to the switching of two colors in the proof of the five color theorem and the false proof of the four color theorem)

And so we have found our set S of vertices of L with maximum deficiency. It consists of the unmatched vertices in L plus all those vertices in L who are matched to the labeled vertices in R.

And what is the union of their neighborhoods? It consists of the labeled vertices in R, The size of this union is the number of labeled vertices in this S ; (which is the cardinality of S) less the number of its unmatched vertices. The entire number of unmatched vertices of L is therefore the deficiency of S , which is what we set out to prove.

Not only does this argument permit us to find S , it allows us to find a maximum size matching, starting from nothing. We can do so by labeling until we find an augmentation, then doing it again, and repeating until there are no more augmentations possible.

The procedure just described is a bit vague because there are lots of ways to label neighbors in a graph. That is, there are lots of ways to choose the order in which you assign labels to neighbors. Among these are “depth first search” in which having labeled a vertex you would, if it is paired in the partial matching you are trying to augment, label a neighbor of its partner next. In a breadth first search you label all immediate neighbors of unmatched vertices first, then those who are three steps away from an unmatched vertex next, then five steps away next, etc.

Here breadth first search is most efficient. We can show that if we first find a maximal set of 1 step augmentations, then a maximal set of 3 step augmentations, and so on, we can increase the size of the minimum length augmentation by at least 2 using only one labeling procedure. The number of labelings (each of which could at worst require us to

look at all the edges) is then at most the number of different sizes of augmentations that we encounter. And this number is at most on the order of $V^{1/2}$, so that the number steps necessary to find a maximum sized matching this way is at most on the order of $EV^{1/2}$.

Why is the number of augmentation lengths at most on the order of $V^{1/2}$?

(If you look at the edges that are in a partial matching and not some maximum sized matching and vice versa, they have degree 0 1 or 2 at each vertex. They therefore consist of a set of cycles and paths in G ; the maximum matching will be bigger in cardinality than the partial matching by at most the number of those (disjoint) paths here that have more maximum matching edges than partial matching edges, and each such path is a potential augmenting path. There are only $N^{1/2}$ possible sizes of augmenting paths less than $N^{1/2}$. When all augmenting paths contain at least $N^{1/2}$ Left vertices, and they are all disjoint x such augmenting paths use at least $N^{1/2}x$ Left vertices. Since there are only N such vertices, x can be at most $N^{1/2}$, and there can be at most that many sizes of augmentations greater than $N^{1/2}$ for a total of at most $2N^{1/2}$ augmentation sizes in all.)

A Useful Fact about Graphs

Given a bipartite graph, the following simple fact is often useful:

FACT: The number of vertices in the Left part multiplied by their average degree must be the same as the same product for the Right part.

This statement follows immediately if we notice that both of these products count the number of edges of the bipartite graph.

You can use this fact, and the Matching Theorem stated above (often called the Philip Hall marriage theorem) to prove the following statement. Do so for homework.

3. If all vertices of G have the same degree (number of incident edges) and G is bipartite, then there is a complete matching in G . (This is a set of pairs such that each pair is an edge of G , and every vertex of G is paired/) Further, the edges of G can be partitioned into complete matchings.

4. A latin square is an N by N square array of integer each of which is in the range from 1 to N so that no integer appears twice in a row or column.

Prove: If you have a k by N array of integers from 1 to N obeying the same condition (with k less than N) you can complete it to a latin square.

Hint (to fill in one additional row with integers, the integers and places in the row form a bipartite graph defined by eligibility of an integer to appear in that place.)

The Stable Matching Problem

The matching results discussed above are sometimes used to assign jobs to machines or to people in a factory. A marriage broker might want to use such things to arrange marriages, but there is a complication: If L and R represent boys and girls who

you seek to pair off, your pairing will be unstable if a boy, v and a girl w prefer each other to those you wish to pair them with. Rather than marrying unwanted spouses they might elope together and deprive you of your commission..

The stable marriage problem is that of finding a maximal set of pairings that are stable, so that this never happens. To describe the problem we need not a compatibility graph, but a list of preferences for each vertex. We suppose first that each vertex v in L orders the vertices in L in some way, indicating with w above x on v 's list if v prefers w to x . And similarly each vertex in R orders the vertices in L according to its preferences.

The basic questions relevant in this situation are: Can one always find a complete set of pairings that are all stable? If so how?

The answer is yes, and the following algorithm, (which conforms in a way with actual human behavior) Each vertex in L (boy?) proposes to his favorite girl, in the first round, Each girl rejects all but her top ranked suitor, and each of those rejected x 's the rejectress from his list, and proposes to the next entry on it.

The procedure continues in this way until in some round there are no more rejections. At this point the not yet rejected boys and the girls that each has not been rejected by form a stable set of marriages.

Why is that?

We have been assuming that there are the same number of boys and girls. If so, and in some round nobody is rejected, then every boy is paired with some girl.

But are these pairings stable?

They have to be. If v and w are paired at this point. Every boy other than v has been either rejected by w or has not proposed to her, which means she is further down his list than is the girl he is to be paired with. So there is no unpaired twosome x and w who wish to elope.

