

Linear Methods in Discrete Mathematics

1 Eventown and Oddtown

We begin with a metaphor for a large class of problems, which was devised by László Babai. A town has 32 inhabitants; their principal activity seems to be forming clubs. The town council passes three rules for club formation.

(1E) Each club must have an *even* number of members.

(2E) Each pair of clubs must share an *even* number of members.

(3E) No two clubs are allowed to have identical membership

(It is legal to have the “club with no members”.) These are called the “Eventown Rules”.

There are two fundamental questions. What is the maximum number of clubs which can be formed under Eventown Rules? Is there a simple strategy (i.e., algorithm) for producing a maximum club formation?

Suppose this is an “industrial problem”. You are not looking for theorems. How do you attack the problem? Make some “back-of-the-envelope” estimates and see if you can later improve them.

Can we devise a scheme to guarantee (2E) cheaply? Pair off the citizens of the town into 16 couples and insist whenever a person joins a club, that the person’s partner join the club. This scheme also guarantees (1E). How many different clubs are there with the couple requirement? 2^{16} . Indeed, there is a one-to-one correspondence between 16-bit words and couple clubs.

So we can form 65,536 clubs. Is it possible to form more? There are two ways of thinking about this question. Can we create additional clubs so that the couple clubs together with the extra ones still satisfy Eventown Rules? Or does the scheme produce a *maximal* configuration of clubs? Even if the answer to the last question is “yes”, it may be that there is a completely different scheme which produces more clubs. There is a distinction, which will appear quite often, between *maximal* and *maximum*.

The distinction between maximal and maximum is fairly subtle. Consider the sequence

1, 2, 3, 4, 5, 6, 7.

Let's say a subsequence is "skippy" provided the difference between successive numbers on the list is always two. For example,

2, 4, 6

is a skippy subsequence. It is a maximal skippy subsequence because no one can insert any additional numbers between 1 and 7 to make it longer. On the other hand,

1, 3, 5, 7

is also maximal skippy. The second list is a maximum skippy subsequence but the first is not.

Is the couple scheme maximal? If there is an extra club, this club splits up at least one couple. Let's call these guys Bill and Hillary. There is an old club whose only members are Bill and Hillary. Compare the old club and the extra club in light of (2E).

The number 65,536 looks pretty good. Wouldn't it be nice if there were situations in which maximal always implied maximum and this was one of them? We will spend a lot of time exploring this possibility.

Since our "solution" to the club formation problem with Eventown Rules may be "good enough for many practical purposes", we are going to set it aside for the time being. The town council found it too unmanageable to register clubs under the original rules. They proposed a new law replacing the word EVEN with the word ODD in the first rule. The result was "Oddtown Rules".

(1O) Each club must have an *odd* number of members.

(2O) Each pair of different clubs must share an *even* number of members.

The council was particularly pleased to avoid a third rule. (How?)

Did the town legislators drastically reduce the maximum number of clubs? Again, we need a scheme for producing clubs subject to Oddtown Rules. Here is an unsatisfactory one: there are 32 one-member clubs. Can we find a more rational scheme? Can we improve on 32 – after all, it is grossly smaller than 2^{16} . If the answer really is 32, can we effectively generate all 32-club configurations?

We will focus for some time on the Oddtown problem. The first issue is to turn it into a suitable mathematical problem. This is a modeling issue. What is the appropriate notational environment for this problem? We take our cue from the Eventown count. We can represent each club by a 32-bit

word. (List the town's inhabitants $1, 2, \dots, 32$. A club word has a 1 in the j^{th} bit if and only if person j belongs to the club.)

We need to interpret the rules using this representation of clubs. Rule (1O) says that the total number of 1's in a word is odd. This works but it is cumbersome; why count the total number of 1's when you don't require this possibly large number? Let's work with Rule (2O). Is there an arithmetic way to test it?

You may be reminded of the AND bit operation. Better yet, you might see the dot product. We can think of Rule (2O) as a statement about the dot product with mod 2 arithmetic:

(2V) The mod 2 dot product of any two different words is 0.

Here **V** stand for vector. From now on, we will also think of clubs as vectors. What about the first rule?

(1V) The mod 2 dot product of any club vector with itself is 1.

We now have a sheerly mathematical problem in linear algebra with no reference to clubs. Find the maximum number of 32-bit vectors subject to (1V) and (2V). Have we actually made any progress? Yes, on two counts. First, it is now easier to run computer experiments, to collect data. Second, we might be able to apply intuition we have about dot product for real vector spaces.

What does it mean for two nonzero vectors in \mathbf{R}^n to have dot product zero? dot product 1? You might remember that two vectors have dot product zero precisely when they are perpendicular. What is the maximum number of perpendicular vectors one can have in \mathbf{R}^2 ? in \mathbf{R}^3 ? Any guesses about \mathbf{R}^n ? How about 32-tuples over \mathbf{F}_2 , the integers modulo 2? If all of our analogies hold, then 32 should be the maximum number of clubs under Oddtown Rules.

We have a program. The first step is to understand thoroughly what we think we know about vector geometry over the real numbers. The last step is to try to adapt what works for real numbers to arithmetic mod 2. Fundamental to the first step is the notion of "linear independence". Some of you may have studied this concept before and some may not have. Whatever the case, our perspective is not going to be the traditional one from linear algebra. We will take the viewpoint of discrete mathematicians whenever possible.

We will be very concrete. For the time being, there are three possible fields K of scalars: the real numbers \mathbf{R} , the complex numbers \mathbf{C} , and \mathbf{F}_2 .

It is important that scalars satisfy the “usual rules for algebra” and share the property that every nonzero scalar has a multiplicative inverse. A *vector space of m -tuples* in K^m is a nonempty subset closed under addition and scalar multiplication. (Addition and scalar multiplication in K^m are “coordinate-wise”.) We usually say that such a subset is a *subspace* of K^m . For example, the xy -plane is a subspace of \mathbf{R}^3 . We want to examine the statement:

The xy -plane is generated by the vectors $(1, 0, 0)$ and $(0, 1, 0)$.

Every vector in the plane has the form $(a, b, 0)$ and

$$(a, b, 0) = a(1, 0, 0) + b(0, 1, 0) .$$

In general, if v_1, \dots, v_n are vectors in some vector space and $\lambda_1, \dots, \lambda_n$ are scalars then an expression

$$\lambda_1 v_1 + \dots + \lambda_n v_n$$

is called a *linear combination* of v_1, \dots, v_n . The collection of all linear combinations of v_1, \dots, v_n is the *span* of v_1, \dots, v_n . Here is an easy but fundamental exercise — the span of a list of vectors is always a subspace.

Try these examples/exercises.

1. \mathbf{R}^3 is spanned by $(1, 0, 0)$, $(1, 1, 0)$, and $(1, 1, 1)$.
2. $\{v \in \mathbf{F}_2^4 \mid v \cdot (1, 1, 1, 1) = 0\}$ is a subspace. Can you find a small list of vectors which span it?
3. $(1, 2, 0)$, $(1, 4, 0)$, and $(1, 3, 0)$ span the xy -plane in \mathbf{R}^3 .

Notice that the third example is not very efficient. We don't need $(1, 3, 0)$. There are two ways of thinking about this. First, $(1, 3, 0)$ is already a linear combination of $(1, 2, 0)$ and $(1, 4, 0)$.

Theorem 1.1 *If v_1, \dots, v_n span a subspace W and v_n is a linear combination of v_1, \dots, v_{n-1} then v_1, \dots, v_{n-1} span W .*

Proof: The argument is a simple matter of substitution in a clutter of subscripts. To say that v_1, \dots, v_n spans W just means that if we pick an arbitrary w in W then it is possible to write the vector as

$$w = \lambda_1 v_1 + \dots + \lambda_n v_n$$

for some choice of scalars λ_i . But v_n is a linear combination of v_1, \dots, v_{n-1} , so

$$v_n = \alpha_1 v_1 + \dots + \alpha_{n-1} v_{n-1}$$

with $\alpha_1, \dots, \alpha_{n-1}$ scalars. Substituting,

$$w = (\lambda_1 + \lambda_n \alpha_1) v_1 + (\lambda_2 + \lambda_n \alpha_2) v_2 + \dots + (\lambda_{n-1} + \lambda_n \alpha_{n-1}) v_{n-1} .$$

We have written an arbitrary vector in W as a linear combination of the first $n - 1$ vectors on the list: v_1, \dots, v_{n-1} . ■

The second way of regarding the situation is that we have more than one way to express some members of W as a linear combination of vectors on the list,

$$(1, 3, 0) = 0 \cdot (1, 2, 0) + 0 \cdot (1, 4, 0) + 1 \cdot (1, 3, 0)$$

$$(1, 3, 0) = \frac{1}{2} \cdot (1, 2, 0) + \frac{1}{2} \cdot (1, 4, 0) + 0 \cdot (1, 3, 0)$$

Equivalently, we can subtract and write the zero vector in a peculiar way:

$$\mathbf{0} = \left(-\frac{1}{2}\right) \cdot (1, 2, 0) + \left(-\frac{1}{2}\right) \cdot (1, 4, 0) + 1 \cdot (1, 3, 0) .$$

2 Linear Independence

We are ready for the official definition.

Definition 2.1 *A list of vectors w_1, \dots, w_n in a vector space W is **linearly independent** provided that*

$$\lambda_1 w_1 + \dots + \lambda_n w_n = 0$$

for scalars λ_j forces

$$\lambda_1 = \dots = \lambda_n = 0 .$$

In other words, there are no peculiar ways to write the zero vector as a linear combination of linearly independent vectors. (Conversely, if w_1, \dots, w_n are vectors and it so happens that

$$\lambda_1 w_1 + \dots + \lambda_n w_n = 0$$

with at least one scalar λ_j nonzero then we will often refer to the linear combination in this equality as a **linear dependence** among w_1, \dots, w_n . In this case, we say w_1, \dots, w_n are **linearly dependent**.) We stress, in the next observation, that both of our ways to think about linear independence are the same.

Theorem 2.1 *The vectors v_1, \dots, v_n are linearly dependent if and only if some v_j is a linear combination of the remaining vectors on the list.*

Proof: Suppose that the list v_1, \dots, v_n is linearly dependent. Then there exists some choice of scalars $\alpha_1, \dots, \alpha_n$ which are not all zero but, nonetheless,

$$\alpha_1 v_1 + \dots + \alpha_n v_n = 0 .$$

Say α_j is one of the coefficients which is not zero. Then we can solve for v_j , obtaining:

$$v_j = (\alpha_j)^{-1}(-\alpha_1)v_1 + \dots + (\alpha_j)^{-1}(-\alpha_{j-1})v_{j-1} + (\alpha_j)^{-1}(-\alpha_{j+1})v_{j+1} + \dots + (\alpha_j)^{-1}(-\alpha_n)v_n .$$

Conversely, if $v_j = \beta_1 v_1 + \dots + \beta_n v_n$ where a term for v_j is not in the sum then we can put all of the vectors on one side of the equation,

$$0 = \beta_1 v_1 + \dots + \beta_{j-1} v_{j-1} + (-1) \cdot v_j + \beta_{j+1} v_{j+1} + \dots + \beta_n v_n$$

and -1 is certainly not zero. ■

The second half of the argument is a special instance of the observation that if a vector can be written in two different ways as a linear combination of v_1, \dots, v_n then this list is linearly dependent. This seemingly abstract formulation can be turned into a practical estimate which is useful in cryptography. As we shall see later, a typical problem reduces to a search for manageable linear dependence relations among a list of vectors with integer coordinates. The following result guaranteeing the existence of “generic” small relations, essentially says that if the vectors are of moderate magnitude and linearly dependent then there has to be a dependence relation with integer coefficients of moderate size.

Lemma 2.1 *Assume that v_1, v_2, \dots, v_n are vectors in \mathbf{R}^t with integer coordinates and that $t < n$. Let M be an upper bound for the absolute values of all coordinates of all v_i 's. Then there exists a linear dependence*

$$\alpha_1 v_1 + \alpha_2 v_2 + \dots + \alpha_n v_n = 0$$

such that all α_i are integers, at least one is nonzero, and all $|\alpha_j|$ are bounded in size by B where

$$\log B = t \frac{\log M + \log n + 1}{n - t} .$$

(Here the logarithm uses the base 2 and you are seeing an estimate on bit size.)

Proof: Consider all possible integer linear combinations

$$\beta_1 v_1 + \beta_2 v_2 + \cdots + \beta_n v_n$$

with the restriction that $0 \leq \beta_i < B$. How many expressions are there like this? B^n . On the other hand, if we think of each such combination as a vector in \mathbf{R}^t then each coordinate is smaller than nBM in absolute value. Since there are less than $(2nBM)^t$ choices, there must be a coincidence of expressions in the case that

$$(2nBM)^t = B^n.$$

Of course, we are in this situation, as can be seen by taking logarithms.

Finally, if two of these linear combinations are equal then their difference has each coefficient of v_j in the open interval from $-B$ to B . ■

The justification for developing the notion of linear independence is supposed to be that it will allow us to handle mutually perpendicular vectors in an algebraic manner.

Theorem 2.2 *If v_1, \dots, v_n are vectors such that*

- (1) $v_j \cdot v_j = 1$ for all j and
- (2) $v_i \cdot v_j = 0$ whenever $i \neq j$

then v_1, \dots, v_n are linearly independent.

Proof: How can

$$\alpha_1 v_1 + \cdots + \alpha_n v_n = 0$$

for scalars α_j ? The dot product of any vector with the zero vector is 0, so for each choice of j

$$0 = (v_j) \cdot (\alpha_1 v_1 + \cdots + \alpha_n v_n) .$$

It is easy to check that the dot product is distributive: $u \cdot (v + w) = (u \cdot v) + (u \cdot w)$. Also, one can pull out scalars: $u \cdot (\lambda w) = \lambda(u \cdot w)$. Hence

$$0 = \alpha_1(v_j \cdot v_1) + \cdots + \alpha_n(v_j \cdot v_n) .$$

But all of the dot products are 0 except for one. In other words,

$$0 = \alpha_j(v_j \cdot v_j) .$$

We conclude that $\alpha_j = 0$ just as we wanted. ■

The Oddtown problem has been reduced to showing that the vector space \mathbf{F}_2^{32} cannot have more than 32 linearly independent vectors inside. Forget about club vectors for a moment. We are going to find one maximal list of linearly independent vectors in \mathbf{F}_2^m . Consider the “standard basis”

$$(1, 0, 0, \dots, 0), (0, 1, 0, \dots, 0), \dots, (0, 0, 0, \dots, 1) .$$

These vectors have the dot product property described in the theorem, so they are linearly independent. When we try to add any another vector to the list, that vector will be a linear combination of these special ones. By theorem 2.1, the enlarged list cannot be linearly independent. To summarize, \mathbf{F}_2^m has a maximal linearly independent set with m vectors.

Imagine that we can prove that every maximal set of independent vectors in \mathbf{F}_2^m is a *maximum* set of linearly independent vectors. Take any list of club vectors in \mathbf{F}_2^{32} . Since the previous theorem says it is a linearly independent set, we can enlarge it to a maximal linearly independent set. We have also constructed an explicit standard maximal linearly independent set of 32 vectors. If “maximal” is “maximum” for linear independence then the enlarged list also has 32 vectors in it. Therefore there are no more than 32 club vectors in the original list. This solves the Oddtown problem. (Of course, we still need good algorithms to produce 32-club configurations.)

Our approach to the “maximal/maximum” conjecture will be to turn algebra into combinatorics. The strategy is one of the versions of what is sometimes called “Steinitz exchange”.

Theorem 2.3 (Replacement Principle For Vectors) *Assume that v_1, \dots, v_n are vectors in the space W and that w is a linear combination of v_1, \dots, v_n :*

$$w = \beta_1 v_1 + \dots + \beta_n v_n .$$

If $\beta_t \neq 0$ then we can replace v_t with w in the list and

- (i) if the old list is linearly independent then the new list will still be linearly independent.*
- (ii) if the old list spans W then the new list spans W .*

Proof: To simplify notation, we'll assume $\beta_1 \neq 0$. For the first assertion, we need to check that w, v_2, v_3, \dots, v_n are linearly independent. Suppose

$$\alpha w + \lambda_2 v_2 + \dots + \lambda_n v_n = 0 .$$

Substitute!

$$\alpha\beta_1 v_1 + (\alpha\beta_2 + \lambda_2)v_2 + \dots + (\alpha\beta_n + \lambda_n)v_n = 0 .$$

Since v_1, \dots, v_n are linearly independent, $\alpha\beta_1 = 0$. Hence $\alpha = 0$. So we really have $\lambda_2 v_2 + \dots + \lambda_n v_n = 0$. Use the linear independence of v_1, \dots, v_n again to conclude that $\lambda_2 = \dots = \lambda_n = 0$.

As to the second part, use the formula for w to solve for v_1 as a linear combination of w, v_2, v_3, \dots, v_n . Then any vector which is a linear combination of $v_1, v_2, v_3, \dots, v_n$ is also a linear combination of w, v_2, v_3, \dots, v_n by virtue of a direct substitution. ■

Corollary 2.1 *Every finite maximal linearly independent set in a vector space is a maximum linearly independent set.*

Proof: Assume that v_1, \dots, v_m and w_1, \dots, w_n are two maximal linearly independent lists with $m \leq n$. We are going to make a sequence of replacements so that all of the v_j get moved to the second list.

Reorder, if necessary, so that any vectors in common to the two lists are at the beginning of each. (There may be none in common.) The lists are

$$v_1, \dots, v_m \quad \text{and} \quad v_1, \dots, v_t, w_{t+1}, \dots, w_n .$$

Consider the temporary list $v_1, \dots, v_t, w_{t+1}, \dots, w_n, v_{t+1}$. It is not linearly independent by the maximality of the second list. Thus v_{t+1} is a linear combination of members of the second list. Not all of the vectors which appear with nonzero coefficients in this combination can be v_i 's. (Why?) Replace one of the w_i having a nonzero coefficient with v_{t+1} and renumber. We obtain the "improved" second list

$$v_1, \dots, v_{t+1}, w_{t+2}, \dots, w_n$$

which remains maximal linearly independent. (Make sure you see how both parts of the replacement principle are being applied.) Iterate the process. (To be more sophisticated, argue by induction on the number of common vectors.) Eventually, we obtain the super improved linearly independent list

$$v_1, \dots, v_m, w_{m+1}, \dots, w_n .$$

This violates maximality of the first list unless $m = n$. ■

Finally, a warning. We solved the Oddtown club problem by examining the issue of maximality. It is very easy to fall into the trap of presuming that we now know that maximal implies maximum for club formation. This is not what we proved. Our argument was that any maximal configuration of clubs corresponds to a linearly independent list of vectors in \mathbf{F}^{32} and there could be at most 32 vectors in such a list. We also were able to exhibit a configuration with 32 clubs. But it is *not* true that every maximal collection of Oddtown clubs is maximum. Consider the following two clubs. One consists of the mayor alone and one has the remaining 31 ordinary citizens. These two clubs satisfy Oddtown Rules. In fact, they are a maximal configuration! Suppose we try to find another club to add. The mayor cannot be a member of the new club for, otherwise, the new club has an intersection of 1 with the mayor's club. So all of the members of the new club are ordinary citizens. But then the cardinality of its intersection with the club of all ordinary citizens is both the size of the new club – odd – and the size of any intersection – even.

