Lecture 1: Latin Squares (Enumeration, Partial, Graphs)

Week 2 *Mathcamp 2012*

The aim of the following ten talks, roughly speaking, is to simutaneously give you a **deep** understanding of what Latin squares are and what their importance is in combinatorics, while simultaneously providing a **broad** overview of many different subfields of combinatorics and mathematics. In theory, over the next two weeks we will study affine geometries, error-correcting codes, graph theory, number theory, cryptography, statistics, chess, and Sudoku (amongst other things) all as applications of the concept of Latin squares; in doing so, we will simultaneously provide examples of currently open problems in combinatorics, prove classical and recent results, and hopefully build some intuition for how to use this beautiful class of objects in your own work.

The main goal for this lecture is to introduce the idea of Latin squares, and try to give you as many ways of looking at the objects themselves as possible; ideally this lecture will help you understand just how many strange ways there are of looking at a Latin square, and help motivate how we can study so many varied fields in combinatorics with Latin squares.

1 Latin Squares: Definitions

Definition. A **latin square** of order n is a $n \times n$ array filled with n distinct symbols (by convention $\{1, \dots n\}$), such that no symbol is repeated twice in any row or column.

Example. Here are all of the latin squares of order 2:

$$\begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix} \quad \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}.$$

A quick observation we should make is the following:

Proposition. Latin squares exist for all n.

Proof. Behold!

$$\begin{bmatrix} 1 & 2 & \dots & n-1 & n \\ 2 & 3 & \dots & n & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ n & 1 & \dots & n-2 & n-1 \end{bmatrix}$$

Given this observation, a natural question to ask might be "How many Latin squares exist of a given order n?" And indeed, this is an excellent question! So excellent, in fact, that it turns out that we have no idea what the answer to it is; indeed, we only know the

true number of Latin squares of any given order up to 11!

n	reduced Latin squares of size n[1]	all Latin squares of size n
1	1	1
2	1	2
3	1	12
4	4	576
5	56	161280
6	9408	812851200
7	16942080	61479419904000
8	535281401856	108776032459082956800
9	377597570964258816	5524751496156892842531225600
10	7580721483160132811489280	9982437658213039871725064756920320000
11	536393777327737129811967354077184 0	776966836171770144107444346734230682311065600000
12	?	?

Asymptotically, the best we know (and you could show, given a lot of linear algebra tools) that

$$L(n) \sim \left(\frac{n}{e^2}\right)^{n^2}.$$

Instead of this question, we're going to spend this lecture starting to study the concept of **partial Latin squares**, which we discuss below:

2 Partial Latin Squares

Definition. A **partial latin square** of order n is a $n \times n$ array where each cell is filled with either blanks or symbols $\{1, \dots n\}$, such that no symbol is repeated twice in any row or column.

Example. Here are a pair of partial 4×4 latin squares:

$$\begin{bmatrix} & & & 4 \\ 2 & & & \\ 3 & 4 & & \\ 4 & 1 & 2 & \end{bmatrix} \qquad \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 2 \end{bmatrix}$$

The most obvious question we can ask about partial latin squares is the following: when can we complete them into filled-in latin squares? There are clearly cases where this is possible: the first array above, for example, can be completed as illustrated below.

$$\begin{bmatrix} & & & 4 \\ 2 & & & \\ 3 & 4 & & \\ 4 & 1 & 2 & \end{bmatrix} \mapsto \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 1 \\ 3 & 4 & 1 & 2 \\ 4 & 1 & 2 & 3 \end{bmatrix}$$

[1] (A **reduced** Latin square of size n is a Latin square where the first column and row are both $(1, 2, 3 \dots n)$.)

However, there are also clearly partial Latin squares that cannot be completed. For example, if we look at the second array

$$\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 2 \end{bmatrix},$$

we can pretty quickly see that there is no way to complete this array to a Latin square: any 4×4 Latin square will have to have a 1 in its last column somewhere, yet it cannot be in any of the three available slots in that last column, because there's already a 1 in those three rows.

Deciding whether a given partial Latin square is completeable to a Latin square is, practically speaking, a useful thing to be able to do. Consider the following simplistic model of a **router**:

- Setup: suppose you have a box with n fiber-optic cables entering it and n fiber-optic cables leaving it. On any of these cables, you have at most n distinct possible wavelengths of light that can be transmitted through that cable simultaneously. As well, you have some sort of magical/electronical device that is capable of "routing" signals from incoming cables to outgoing cables: i.e. it's a list of rules of the form (r, c, s), each of which send mean "send all signals of wavelength s from incoming cable r to outgoing cable s." These rules cannot conflict: i.e. if we're sending wavelength s from incoming cable r to outgoing cable s, we cannot also send s from r to t, for some other outgoing cable t. (Similarly, we cannot have two transmits of the form $\{(r, c, s), (r, t, s)\}$ or $\{(r, c, s), (t, c, s)\}$.)

- Now, suppose that your box currently has some predefined set of rules it would like to keep preserving: i.e. it already has some set of rules $\{(r_1, c_1, s_1), \ldots\}$. We can model this as a **partial Latin square**, by simply interpreting each rule (r, c, s) as "fill entry (r, c) of our partial Latin square with symbol s."

