Staff Solutions to In-Class Problems Week 9, Mon.

STAFF NOTE: Simple Graphs: Bipartite Matching Ch. 12.5

Coaches: Don’t let your team struggle for more than a few minutes trying to model the questions as matching problems; after that tell them what the boys and girls should be, and let them work out the rest.

Problem 1.
A certain Institute of Technology has a lot of student clubs; these are loosely overseen by the Student Association. Each eligible club would like to delegate one of its members to appeal to the Dean for funding, but the Dean will not allow a student to be the delegate of more than one club. Fortunately, the Association VP took Math for Computer Science and recognizes a matching problem when she sees one.

(a) Explain how to model the delegate selection problem as a bipartite matching problem. (This is a modeling problem; we aren’t looking for a description of an algorithm to solve the problem.)

COMMENTS:

- CP_student_clubs
- from: S09.cp6r
- from: S07.cp6w (slightly edited/shortened)

keywords = [ bipartite_matching degree-constrained Halls_Theorem ]

Solution. Define a bipartite graph with the student clubs as one set of vertices and everybody who belongs to some club as the other set of vertices. Let a club and a student be adjacent exactly when the student belongs to the club. Now a matching of clubs to students will give a proper selection of delegates: every club will have a delegate, and every delegate will represent exactly one club.

(b) The VP’s records show that no student is a member of more than 9 clubs. The VP also knows that to be eligible for support from the Dean’s office, a club must have at least 13 members. That’s enough for her to guarantee there is a proper delegate selection. Explain. (If only the VP had taken an Algorithms class, she could even have found a delegate selection without much effort.)

Solution. The degree of every club is at least 13, and the degree of every student is at most 9, so the graph is degree-constrained, which implies there will be no bottlenecks to prevent a matching. Hall’s Theorem then guarantees a matching.

Problem 2.
A Latin square is \( n \times n \) array whose entries are the number 1, \ldots, \( n \). These entries satisfy two constraints: every row contains all \( n \) integers in some order, and also every column contains all \( n \) integers in some order.
Latin squares come up frequently in the design of scientific experiments for reasons illustrated by a little story in a footnote

For example, here is a $4 \times 4$ Latin square:

\[
\begin{array}{cccc}
1 & 2 & 3 & 4 \\
3 & 4 & 2 & 1 \\
2 & 1 & 4 & 3 \\
4 & 3 & 1 & 2 \\
\end{array}
\]

(a) Here are three rows of what could be part of a $5 \times 5$ Latin square:

\[
\begin{array}{ccccc}
2 & 4 & 5 & 3 & 1 \\
4 & 1 & 3 & 2 & 5 \\
3 & 2 & 1 & 5 & 4 \\
\end{array}
\]

Fill in the last two rows to extend this “Latin rectangle” to a complete Latin square.

**COMMENTS:**

- CP\_latin\_squares
- from: S09.cp6r
- from: S04.cp5f (edited ARM in S08)
- This problem has a lot of commented out material that should be cleaned up.

**keywords** = [ bipartite\_matching degree-constrained graph\_coloring ]

**Solution.** Here is one possible solution:

\[
\begin{array}{ccccc}
2 & 4 & 5 & 3 & 1 \\
4 & 1 & 3 & 2 & 5 \\
3 & 2 & 1 & 5 & 4 \\
1 & 5 & 2 & 4 & 3 \\
5 & 3 & 4 & 1 & 2 \\
\end{array}
\]

\[1]At Guinness brewery in the early 1900's, W. S. Gosset (a chemist) and E. S. Beavan (a "malster") were trying to improve the barley used to make the brew. The brewery used different varieties of barley according to price and availability, and their agricultural consultants suggested a different fertilizer mix and best planting month for each variety.

Somewhat sceptical about paying high prices for customized fertilizer, Gosset and Beavan planned a season long test of the influence of fertilizer and planting month on barley yields. For as many months as there were varieties of barley, they would plant one sample of each variety using a different one of the fertilizers. So every month, they would have all the barley varieties planted and all the fertilizers used, which would give them a way to judge the overall quality of that planting month. But they also wanted to judge the fertilizers, so they wanted each fertilizer to be used on each variety during the course of the season. Now they had a little mathematical problem, which we can abstract as follows.

Suppose there are $n$ barley varieties and an equal number of recommended fertilizers. Form an $n \times n$ array with a column for each fertilizer and a row for each planting month. We want to fill in the entries of this array with the integers $1, \ldots, n$ numbering the barley varieties, so that every row contains all $n$ integers in some order (so every month each variety is planted and each fertilizer is used), and also every column contains all $n$ integers (so each fertilizer is used on all the varieties over the course of the growing season).
(b) Show that filling in the next row of an \( n \times n \) Latin rectangle is equivalent to finding a matching in some \( 2n \)-vertex bipartite graph.