3 Matroids

Let's re-examine the corollary, the punchline to the Oddtown problem. If we remove the maximality requirement from both lists and don't discard any w_j when the temporary list remains linearly independent, we can still move all of the v_j 's to the second list. At the end, we will have proved that

If v_1, \dots, v_m and w_1, \dots, w_n are linearly independent lists with $m < n$ then there exists some w_t with v_1, \dots, v_m, w_t linearly independent.

This statement is our motivation for the versatile notion of “matroid”.

Definition 3.1 *A matroid consists of a finite set S and a collection \mathcal{I} of subsets of S called **independent sets** such that*

Subset Rule: If $X \in \mathcal{I}$ and $Y \subseteq X$ then $Y \in \mathcal{I}$. In particular, $\emptyset \in \mathcal{I}$.

Expansion Rule: If A and B are members of \mathcal{I} with $\#(B) = \#(A) + 1$ then there exists some $z \in B \setminus A$ with $A \cup \{z\} \in \mathcal{I}$.

The point is that if V is a vector space then \emptyset together with the collection of all finite linearly independent subsets of some finite source S of vectors

constitutes a matroid. We are going to discuss some properties of matroids in general and then apply them to two additional examples.

Theorem 3.1 *Any two maximal independent sets in a matroid have the same cardinality.*

This theorem says that a maximal independent set in a matroid is a maximum independent set. We already have this result for vector spaces. If $T = \{v_1, \dots, v_m\}$ is a maximal linearly independent set in K^n , let the source set S be the union of T and the standard basis of n vectors. Then $m = n$; every maximal linearly independent set has n vectors. A maximal linearly independent set in any vector space is referred to as a *basis* of the space and its cardinality is the *dimension* of the space.

Proof of the theorem: Suppose that A and C are maximal independent sets in a matroid. We compare their cardinalities. If $\#(C) > \#(A)$ we can use the Subset Rule to find an independent subset B of C with $\#(A) + 1$ members. The Expansion Rule tells us that there is a $z \in B$ with $z \notin A$ so that $A \cup \{z\}$ is independent. But $A \cup \{z\}$ has more elements than A does. This violates the assumption that A is maximal independent. Thus $\#(C) \leq \#(A)$. Now switch the roles of A and C to get $\#(A) \leq \#(C)$. Putting the two inequalities together, we conclude that $\#(A) = \#(C)$. ■

A statement such as “maximal is maximum” is an example of a *greedy* statement. If at some stage you can’t do better then you have done best possible. In a *greedy algorithm*, if you optimize each individual step then you have a globally optimal solution. Most algorithms are not greedy; this is not an effective way to play chess! However, when you can find a greedy algorithm it is particularly easy to implement. Matroids are a source of greedy algorithms. We illustrate this by assigning a “cost” to the objects in S so we have a way to measure how well we are doing at each step in producing a maximal independent set.

Assume that each member $x \in S$ is assigned a cost $f(x)$ which is a non-negative real number. We make it additive in the sense that for any subset $T \subseteq S$ we announce that the cost of T is the sum of the costs of the members of T . The “greedy algorithm” for producing a cheap maximal independent set is as follows:

STEP 1. Set $I = \emptyset$ and $X = S$.

Repeat STEPS 2 through 4 until X becomes empty.

STEP 2. Pick an element $x \in X$ with minimum cost.

STEP 3. If $I \cup \{x\}$ is independent replace I with $I \cup \{x\}$.

STEP 4. Delete x from X .

Theorem 3.2 *The greedy algorithm produces a maximal independent set of minimum cost.*

Proof: According to the construction, the final set I is maximal independent. List the members of I ,

$$b_1, \dots, b_k$$

in the order they were chosen by the algorithm. Then

$$f(b_1) \leq f(b_2) \leq \dots \leq f(b_k) .$$

Suppose that J is another maximal independent set and list its members,

$$a_1, \dots, a_k \text{ with } f(a_1) \leq \dots \leq f(a_k) .$$

The greedy algorithm requires that $f(b_1) \leq f(a_1)$. If $f(b_r) \leq f(a_r)$ for all possible r then J is no cheaper than I , as we hope. Suppose otherwise. Let j be the smallest index with $f(a_j) < f(b_j)$.

The Subset Rule tells us that

$$\{a_1, a_2, \dots, a_j\} \text{ and } \{b_1, \dots, b_{j-1}\}$$

are both independent sets. Apply the Expansion Rule. We can find t with $1 \leq t \leq j$ such that $a_t \notin \{b_1, \dots, b_{j-1}\}$ and

$$\{b_1, \dots, b_{j-1}, a_t\}$$

is independent. But consider costs.

$$f(a_t) \leq f(a_j) < f(b_j) .$$

In other words, the algorithm should have chosen a_t over b_j at the j^{th} stage.

■

The most interesting thing about the proof we just completed should be its familiarity. It is Kruskal's Algorithm for minimal spanning trees stripped

to its essentials. We pause to make the connection between linear algebra and graph theory via matroids.

Suppose that G is a multigraph. A *cycle* in G is a walk from vertex v_0 to vertex v_m such that v_1, \dots, v_m are distinct and v_0 coincides with v_m . A *forest* is a multigraph with no cycles; a connected forest is called a *tree*. We frequently use the observation that if an edge on a cycle with endpoints x and y is blown up then it is still possible to walk from x to y the “long way around”.

Set S to be the set of edges of the multigraph G . The independent sets for the *graph matroid* associated to G consist of \emptyset and all forests in G . We temporarily delay checking the two axioms for a matroid.

Lemma 3.1 *If the multigraph G is connected then the maximal independent sets are precisely the spanning trees.*

Proof: Suppose that M is a maximal independent set of edges. We first argue that M is connected. If not, there is a walk in G which starts in one connected component C of M and ends in another. Let e be the first edge in the walk with a vertex outside of C . Then e has one vertex a in C and one vertex b not in C . Consider $M \cup \{e\}$. If it creates a cycle then that cycle includes e . Blow up e . There is still a path in M from a to b . It follows that a and b are in the same connected component C , a contradiction. Thus $M \cup \{e\}$ is a larger forest than M , a violation of maximality. We conclude that M is a tree. As to spanning, if M is missing a vertex c in G consider a walk from any vertex in M to c . Some edge f of the walk has one vertex in M and one vertex not in M . It is easy to see that $M \cup \{f\}$ cannot have a cycle.

Conversely, suppose that T is a spanning tree. Can there be an edge g not in T such that $T \cup \{g\}$ is still a forest? The edge g has vertices u and v . There is a walk in T from u to v without any repeated vertices. Glue g onto this walk and we obtain a cycle in $T \cup \{g\}$. ■

You should make sure you understand how the greedy algorithm produces a minimum cost spanning tree and look up Kruskal’s Algorithm. We turn to the verification that the graph matroid is truly a matroid. The Subset Rule is nearly free: a subgraph of a graph with no cycles has no cycles. The Expansion Rule is much more of a challenge. We mimic our work in vector spaces.

Theorem 3.3 (Replacement Principle For Edges) *Assume that F is a forest in a graph G . If e is an edge in G such that $F \cup \{e\}$ creates a cycle*

C then we can replace any edge of F on C with e and the new subgraph obtained from F by this exchange will still be a forest.

Proof: Let f be an edge in F which lies on the cycle C . We wish to replace f with e . In particular, we want to know what goes wrong if we think that the graph after replacement fails to be a forest. So assume that the graph $(F \setminus \{f\}) \cup \{e\}$ has a cycle D . Since F is a forest, D must contain e . If the endpoints of e are x and y then there is a long way around walk in D , consisting of edges in F , which does not pass through e . There is a similar walk from x to y on C . The second walk passes through the edge f and the first does not. If we glue these walks together at x and y we obtain a round-trip walk in F which passes through f exactly once. The smallest round-trip subwalk which passes through f exactly once is a cycle. (This takes a bit of thought.) We reach the contradiction that F has a cycle. ■

Corollary 3.1 *Forests in a graph determine the independent sets for a matroid on the edges of the graph.*

Proof: We must verify the Expansion Rule. We follow the procedure we already have seen for vectors. Assume that F_1 and F_2 are two forests and there are more edges in F_2 than there are in F_1 . We argue by induction on the number of edges in F_1 which do not appear in F_2 that there exists a new edge in the larger forest which can be appended to the smaller and still result in a forest.

If the number is zero then F_1 is a subgraph of F_2 and we are done. For the induction step, suppose that f is an edge of F_1 which is not an edge of F_2 . Create the temporary graph F_{temp} by adding f to F_2 . If F_{temp} is a forest then we can use induction on the pair F_1 and F_{temp} to find an edge e in F_{temp} which is not in F_1 such that $F_1 \cup \{e\}$ is a forest. Since e is in F_2 we have “expanded” as we were supposed to.

But F_{temp} may not be a forest. If it has a cycle then the cycle must contain f and cannot consist entirely of edges from F_1 . Thus we may replace an edge in the cycle which is not in F_1 with f , in the forest F_2 ; according to the theorem, the result of making this exchange in F_2 results in a new forest. Apply induction to F_1 and the new forest. ■

Our third example of a matroid comes from the theory of matchings. Recall that a *bipartite graph* has vertices which come in two colors, **red** and **blue**. Every edge in the graph has one red endpoint and one blue endpoint

and no pair of vertices is connected by more than one edge. A *matching* is a subgraph in which every vertex has degree 0 or degree 1. It is helpful to use the metaphor in which red vertices are people and blue vertices are jobs. A person is connected to a job by an edge provided that person is qualified for the job. A matching occurs when no person is assigned to more than one job and no job is filled by more than one person. One problem is to find the largest matching. We can make the problem more interesting. Suppose that each job applicant demands a certain minimum salary. Among all maximum matchings, we may want to find the cheapest one(s).

The bipartite graph can be recorded as a matrix. The rows are indexed by red vertices and the columns by blue vertices. Place a 1 in the (i, j) -position if the i^{th} red vertex is connected by an edge with the j^{th} blue vertex. For example, consider the matrix

$$\begin{pmatrix} 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

Assume that the salary demands in \$10,000 units are (60, 20, 50, 30, 40). One maximum match is

person 1 to job 1
 person 2 to job 2
 person 3 to job 3
 person 5 to job 4

The total cost is \$170,000. A second matching is

person 1 to job 1
 person 2 to job 2
 person 4 to job 3
 person 5 to job 5

This time the total cost is less: \$150,000. This example is small enough so that one can figure out in an *ad hoc* way that the second solution gives the minimum cost for a maximum matching. To understand what is going on in general, we turn the set-up into a matroid.

Let G be a bipartite graph with **red** and **blue** vertices. Independent sets for the *matching matroid* consist of \emptyset together with those subsets of

red vertices which can be legally matched with blue vertices. A maximal independent set obviously is the red contribution to a maximal matching. Assuming this really is a matroid, we can use our usual greedy algorithm to produce a maximum matching with minimum cost.

The Subset Rule holds trivially. The Expansion Rule is quite subtle. We present a general verification illustrated by an example; we urge the reader to sketch the actual bipartite graphs.

We assume that the bipartite graph has two matchings, one with $n + 1$ edges which is “sweet” and one with n edges which is “sour”, e.g.,

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

and

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

In the example, the sweet independent set of red vertices is $\{1, 2, 4, 5, 6, 7, 8\}$ while the sour independent set of red vertices is $\{1, 3, 4, 6, 7, 8\}$. If we simply add the sweet vertex 2 with its sweet edge to the sour matching, the second blue vertex will have two edges attached to it. A similar problem arises with sweet vertex number 5. So we will have to be more clever to enlarge the sour independent set of red vertices.

By the “taste subgraph” we will mean the subgraph of the original bipartite graph which consists of all original vertices and all edges which are either sweet or sour but not sweet-and-sour. Every vertex in the taste subgraph has degree 0, 1, or 2. Consequently, any connected component of the graph is either a singleton vertex with no edges, a cycle, or a path with no

repeated vertices. In the last two cases, the walks alternate sweet and sour. Since the taste subgraph has one more sweet edge than sour, at least one connected component must have one more sweet than sour edge. Pick one of these components and call it a “delicious path”. In the example, red 5 to blue 4 to red 6 to blue 6 is a delicious path with sweet-sour-sweet edges.

Define a new matching according to the following scheme:

- Every sour edge in the original graph which is not on the delicious path is in the new matching.
- Every sweet edge on the delicious path is in the new matching.

We must check three things. We must have a matching, all sour red vertices must still be included, and one new sweet red vertex must appear. We leave the first two as exercises. As to the last, the delicious path has one terminal red vertex attached to a sweet edge and one terminal blue vertex attached to a sweet edge. The terminal red vertex is in the new matching whereas it was not in the sour matching. (In the example, the terminal red is vertex number 5.) Indeed, if the terminal red was in the sour matching either the incident sour edge would be sweet-and-sour or it would give the terminal red vertex degree 2 in the taste subgraph; both scenarios are impossible.

4 Lattices

There are many problems which require algebra over the integers rather than over a scalar field. (The issue of integer linear combinations already appears in Lemma 2.1.) This variation of linear algebra comes with its own terminology. A *lattice* is the the collection of all integer linear combinations of some finite set of vectors in \mathbf{R}^n which are linearly independent. In case all of the vectors lie in \mathbf{Z}^n (i.e., every lattice vector has integer coordinates), we refer to an *integer lattice*. When we say vectors “span” a lattice, we mean in this context that all admissible coefficients in a linear combination are integers.

Suppose v_1, \dots, v_n are vectors in \mathbf{R}^d . Consider all integer linear dependencies

$$c_1 v_1 + \dots + c_n v_n = 0$$

with each $c_j \in \mathbf{Z}$. The dependency is encoded by a vector

$$(c_1, c_2, \dots, c_n) \in \mathbf{Z}^n.$$

We are about to show that the collection of all such “dependency vectors” make up a lattice. However, at this stage it is not at all clear where we will find a finite list of spanners, each of which is a dependency vector. On the positive side, this collection of integer vectors has two properties which are easy to check: it is closed under addition and closed under multiplication by integer scalars. This is all we need.

Theorem 4.1 *Any nonempty subset of \mathbf{Z}^n closed under addition and multiplication by integer scalars is a lattice.*

Proof: Let’s temporarily call a subset of \mathbf{Z}^n with the two closure properties a “virtual lattice”. We argue by mathematical induction on n that a virtual lattice L in \mathbf{Z}^n can be spanned by n vectors or less.

We first indicate how to drop in dimension. Set L' to be the subset of L consisting of all vectors whose first coordinate is 0. Since $0 \in L$ (why?), L' is nonempty. It is easy to check that L' is a virtual lattice, too. But for all practical purposes, the first coordinate of vectors in L' is irrelevant. We might as well erase this first zero and regard L' as living inside \mathbf{Z}^{n-1} . By induction, L' is spanned by $n - 1$ vectors from \mathbf{Z}^{n-1} . Regluing a zero at the end of these spanners, we see that the unexpurgated L' is spanned by integer vectors w_2, w_3, \dots, w_n .

Now go back to all of L and focus on the set F of first coordinates for all vectors in L . It is easy to check that F is a nonempty set of \mathbf{Z} closed under addition and closed under multiplication by any integer. If F consists of 0 alone then $L = L'$ and we’re done. Otherwise, F contains a smallest positive integer s (why?). Let w_1 be a vector in L with first coordinate s . If v is any vector in L and it has first coordinate t , perform long division:

$$t = sq + r \quad \text{with } 0 \leq r < s.$$

(This works even if t is negative.) The closure properties of F tell us that $r \in F$ because $t, s \in F$. By the minimality of s , we must have $r = 0$. In other words,

$$v - qw_1 \in L'.$$

Write $v - qw_1 = b_2w_2 + \dots + b_nw_n$ by the previous paragraph. Then

$$v = qw_1 + b_2w_2 + \dots + b_nw_n.$$

The two paragraphs above actually describe an algorithm which produces specific lattice vectors w'_1, w'_2, \dots, w'_k for some $k \leq n$, which span L and have

zeros in strategic places. Suppose we recorded the first vector on this list as the first row of a matrix, the second vector as the second row, etc. Then each row has a nonzero leading entry and all entries directly underneath a leading entry are zeroes. (This may remind you of the rows of a matrix in row echelon form.) Can you successively use the Replacement Principle, starting with the subset of the standard basis with ones only in the positions of leading entries, to show that w'_1, \dots, w'_k are linearly independent? ■

One of the fundamental problems for lattices is the identification of a shortest vector inside, by means of a practical algorithm. When finding such a vector is too hard, we often settle for a “pretty short” vector. The strategy is to find a clever way to produce a spanning set with short integer vectors. Fortunately, we can still use linear combinations to improve a basis for a lattice because the Replacement Principle for vectors needs only slight modification. Assume that v_1, \dots, v_n span a lattice and

$$w = \beta_1 v_1 + \dots + \beta_n v_n$$

for integers β_j . In order to divide at the appropriate spot in the verification of the Principle and yet remain with integer coefficients, we insist that v_t is only replaced with w when $\beta_t = \pm 1$. Of course, the availability of Replacement does not tell us at all how to replace integer vectors in a helpful, shortening way.

Let's return to the issue of computing an integer linear dependency for $v_1, \dots, v_n \in \mathbf{Z}^d$ of shortest possible Euclidean length. We could use the previous theorem to find a basis for the lattice of dependencies and then play with the basis to obtain the shortest vector in the lattice. In practice, the first step can be finessed.

First use something like Lemma 2.1 to make an *a priori* back-of-the-envelope estimate of the length of a shortest integer dependency.

$$\alpha_1 v_1 + \dots + \alpha_n v_n = 0$$

for integers α_j not all zero with $\|(\alpha_1, \dots, \alpha_n)\| < K$. (Here K is some large integer.) Now pad each vector with n additional coordinates by setting

$$w_1 = (Kv_1, 1, 0, \dots, 0); w_2 = (Kv_2, 0, 1, \dots, 0); \dots; w_n = (Kv_n, 0, \dots, 0, 1).$$

Each of the new vectors lies in \mathbf{Z}^{d+n} . Since

$$\alpha_1 K v_1 + \dots + \alpha_n K v_n = 0,$$

we have

$$\alpha_1 w_1 + \cdots + \alpha_n w_n = (0, \dots, 0, \alpha_1, \alpha_2, \dots, \alpha_n).$$

Thus $\|\alpha_1 w_1 + \cdots + \alpha_n w_n\| < K$. Consequently, the shortest nonzero vector in the lattice L spanned by w_1, \dots, w_n must have length smaller than K .