- With this analogy made, adding more symbols to our partial Latin square is equivalent to increasing the amount of traffic being handled by our router.

This example hopefully motivates why we care about completing partial Latin squares, which lets us turn to the "what" part of our mathematical question: what kinds of partial Latin squares have completions?

Let's start simple.

Question. Suppose that we have a $n \times n$ partial Latin square L in which we've already filled in the first $n-1$ rows. Can we always complete this partial Latin square to a complete Latin square: i.e. can we always fill in the last row?

Proof. The answer to this question is yes (as trying a few examples may lead you to believe.) To prove this, consider the following simple algorithm for filling in our partial Latin square:

- Look at row n.
- For each cell (n, i) in row n, look at the column i.
- There are $n-1$ distinct symbols in column i, and therefore precisely one symbol s that is **not** present in column i. Write symbol s in the cell (n, i).

We claim that this algorithm creates a Latin square. To check this, it suffices to check whether any symbols are repeated in any row or column. By construction, we know that our choice of symbol s does not cause any repetition of symbols in any column; as well, we know that no symbol is repeated in any row other than possibly the n-th row, because we started with a partial Latin square. Therefore, it suffices to check the n-th row for any repeated symbols.

To do this, proceed by contradiction: i.e. suppose not, that there are two cells in the bottom row such that we've placed the same symbols in those two cells. This means that there is some symbol s that we've **never** written in our last row. But this means that this symbol s is used somewhere in all n columns within the first $n-1$ rows, which forces some row in those first $n-1$ to contain two copies of s, a contradiction.

Therefore, our algorithm works!

(As an aside, the above tactic of "make an algorithm" is ridiculously useful, and is something we should remember and make frequent use of.)

Given our success above, it's tempting to ask whether we can extend this result to partial Latin squares where we've (say) filled in just the first $n-2$ rows, or even in general a partial Latin square where we've filled in the first k rows, for any value of k. As it turns out, this is also possible! Consider the following definition and theorem:

Definition. A $n \times n$ **Latin rectangle** is a $n \times n$ partial Latin square in which the first k rows are completely filled and the remaining $n-k$ rows are completely empty, for some value of k.

Theorem 1 *Every Latin rectangle can be completed to a partial Latin square.*

In order to prove this theorem, we're going to need to use Hall's marriage theorem, a remarkably powerful result from **graph theory!** We present a number of definitions here, then state and prove Hall's theorem, and then use it for our result on Latin rectangles:

2.1 A Brief Detour into Graph Theory

Definition. A **simple** graph G with n vertices and m edges consists of the following two objects:

1. a set $V = \{v_1, \ldots v_n\}$, the members of which we call G's **vertices**, and
2. a set $E = \{e_1, \ldots e_m\}$, the members of which we call G's **edges**, where each edge e_i is an unordered pair of distinct elements in V, and no unordered pair is repeated. For a given edge $e = \{v, w\}$, we will often refer to the two vertices v, w contained by e as its endpoints. If $\{v, w\}$ is an edge, we will also often say that v and w are **adjacent**.

A **path** in a graph $G = (V, E)$ is a alternating sequence of vertices and edges $v_0, \{v_0, v_1\}, v_1, \{v_1, v_2\}, v_2, \ldots v_n$ such that all of these edges are elements of E.

We call a graph **bipartite** if its vertices V can be split up into two sets V_1, V_2 such that none of V's edges have both endpoints in V_1 or V_2.

Given a graph $G = (V, E)$ and a vertex $v \in V$, let $n(v)$ denote the **neighbors** of v, i.e. the collection of vertices connected to v via edges in E. The **degree** of a vertex is the size of $n(v)$.

A **subgraph** H of a graph $G = (V, E)$ is a graph $H = (V', E')$ such that $V \subset V'$ and $E \subset E'$: i.e. H is a graph we can make by deleting some edges and vertices from G.

A **matching** of a graph G is a collection of edges M, such that every vertex of G is contained in at most one edge in M. Finally, a **perfect matching** of a graph $G = (V, E)$ is a collection of edges P such that every vertex $v \in V$ is contained in precisely one edge in P.

Theorem 2 *Take a bipartite graph $G = (V, E)$ with bipartition $V = A \cup B$. Then, the following conditions are equivalent:*

- *G has a perfect matching: i.e. there's a way to pair off all of the elements of A with the elements of B.*
- *(Hall's condition): For any subset $X \subset A$ or $X \subset B$, if $N(X)$ denotes the neighbors of X, then $|X| \leq |N(X)|$. In other words, any collection of vertices from one side of our bipartition cannot have less neighbors on the other side than it has elements.*

Proof. Because this is an equivalence proof, we need to prove two statements: that the second statement implies the first, and that the first implies the second. We start with the easier direction, that the existence of a perfect matching implies Hall's condition:

($\Rightarrow$): Suppose that G has a perfect matching. Because G is bipartite, such a perfect matching is just a pairing-up of vertices in A to vertices in B along edges in G. Thus, for any subset $X \subset A$, because $N(X)$ must contain the edges in this perfect matching, we have that $|X| \leq N(X)$ (and similarly for $X \subset B$.)