Solution. Construct a bipartite graph as follows. One set of vertices are the columns of the Latin rectangle, and the other set is the numbers 1 to \( n \). Put an edge between a column and a number if the number has *not yet appeared* in the column. Thus, a matching in this graph would associate each column with a distinct number that has not yet appeared in that column. These numbers would form the next row of the Latin rectangle.

(c) Prove that a matching must exist in this bipartite graph and, consequently, a Latin rectangle can always be extended to a Latin square.

Solution. Suppose the Latin rectangle has \( k \) rows of width \( n \). Then each column-vertex has degree \( n - k \) because its edges go to the \( n - k \) numbers missing from the column. Also, each number-vertex also has degree \( n - k \). That’s because each number appears exactly once in each of the \( k \) rows and at most once in each column, so each number must be missing from exactly \( n - k \) columns.

So the graph is degree-constrained and therefore has a matching. This implies that we can add rows to the Latin rectangle by the procedure described above as long as \( k < n \). At that point, we have a Latin square.

Problem 3.
Take a regular deck of 52 cards. Each card has a suit and a value. The suit is one of four possibilities: heart, diamond, club, spade. The value is one of 13 possibilities, \( A, 2, 3, \ldots, 10, J, Q, K \). There is exactly one card for each of the \( 4 \times 13 \) possible combinations of suit and value.

Ask your friend to lay the cards out into a grid with 4 rows and 13 columns. They can fill the cards in any way they’d like. In this problem you will show that you can always pick out 13 cards, one from each column of the grid, so that you wind up with cards of all 13 possible values.

(a) Explain how to model this trick as a bipartite matching problem between the 13 column vertices and the 13 value vertices. Is the graph necessarily degree-constrained?

COMMENTS:

- from: S09.ps6
- from: F07.ps6 (revised)

keywords = [ bipartite_matching Halls_Theorem ]

Solution. Create a simple bipartite graph with 13 column vertices and 13 value vertices. Connect a column to a value by a single edge iff a card with that value is contained in that column. A perfect matching would then indicate the value of the card you would choose from each column.

The graph may not be degree-constrained if any one of the columns contains more than one card with the same value. In the case where the matching indicates a value that appears more than once in the column it is matched to, you can arbitrarily pick any card of that value in that column.

(b) Show that any \( n \) columns must contain at least \( n \) different values and prove that a matching must exist.
Solution. If \( S \) is a set of columns, they contain \( 4|S| \) cards. No card value repeats more than four times, so at least \( |S| \) values must appear among those cards. Thus \( |N(S)| \geq |S| \) and Hall’s theorem gives us a matching.

**Problem 4.**
A simple graph is called regular when every vertex has the same degree. Call a graph balanced when it is regular and is also a bipartite graph with the same number of left and right vertices.

Prove that if \( G \) is a balanced graph, then the edges of \( G \) can be partitioned into blocks such that each block is a perfect matching.

For example, if \( G \) is a balanced graph with \( 2k \) vertices each of degree \( j \), then the edges of \( G \) can be partitioned into \( j \) blocks, where each block consists of \( k \) edges, each of which is a perfect matching.

**STAFF NOTE:** *Hint:* Induction on degree.

I don’t see a way to do this by induction on number of vertices—ARM.

**COMMENTS:**
- CP\_degree\_constrained\_induction
- ARM 4/8/14
- S16.cp9m, F12.rec6

**keywords** = [ graph degree regular matching induction ]

**Solution. Proof.** The proof is by induction on the degree \( d \) of the vertices in a balanced graph. The induction hypothesis is

\[ P(d) := \text{If } G \text{ is balanced graph with vertices of degree } d, \text{ then } E(G) \text{ can be partitioned into } d \text{ perfect matchings.} \]

**Base case:** \((d = 0)\). If \( G \) is regular with degree 0 vertices, then it has no edges, so the empty partition with 0 blocks does the job. Note that all the blocks are vacuously perfect matchings, since there are no blocks.

**Induction step** Let \( G \) be a balanced graph with vertices of degree \( d + 1 \). We need to prove that \( E(G) \) can be partitioned into \( d + 1 \) perfect matchings.

Since all vertices have the same degree, \( G \) is degree-constrained and so has a perfect matching \( M \) by Theorem 12.5.6.

Let \( G - M \) be the graph obtained by removing the edges in \( M \) from \( G \). Now \( G - M \) is balanced with degree \( d \), since, by definition of perfect matching, each vertex is incident to exactly one edge in \( M \). By induction, we may assume that \( E(G - M) \) can be partitioned into \( d \) perfect matchings. Then \( M \), together with these \( d \) matchings, is a partition of \( E(G) \) into \( d + 1 \) perfect matchings.

This proves \( P(d + 1) \), and so by induction, \( P(d) \) holds for all \( d \geq 0 \).