The padding trick we have just exploited really shows that if $(\gamma_1, \dots, \gamma_n)$ describes any integer dependency for v_1, \dots, v_n then $(0, \dots, 0, \gamma_1, \dots, \gamma_n) \in L$. Conversely, if $(0, \dots, 0, \sigma_1, \dots, \sigma_n) \in L$ write

$$(0, \dots, 0, \sigma_1, \dots, \sigma_n) = c_1 w_1 + \cdots + c_n w_n.$$

Then $K(c_1 v_1 + \cdots + c_n v_n) = 0$ from the first d coordinates and

$$\sigma_1 = c_1, \dots, \sigma_n = c_n$$

from the padded coordinates. Thus $(\sigma_1, \dots, \sigma_n)$ describes a linear dependency for v_1, \dots, v_n .

We next argue that any shortest vector in L *must* have its first d coordinates equal to zero. In other words, a shortest vector in L truncates to a shortest integer dependency for v_1, \dots, v_n . So suppose

$$u = (\beta_1, \dots, \beta_{d+n})$$

is a shortest vector in L . Then $\|u\| < K$, so $|\beta_i| < K$ for $i = 1, \dots, d+n$. On the other hand, u is an integer linear combination of w_1, \dots, w_n forcing each of its first d coordinates to be a multiple of K . The only multiple of K with absolute value less than K is zero! Hence

$$u = (0, \dots, 0, \beta_{d+1}, \dots, \beta_{d+n})$$

as claimed.

The only problem with the procedure we have just described is that there is no algorithm which quickly produces a shortest vector in a lattice. However, we have some wiggle room. Later on, we will derive an algorithm that efficiently provides a “pretty short” vector in a lattice. By taking K even larger in the scheme above, we can then produce a pretty short dependency.

Sometimes this is good enough for a complete solution to a problem. We illustrate this idea with a “knapsack” scenario. There is a small company which has sent out bills for the amounts a_1, a_2, \dots, a_n to its different customers. Some time later, they get a bank statement telling them that

a total amount of b dollars has been transferred to their account. To their dismay, this is less than the total outstanding amount

$$a_1 + a_2 + \cdots + a_n.$$

They want to write letters to the customers who have not paid, so they have to figure out which subset of the numbers a_j adds up to b . Formally, we must find a subset

$$\{i(1), \dots, i(k)\}$$

of

$$\{1, 2, \dots, n\}$$

such that

$$a_{i(1)} + \cdots + a_{i(k)} = b.$$

The additional information available is that there is *some* solution. If we assume that the a_j are “randomly” chosen from a large set then we know a bit more: there is a unique solution (x_1, \dots, x_n, y) to the bank equation

$$a_1 X_1 + \cdots + a_n X_n - bY = 0$$

with each x_j and y either 0 or 1. Moreover, any possible integer solution to the equation is either an integer multiple of this solution or has a coordinate whose absolute value is at least 2^n . (We will give a plausibility argument for this assertion when we have completed our analysis.)

We are looking for an integer linear dependency among the *numbers* $a_1, \dots, a_n, -b$ which has such a small length that the coefficients are forced to be 0 or 1. The extra information essentially says that any relatively short solution is determined by the unique shortest solution.

Imagine that we have sufficiently random bills and that we have a practical general algorithm which may not produce the vector of shortest length λ in an $n + 1$ -dimensional lattice, but will produce one of length at most $2^{\frac{n}{2}} \lambda$. We apply this algorithm to the dependency lattice for our $n + 1$ numbers. If the wonderful vector which is produced happens to have length less than 2^n then each entry has absolute value less than 2^n ; by rescaling, we can recover the truly shortest lattice vector. What are the implications should the wonderful vector have length 2^n or greater? Then

$$2^n \leq 2^{\frac{n}{2}} \lambda \quad \text{i.e.,} \quad \lambda \geq 2^{\frac{n}{2}}.$$

A simple induction shows that for $n \geq 2$,

$$2^{\frac{n}{2}} > \sqrt{n + 1},$$

so

$$\lambda > \sqrt{n+1}.$$

Now the length of the unique shortest solution (x_1, \dots, x_n, y) is also

$$\sqrt{x_1^2 + \dots + x_n^2 + y^2}.$$

If the entries are 0's or 1's then the length cannot be more than $\sqrt{n+1}$. In other words, if the wonderful vector has length 2^n or greater then there is no 0,1 solution to the banking problem after all, a contradiction!

As an appendix, we examine the strong uniqueness which is supposed to be a consequence of randomness among large numbers. Suppose that

$$(\overline{x_1}, \dots, \overline{x_n}, \overline{y})$$

is a integer solution to the banking problem with $|\overline{x_j}|, |\overline{y}| < 2^n$ and this solution is not a multiple of our 0,1-solution (x_1, \dots, x_n, y) . Set

$$u_j = \begin{cases} \overline{x_j} - \overline{y} & \text{if } x_j = 1 \\ \overline{x_j} & \text{if } x_j = 0. \end{cases}$$

Since the new solution is not a multiple of the short solution, some u_i is nonzero. We have $-2^{n+1} < u_j < 2^{n+1}$ for all j and

$$a_1 u_1 + \dots + a_n u_n = 0.$$

This is unlikely if the a_j are large in comparison to 2^{n+1} and randomly distributed.

5 Eventown Revisited

Having acquired expertise with vector spaces, we return to the Eventown problem. In vector language, a configuration of Eventown clubs is a subset of \mathbf{F}_2^{32} such that the dot product of any two vectors (including a vector with itself) is zero. Once we start thinking in vector space terms, we might discover the following remarkable property.

Lemma 5.1 *A maximal collection of Eventown vectors constitutes a subspace of \mathbf{F}_2^{32} .*

Proof: Suppose that C is a maximal collection of Eventown club vectors. To show that C is a subspace we need only show that C is closed under addition. (Why don't we have to worry about scalar multiplication?) Suppose that x and y lie in C but $x + y$ does not. If v is any vector in C then

$$v \cdot (x + y) = (v \cdot x) + (v \cdot y) = 0 .$$

In addition,

$$(x + y) \cdot (x + y) = x \cdot x + x \cdot y + y \cdot x + y \cdot y = 0 .$$

Therefore $C \cup \{x + y\}$ satisfies Eventown Rules. But this violates maximality.

■

The lemma already gives us information about any maximal collection of Eventown vectors. Any subspace of \mathbf{F}_2^{32} has a basis. (Give a short and stupid algorithm for producing one.) If the basis is v_1, \dots, v_r then there are exactly 2^r distinct linear combinations which can be constructed. As a consequence, any maximal Eventown collection has cardinality a power of 2. This suggests that our number for the couples configuration, 2^{16} , might be a reasonable candidate for the cardinality of a maximum configuration. Looking at dimensions, we have to understand why an "Eventown subspace" should have half the dimension of the entire space. We are going to explain how to get the second half of \mathbf{F}_2^{32} with the "more the merrier" technique. This will not be the only time we will use this line of argument.

Theorem 5.1 *Suppose that E is a maximum collection of Eventown vectors in \mathbf{F}_2^{2m} . Then E is a subspace of dimension m .*

Proof: We already know from the lemma that E is a subspace. It has the property that the dot product of any two vectors in E is zero. Let v_1, \dots, v_r be a basis of E . The idea of more-the-merrier is to strategically throw in more vectors while maintaining linear independence.

One piece of the argument we will use several times is known as *non-degeneracy*. If $v \in \mathbf{F}_2^{2m}$ is nonzero then we can find some vector w so that $v \cdot w = 1$. Indeed, v must have a nonzero entry somewhere. If there is a 1 in the j^{th} coordinate, let w be the standard basis vector with 1 in the j^{th} coordinate.

Use nondegeneracy to find a vector $w_1 \in \mathbf{F}_2^{2m}$ such that $v_1 \cdot w_1 = 1$. If $v_k \cdot w_1 = 0$ for all other choices of k we are happy; if not we change the basis for E as follows. Find the first $d > 1$ with $v_d \cdot w_1 = 1$. Replace v_d with

$v_d - v_1$ in the basis for E to obtain a new basis. (Caution: Slow down for an application of the Replacement Principle.) It is easy to see that the dot product of $v_d - v_1$ with every vector in E is zero. Moreover, $(v_d - v_1) \cdot w_1 = 0$. Now rename the new vector v_d and find the next vector on the list whose dot product with w_1 is 1. Replace this vector, etc. When we are done, we will have a basis v_1, \dots, v_r for E such that $v_i \cdot w_1 = 0$ for all $i \geq 2$.

Next choose w_2 so that $v_2 \cdot w_2 = 1$. We repeat the process we just went through. If $v_1 \cdot w_2 = 0$ we are happy. If not, replace v_1 with $v_1 - v_2$. We get $(v_1 - v_2) \cdot w_2 = 0$ but we had better make sure we have not messed up the work we just achieved in the previous step.

$$(v_1 - v_2) \cdot w_1 = 1 - 0 = 1 .$$

Go to the next $d > 2$ with $v_d \cdot w_2 = 1$. Make the replacement $v_d \mapsto v_d - v_2$. In the previous step we had $v_d \cdot w_1 = 0$ while now,

$$(v_d - v_2) \cdot w_1 = 0 - 0 = 0 .$$

We have preserved the efforts of step 1. Rename and continue. At the end of step 2, we will have a new basis for E , as well as two vectors w_1 and w_2 such that

$$v_1 \cdot w_1 = v_2 \cdot w_2 = 1 \text{ and } v_i \cdot w_j = 0 \text{ for } i \neq j .$$

By this point you should have figured out what steps 3 through r must be. When the entire algorithm stops we will have concocted r new vectors w_1, \dots, w_r with the properties that

$$v_1 \cdot w_1 = \dots = v_r \cdot w_r = 1 \text{ and } v_i \cdot w_j = 0 \text{ for } i \neq j .$$

We claim that the large list $v_1, \dots, v_r, w_1, \dots, w_r$ is linearly independent. Suppose we have scalars with

$$\alpha_1 v_1 + \dots + \alpha_r v_r + \beta_1 w_1 + \dots + \beta_r w_r = 0 .$$

Take the dot product of each side with v_t . We will obtain $\beta_t = 0$. (What properties of the list are we using?) So far we have $\beta_1 = \dots = \beta_r = 0$. But $\alpha_1 v_1 + \dots + \alpha_r v_r = 0$ forces all of the α_i to be zero because the v_i constitute a basis for E .

Let's see where we stand. At the end of the algorithm, we have a list of $2r$ linearly independent vectors in \mathbf{F}_2^{2m} . Since any maximal list of linearly independent vectors in the ambient space has $2m$ vectors, we must have

$2r \leq 2m$. Hence $2^r \leq 2^m$. But the couples scheme produces 2^m Eventown club vectors. Thus $r = m$. ■

Here is a question which, given the issues so far, should occur to you. What if we replace “maximum” with “maximal” in the last theorem? A subspace of \mathbf{F}_2^n is *totally isotropic* provided that the dot product of any two vectors in the subspace is zero. Is every maximal totally isotropic subspace a maximum one?

6 Block Designs

The subject of block designs was initiated and developed by statisticians to construct valid agricultural experiments. Over the years, the theory has found application to geometry, cryptography, memory design for computers, and a myriad of combinatorial problems.

We begin with a prototype toy example. The requirement is to compare the advantage of one motor oil over another for a particular model car. There are 7 motor oils to analyze. The constraint is that testing causes such wear and tear on a car that only 3 motor oils can be tested on any given car. Since two cars of the same model, style, and year still are not identical, we want to ensure that comparisons of any two oils are fair: given any pair of motor oils, there should be a car which tests these two. For uniformity, we may even ask that each pair be tested in exactly one car. Cars being expensive, we would like to know the least number to purchase in order to satisfy all of the design constraints.

Let’s set the problem up more mathematically. List the motor oils as $\{1, 2, 3, 4, 5, 6, 7\}$. Each car is assigned a different “block” of three elements from this set. Furthermore, we require that each pair of elements from the motor oil set lie in exactly one block. One possible design is

$$\begin{aligned} & \{ 1, 2, 3 \} \\ & \{ 1, 4, 5 \} \\ & \{ 1, 6, 7 \} \\ & \{ 2, 4, 6 \} \\ & \{ 2, 5, 7 \} \\ & \{ 3, 5, 6 \} \\ & \{ 3, 4, 7 \} \end{aligned}$$

Notice a remarkable coincidence. We required 7 cars to test 7 motor oils. We want to study this issue. But first a word from our sponsor, the definition

industry.

Definition 6.1 A set V of v objects called **varieties** is given. A **design** on V is a collection of b subsets of V called **blocks**. The design is **incomplete** provided at least one block is not the entire set V . The design is **k -uniform** if each block has the same size k . It is **balanced** of **index** λ provided that each pair of distinct varieties appears together in exactly λ blocks of the design.

We focus on *balanced incomplete block designs* with parameters (b, v, k, λ) . For example, the motor oil example is a $(7, 7, 3, 1)$ -BIBD. Our second example may remind you of clubs. It may also ring a geometric bell: to say that $\lambda = 1$ is to say that “two varieties determine a block”. Sound familiar?

Consider the varieties to be the nonzero vectors in \mathbf{F}_2^4 . A block is a triple $\{x, y, z\}$ of vectors such that

$$x + y + z = 0 .$$

(Prove that the blocks are all of the two-dimensional subspaces of \mathbf{F}_2^4 with zero thrown out of each.) Since any two distinct vectors in a block determine the third by $z = x + y$, we see that the design is balanced of index 1. You might want to try a similar construction in \mathbf{F}_5^3 where the varieties are all one-dimensional subspaces and the blocks are all two-dimensional subspaces. There are $\frac{5^3-1}{5-1} = 31$ varieties because the intersection of any two distinct one-dimensional subspaces consists of zero alone. How many varieties are there in a block? How many varieties are in the intersection of two blocks? Can you make a connection to Oddtown?

We leave our third example, called the *Fano plane*, as an exercise . The varieties are the seven elements of the integers modulo 7. The blocks have the form $B_x = \{x, x + 1, x + 3\}$ for x in the integers mod 7. Find all of the parameters to verify that this is a BIBD. Can you draw a picture of the incidence relationships? Do you see what appears to be a different solution to the motor oil problem?

Some elementary general observations can be made. Suppose we have a balanced incomplete block design with parameters b, v, k , and λ . We ask how many blocks a particular variety x appears in. Notice that $k > 1$ since blocks of size one cannot have pairs of varieties inside. Consider the collection of ordered pairs (B, y) such that B is a block with $\{x, y\} \subseteq B$. (Here x and y are distinct.) On one hand, the number of pairs is

$$\#(\text{blocks containing } x)(k - 1) .$$

On the other hand, each pair of distinct varieties x and y appear together in λ blocks and there are $v - 1$ choices of y . Thus the number of pairs is also

$$\lambda(v - 1) .$$

Therefore the number of blocks in which x appears is independent of x and equal to $\frac{\lambda(v-1)}{k-1}$.

Let's vary the argument and consider all ordered pairs (B, z) such that B is a block and $z \in B$. The total number of pairs is bk . But we now know that each variety appears in $\frac{\lambda(v-1)}{k-1}$ blocks so the number of pairs is also $v \frac{\lambda(v-1)}{k-1}$. Therefore $bk(k - 1) = \lambda v(v - 1)$.

What we would really like to know is whether we always need at least as many blocks as varieties. The expression $b \geq v$ is known as *Fisher's inequality*. We are going to use "club" methods to analyze it. The constraint on pairs of varieties is like the constraint on pairs of clubs. We can make this precise by inventing clubs whose members are blocks, inspired by the calculations we just finished. For each variety x the x -club $C(x)$ has as members those blocks which contain x . If x and y are distinct then

$$\#(C(x) \cap C(y)) = \lambda \text{ and } 1 \leq \lambda < b .$$

We remark that two distinct varieties x and y cannot determine the same club. For if this is the case, the intersection equality above implies that $\#(C(x)) = \lambda$. But the number of blocks containing x is independent of x . Thus every club named after a variety contains λ blocks as members. A second application of the intersection equality then tells us that $C(u) = C(w)$ for each pair of varieties u and w . In other words, every block consists of every variety! This violates the I in BIBD.

Theorem 6.1 (Fisher's Inequality) *Let C_1, \dots, C_v be distinct subsets of a set with b elements. Assume that for every $i \neq j$ there are uniform intersections,*

$$\#(C_i \cap C_j) = \lambda$$

where λ is some number satisfying $1 \leq \lambda < b$. Then $v \leq b$.

Proof: We first eliminate a degenerate case which occurs in our abstract version of Fisher's inequality but never comes up for BIBD's. Suppose one of the sets, say C_r , has λ elements. Then every C_j must contain this particular

C_r . More can be said because the sets are all different: if $j \neq r$ then C_j contains C_r and at least one element not in any other C_* . Hence

$$b \geq \lambda + (v - 1),$$

counting the elements of C_r and the extra element per other set which appears only in that set. Moving around the symbols we obtain

$$v \leq b - \lambda + 1 \leq b .$$

From now on, we will assume that $\#(C_i) - \lambda$ is strictly positive for each i . Call this discrepancy d_i . List the b elements in our ambient set so that we can represent each C_j as an incidence vector c_j in \mathbf{R}^b . Our hypothesis is that $c_i \cdot c_j = \lambda$ for $i \neq j$. With our special notation, $c_j \cdot c_j = \lambda + d_j$. The inequality we need will follow from what we know about maximal independent sets once we show that c_1, \dots, c_v is linearly independent!