Now, we move to the harder condition, showing that Hall's condition implies that there is a perfect matching.

($\Leftarrow$): Take any matching M, not necessarily perfect, for G. If M is a perfect matching, then we're done! Otherwise, consider the following algorithm for creating a path between vertices in A and B:

1. Pick some $a_0 \in A$ that's not involved in M. We can do this because M is not a perfect matching, and therefore there is some vertex in our graph that's not hit by one of M's edges (and it might as well be an element from A, because our labelings of A and B are arbitrary.)

2. We seek to create a path starting from a_0 that's as long as possible. We do so by creating an alternating sequence of vertices from A and B, along with a collection of specially marked edges that we will use to find this path.

3. Suppose that the sequence $a_0 b_1 a_1 b_2 a_2 \dots b_{k-1} a_{k-1}$ has been created. Then, because the set $\{a_0 \dots a_{k-1}\}$ of chosen A-vertices is strictly larger than the set $\{b_1 \dots b_{k-1}\}$of chosen B-vertices, there must be some element $b_k \in B$ that's connected by some edge $\{b_k, a_i\}$ to some previously-chosen a_i, by Hall's condition. Define e_k to be this edge $\{b_k, a_{f(k)}\}$.

4. If b_k is in M, let a_k be the vertex across from b_k in M, and return to (2) to continue to grow our sequence. Otherwise, end our sequence!

5. By construction, we know that b_k is unique amongst the previously chosen b_i's; similarly, because we picked the a_i up to this point by using the matching M, we know that they're all distinct. So this is still a sequence of distinct vertices!

6. Using this sequence, we can construct the following path, made by alternately following the edges of M and the edges recorded by the function f:

   (a) Start at b_k.

   (b) If you are at some element b_j in B, travel along the specially marked edge e_j to an element in A, and go to step (c).

   (c) Otherwise, if you are at an element a_i in A that's not a_0, by construction there is some element b_i that you're directly across from using an edge in M. Travel along that edge to b_j, and then go to step (b).

   (d) Otherwise, you're at a_0. Stop.

   How can we use this path to make M a perfect matching? Well, take M, and (1) delete all of the M edges we used above, while (2) adding to M all of the e_i-edges. This collection has precisely one more edge than the old collection, and only deals with the vertices a_0, b_k (which weren't in M anyways) and $a_1 \dots a_{k-1}, b_1 \dots b_{k-1}$ (which were only involved in edges we removed from M) — so it preserves M's status as a matching! Therefore, this matching is strictly larger than the one it replaced. If it is now a perfect matching, stop! Otherwise, repeat this process to keep adding edges until it becomes a perfect matching.

Corollary 3 *Suppose that $G = (V, E)$ is a bipartite graph with bipartition $V = A \cup B$, such that every vertex of V has degree k. Then G can be decomposed into k different perfect matchings, none of which have any edges in common.*

Proof. Pick any subset $X \subset A$ or $X \subset B$ of size n. Because every vertex in G has degree k, there are kn distinct edges leaving X and entering $N(X)$. Consequently, as each vertex has degree k, there must be at least n vertices in $N(X)$ to absorb these edges! – so $|N(X)| \geq |X|$.

Thus, by Hall's Marriage theorem, there is a perfect matching in G. Deleting it from G leaves a graph in which every vertex has degree $k - 1$; therefore, we can simply repeat this process until we get the desired k disjoint perfect matchings.

Excellent. So... how does this help us with Latin rectangles?

2.2 Completing Latin Rectangles

As it turns out, Hall's marriage theorem is useful to us because another way to view a Latin square (or indeed a partial Latin square) is as a graph! Specifically, given a $n \times n$ Latin rectangle L with its first k rows filled, we can create a bipartite graph $G = (V, E)$ with $V = A \cup B, E$ defined as follows:

- $A = \{1 \dots n\}$.
- $B = \{B_1 \dots B_n\}$.
- Associate to each B_j the collection of elements that don't occur in column j. Draw an edge from $i \in A$ to $B_j \in B$ if and only if i is one of those symbols that does not occur in B_j.

By construction, the degree of any B_j is just the number of elements that don't occur in a given column: i.e. $n-k$. As well, the degree of any $i \in A$ is just the number of columns that i doesn't show up, which is **also** $n - k$; so this is a bipartite graph in which every vertex has degree k.

We claim that this graph satisfies Hall's marriage theorem. To see why, pick any subset $X \subset A$ or $X \subset B$ of size n. Because every vertex in G has degree $n - k$, there are $n(n - k)$ distinct edges leaving X and entering $N(X)$. Consequently, as each vertex has degree $n-k$, there must be at least n vertices in $N(X)$ to absorb these edges! Therefore, $|N(X)| \geq |X|$.

If we apply Hall's marriage theorem, then, we have a perfect matching: i.e. a bijection between elements in $\{1, \dots n\}$ and columns where they do not occur. Use this perfect matching to fill in row $k + 1$, by placing in each column the element given by our bijection that does not occur in that column. This gives us another Latin rectangle: by repeating this process, we get a Latin square.