A Slight Variant of the Stable Marriage Problem

The traditional form of this problem is a bit unrealistic in practice, since it assumes that every boy ranks every girl and vice versa. In practice many people find certain others totally unacceptable, and prefer to be single than to marry them. This people tend to rank only a subset of the vertices on the other side.

In that case we cannot expect that every vertex becomes paired. (A vertex with an empty list, for example, will not be paired.) But we can ask: does there always, with every possible set of preferences, exist a set of stable marriages, such that no unpaired v and w prefers each other to their partners, and no unpaired v and w both find the other acceptable at all?

The answer is, yes there is always a set of stable marriages, and the same arguments apply. The only difference is there here may be unpaired boys, who are rejected by everyone on their respective lists, and unpaired girls who receive no proposals.

An Application: (by Fred Galvin)

The algorithm described above, which is due to Gale and Shapley, has actual uses in matching new MD's to residencies, and for admission to certain kinds of graduate schools, and for assignment of students to dormitory rooms. We here consider an application to proving a statement about a certain kind of graph.

Suppose we have a graph on N^2 vertices, which vertices are described by two indices which are numbers each of which runs from 1 to N . Two vertices are adjacent when they have the same first index, or the same second index. If the indices represent coordinates in the plane, then two vertices in this square array are adjacent if they lie in the same row or in the same column. (This kind of graph is called the direct product of two complete graphs K_N . A direct product of two graphs is one in which each vertex corresponds to a pair of vertices, one from each factor, and two are adjacent if the vertices are identical in one factor and adjacent in the other.)

The question to be addressed is: what is the list coloring number of this graph?

The ordinary coloring number of a graph G is the smallest number of colors needed to color all the vertices of G such that no edge has both of its vertices of the same color.

Suppose that each vertex v of G has a separate list of k colors and we require that v be colored only by one of the colors on its list. Then the list coloring number of G is no more than k if for every possible set of lists of length k all the vertices of G can be colored so that no edge has both ends the same color.

Since it takes N colors to color K_N , the list coloring number of this direct product is also at least N , And here is a proof.

We begin by choosing an arbitrary N by N Latin Square L_{ij} and use it to define preference lists for each row among the columns, and vice versa. The rule is that each row i prefers column j over column k when L_{ij} is larger than L_{ik} and column j prefers row i to row k when L_{ij} is less than L_{kj} .

Next you may arrange the possible colors on all the lists in your favorite order, and starting with the first color take all the vertices that have that color on their lists and find a set of stable row-column pairings among them Color the (i,j) vertex in that color if row i is paired with column j in this stable pairing. If so remove the remaining colors from the lists for the vertices you have colored, and do the same thing for the next color, and the next, until you have done so for all colors that are on any list.

We now claim that every pair (i, j) has received some color.

Why?

This vertex had N colors on its list. If it does not get colored it must have been rejected in each color. But its row would only reject in a color if there were a preferred pairing in that color in that row (a column that the row preferred to the j -th column); which means an index k in row i with $L_{ik} > L_{ij}$, and it can only be rejected in column j if there is an entry in that column that causes its row to be preferred over the i -th row. This would be a row k for which L_{ki} is less than L_{ij} . Each entry in row i or column j can only cause rejection in one color, since once it is colored, it is removed from all subsequent lists. And there are only $N-1$ integers less than L_{ij} in row i or greater than it in column j , so the pair (i, j) cannot be rejected in all N colors on its list.

This proves that the direct product of two K_N 's is N list colorable.

Eulerian Paths.

One of the earliest questions ever raised about graphs was as follows:

For what graphs G can we start at a vertex v and find a path through G that goes through every edge of G exactly once?

Obviously G must be connected, or at least can have only one connected component with any edges in it. Among connected graphs, Euler noted that the only graphs that have such a path (called an Eulerian path) are those with at most two vertices of odd degree, and that any connected graph with this property possesses such a path. If all degrees are even the Eulerian path can be closed to form an Eulerian cycle. In most cases this path or cycle is not simple, the only exception are those graphs that are cycles or paths to begin with.

We can prove that any graph with only two odd degree vertices or less has an Eulerian path, by wandering through the graph starting from one of the odd degree vertices. If we enter and leave a vertex and remove the edges used to do so, we reduce the degree of the remaining edges at that vertex by 2. Since the degree of the vertex was even, we can always leave any vertex we enter, unless we stumble upon the other vertex of odd degree

So we can wander through G . Every time we complete a cycle we can remove it and that will reduce the degrees of every vertex on it by 2 leaving them still even. At some stage we will find a path from one odd degree vertex to the other.

We are done if we can show how to put all the cycles we encounter and the path between odd degree vertices, together to form one path. Suppose we have two cycles that intersect at some vertex v . Then we can traverse one cycle up to v then switch to the other, and go around it completely, returning to v and then going back to the first cycle. Using this kind

of merging we can put the cycles together, with the path if any, to form a path or cycle.
And that completes the proof/

Exercise: Show how to find a matching in a bipartite graph having N vertices in each part in which each vertex has degree d obeying $d=2^k$ by wandering over a total of $2Nd$ edges. (Hint find cycles and modify them)

—