Suppose that $\sum_{j=1}^v \alpha_j c_j = 0$ for some choice of real scalars α_j . How did we establish linear independence for clubs? Take the dot product of this expression for zero with each c_k .

$$\left(\sum \alpha_i\right)\lambda + (\alpha_k d_k) = 0 .$$

We list this information in a table:

$$\begin{array}{ccccccc} \alpha_1 & \alpha_2 & \cdot & \cdot & \cdot & \alpha_v & \\ \alpha_1 d_1 + \lambda \Sigma & \alpha_2 d_2 + \lambda \Sigma & \cdot & \cdot & \cdot & \alpha_v d_v + \lambda \Sigma & \end{array}$$

where the bottom row “really” consists entirely of zeroes. Our last trick is to regard the two rows as vectors and take their dot product again.

$$\alpha_1^2 d_1 + \alpha_2^2 d_2 + \dots + \alpha_v^2 d_v + \lambda \left(\sum \alpha_j\right)^2 = 0$$

Each term in this sum is non-negative and yet they add to zero. The only way this can happen is if each term is zero. In particular, $\alpha_j^2 d_j = 0$ for each j . Since d_j is never zero, we conclude that $\alpha_j = 0$ for all j . ■

The argument we have just presented is clever and mysterious. Since it might help to see it more than once, we are going to make a digression from block designs to study a geometric puzzle which turns out to have nearly the same solution as the previous theorem. What is the largest possible number of points in the plane which are all equidistant from each other? The equilateral triangle probably came to mind, which means the answer is

likely 3. What is the answer in space? What is the answer in \mathbf{R}^n ? Intuitively, the answer is $n + 1$. We are going to look at the problem very carefully in light of inner products. The connection lies in the definition of the length of a vector. If $w \in \mathbf{R}^n$ then the length of w comes from the Pythagorean Theorem: the length of $w = (w_1, \dots, w_n)$ is the square root of

$$w_1^2 + \dots + w_n^2 .$$

In other words,

$$\|w\| = \sqrt{w \cdot w} .$$

With this notation, the distance between vector v and vector w is

$$\|v - w\| = [(v - w) \cdot (v - w)]^{\frac{1}{2}} .$$

We make two reductions in the equidistance problem for \mathbf{R}^n . First, we can drag the configuration of points so that one of the points is the origin. Second, we can rescale so that the common distance is always 1. Under these conditions, suppose that we have $t + 1$ distinct points

$$v_0 = 0, v_1, \dots, v_t$$

such that the distance between any two different vectors is 1. If $j \neq 0$ then $\|v_j\| = \|v_j - v_0\| = 1$. In other words, $v_j \cdot v_j = 1$ whenever j is not 0. If both i and j are nonzero then

$$1 = (v_i - v_j) \cdot (v_i - v_j) = 2 - 2(v_i \cdot v_j) .$$

In summary, for v_1, \dots, v_t we have

$$v_j \cdot v_j = 1 \quad \text{and} \quad v_i \cdot v_j = \frac{1}{2} \quad \text{for all } i \neq j .$$

This should remind us of the set-up for Fisher's inequality with $\lambda = \frac{1}{2}$ and discrepancy always $\frac{1}{2}$.

Proposition 6.1 *Suppose that v_1, \dots, v_t are vectors in \mathbf{R}^n such that*

$$v_j \cdot v_j = 1 \quad \text{and} \quad v_i \cdot v_j = \frac{1}{2} \quad \text{for all } i \neq j .$$

Then this list of vectors is linearly independent. In particular, there are at most $n + 1$ equidistant vectors in \mathbf{R}^n .

Proof: Suppose that $\sum_{j=1}^t \alpha_j v_j = 0$ for some choice of real scalars α_j . Take the dot product of this expression for zero with each v_k .

$$\frac{1}{2}\alpha_k + \frac{1}{2}\sum \alpha_i = 0 .$$

(Remember that we get an extra quantity from the dot product of v_k with itself.) We list this information in a table:

$$\begin{array}{ccccccc} \alpha_1 & \alpha_2 & \cdot & \cdot & \cdot & \alpha_t & \\ \frac{1}{2}\alpha_1 + \frac{1}{2}\Sigma & \frac{1}{2}\alpha_2 + \frac{1}{2}\Sigma & \cdot & \cdot & \cdot & \frac{1}{2}\alpha_t + \frac{1}{2}\Sigma & \end{array}$$

As before, every number in the bottom row is zero. Dot product the rows.

$$\frac{1}{2}\alpha_1^2 + \frac{1}{2}\alpha_2^2 + \cdots + \frac{1}{2}\alpha_t^2 + \frac{1}{2}(\sum \alpha_j)^2 = 0$$

A sum of squares is zero if and only if each square is zero. Thus $\alpha_j = 0$ for all j . ■

7 Polynomial Spaces

Here is a naive follow-up to the last proposition. What is the maximum number of points for a configuration in \mathbf{R}^n with only *two* choices for the distances between all pairs points? This time we can use the square as an example of such a configuration in the plane. If we can't find the exact maximum, we'd at least like a good bound.

Suppose that the two available positive distances are d and e . If u and v are any two different points on the configuration then either $\|u - v\|$ is d or e . A fancy but compact way of writing this is

$$(\|u - v\|^2 - d^2)(\|u - v\|^2 - e^2) = 0 .$$

If we set $X = (X_1, \dots, X_n)$ and $Y = (Y_1, \dots, Y_n)$ then we obtain a polynomial

$$F(X, Y) = (\|X - Y\|^2 - d^2)(\|X - Y\|^2 - e^2)$$

in $2n$ variables with the following property. If $v[1], \dots, v[k]$ are the points in a 2-distance configuration then

$$F(v[i], v[j]) = \begin{cases} d^2 e^2 & \text{if } i = j; \\ 0 & \text{if } i \neq j. \end{cases}$$

Notice that $d^2e^2 \neq 0$. Now stare at the formula above and imagine that the F vanishes into thin air. The formula is the dot-product version of Oddtown Rules! Can we make this analogy work legitimately?

The new idea is to use polynomials as vectors. Think of a polynomial

$$a_0 + a_1X + a_2X^2 + \cdots + a_dX^d .$$

It is the zero polynomial when each of the coefficients is zero. This should remind you of linear independence. If we think about the algebra of vector spaces we realize that we did not use the fact that we had n -tuples in any significant way. The important properties of vectors for spanning and linear independence were that they could be added and multiplied by scalars. Polynomials have these properties. Moreover, the algebraic rules (like distributive laws, etc.) which allowed us to work with vectors also hold for polynomials. So we could abstract the notion of vector space to include polynomials. Rather than take this detour, we will just expand our language to include polynomial spaces. For example, the list

$$1, X, X^2, X^3, \dots$$

is an independent set in the vector space of all polynomials in one variable. More generally, if X_1, X_2, \dots, X_n are variables then the list of all monomials

$$X_1^{e(1)} X_2^{e(2)} \cdots X_n^{e(n)} \quad \text{for non-negative integers } e(1), \dots, e(n)$$

is a basis for the vector space of all polynomials in n variables.

Proposition 7.1 *Suppose that $v[1], \dots, v[k]$ are vectors in \mathbf{R}^n such that the distance between any two of them is one of two positive numbers. Then $k \leq \frac{(n+1)(n+4)}{2}$.*

Proof: We use the notation d , e , and F we have already introduced. For each t with $1 \leq t \leq k$ we describe a new polynomial in n variables by

$$f_t(X) = F(X, v[t]) .$$

(Remember that X is shorthand for (X_1, \dots, X_n) .) We first argue that f_1, \dots, f_k are linearly independent polynomials using the “fake” dot product. Suppose that

$$\alpha_1 f_1(X) + \cdots + \alpha_k f_k(X) = 0$$

for some choice of real numbers α_t . Make a choice of t and plug in $v[t]$ for X ; substitute the first coordinate of $v[t]$ for X_1 , etc. Explicitly,

$$f_j(v[t]) = F(v[t], v[j]) .$$

Using our Oddtown-like formula, we see that

$$0 = \alpha_1 f_1(v[t]) + \cdots + \alpha_k f_k(v[t]) = \alpha_t d^2 e^2 .$$

Therefore $\alpha_t = 0$ for all t .

The next step is to find a manageable vector space in which the polynomials $f_1(X), \dots, f_k(X)$ live. We will want to compare the dimension of this space with the size of our independent list. What does a typical $f_t(X)$ really look like? Let's write out

$$v[t] = (b_1, \dots, b_n) .$$

Then $f_t(X)$ is

$$\left((X_1 - b_1)^2 + \cdots + (X_n - b_n)^2 - d^2 \right) \left((X_1 - b_1)^2 + \cdots + (X_n - b_n)^2 - e^2 \right) .$$

Imagine expanding this expression (or use Mathematica!). You will get a summand $(X_1^2 + \cdots + X_n^2)^2$, linear combinations of all $(X_1^2 + \cdots + X_n^2)X_j$, of all $X_i X_j$, of all X_j , and a scalar multiple of 1. The total number of such monomials is $1 + n + \frac{n(n+1)}{2} + n + 1$. In other words, f_1, \dots, f_k lie in the subspace spanned by these $\frac{(n+1)(n+4)}{2}$ monomials. Since the dimension of this subspace cannot exceed the cardinality of a spanning set (why?), we are done. (In fact, you should be able to show that these particular monomials constitute a basis for the subspace.) ■

The bound we currently have can be rewritten $\binom{n+2}{2} + (n+1)$. We are going to show that it can be significantly improved to simply $\binom{n+2}{2}$. To establish this we will use fake dot products, genuine dot products, more-the-merrier, and calculus. Since we will be continuing the earlier argument, we borrow all of the notation.

Theorem 7.1 *Suppose that $v[1], \dots, v[k]$ are vectors in \mathbf{R}^n such that the distance between any two of them is one of two positive numbers. Then $k \leq \frac{(n+1)(n+2)}{2}$.*

Proof: The big picture is a more-the-merrier argument which shows that

$$f_1, \dots, f_k, X_1, \dots, X_n, 1$$

is linearly independent. Since this entire list lives in the previously described subspace of dimension $\binom{n+2}{2} + (n+1)$ we will have

$$k + (n+1) \leq \binom{n+2}{2} + (n+1) .$$

The desired inequality follows by subtracting $n+1$ from each side.

So suppose that

$$\alpha_1 f_1 + \dots + \alpha_k f_k + \beta_0 \cdot 1 + \beta_1 X_1 + \dots + \beta_n X_n = 0 .$$

The first trick is to get rid of all of the linear terms (e.g., anything involving a β) by taking derivatives twice. For each $i < j$, apply $\frac{\partial^2}{\partial X_i \partial X_j}$ to the expression for zero,

$$\sum_{t=1}^k \alpha_t \frac{\partial^2}{\partial X_i \partial X_j} f_t = 0 .$$

With the notation $v[t] = (v_1[t], \dots, v_n[t])$ and after dividing by 8, we obtain

$$\sum_{t=1}^k \alpha_t (X_j - v_j[t])(X_i - v_i[t]) = 0 .$$

The left-hand side of the equation is a linear combination of four monomials: $X_j X_i$, X_j , X_i , and 1. Since these monomials are linearly independent, their coefficients must be zero. We single out the first two,

$$\sum_{t=1}^k \alpha_t = 0 \quad \text{and} \quad \alpha_1 v_i[1] + \alpha_2 v_i[2] + \dots + \alpha_k v_i[k] = 0$$

for all choices of i .

Second, we use the fake dot product trick and plug in $v[t]$ for X in the original linear combination equal to zero. The upshot is that all but one $f_s(v[t])$ disappears.

$$d^2 e^2 \alpha_t + \beta_0 + \beta_1 v_1[t] + \dots + \beta_k v_k[t] = 0 .$$

For typesetting purposes, we set $C = d^2 e^2$.

Third, we use a *bona fide* dot product. As we have done earlier, we line up our data in two rows where each entry of the bottom row is an expression for zero:

$$C\alpha_1 + \beta_0 + \sum \beta_s v_s[1] \quad C\alpha_2 + \beta_0 + \sum \beta_s v_s[2] \quad \cdots \quad C\alpha_k + \beta_0 + \sum \beta_s v_s[k]$$

The dot product of the rows is

$$0 = C \sum_t \alpha_t^2 + \beta_0 \sum_t \alpha_t + \sum_t \sum_{s \geq 1} \alpha_t \beta_s v_s[t].$$

The second term is zero by the partial derivative calculation. The third term will be more comprehensible if we write all of the terms out:

$$\begin{aligned} & \alpha_1 \beta_1 v_1[1] + \alpha_1 \beta_2 v_2[1] + \cdots + \alpha_1 \beta_k v_k[1] \\ + & \alpha_2 \beta_1 v_1[2] + \alpha_2 \beta_2 v_2[2] + \cdots + \alpha_2 \beta_k v_k[2] \\ & \vdots \\ + & \alpha_k \beta_1 v_1[k] + \alpha_k \beta_2 v_2[k] + \cdots + \alpha_k \beta_k v_k[k] \end{aligned}$$

If we look down each column we will see β_j multiplying the second expression equal to zero which came from taking partial derivatives. We are left with

$$0 = C \sum_t \alpha_t^2.$$

Therefore $\alpha_t = 0$ for all t . This means that the original linear combination set to zero is

$$\beta_0 \cdot 1 + \beta_1 X_1 + \cdots + \beta_n X_n = 0.$$

It follows that each $\beta_j = 0$ from the linear independence of $1, X_1, \dots, X_n$ in the vector space of polynomials. ■

8 Club Rules in Other Contexts

Our analysis of Fisher's inequality suggests that club techniques can be extended to more elaborate and applicable configurations of sets. We will describe a family of theorems, all variations of powerful estimates by Ray-Chaudhuri and R.M. Wilson, which synthesize the ideas and methods we have developed to this point.

For the main result of this section, we expand our repertoire to include finite fields of scalars \mathbf{F}_p , the integers modulo a prime p . All of the arithmetic we need to handle scalars for a vector space works for \mathbf{F}_p . Our “clubs” are subsets $A[1], \dots, A[m]$ of a set with n elements. The role of “even” and “odd” is played by an indicator L which consists of s integers. The club rules become

- $\#(A[j])$ is never congruent modulo p to an integer in L .
- If $i \neq j$ then $\#(A[i] \cap A[j])$ is always congruent modulo p to an integer in L .

The goal is to find a good upper bound for the number of possible clubs m , in terms of the parameters n and s . The second club rule should remind you of the two-distances constraint; this time there are s cardinalities instead of two distances. We require that the dot product (rather than distance) between different club vectors in \mathbf{F}_p^n be one of these values.

Represent each $A[j]$ by an incidence vector $v[j]$ in \mathbf{F}_p^n . Let $X_1, \dots, X_n, Y_1, \dots, Y_n$ be variables and then set $X = (X_1, \dots, X_n)$ and $Y = (Y_1, \dots, Y_n)$. Construct a function similar to the two-distance polynomial,

$$F(X, Y) = \prod_{\lambda \in L} (X \cdot Y - \lambda) \in \mathbf{F}_p[X_1, \dots, X_n, Y_1, \dots, Y_n] .$$

Finally, define $f_j(X) = F(X, v[j])$. Club rules amount to the fake dot product properties

$$f_j(v[i]) = \begin{cases} \text{nonzero} & \text{if } i = j; \\ 0 & \text{if } i \neq j. \end{cases}$$

Now we introduce a new trick called “exponent smashing”. It is based on the observation that the club vectors plugged into f_j are quite special; all entries are 0 or 1. Any positive power of 0 is 0 and any power of 1 is 1. So 0 and 1 cannot tell the difference between X_t^d for $d > 0$ and X_t in any polynomial they are being plugged into. If $g(X_1, \dots, X_n)$ is an arbitrary polynomial, the “smashed” polynomial $\tilde{g}(X_1, \dots, X_n)$ is obtained from g by removing every positive exponent from a monomial which appears in g . For example, if

$$g = X_1 X_2^2 + 3X_1^2 X_2 - X_3^7$$

then

$$\tilde{g} = 4X_1 X_2 - X_3 .$$

Notice that \tilde{g} is always in the span of monomials which have the property that each variable appears at most once. Moreover, the construction was designed so that

$$f_j(v[i]) = \tilde{f}_j(v[i])$$

for all choices of i and j .

Theorem 8.1 *Assume that $A[1], \dots, A[m]$ are distinct subsets of a set with n members which satisfy mod p club rules for an indicator set L of cardinality s . Then*

$$m \leq \binom{n}{s} + \binom{n}{s-1} + \dots + \binom{n}{0} .$$

Proof: Using the notation we have already developed, the first step is to find a subspace of polynomials which contains each smashed polynomial $\tilde{f}_1, \dots, \tilde{f}_m$ and is easy to describe. What is the largest number of variables, counting repeats, in a monomial which appears in some f_j ? We get one X_t for each factor in the big product defining F . Hence the number of variables in a monomial is at most s , the size of L . In other words, $\tilde{f}_1, \dots, \tilde{f}_m$ live in the subspace of polynomials with a basis consisting of all

$$X_{h(1)} X_{h(2)} \cdots X_{h(w)}$$

with $h(1) < h(2) < \dots < h(w)$ and $0 \leq w \leq s$. Let's count these basis monomials by length.

There is $1 = \binom{n}{0}$ monomial of length zero, namely 1.

There are $\binom{n}{1}$ monomials of length 1, the variables themselves.

There are $\binom{n}{2}$ monomials of length 2 with two different variables.

Etc.

Thus all of the \tilde{f}_j live in a subspace of dimension

$$\binom{n}{s} + \binom{n}{s-1} + \dots + \binom{n}{0} .$$

We complete the proof in the usual fashion. If

$$\tilde{f}_1, \dots, \tilde{f}_m$$

are linearly independent, we are done. But this follows immediately from the fake dot product perpendicularity after smashing. ■

Before we look at applications of the theorem, it would be nice to get a feel for the magnitude of

$$\binom{n}{s} + \binom{n}{s-1} + \cdots + \binom{n}{0} .$$

We expect s to be much smaller than n , so *let's assume that* $s \leq \frac{1}{4}n$. We compare $\binom{n}{k-1}$ and $\binom{n}{k}$.

$$\binom{n}{k-1} = \frac{k}{n-k+1} \binom{n}{k} \leq \frac{s}{n-s+1} \binom{n}{k}$$

whenever $k \leq s$. Set $\epsilon = \frac{s}{n-s+1}$.

$$4s \leq n \Rightarrow 3s \leq n+1 \Rightarrow s \leq n-2s+1 \Rightarrow \frac{s}{n-2s+1} \leq 1 .$$

In particular, $\epsilon < 1$. Thus

$$\begin{aligned} \binom{n}{s} + \binom{n}{s-1} + \cdots + \binom{n}{0} &\leq (1 + \epsilon + \epsilon^2 + \cdots + \epsilon^s) \binom{n}{s} \\ &< (1 + \epsilon + \epsilon^2 + \cdots) \binom{n}{s} \\ &= \frac{1}{1 - \epsilon} \binom{n}{s} \\ &= \left(1 + \frac{s}{n-2s+1}\right) \binom{n}{s} \\ &\leq 2 \binom{n}{s} . \end{aligned}$$

The first application of this theorem will be to constructive Ramsey theory. To introduce the subject, we begin with a game and a question. Six dots are drawn on a piece of paper. Two players take turns drawing a line which connects two points, with the rules that no one can connect a pair of points already connected and one player uses a red crayon while the other uses blue. Whoever first completes a triangle of his or her own color loses. Can the game ever end in a draw?

You should convince yourself that the answer is “no”. (But try five dots to make sure you see that there is an issue.) We recast the the game as a problem in graph theory. Recall that the *complete graph* on n vertices is the graph with one edge connecting each pair of different vertices. *If we color the 15 edges of the complete graph on 6 vertices red and blue in any way then a monochromatic triangle will necessarily be present.*

The philosophy of Ramsey theory is that any configuration which is large enough must have regular patterns. More precisely, any large configuration which has been partitioned (i.e., colored) has a monochromatic pattern of a certain type. One of the constructive practical issues which comes up is “how large is large”? Here is an example which addresses the question for generalizations of our game.

Proposition 8.1 *Let t be a positive integer. There is a blue and red coloring of all edges in the complete graph on $\binom{t}{3}$ vertices which does not have any monochromatic complete subgraphs on $t + 1$ vertices.*

Proof: The binomial coefficient $\binom{t}{3}$ counts the number of subsets of $\{1, 2, \dots, t\}$ with three elements. So label each vertex in the graph with a different three-element subset, in some fashion. Color the edge joining two vertices red when their labels have exactly one element in common. Color all other edges blue.

What can we say about a complete subgraph on m vertices all of whose edges are red? The list of labels consists of m three-element subsets such that any pair has one element in common. Fisher’s inequality applies! It tells us that $m \leq t$.

What can we say about a complete subgraph on m vertices all of whose edges are blue? The list of labels consists of m three-element subsets such that any pair has an even intersection, namely 0 or 2 elements. Think three member clubs in a town of t people! By the Oddtown problem, again $m \leq t$.

■

If we regard the main theorem of this section as a generalization of the Oddtown solution as well as Fisher’s inequality then we should not be surprised that the proposition belongs to a large family of examples. Let p be a prime number and assume that $n > 2p^2$. Identify each subset of $\{1, 2, \dots, n\}$ containing $p^2 - 1$ elements with a vertex in the complete graph on $\binom{n}{p^2-1}$ vertices. Color the edge joining two vertices labeled by subsets A and B red when

$$\#(A \cap B) \not\equiv -1 \pmod{p} .$$

Color all other edges blue.

Corollary 8.1 *In the complete graph on $\binom{n}{p^2-1}$ vertices with the coloring described above, there is no monochromatic complete subgraph on more than $2 \cdot \binom{n}{p-1}$ vertices.*

Proof: Suppose there is a complete subgraph with red edges which uses m vertices. We regard the labels for vertices in this subgraph as clubs. We apply the main theorem of this section with indicator set

$$L = \{0, 1, \dots, p-2\} .$$

Each label has $p^2 - 1$ elements in it. Thus the number of members in a club is congruent to $p - 1$ modulo p ; this is a value which is not in L . However, the intersection of any two different clubs is definitely in L by the red edge criterion. Using our inequality, we see that $m \leq 2 \cdot \binom{n}{\#(L)}$.

Next, we study a complete subgraph with blue edges and m vertices. If A and B represents different labels for vertices in this subgraph then $\#(A \cap B) \equiv -1 \pmod{p}$. The largest value for the size of the intersection is less than $p^2 - 1$. In other words, for the second case we could use the indicator set

$$L = \{p-1, 2p-1, \dots, p^2-p-1\},$$

a set with $p-1$ members. Notice that the cardinality of a label is $p^2 - 1$ which is not a number on this list. The trick is *not* to use the prime p in applying the main theorem. Instead, pick a prime q larger than $p^2 - 1$. The same inequality as in the first paragraph now follows with $s = p - 1$. ■

9 The Ray-Chaudhuri Wilson Theorem

We are about to prove the last of our linear bound results. The proof we present involves all sorts of techniques we have already seen.

Theorem 9.1 *Let L be a set of s non-negative integers each less than k . Suppose that $A[1], \dots, A[m]$ are distinct subsets of a set with n elements and all of these sets have the same cardinality k . If $\#(A[i] \cap A[j]) \in L$ for all $i \neq j$ then*

$$m \leq \binom{n}{s} .$$

We almost have this result already. Choose a prime p larger than k and apply the main theorem of the previous section. We obtain

$$m \leq 2 \cdot \binom{n}{s},$$

off by a factor of two.

Proof of the theorem: Since $L \subseteq \{0, 1, \dots, k-1\}$ we see that $s \leq k$. We will frequently use the equivalent statement that $s-1 < k$.

For notational simplicity, we will assume that the universe of n elements referred to in the theorem is $\{1, 2, \dots, n\}$. For any subset I let X_I denote the monomial which is the product of those X_j with $j \in I$. For example, $X_{\{1,2,4\}} = X_1 X_2 X_4$ and $X_\emptyset = 1$. Define

$$Q_I(X_1, \dots, X_n) = X_I \left(\sum_{j=1}^n X_j - k \right)$$

and let \tilde{Q}_I be the polynomial obtained from Q_I after exponent smashing. We delay a proof of the following assertion until we are done with the interesting stuff:

$$\{\tilde{Q}_I \mid \#(I) \leq s-1\} \text{ is linearly independent.}$$

We have copied a paragraph from the beginning of the last section nearly verbatim. Represent each $A[j]$ by an incidence vector $v[j]$ in \mathbf{R}^n . Let $X_1, \dots, X_n, Y_1, \dots, Y_n$ be variables and then set $X = (X_1, \dots, X_n)$ and $Y = (Y_1, \dots, Y_n)$. Construct the usual function,

$$F(X, Y) = \prod_{\lambda \in L} (X \cdot Y - \lambda) \in \mathbf{R}[X_1, \dots, X_n, Y_1, \dots, Y_n].$$

Finally, define $f_j(X) = F(X, v[j])$. Club rules amount to the fake dot product properties

$$\tilde{f}_j(v[i]) = \begin{cases} \text{nonzero} & \text{if } i = j; \\ 0 & \text{if } i \neq j. \end{cases}$$

(Here we have used the fact that $F(v[t], v[t]) = \prod_{\lambda \in L} (k - \lambda)$ and $k \notin L$.) We argue that

$$\{\tilde{f}_1, \dots, \tilde{f}_m\} \cup \{\tilde{Q}_I \mid \#(I) \leq s-1\}$$

is a linearly independent set of

$$m + \binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{s-1}$$

polynomials.

Suppose $\alpha_1 \tilde{f}_1 + \cdots + \alpha_m \tilde{f}_m + \sum_I \beta_I \tilde{Q}_I = 0$ for scalars α_j and β_I , where I runs over sets of cardinality less than s . Plug $v[t]$ into each side of the equation. Since $v[t]$ is a vector with entries 0 and 1, the effect of substituting it into \tilde{Q}_I is the same as substituting it in Q_I . But $v[t]$ has precisely k ones so the result of plugging it into $\sum_{j=1}^n X_j - k$ results in zero. In other words, $\tilde{Q}_I(v[t]) = 0$ for all I . Thus we obtain

$$\alpha_t = 0$$

from the original linear combination set equal to zero. It follows that $\sum_I \beta_I \tilde{Q}_I = 0$. We are done: each $\beta_I = 0$ from our as yet unproved linear independence assertion. The count of polynomials is an immediate consequence of there being $\binom{n}{d}$ subsets of $\{1, 2, \dots, n\}$ with cardinality d .

All of the \tilde{f}_j and all of the \tilde{Q}_I live in the subspace of polynomials spanned by monomials of the form $X_{h(1)} X_{h(2)} \cdots X_{h(d)}$ where

$$h(1) < h(2) < \cdots < h(d) \leq s .$$

Hence

$$m + \binom{n}{0} + \cdots + \binom{n}{s-1} \leq \binom{n}{0} + \cdots + \binom{n}{s-1} + \binom{n}{s} .$$

Subtract to bound m . ■

We are left with the claim that the list of \tilde{Q}_I as I varies over all sets I with $\#(I) \leq s-1$ is linearly independent. First, let's take a closer look at \tilde{Q}_I . Recall that

$$Q_I(X_1, \dots, X_n) = X_I(X_1 + \cdots + X_n - k) .$$

If $j \in I$ then $X_I X_j$ becomes X_I after exponent smashing. Hence

$$\tilde{Q}_I(X_1, \dots, X_n) = (\#(I) - k) X_I + \sum_J X_J$$

where J runs over all subsets containing I with $\#(J) = \#(I) + 1$. Moreover, $\#(I) - k \neq 0$ because $\#(I) \leq s - 1 < k$.

The collection of monomials X_I for all possible choices of subsets I is a basis for the space spanned by all smashed polynomials. List them so that if $\#(A) < \#(B)$ we place X_A to the left of X_B . (We don't care about the relative order among those X_C with C of a fixed cardinality.) We shall apply the Replacement Principle to this list repeatedly, going down from large subsets to small.

Let X_M be the right-most monomial on the list with $\#(M) \leq s - 1$. We can replace X_M with \tilde{Q}_M . Move one monomial to the left and repeat. Since we are replacing right to left, the smashed polynomials from \tilde{Q}_N on toward the right in any temporary list is a basis for the subspace spanned by all monomials X_N rightward on the original list. So suppose that X_I is the monomial examined at a given step. Then \tilde{Q}_I is a nonzero scalar multiple of X_I plus a linear combination of monomials with greater length than X_I . Consequently, it is also a nonzero scalar multiple of X_I plus a linear combination of basis polynomials to the right of X_I in the temporary list. Thus we can legally replace X_I with \tilde{Q}_I . When the algorithm stops, the desired list of all \tilde{Q}_I will be a subset of the final list.

10 Bell Lab's Loop Switching Problem

Consider the following communication issue. For a telephone or computer network (i.e., a large graph), a connection between node A and node B is established before any messages are sent. When we do send a message from A to B we would like the address for B to indicate the path by which the message should be routed.

One simple-minded solution proceeds as follows. Assign each vertex an n -bit word and define the *Hamming distance* between two such bit strings to be the number of bits in which the strings differ. Now arrange the addressing so that the distance between any two vertices A and B in the graph (which means the least number of edges in a path from A to B) is the same as the Hamming distance between their addresses. If a message is to be sent from A to B , at each intermediate vertex C the addresses of all adjacent vertices are scanned. The one which has Hamming distance one closer to B than the address of C is the next vertex on the route.

As an example, consider any tree. We describe an addressing scheme inductively. For the tree with two vertices, label one 0 and the other 1.

Assume that we can find an addressing for any tree with k vertices which uses $k - 1$ bits. Suppose now that we have a tree with $k + 1$ vertices. It must have a leaf L , a vertex of degree one. When we remove L and its adjacent edge we obtain a tree with k vertices; address it. For each vertex in this subtree append the bit 0 to the left. Assign L the address of its unique adjacent vertex but make its appended bit 1. It is easy to see that Hamming distance is path distance for vertices in the pruned tree; if one of the vertices is the leaf, notice that any path to the leaf must pass through the unique vertex adjacent to it.

However, there is a fatal flaw in any generalization of this scheme. Consider the complete graph on three vertices, otherwise known as the triangle. Call the vertices A , B , and C . The addresses for A and B must differ in exactly one bit because they are a distance of 1 apart. Let's say A has a 0 in the k^{th} bit and B has a 1 in that bit. The k^{th} bit of C is either 0 or 1. Suppose it is 0. Since it has distance one from A , there is some other bit in which it differs from A . But then C has two bits in which it differs from B . If the k^{th} bit of C is 1 then Obviously, no suitable addressing is possible.

The solution proposed to compensate for this difficulty is to use a "joker" symbol $*$. In other words, the new alphabet is $\{0, 1, *\}$ and we form n -bit addresses in this alphabet. The $*$ -Hamming distance between two such strings is defined to be the number of bits in which one string has a 0 and the other has a 1; the $*$ does not contribute at all to the distance. We say that we have a $*$ -addressing of a graph provided we can assign each vertex an address so that the $*$ -Hamming distance between any two vertices is the same as the distance in the graph. You should check that the triangle has a $*$ -addressing using 2 bits.

Can we $*$ -address any connected graph? Suppose that there are k vertices which are named after the numerals $1, 2, \dots, k$. Imagine an N -bit string partitioned into blocks, one for each two-element subset of vertex names. The block indexed by $\{x, y\}$ has $d(x, y)$ bit positions where $d(-, -)$ denotes path distance. Thus

$$N = \sum_{\{x,y\}} d(x, y) ,$$

the sum taken over all two-element subsets of vertices. To address the vertex named s , put a 1 in every bit of the $\{s, t\}$ -block when $s < t$, put a 0 in every bit of the $\{s, t\}$ -block when $s > t$ and place a $*$ in every other bit of the address. Let's compute the $*$ -Hamming distance between distinct vertices x

and y . The only block in which **both** addresses have entries which are not $*$ is the $\{x, y\}$ -block; the number of differing bits is the size of the block, which is precisely the path distance between x and y .

Now that we know that $*$ -addressings are always possible, we can be more picky. What is the least number of bits we can get away with? For a tree, we required at most one less bit than the number of vertices. Can we do better? We required a ridiculously large number of bits for our general argument. However, for years there was a conjecture that smallest the number of bits required for a $*$ -addressing never exceeds the number of vertices minus 1 for *any* connected graph. Eventually, a clever addressing strategy was found that settled the conjecture positively. We are going to devote some time to examining upper and lower bounds for the smallest number of required bits, looking at various classes of graphs.

As a warm-up exercise, we ask for the least number of bits in a $*$ -addressing of the complete graph on n vertices. In this case, the problem has an interesting translation. Suppose that we are given a k -bit $*$ -addressing. For each j between 1 and k inclusively, we describe the j flavored bipartite subgraph of the complete graph. Its red vertices are those vertices whose j^{th} address bit is 0. Its blue vertices consist of those vertices in the complete graph whose j^{th} address bit is 1. The edges of the subgraph are all possible edges connecting a red and blue vertex. Technically speaking, each flavored subgraph is the complete bipartite graph on some disjoint collection of red and blue vertices. We claim that each edge in the complete graph has exactly one flavor. Indeed, if some edge belongs to subgraphs of two different flavors then the endpoints of the edge have $*$ -addresses with $*$ -Hamming distance at least 2 apart. But the path distance between any two distinct vertices in the complete graph is 1. Does every edge have a flavor? Again, the answer is “yes” because the $*$ -Hamming distance between any two different vertices must be 1. We may rephrase our flavor decomposition as follows. Every $*$ -addressing of a complete graph which requires k bits gives rise to an edge-disjoint partition of the graph into k complete bipartite subgraphs.

Conversely, suppose that a complete graph can be edge partitioned into m complete bipartite subgraphs. We can build a $*$ -addressing with m bits. List the subgraphs $1, 2, \dots, m$. The j^{th} bit of the address of a vertex is 0 when the vertex is red in the j^{th} subgraph, is 1 when the vertex is blue in the j^{th} subgraph, and is $*$ in all other cases. We leave it to the reader to check that the $*$ -Hamming distance between any two distinct vertices is 1.

With this equivalent description of $*$ -addressing, we are ready to find the minimum number of bits required to handle a complete graph. However, we

first need a refinement of linear dependence.

Lemma 10.1 *Suppose that $v[1], \dots, v[n]$ are vectors in some space such that any $n - 1$ of them are linearly dependent. Then it is possible to find scalars $\alpha_1, \dots, \alpha_n$ which are not all zero such that $\sum \alpha_j v[j] = 0$ and $\sum \alpha_j = 0$.*

Proof: We are assuming that the span of $v[1], \dots, v[n]$ has dimension at most $n - 2$. Thus we may renumber so that

$$v[n] = \beta_1 v[1] + \dots + \beta_{n-2} v[n-2] \quad \text{and} \quad v[n-1] = \gamma_1 v[1] + \dots + \gamma_{n-2} v[n-2]$$

for scalars β_i and γ_j . Set

$$b = \beta_1 + \dots + \beta_{n-2} - 1 \quad \text{and} \quad c = \gamma_1 + \dots + \gamma_{n-2} - 1 .$$

If $b = 0$ we use the linear combination

$$\beta_1 v[1] + \dots + \beta_{n-2} v[n-2] + (-1)v[n]$$

to complete the lemma. If $b \neq 0$, consider

$$c(\beta_1 v[1] + \dots + \beta_{n-2} v[n-2] - v[n]) - b(\gamma_1 v[1] + \dots + \gamma_{n-2} v[n-2] - v[n-1]) = 0 .$$

The sum of the coefficients is $cb - bc = 0$. Moreover, the coefficient of $v[n-1]$ is nonzero. ■

Theorem 10.1 *The minimum number of pairwise edge-disjoint complete bipartite subgraphs in a partition of the complete graph on n vertices is $n - 1$.*

Proof: We first produce a partition into $n - 1$ subgraphs inductively. The case $n = 2$ is trivial. Assume that we know how to partition the complete graph on m vertices. The complete graph on $m + 1$ vertices can be obtained by appending one new vertex and m edges, one from the new vertex to each old vertex. But this describes a single complete bipartite graph which is disjoint from the old graph. (You should write down some of the *addressings which arise from this procedure.)

The more challenging portion of the proof requires us to show that we cannot do better than $n - 1$. This involves yet another linear bound estimate, using the lemma. Suppose that the complete graph on n vertices is partitioned into k complete bipartite subgraphs; number them $1, \dots, k$. We first define the j^{th} red vector $r[j]$ in \mathbf{R}^n as the incidence vector with

a 1 in every coordinate indexed by a red vertex in the j^{th} subgraph and 0 elsewhere. For $t = 1, \dots, n$, then set

$$v[t] = \sum_{\text{blue } t \text{ in } j} r[j] ,$$

where the sum is taken over all j such that vertex numbered t is a blue vertex in the j^{th} subgraph. In words, $v[t]$ is the incidence vector which records all red vertices adjacent to t in some bipartite subgraph. Of course, there is only one flavor per edge. So $v[t]$ is a vector whose p th coordinate is either 0 or 1. That coordinate is one if and only if the unique flavored edge between t and p has a blue t connected to a red p .

Our subgoal is to show that it is possible to find $n-1$ linearly independent vectors on the list

$$v[1], \dots, v[n] .$$

What would this tell us? Since the $v[t]$ are linear combinations of red vectors, there must be $n-1$ linearly independent red vectors. On the other hand, there are only k red vectors available. Thus $k \geq n-1$, just what we are hoping for.

We argue by contradiction. Suppose there are not $n-1$ linearly independent vectors on the list. The lemma implies that there exist scalars $\alpha_1, \dots, \alpha_n$ which are not all zero and satisfy both $\sum \alpha_i = 0$ and

$$\alpha_1 v[1] + \dots + \alpha_n v[n] = 0 .$$

We take the last equality and write it out coordinate-wise in a table. As usual, each entry in the bottom row is a complicated expression for zero.

$$\begin{array}{cccc} \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \sum_j \alpha_j v_1[j] & \sum_j \alpha_j v_2[j] & \cdots & \sum_j \alpha_j v_n[j] \end{array}$$

When we take dot product of the rows and organize the sums in an array, we obtain

$$\begin{array}{ccccccc} & \alpha_1 \alpha_1 v_1[1] & + & \alpha_1 \alpha_2 v_1[2] & + & \cdots & + & \alpha_1 \alpha_n v_1[n] \\ + & \alpha_2 \alpha_1 v_2[1] & + & \alpha_2 \alpha_2 v_2[2] & + & \cdots & + & \alpha_2 \alpha_n v_2[n] \\ & \cdot & & & & & & \cdot \\ & \cdot & & & & & & \cdot \\ & \cdot & & & & & & \cdot \\ + & \alpha_n \alpha_1 v_n[1] & + & \alpha_n \alpha_2 v_n[2] & + & \cdots & + & \alpha_n \alpha_n v_n[n] \end{array}$$

Now add down the rows to get

$$(\diamond) \sum_{j=1}^n \alpha_j (\alpha_1 v_1[j] + \alpha_2 v_2[j] + \cdots + \alpha_n v_n[j]) = 0 .$$

We interpret $\alpha_1 v_1[j] + \cdots + \alpha_n v_n[j]$ combinatorially. It can be rewritten $\sum \alpha_x$ where x runs over all red vertices connected to a blue j . What can we say about a vertex y for which α_y is missing from the sum? It can be j itself because the graph has no loops. Otherwise, we recall that j is connected by an edge to *every* other vertex so a remaining vertex y would have to be blue and j red in a bipartite subgraph. To summarize

$$\alpha_1 v_1[j] + \cdots + \alpha_n v_n[j] = \sum_{\text{all } d} \alpha_d - \alpha_j - \sum_{\text{blue } y \text{ adjacent to red } j} \alpha_y .$$

The first sum to the right of the equal sign is 0. The last sum is

$$\sum_{y=1}^n \alpha_y v_j[y] .$$

It is also 0 because it displays the j th coordinate of a linear combination which vanishes. Thus formula (\diamond) from the dot product of the two rows described above becomes

$$\sum_{j=1}^n -\alpha_j^2 = 0 .$$

We conclude that $\alpha_j = 0$ for all choices of j . ■

11 Encoding With Quadratic Forms

We would like to turn the problem of finding the least number of bits for a *-addressing of a particular graph into a mechanical algebraic problem. The following idea was developed in Bell Labs. Label the vertices of a graph $1, 2, \dots, n$ and choose variables X_1, X_2, \dots, X_n . Define the distance from to be

$$\sum_{\{i,j\}} (\text{path distance from vertex } i \text{ to } j) X_i X_j$$

where the sum is taken over all two-element subsets of vertices. For example, the complete graph on three vertices has distance form

$$X_1 X_2 + X_2 X_3 + X_1 X_3$$

while if we remove the edge between vertex 1 and vertex 3 we have the form

$$X_1X_2 + X_2X_3 + 2X_1X_3 .$$

Now suppose that the graph has a *-addressing. To illustrate the effect on our form, let's assume there are addresses

$$\text{vertex } i : 1 \ 1 \ 0 \ * \ 0 \ 1 \ *$$

$$\text{vertex } j : 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1$$

The *-Hamming distance between them is three and must coincide with the path distance. Notice that the coefficient 3 for X_iX_j is contributed by the first, third, and fifth columns. This suggests that we can keep account of distance contributions one column at a time. So imagine that the addresses for all of the vertices are listed, giving us a matrix of 0's, 1's, and *'s. The first column contribution to the distance form is

$$\left(\sum X_{s(1)}\right)\left(\sum X_{t(1)}\right)$$

where the first sum is taken over those $s(1)$ which account for a 1 in the $s(1)$ -entry of the first column while the second sum is taken over all coordinates $t(1)$ in which there is a 0 in the first column. Thus the entire distance form has the shape

$$\sum_{j=1}^k \left(\sum X_{s(j)}\right)\left(\sum X_{t(j)}\right)$$

so that the number k of summands is the number of bit positions and for each fixed j , no $s(j)$ can also be a $t(j)$.

For example, the bipartite partition of the complete graph on four vertices yields the *-addressing

$$\begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ * & 1 & 0 \\ * & * & 1 \end{pmatrix}$$

We can write its distance form consistent with the addressing as

$$(X_2)(X_1) + (X_3)(X_1 + X_2) + (X_4)(X_1 + X_2 + X_3) .$$

The trick is to apply the “difference of two squares” identity to make a change of variables so that sums of squares appear. The relevant high school algebra is

$$(a + b)(a - b) = a^2 - b^2 .$$

If we set $u = a + b$ and $v = a - b$ then

$$uv = \left(\frac{1}{2}u + \frac{1}{2}v\right)^2 - \left(\frac{1}{2}u - \frac{1}{2}v\right)^2.$$

In our particular case, u and v will each be sums of different variables so neither $\frac{1}{2}u + \frac{1}{2}v$ nor $\frac{1}{2}u - \frac{1}{2}v$ can be zero. (A linear combination of variables is called a *linear form*. We always get two squares of linear forms for each term in the expression obtained from the address matrix.) Thus given a *-addressing, the distance form can be made to look like

$$\sum (\text{nonzero linear form})^2 - \sum (\text{linear linear form})^2$$

where the number of terms in each sum coincides and equals the number of bit positions.

Just where do sums of squares fit into our mathematical universe? They appear in the Euclidean distance formula which just happens to be derived from the dot product. We begin by placing this relationship in a formal context.

Definition 11.1 *Let V be a real vector space. A symmetric bilinear form $B : V \times V \rightarrow \mathbf{R}$ is a function such that*

- $B(v, w) = B(w, v)$;
- $B(\lambda v, w) = \lambda B(v, w)$; and
- $B(v + v', w) = B(v, w) + B(v', w)$

for all $v, v', w \in V$ and all scalars λ .

For instance, the dot product defines a bilinear form via $(v, w) \mapsto v \cdot w$. If B is a symmetric bilinear form, we can assign it a *quadratic form* Q much the same way we associate distance to the dot product. If $v \in V$ we set $Q(v) = B(v, v)$. There is an alternate way to view Q . Suppose that $\{e_1, \dots, e_n\}$ is a fixed basis for V . A “generic” vector looks like

$$X_1e_1 + \dots + X_n e_n$$

so that the generic formula for the quadratic form becomes

$$Q(X_1e_1 + \dots + X_n e_n) = B\left(\sum_i X_i e_i, \sum_j X_j e_j\right) = \sum_{i,j} B(e_i, e_j) X_i X_j$$

for scalars $B(e_i, e_j)$. To simplify notation, we will abbreviate

$$Q(X_1 e_1 + \cdots + X_n e_n) = Q(X_1, \dots, X_n) .$$

In the case of the dot product and the standard basis, we obtain the familiar formula

$$Q(X_1, \dots, X_n) = X_1^2 + \cdots + X_n^2 .$$

Conversely, it is possible to realize every expression which looks like a distance form,

$$G(X_1, \dots, X_n) = \sum_{i \leq j} g_{ij} X_i X_j$$

for real numbers g_{ij} , as the quadratic form associated to some symmetric bilinear form relative to the standard basis on \mathbf{R}^n . As a simple example, suppose that $G(X_1, X_2) = X_1 X_2$. We are looking for a bilinear form B on \mathbf{R}^2 so that

$$B(X_1 e_1 + X_2 e_2, X_1 e_1 + X_2 e_2) = X_1 X_2 .$$

Here $e_1 = (1, 0)$ and $e_2 = (0, 1)$. A simple calculation shows that

$$B(e_1, e_1) = B(e_2, e_2) = 0 \quad \text{and} \quad B(e_1, e_2) + B(e_2, e_1) = 1 .$$

Since B is symmetric, we must have $B(e_1, e_2) = \frac{1}{2}$. Putting everything together, we have

$$B((a_1, a_2), (b_1, b_2)) = \frac{1}{2} a_1 b_2 + \frac{1}{2} a_2 b_1 .$$

The general picture should now be obvious. Given G as above, define the symmetric form B by

$$B(e_i, e_j) = \begin{cases} g_{ii} & \text{if } i = j \\ \frac{1}{2} g_{ij} & \text{if } i \neq j \end{cases} .$$

Then G is the associated quadratic form for B .

12 Sylvester's Law of Inertia

The basic result we need is a dusty old theorem of nineteenth century mathematics. It will allow us to capture the number of summands (i.e., bits) in our sums and differences of the squares of linear forms.

Theorem 12.1 (Sylvester’s Law of Inertia) *Assume that*

$$e_1, \dots, e_p, e_{p+1}, \dots, e_{p+q}, e_{p+q+1}, \dots, e_n$$

is a basis for an n -dimensional real vector space and assume that B is a symmetric bilinear form on the space so that the associated generic quadratic form with respect to this basis is

$$Q(X_1, \dots, X_n) = X_1^2 + \dots + X_p^2 - X_{p+1}^2 - \dots - X_{p+q}^2 .$$

If the quadratic form can also be written

$$Q(X_1, \dots, X_n) = L_1^2 + \dots + L_c^2 - M_1^2 - \dots - M_d^2$$

for linear forms $L_1, \dots, L_c, M_1, \dots, M_d$ which are not necessarily linearly independent then

$$c \geq p \text{ and } d \geq q .$$

Before jumping into a proof, we discuss a strategy for using this remarkable theorem. Let’s indulge in some wishful thinking. Suppose Sylvester’s Theorem said the following: If D is the distance form for a graph, there are magic numbers p and q such that whenever D can be written

$$D(X_1, \dots, X_n) = L_1^2 + \dots + L_c^2 - M_1^2 - \dots - M_d^2$$

for linear forms L_i, M_j then

$$c \geq p \text{ and } d \geq q.$$

The *-addressing gives us one choice of L_i ’s and M_j ’s where $c = d = k$, the number of bits for an address. Thus the “theorem” would tell us that

$$k \geq \max\{p, q\}.$$

Any *-addressing of the graph would require at least $\max\{p, q\}$ bits.

Unfortunately, the distance form doesn’t look anything like

$$X_1^2 + \dots + X_p^2 - X_{p+1}^2 - \dots - X_{p+q}^2.$$

Nonetheless, Sylvester’s Theorem does leave us some room in which to maneuver. We can regard the distance form as an associated generic quadratic form for some bilinear form, where we write out a generic vector using the

standard basis. Suppose we can find a *better* basis with respect to which the associated generic quadratic form, D_{better} , has the shape required by Sylvester. If the bit equation transforms to

$$D_{\text{better}} = \sum_{j=1}^k (\text{new linear forms})^2 - \sum_{j=1}^k (\text{other new linear forms})^2$$

then we have our p and q as well as a bound for k .

Now that we have a strategy which justifies learning Sylvester's Theorem, we turn to a proof of the result.

Lemma 12.1 *Let v_1, \dots, v_n be a basis for a vector space Y . Fix scalars b_1, \dots, b_n and set*

$$Y' = \left\{ \sum r_j v_j \in Y \mid r_1 b_1 + \dots + r_n b_n = 0 \right\} .$$

Then Y' is a subspace of Y whose dimension is at least $n - 1$.

Proof: It is easy to check that Y' is a subspace. It is even easier to see that if all of the scalars b_j are zero then Y' is all of Y ; in this case the dimension of Y' is n .

So assume that at least one of the fixed scalars is nonzero. Renumbering, we may suppose that $b_1 \neq 0$. Any vector $v \in Y$ can be written

$$v = a_1 v_1 + \dots + a_n v_n .$$

A straightforward calculation then shows that

$$v - \left(\frac{\sum_j a_j b_j}{b_1} \right) v_1 \in Y' .$$

Thus if we choose a basis for Y' then every vector $v \in Y$ can be spanned by the members of this basis along with v_1 . Thus

$$n = \dim Y \leq 1 + \dim Y' . \blacksquare$$

Lemma 12.2 *Let V and W be subspaces of the n -dimensional vector space Z . If*

$$\dim V + \dim W > n$$

then there is a nonzero vector which is simultaneously in V and W .

Proof: Let e_1, \dots, e_p be a basis for V and let f_1, \dots, f_q be a basis for W . Then

$$e_1, \dots, e_p, f_1, \dots, f_q$$

cannot be linearly independent. (This is an application of ‘maximal is maximum’ for independent sets. How?) Thus there are scalars $\lambda_1, \dots, \lambda_p$ and μ_1, \dots, μ_q such that at least one λ_i and at least one μ_j are nonzero and

$$\lambda_1 e_1 + \dots + \lambda_p e_p + \mu_1 f_1 + \dots + \mu_q f_q = 0.$$

It follows that

$$\lambda_1 e_1 + \dots + \lambda_p e_p = -\mu_1 f_1 - \dots - \mu_q f_q \in V \cap W$$

is the desired nonzero vector. ■

We can now complete the proof of Sylvester’s Theorem. We are given a generic quadratic form Q written in two ways as a sum and difference of linear forms squared. The underlying basis used in the argument is e_1, \dots, e_n . If the conclusion fails then either $c < p$ or $d < q$. We will argue that the first possibility does not occur; the second case is handled similarly. Write out the linear form

$$L_h = \sum_{j=1}^n b_{hj} X_j$$

and set

$$W = \left\{ \sum_{i=1}^n r_i e_i \mid r_1 b_{h1} + r_2 b_{h2} + \dots + r_n b_{hn} = 0 \text{ for } h = 1, \dots, c \right\}.$$

There are c linear constraints on W . By repeated use of the first lemma we see that W is a subspace and $\dim W \geq n - c$. Set V to be the span of e_1, \dots, e_p . Then

$$\dim V + \dim W \geq p + n - c > n,$$

so the second lemma applies. There is a common nonzero vector $z \in V \cap W$.

On one hand, we can write $z = m_1 e_1 + \dots + m_p e_p$ with some coefficient nonzero. It follows that

$$B(z, z) = m_1^2 + \dots + m_p^2 > 0.$$

On the other hand, we can also write $z = r_1 e_1 + \dots + r_n e_n$ with $\sum r_i b_{hi} = 0$ for all $h = 1, \dots, c$. This means that when we substitute $X_i \mapsto r_i$ in L_h we

get zero. Thus if we make this substitution into the entire L - M expression for Q we obtain

$$B(z, z) = -(M_1(r_1, \dots, r_n))^2 - \dots - (M_d(r_1, \dots, r_n))^2 \leq 0 .$$

The two inequalities are incompatible. ■

13 Change of Variables and Sums of Squares

We already have some experience building a new vector space basis from an old one, thanks to the Replacement Theorem. We'd like to know how this process affects quadratic forms. To be specific, suppose that

$$\mathbf{e} : e_1, e_2, \dots, e_n$$

is the starting basis for \mathbf{R}^n (e.g., the standard basis) and

$$\mathbf{f} : f_1, f_2, \dots, f_n$$

is a second basis. If B is a symmetric bilinear form on \mathbf{R}^n then it can be associated with two quadratic forms,

$$\begin{aligned} Q_{\mathbf{e}}(X_1, \dots, X_n) &= B\left(\sum X_i e_i, \sum X_j e_j\right) \text{ and} \\ Q_{\mathbf{f}}(Y_1, \dots, Y_n) &= B\left(\sum Y_i f_i, \sum Y_j f_j\right) . \end{aligned}$$

We will say that these two quadratic forms are *equivalent*.

If we know that

$$Q_{\mathbf{e}}(X_1, \dots, X_n) = \sum_{i \leq j} g_{ij} X_i X_j$$

we would like to quickly calculate $Q_{\mathbf{f}}$. Each vector in \mathbf{f} is a linear combination of starting basis vectors, say $f_j = \sum_i \alpha_{ij} e_i$. Then

$$\begin{aligned} Q_{\mathbf{f}}(Y_1, \dots, Y_n) &= B\left(\sum Y_i f_i, \sum Y_j f_j\right) \\ &= B\left(\sum_{i,s} \alpha_{si} Y_i e_s, \sum_{j,t} \alpha_{tj} Y_j e_t\right) \\ &= B\left(\sum_s \left(\sum_i \alpha_{si} Y_i\right) e_s, \sum_t \left(\sum_j \alpha_{tj} Y_j\right) e_t\right) \\ &= Q_{\mathbf{e}}\left(\sum_i \alpha_{1i} Y_i, \sum_i \alpha_{2i} Y_i, \dots, \sum_i \alpha_{ni} Y_i\right) \\ &= \sum_{s \leq t} g_{st} \left(\sum_i \alpha_{si} Y_i\right) \left(\sum_i \alpha_{ti} Y_i\right) . \end{aligned}$$

In other words, take the expression G and replace each X_j with

$$\alpha_{j1}Y_1 + \alpha_{j2}Y_2 + \cdots + \alpha_{jn}Y_n .$$

This process has a direct application to the distance form in the presence of a $*$ -addressing. Suppose that $D(X_1, \dots, X_n)$ is the distance form for a connected graph. Assume that, by virtue of an addressing, it can be written

$$D(X_1, \dots, X_n) = L_1(X_1, \dots, X_n)^2 + \cdots + L_b(X_1, \dots, X_n)^2 \\ - M_1(X_1, \dots, X_n)^2 - \cdots - M_b(X_1, \dots, X_n)^2$$

for linear forms L_i, M_j and bit length b . If we regard D as a quadratic form relative to the standard basis then relative to any other basis \mathbf{f} , the expression for D is transformed to

$$D'(Y_1, \dots, Y_n) = L'_1(Y_1, \dots, Y_n)^2 + \cdots + L'_b(Y_1, \dots, Y_n)^2 \\ - M'_1(Y_1, \dots, Y_n)^2 - \cdots - M'_b(Y_1, \dots, Y_n)^2$$

where $L'_j(Y_1, \dots, Y_n)$ (and, similarly, M'_j) is another linear form given by

$$L'_j(Y_1, \dots, Y_n) = L_j\left(\sum_t \alpha_{t1}Y_t, \dots, \sum_t \alpha_{tn}Y_t\right) .$$

So changing the basis does not affect the ability to write the quadratic form in this special way. (Here is an exercise in keeping track of subscripts: If

$$c_1X_1 + \cdots + c_nX_n$$

is a *nonzero* linear form then after replacing each X_j with the appropriate linear combination of Y_1, \dots, Y_n the transformed linear form is nonzero as well. The argument will require you to use the fact that e_1, \dots, e_n are linearly independent.) The upshot is that we may estimate b after we have transformed the distance form so that the hypothesis of Sylvester's Law of Inertia holds. So our next goal is to try to find a particularly good basis so that a quadratic form G can be transformed to one which looks like

$$Y_1^2 + \cdots + Y_p^2 - Y_{p+1}^2 - \cdots - Y_{p+q}^2 .$$

It turns out that this problem has a long history and many different sorts of solutions. We will discuss several of these, but not in complete detail. The first method we look at is "completing the square", which is

sometimes attributed to the Babylonians. We analyze an example and leave a general algorithm as a long exercise.

We begin with the quadratic form

$$2X_1X_2 + 4X_1X_3 - 2X_2^2 - 8X_3^2 .$$

We would like to have a square involved in the first variable, so we make the obvious change of basis $f_1, f_2, f_3 \mapsto f_2, f_1, f_3$. The quadratic form after change of variable is

$$-2X_1^2 + 2X_1X_2 + 4X_2X_3 - 8X_3^2 .$$

Now look at all terms with an X_1 and force the appearance of a perfect square.

$$\begin{aligned} & -2(X_1^2 - X_1X_2) + 4X_2X_3 - 8X_3^2 \\ = & -2(X_1^2 - X_1X_2 + \frac{1}{4}X_2^2) + \frac{1}{2}X_2^2 + 4X_2X_3 - 8X_3^2 \\ = & -2(X_1 - \frac{1}{2}X_2)^2 + \frac{1}{2}X_2^2 + 4X_2X_3 - 8X_3^2 . \end{aligned}$$

We now use the full force of our formula for change of variables. If the basis at this point is f_1, f_2, f_3 then we'd like a new basis f'_1, f'_2, f'_3 with $f'_j = \sum_i \alpha_{ij}f_i$ so that the quadratic form becomes

$$-2Y_1^2 + \frac{1}{2}Y_2^2 + 4Y_2Y_3 - 8Y_3^2 .$$

The requisite change of variables is

$$\begin{aligned} X_1 & \mapsto Y_1 + \frac{1}{2}Y_2 \\ X_2 & \mapsto Y_2 \\ X_3 & \mapsto Y_3 . \end{aligned}$$

This means

$$\begin{aligned} \alpha_{11} &= 1; & \alpha_{12} &= \frac{1}{2}; & \alpha_{13} &= 0 \\ \alpha_{21} &= 0; & \alpha_{22} &= 1; & \alpha_{23} &= 0 \\ \alpha_{31} &= 0; & \alpha_{32} &= 0; & \alpha_{33} &= 1 \end{aligned}$$

Thus $f'_1 = f_1, f'_2 = \frac{1}{2}f_1 + f_2, f'_3 = f_3$. According to the Replacement Principle, f'_1, f'_2, f'_3 is indeed a basis. After changing Y 's back to X 's the quadratic

form becomes

$$\begin{aligned}
 & -2X_1^2 + \frac{1}{2}X_2^2 + 4X_2X_3 - 8X_3^2 \\
 = & -2X_1^2 + \frac{1}{2}(X_2^2 + 8X_2X_3 + 16X_3^2) - 16X_3^2 \\
 = & -2X_1^2 + \frac{1}{2}(X_2 + 4X_3)^2 - 16X_3^2 .
 \end{aligned}$$

After a second change of variables similar to the first, we get the new form

$$\begin{aligned}
 & -2X_1^2 + \frac{1}{2}X_2^2 - 16X_3^2 \\
 = & -(\sqrt{2}X_1)^2 + \left(\frac{1}{\sqrt{2}}X_2\right)^2 - (4X_3)^2 .
 \end{aligned}$$

Finally, make the change of basis (including renumbering) to arrive at

$$X_1^2 - X_2^2 - X_3^2 .$$

This example reflects a general procedure except for one peculiarity. It is possible that one cannot begin the algorithm. What do you do if there are no squares in sight, e.g., the given quadratic form is

$$3X_1X_2 - X_2X_3 ?$$

In this case, we reverse the difference-of-two-squares trick. For the example, we'd like

$$X_1 \mapsto Y_1 + Y_2; \quad X_2 \mapsto Y_1 - Y_2; \quad X_3 \mapsto Y_3 .$$

You should check that this change of variables really does come from a change of basis.

As an exercise, consider the distance form for the Y-shaped tree on four vertices,

$$X_1X_2 + X_2X_3 + X_2X_4 + 2X_1X_3 + 2X_1X_4 + 2X_3X_4 .$$

Iteratively complete squares to produce a quadratic form which looks like $\sum \pm Y_j^2$. Then estimate the minimum number of bits in a *-addressing using Sylvester's Law. We will refine this example in the next section.

14 Addressing Trees

After all of our work on quadratic forms, we are finally prepared to establish that the minimum number of bits required for a *-addressing of a tree with n vertices is, as expected, $n - 1$.

Start with a tree T that has n vertices. Set $T_n = T$ and pick a leaf P_n of T_n . We now prune the tree inductively. In general, T_j is obtained from T_{j+1} by removing the leaf P_{j+1} with its incident edge. Then identify a leaf P_j of T_j . When we are done, the vertices of T are numbered P_1, P_2, \dots, P_n .

Denote by $e(j)$ the number such that $P_{e(j)}$ is the vertex in T_j adjacent to the designated leaf P_j . Certainly $e(j) < j$. Last but not least, we'll write $D_j(X_1, \dots, X_j)$ for the distance form associated to T_j .

Lemma 14.1 *For each $0 \leq k \leq n - 1$ the quadratic form D_n is equivalent to*

$$D_{n-k}(X_1, \dots, X_{n-k}) + \left(\sum_{p=1}^{n-k} X_p \right) \left(\sum_{q=n-k+1}^n X_q \right) - \sum_{s=n-k+1}^n X_s^2.$$

Proof: We argue by induction on k . For the case $k = 0$, the last two terms are zero because the sums don't exist. Assume the forms are equivalent for $k = m$. We first identify all terms in the expansion of D_{n-m} which involve either X_{n-m} or $X_{e(n-m)}$. Let's abbreviate $e(n-m)$ as e and write d_i for the distance from P_i to P_e in T_{n-m} . The terms in question come in three varieties:

$$\begin{aligned} d_i X_i X_e & \text{ for } i \neq e \text{ and } i < n - m \\ (d_i + 1) X_i X_{n-m} & \text{ for } i \neq e \text{ and } i < n - m \\ X_e X_{n-m} & \end{aligned}$$

The presence of $d_i + 1$ is due to the fact that a minimal path in T_{n-m} from P_i to P_{n-m} must pass through P_e .

We will make the change of variable $Y_e = X_e + X_{n-m}$, which comes from a change of basis. The sum of the terms we have listed above equals the sum of terms

$$\begin{aligned} d_i X_i Y_e & \text{ for } i \neq e \text{ and } i < n - m \\ X_i X_{n-m} & \text{ for } i \neq e \text{ and } i < n - m \\ Y_e X_{n-m} - X_{n-m}^2 & \end{aligned}$$

Next, we record the effect of this change of variable on $\sum_{p=1}^{n-m} X_p$; replace X_e with Y_e and subtract X_{n-m} . Now carefully *rename* Y_e as X_e to see that

the expression we started with,

$$D_{n-m} + \left(\sum_{p \leq n-m} X_p \right) \left(\sum_{q > n-m} X_q \right) - \sum_{s > n-m} X_s^2$$

is equivalent to

$$\begin{aligned} D_{n-m-1} + \sum_{i \leq n-m-1} X_i X_{n-m} - X_{n-m}^2 \\ + \left(\sum_{p \leq n-m-1} X_p \right) \left(\sum_{q > n-m} X_q \right) - \sum_{s > n-m} X_s^2. \end{aligned}$$

A bit of algebra shows that this last quadratic form equals

$$D_{n-m-1} + \left(\sum_{p \leq n-m-1} X_p \right) \left(\sum_{q > n-m-1} X_q \right) - \sum_{s > n-m-1} X_s^2. \quad \blacksquare$$

Theorem 14.1 *If T is any tree with $n \geq 2$ vertices then the distance form for T is equivalent to*

$$Z_1^2 - Z_2^2 - Z_3^2 - \dots - Z_n^2.$$

Proof: Recall that $D_1 = 0$. As a consequence, the lemma for $k = n - 1$ tells us that the distance form for T is equivalent to

$$X_1 X_2 + X_1 X_3 + \dots + X_1 X_n - X_2^2 - X_3^2 - \dots - X_n^2,$$

Make the change of variables

$$Z_j = X_j - \frac{1}{2} X_1$$

for $j = 2, \dots, n$. We are, of course, completing the square:

$$-X_1 X_j + X_j^2 = \left(X_j - \frac{1}{2} X_1 \right)^2 - \frac{1}{4} X_1^2.$$

Thus the distance form is equivalent to

$$\frac{n-1}{4} X_1^2 - Z_2^2 - Z_3^2 - \dots - Z_n^2.$$

Finally, set $Z_1 = \frac{1}{2} \sqrt{n-1} X_1$. \blacksquare

Corollary 14.1 *The minimum number of bits required to *-address any tree with n vertices is $n - 1$, \blacksquare*

15 Orthonormal Bases Again

We step back for a moment and review the sum-and-difference of squares problem. A bilinear form B on a finite-dimensional real vector space is given. A quadratic form Q_e is constructed relative to a basis $\mathbf{e} : e_1, \dots, e_n$. We are looking for a better basis. In fact, it is enough to find a basis $\mathbf{f} : f_1, \dots, f_n$ such that $B(f_i, f_j) = 0$ for $i \neq j$. For in this situation, Q_f is a linear combination of squares. By taking the square root of the absolute value of the coefficients and making the change variables seen in the last step of the main example in the previous section, we arrive at a quadratic form $\sum X_i^2 - \sum X_j^2$.

The requirement that $B(f_i, f_j) = 0$ for $i \neq j$ should remind us of the perpendicularity which appeared in Oddtown and Eventown Rules. In some sense, the search for a better basis amounts to finding a scheme which makes a basis of vectors perpendicular. For two vectors in Euclidean space there is a simple geometric procedure. If v and w are vectors which are not scalar multiples of one another then v and

$$w - (\text{the perpendicular projection of } w \text{ on } v)$$

are orthogonal. Let's pursue this formally.

Lemma 15.1 (Orthogonal Projection) *Assume B is a bilinear form on the finite-dimensional vector space V . If $v, w \in V$ and $B(v, v) \neq 0$ then there exists a scalar α such that*

$$B(w - \alpha v, v) = 0 .$$

Consequently, the span of v and w is the same as the span of $w - \alpha v$ and v , where the latter two vectors are orthogonal with respect to B .

Proof: We want $B(w, v) - \alpha B(v, v) = 0$. Solve!

$$\alpha = \frac{B(w, v)}{B(v, v)} . \blacksquare$$

We will “construct” a basis for the symmetric bilinear form B on the finite-dimensional real vector space V with the property that any two distinct basis vectors are orthogonal with respect to B . If $B(v, w) = 0$ for every choice of $v, w \in V$ then any basis will do. Assume otherwise. To apply the

lemma, we need a vector z with $B(z, z) \neq 0$. Choose any two vectors v and w with $B(v, w) \neq 0$. If either $B(v, v)$ or $B(w, w)$ are nonzero we have found z . If both values are zero then $B(v + w, v + w) = 2B(v, w) \neq 0$ so choose $z = v + w$.

Since z is not zero, we may use it as the first member of a basis for V , say z, v_2, \dots, v_n . We next apply the Replacement Principle many times so that for each $j \geq 2$, the basis vector v_j is replaced with

$$v'_j = v_j - \frac{B(v_j, z)}{B(z, z)} z .$$

The improved basis z, v'_2, \dots, v'_n has the property that $B(z, v'_j) = 0$ for all $j \geq 2$. Set W to be the span of v'_2, \dots, v'_n . Then W has smaller dimension than V and $B(z, w) = 0$ for all $w \in W$. Inductively find an orthogonal basis w_2, \dots, w_n for W . Then

$$z, w_2, \dots, w_n$$

is an orthogonal basis for V .

We urge the reader to try this second algorithm on the Y-shaped tree used in a similar exercise in the last section. For hand calculations, it can be helpful to remember that the orthogonality of a set of vectors is unaffected by scalar multiplication of any vectors on the list; repeated application of this observation eliminates unsightly fractions.

In the special case of the dot product, this procedure can be modified to turn any basis for V into an orthonormal basis. The algorithm, known as Gram-Schmidt, is quite prominent. Recall that an orthonormal basis, relative to the dot product, is a basis f_1, \dots, f_n such that

$$f_i \cdot f_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

The first requirement is usually achieved by “normalization”: if v is any nonzero vector then

$$\frac{1}{\|v\|} v \cdot \frac{1}{\|v\|} v = 1$$

because $v \cdot v = \|v\|^2$.

Here is one version of Gram-Schmidt. Start with any basis

$$v_1, \dots, v_n$$

for a real vector space. Let w_1 be the normalization of v_1 . By the Replacement Principle,

$$w_1, v_2, \dots, v_n$$

is a basis. Apply the Orthogonal Projection Lemma by setting

$$v'_2 = v_2 - (v_2 \cdot w_1)w_1$$

and normalize to obtain w_2 . Use the Replacement Principle again to see that

$$w_1, w_2, v_3, \dots, v_n$$

is a basis with $w_1 \cdot w_1 = w_2 \cdot w_2 = 1$ and $w_1 \cdot w_2 = 0$. For the general step, suppose that

$$w_1, \dots, w_{k-1}, v_k, \dots, v_n$$

is a basis such that

$$w_i \cdot w_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

If $v'_k = v_k - (v_k \cdot w_1)w_1 - (v_k \cdot w_2)w_2 - \dots - (v_k \cdot w_{k-1})w_{k-1}$ then it is easy to check that

$$v'_k \cdot w_j = v_k \cdot w_j - (v_k \cdot w_j)1 = 0$$

for $j = 1, \dots, k-1$. Normalize v'_k to obtain w_k and replace v_k with w_k .

For our last approach to rewriting quadratic forms as sums and differences of squares, we will need a brief detour to the complex scalars. Complex numbers were “invented” to ensure that every real polynomial has a root, albeit not necessarily a real root. For vectors, this principal can be formulated as follows:

Let A be an $n \times n$ matrix with complex entries and let V be a nonzero subspace of \mathbf{C}^n such that $Av \in V$ for all $v \in V$. Then there exists a nonzero vector $w \in V$ and a complex scalar λ such that

$$Aw = \lambda w.$$

The vector w is called an *eigenvector* for A and λ is called an *eigenvalue* for A (associated to w). We will assume this result.

Recall that if $\alpha = r + \imath s$ is a complex number ($r, s \in \mathbf{R}$) then its complex conjugate is $\bar{\alpha} = r - \imath s$. The following properties are easy to check.

- $\overline{\alpha + \beta} = \bar{\alpha} + \bar{\beta}$ for all complex numbers α and β .

- $\overline{\alpha\beta} = \overline{\alpha}\overline{\beta}$ for all complex numbers α and β .
- $\alpha = \overline{\alpha}$ if and only if α is a real number.
- $\alpha\overline{\alpha} = r^2 + s^2$ is a nonzero positive real number whenever $\alpha \neq 0$.

In some sense, the last fact is the complex version of the observation for real numbers that a^2 is always a nonzero positive real number when $a \neq 0$. This leads to the notion of *complex dot product*. If $v = (v_1, \dots, v_n) \in \mathbf{C}^n$ and $w = (w_1, \dots, w_n) \in \mathbf{C}^n$ then

$$v \cdot w = v_1\overline{w_1} + v_2\overline{w_2} + \cdots + v_n\overline{w_n}.$$

Notice that if v is a nonzero vector then $v \cdot v$ is a nonzero positive real number.

Lemma 15.2 *Let A be an $n \times n$ symmetric real matrix. (That means each entry of A is a real number and its (i, j) -entry is the same as its (j, i) -entry.) If $v, w \in \mathbf{C}^n$ then the following two complex dot products coincide:*

$$(Av) \cdot w = v \cdot (Aw).$$

Proof: Write $A = (a_{ij})$, $V = (v_1, \dots, v_n)$, and $w = (w_1, \dots, w_n)$. When we multiply the matrix A with w we think of w as a column vector; its t^{th} coordinate is

$$a_{t1}w_1 + a_{t2}w_2 + \cdots + a_{tn}w_n.$$

The algebra of complex numbers tells us that

$$\overline{a_{t1}w_1 + a_{t2}w_2 + \cdots + a_{tn}w_n} = a_{t1}\overline{w_1} + a_{t2}\overline{w_2} + \cdots + a_{tn}\overline{w_n}.$$

We compute the complex dot product $v \cdot (Aw)$ as we have in the real case by lining up the two row vectors.

$$\begin{array}{cccc} & v_1 & v_2 & \cdots & v_n \\ a_{11}\overline{w_1} + \cdots + a_{1n}\overline{w_n} & a_{21}\overline{w_1} + \cdots + a_{2n}\overline{w_n} & \cdots & a_{n1}\overline{w_1} + \cdots + a_{nn}\overline{w_n} \end{array}$$

to obtain

$$\begin{array}{ccccccc} & a_{11}v_1\overline{w_1} & + & a_{12}v_1\overline{w_2} & + & \cdots & + & a_{1n}v_1\overline{w_n} \\ + & a_{21}v_2\overline{w_1} & + & a_{22}v_2\overline{w_2} & + & \cdots & + & a_{2n}v_2\overline{w_n} \\ & \cdot & & & & & & \cdot \\ & \cdot & & & & & & \cdot \\ & \cdot & & & & & & \cdot \\ + & a_{n1}v_n\overline{w_1} & + & a_{n2}v_n\overline{w_2} & + & \cdots & + & a_{nn}v_n\overline{w_n} \end{array}$$

Now switch the order of the subscripts for each a_{ij} and add down columns. The original dot product equals

$$(a_{11}v_1 + \cdots + a_{1n}v_n)\overline{w_1} + (a_{21}v_1 + \cdots + a_{2n}v_n)\overline{w_2} + \cdots + (a_{n1}v_1 + \cdots + a_{nn}v_n)\overline{w_n}$$

This last expression happens to be $(Av) \cdot w$. ■

The next result is sometimes called the “Principal Axis Theorem”.

Theorem 15.1 *Let A be an $n \times n$ symmetric real matrix. Then*

1. *all eigenvalues of A are real numbers, and*
2. *there is an orthogonal basis for \mathbf{R}^n which consists of eigenvectors for A .*

Proof: The first step is to show that each eigenvalue of A is real. So suppose that

$$Av = \lambda v$$

for some nonzero vector $v \in \mathbf{C}^n$ and some complex number λ . According to the lemma,

$$(Av) \cdot v = v \cdot (Av), \quad \text{so} \quad (\lambda v) \cdot v = v \cdot (\lambda v).$$

It follows that $\lambda(v \cdot v) = \overline{\lambda}(v \cdot v)$. Since $(v \cdot v) \neq 0$, we see that $\lambda = \overline{\lambda}$.

The second step is to prove that if $Av = \lambda v$ then we can choose v to be a nonzero *real* vector. Indeed, write

$$v = x + iy$$

where $x, y \in \mathbf{R}^n$. If $v \neq 0$ then at least one of x or y is nonzero. We have

$$Ax + iAy = \lambda x + i\lambda y.$$

Since λ is a real number, we are exhibiting the real and imaginary “parts” on each side of the equality. Thus

$$Ax = \lambda x \quad \text{and} \quad Ay = \lambda y.$$

We may choose x or y to be our nonzero real eigenvector.

To prove part 2., we argue by induction on the dimension of W that if W is a nonzero subspace of \mathbf{R}^n with $Aw \in W$ for all $w \in W$ then W has an orthogonal basis of eigenvectors for A . When the dimension of W reaches n , we have completed the proof of the theorem.

If the dimension of W is one then every nonzero vector in W is an eigenvector. Choose any one of them as a basis for W . Now suppose the dimension of W is k . Choose a nonzero vector $w \in W$ with $Aw = \lambda w$ for some eigenvalue λ . According to Gram-Schmidt, w is the first vector of some orthogonal basis

$$w, u_2, u_3, \dots, u_k$$

for W . If we set

$$U = \{z \in W \mid w \cdot z = 0\}$$

then U is a subspace of W and u_2, \dots, u_k is a basis for U . In order to apply induction to the smaller subspace U , we need to know that $Au \in U$ for all $u \in U$. Equivalently, we must show that

$$w \cdot (Au) = 0$$

for $u \in U$. But the lemma implies that

$$w \cdot (Au) = (Aw) \cdot u = (\lambda w) \cdot u = \lambda(w \cdot u) = 0.$$

By induction, U has an orthogonal basis of eigenvectors w_2, \dots, w_k . It is immediate that

$$w, w_2, \dots, w_k$$

is an orthogonal basis for W consisting of eigenvectors for A . ■

The relevance to *-addressing is that every symmetric bilinear form B can be encoded by a symmetric matrix: if e_1, \dots, e_n is a basis for the underlying real vector space then the $n \times n$ matrix $((B))$, whose (i, j) -entry is $B(e_i, e_j)$, will be symmetric. For the crucial example of the distance form for a connected graph, squares of variables never appear. Thus the matrix $((B))$ which induces the distance form looks like $\frac{1}{2}C$ where the (i, j) -entry of C is the distance between vertex i and vertex j (and there are zeroes down the diagonal).

As a consequence of the Principal Axis Theorem we can find a second orthonormal basis

$$\mathbf{v} : v_1, \dots, v_n$$

such that $((B))v_t = \lambda_t v_t$ for some real number λ_t . We are going to explore what this means.

Expand $v_t = \sum_{d=1}^n \alpha_{dt} e_d$ so we see each v_t as a linear combination of vectors in the original basis. The eigenvector equation for $((B))$ becomes

$$\sum_j B(e_i, e_j) \alpha_{jt} = \lambda_t \alpha_{it} .$$

But

$$\sum_j B(e_i, e_j) \alpha_{jt} = B(e_i, \sum_j \alpha_{jt} e_j) = B(e_i, v_t) .$$

We are now ready to evaluate

$$\begin{aligned} B(v_s, v_t) &= B\left(\sum_i \alpha_{is} e_i, v_t\right) \\ &= \sum_i \alpha_{is} B(e_i, v_t) \\ &= \sum_i \alpha_{is} \lambda_t \alpha_{it} \\ &= \lambda_t \sum_i \alpha_{is} \alpha_{it} \\ &= \lambda_t (v_s \cdot v_t) . \end{aligned}$$

Hence $B(v_s, v_t) = 0$ whenever $s \neq t$ and $B(v_t, v_t) = \lambda_t$ in the remaining case.

It follows that the associated quadratic form for B relative to the basis \mathbf{v} is

$$\mathcal{Q}_{\mathbf{v}}(X_1, \dots, X_n) = \sum_{t=1}^n \lambda_t X_t^2 .$$

It's time to put together a lot of our work on quadratic forms.

Theorem 15.2 *Let b be the number of bits for a $*$ -addressing of some connected graph on n vertices. Let B denote the symmetric bilinear form, relative to the standard basis for \mathbf{R}^n , which gives rise to the distance form on the graph. If $\#PE$ is the number of positive eigenvalues for $((B))$ and $\#NE$ is the number of negative eigenvalues then*

$$b \geq \max(\#PE, \#NE) .$$

Proof: Let D be the distance form for the given graph. We regard it as the quadratic form associated to the bilinear form B , relative to the standard basis \mathbf{e} . The matrix $((B))$ can be described directly: its (i, j) -entry is half the path distance from vertex i to vertex j (with zeroes down the diagonal). The $*$ -addressing allows us to write

$$\begin{aligned} D(X_1, \dots, X_n) &= \mathcal{Q}_{\mathbf{e}}(X_1, \dots, X_n) \\ &= \sum_{k=1}^b L_k(X_1, \dots, X_n)^2 - \sum_{k=1}^b M_k(X_1, \dots, X_n)^2 \end{aligned}$$

where L_k and M_k are linear forms.

Now use the orthonormal basis \mathbf{f} consisting of eigenvectors for $((B))$. We know, as a consequence of the Principal Axis Theorem, that

$$\mathcal{Q}_{\mathbf{f}}(Y_1, \dots, Y_n) = \sum_{j=1}^n \lambda_j Y_j^2$$

with $\lambda_1, \dots, \lambda_n$ the eigenvalues of $((B))$. Making the change of variables $Z_j = \sqrt{|\lambda_j|} Y_j$ is equivalent to changing the members of the basis \mathbf{f} by a scalar multiple. So there is a basis \mathbf{g} such that

$$\mathcal{Q}_{\mathbf{g}}(Z_1, \dots, Z_n) = \sum_{i=1}^{\#PE} Z_i^2 - \sum_{j=1}^{\#NE} Z_j^2 .$$

According to our results relating change of variables to a linear combination of squares of linear forms, we also have

$$\mathcal{Q}_{\mathbf{g}}(Z_1, \dots, Z_n) = \sum_{k=1}^b L'_k(X_1, \dots, X_n)^2 - \sum_{k=1}^b M'_k(X_1, \dots, X_n)^2$$

for linear forms L'_k and M'_k .

Finally, apply Sylvester's Law to obtain

$$b \geq \#PE \text{ and } b \geq \#NE . \blacksquare$$

As an illustration, you can use your favorite symbolic manipulation package to find the eigenvalues for the symmetric matrix arising from the Y-tree and make yet another estimate for bits in a *-addressing via Sylvester's Law.

16 LLL Lattice Reduction

We have nearly enough tools to return to our old problem of algorithmically finding shortest vectors in a lattice. So suppose we are given a lattice L in \mathbf{R}^n . This means that we have a list of vectors

$$\mathbf{v} : v_1, v_2, \dots, v_r$$

which are linearly independent over \mathbf{R} and L consists of all integer linear combinations of these vectors. The goal is to replace the original basis with a better basis

$$\mathbf{w} : w_1, w_2, \dots, w_r$$

so that w_1 is a fairly short vector in L .

The insight is that if v_1, \dots, v_r is orthogonal with

$$\|v_1\| \leq \|v_2\| \leq \dots \leq \|v_r\|$$

then v_1 is a shortest vector. Indeed, suppose

$$y = a_1 v_1 + \dots + a_r v_r$$

is a nonzero vector in L . Then $a_j \in \mathbf{Z}$ for all j and $|a_t| \geq 1$ for some t .

$$\begin{aligned} \|y\|^2 &= y \cdot y \\ &= a_1^2 \|v_1\|^2 + a_2^2 \|v_2\|^2 + \dots + a_r^2 \|v_r\|^2 \\ &\geq a_t^2 \|v_t\|^2 \\ &\geq \|v_1\|^2. \end{aligned}$$

If v_1, \dots, v_r is not orthogonal, why can't we simply apply Gram-Schmidt? The problem is that Projection Lemma 15.1 involves denominators while scalars for linear combinations in L must be integers. Let's review. If $\mathbf{w} : w_1, \dots, w_r$ is a list of linearly independent vectors let $\mathbf{w}^* : w_1^*, \dots, w_r^*$ be its *Gram-Schmidt orthogonalization*, i.e.,

$$w_1^* = w_1$$

and, iteratively,

$$w_k^* = w_k - \frac{w_k \cdot w_1^*}{\|w_1^*\|^2} w_1^* - \frac{w_k \cdot w_2^*}{\|w_2^*\|^2} w_2^* - \dots - \frac{w_k \cdot w_{k-1}^*}{\|w_{k-1}^*\|^2} w_{k-1}^*.$$

To save notation, in the future we shall write

$$\mu_{kj} = \frac{w_k \cdot w_j^*}{\|w_j^*\|^2}$$

for $j < k$. The scalar μ_{kj} depends on the list \mathbf{w} and will occasionally be written $\mu_{kj}(\mathbf{w})$.

The rough idea of the Lenstra-Lenstra-Lovasz algorithm on \mathbf{v} is to use many applications of the Replacement Principle to produce a basis \mathbf{w} for L so that \mathbf{w}^* has small coefficients. Since \mathbf{w} is nearly orthogonal, it should contain a nearly shortest vector for L .

Definition 16.1 A list $\mathbf{w} : w_1, \dots, w_r$ of linearly independent vectors in \mathbf{R}^n is **LLL-reduced** provided it satisfies two conditions:

$$[\text{Gauss}] \quad |\mu_{kj}| \leq \frac{1}{2} \text{ for all } 1 \leq j < k \leq r;$$

$$[\text{Lovasz}] \quad \|w_j^*\|^2 \geq \left(\frac{3}{4} - \mu_{j,j-1}^2\right) \|w_{j-1}^*\|^2 \text{ for all } j > 1.$$

Lemma 16.1 If $\mathbf{w} : w_1, \dots, w_r$ is LLL-reduced then

$$\|w_k\|^2 \leq 2^{k-1} \|w_k^*\|^2$$

for $k = 1, 2, \dots, r$.

Proof: By definition,

$$w_k = w_k^* + \mu_{k,k-1} w_{k-1}^* + \mu_{k,k-2} w_{k-2}^* + \dots + \mu_{k1} w_1^*.$$

Hence

$$\|w_k\|^2 = \|w_k^*\|^2 + \mu_{k,k-1}^2 \|w_{k-1}^*\|^2 + \mu_{k,k-2}^2 \|w_{k-2}^*\|^2 + \dots + \mu_{k1}^2 \|w_1^*\|^2.$$

According to the Gauss condition, $\mu_{kj}^2 \leq \frac{1}{4}$, so

$$\|w_k\|^2 \leq \|w_k^*\|^2 + \frac{1}{4} (\|w_{k-1}^*\|^2 + \dots + \|w_1^*\|^2).$$

Also, $\frac{3}{4} - \mu_{j,j-1}^2 \geq \frac{3}{4} - \frac{1}{4}$, so the Lovasz condition implies

$$\|w_j^*\|^2 \geq \frac{1}{2} \|w_{j-1}^*\|^2.$$

If we put the last two \mathbf{w} -inequalities together, we obtain

$$\|w_k\|^2 \leq \|w_k^*\|^2 \left(1 + \frac{1}{4}(2 + 2^2 + \dots + 2^{k-1})\right).$$

But the formula for the sum of a geometric series states that $1 + 2 + 2^2 + \dots + 2^{k-2} = 2^{k-1} - 1$. Thus

$$\|w_k\|^2 \leq \|w_k^*\|^2 \left(1 + \frac{1}{2}(2^{k-1} - 1)\right).$$

Finally,

$$1 + \frac{1}{2}(2^{k-1} - 1) \leq 2^{k-1}. \quad \blacksquare$$

Theorem 16.1 Let $\mathbf{w} : w_1, \dots, w_r$ be an LLL-reduced basis for the lattice L and assume

$$\| w_1 \| \leq \| w_2 \| \leq \dots \leq \| w_r \| .$$

Then

$$\| w_1 \| \leq 2^{\frac{r-1}{2}} \| x \|$$

for every nonzero vector x in L .

Proof: We may write

$$x = b_1 w_1 + b_2 w_2 + \dots + b_k w_k$$

for some integers b_1, \dots, b_k with $b_k \neq 0$. Next rewrite x using the orthogonalized basis \mathbf{w}^* .

$$x = c_1 w_1^* + c_2 w_2^* + \dots + c_k w_k^*$$

where c_1, \dots, c_k are real numbers and

$$c_k = b_k.$$

Now

$$\| x \|^2 = c_1^2 \| w_1^* \|^2 + \dots + c_k^2 \| w_k^* \|^2 .$$

This forces

$$\| x \|^2 \geq c_k^2 \| w_k^* \|^2 = b_k^2 \| w_k^* \|^2 \geq \| w_k^* \|^2$$

because b_k is a nonzero integer.

By the lemma,

$$\| w_k^* \|^2 \geq \frac{1}{2^{k-1}} \| w_k \|^2 \geq \frac{1}{2^{k-1}} \| w_1 \|^2 .$$

We have proved that

$$\| w_1 \|^2 \leq 2^{k-1} \| x \|^2 .$$

Finally, notice that $k \leq r$ and take the square root of each side. ■

The message of the corollary is that if we can replace a given basis \mathbf{v} for a lattice with an LLL-reduced basis \mathbf{w} then \mathbf{w} will contain a pretty short vector for the lattice. We shall outline why such a replacement is feasible.

Let's handle the Gauss condition by "correcting" each μ_{rs} inductively. Say an ordered pair (r, s) with $r > s$ is in the *tail* of a list \mathbf{w} provided that $|\mu_{pq}(\mathbf{w})| \leq \frac{1}{2}$ for all $p > r$ and whenever $p = r$ and $q \geq s$. The procedure is to increase the size of the tail at least one ordered pair at a time.

Suppose that $|\mu_{kj}(\mathbf{w})| > \frac{1}{2}$ with (k, j) at the “boundary” of the tail for \mathbf{w} . Let c be the closest integer to μ_{kj} . Notice that $c \neq 0$ and $|\mu_{kj} - c| \leq \frac{1}{2}$. By the Replacement Principle,

$$\mathbf{u} : w_1, w_2, \dots, w_{k-1}, w_k - cw_j, w_{k+1}, \dots, w_r$$

is also a basis for the lattice.

Consider the Gram-Schmidt orthogonalization of \mathbf{u} . After $k-1$ steps one has

$$w_1^*, w_2^*, \dots, w_{k-1}^*, w_k - cw_j, w_{k+1}, \dots, w_r.$$

Since $j < k$, the vector cw_j is a linear combination of w_1^*, \dots, w_{k-1}^* . Indeed,

$$cw_j = c\mu_{j1}w_1^* + \dots + c\mu_{j,j-1}w_{j-1}^* + cw_j^*.$$

In the k^{th} step of orthogonalization, $w_k - cw_j$ is replaced with

$$(w_k - cw_j) - \frac{(w_k - cw_j) \cdot w_1^*}{\|w_1^*\|^2} w_1^* - \dots - \frac{(w_k - cw_j) \cdot w_{k-1}^*}{\|w_{k-1}^*\|^2} w_{k-1}^*.$$

By using the bilinearity of dot product, we see that this expression equals

$$w_k^* - cw_j + (c\mu_{j1}w_1^* + \dots + c\mu_{j,j-1}w_{j-1}^*) + c \frac{w_j \cdot w_j^*}{\|w_j^*\|^2} w_j^* + \sum_{p=j+1}^{k-1} c \frac{w_j \cdot w_p^*}{\|w_p^*\|^2} w_p^*.$$

Now w_j is w_j^* plus a linear combination of w_1^*, \dots, w_{j-1}^* . Hence

$$w_j \cdot w_j^* = \|w_j^*\|^2 \quad \text{and} \quad w_j \cdot w_p^* = 0 \quad \text{for } p > j.$$

Thus every term in the long expression above vanishes except the first. In other words, \mathbf{u}^* is the same as \mathbf{w}^* .

What is $\mu_{rs}(\mathbf{u})$? Formally it is

$$\frac{u_r \cdot w_s^*}{\|w_s^*\|^2}.$$

Thus $\mu_{rs}(\mathbf{u}) = \mu_{rs}(\mathbf{w})$ whenever $r \neq k$. Also,

$$\mu_{ks}(\mathbf{u}) = \mu_{ks}(\mathbf{w}) - c \frac{w_j \cdot w_s^*}{\|w_s^*\|^2}.$$

If $j < s$ then $w_j \cdot w_s^* = 0$. (Why?) We conclude that

$$\mu_{rs}(\mathbf{u}) = \mu_{rs}(\mathbf{w}) \quad \text{for } r > k \text{ or for } r = k \text{ with } s > j.$$

So far, we have shown that the tail of \mathbf{u} is at least as long as the tail of \mathbf{w} . At the crucial (k, j) -coefficient,

$$\mu_{kj}(\mathbf{u}) = \frac{u_k \cdot w_j^*}{\|w_j^*\|^2} = \frac{w_k \cdot w_j^*}{\|w_j^*\|^2} - c \frac{w_j \cdot w_j^*}{\|w_j^*\|^2} = \mu_{kj}(\mathbf{w}) - c.$$

Thus $|\mu_{kj}(\mathbf{u})| \leq \frac{1}{2}$. The tail for \mathbf{u} is at least one pair longer than the tail for \mathbf{w} !

Eventually, we produce a basis for the lattice which is all tail. That is, the Gauss condition holds. With this procedure, it's time to describe the entire LLL algorithm. A lattice basis

$$\mathbf{v} : v_1, \dots, v_r$$

is given. Any list consisting of a single vector is LLL-reduced by default. Suppose we can achieve LLL-reduction via Replacement for any list with $k - 1$ vectors. Then we can produce a new basis $\mathbf{u} : u_1, \dots, u_{k-1}, v_k, \dots, v_r$ so that the sublist consisting of the first $k - 1$ vectors is LLL-reduced. Using the Gauss subalgorithm, we can replace v_k with some

$$u_k = v_k - \sum_{j=1}^{k-1} c_j u_j$$

with the result that

$$\mathbf{u} : u_1, \dots, u_{k-1}, u_k$$

satisfies the Gauss condition. If

$$\|u_k^*\|^2 \geq \left(\frac{3}{4} - \mu_{k,k-1}^2\right) \|u_{k-1}^*\|^2$$

then \mathbf{u} is LLL-reduced and we have handled sublists of length k . If not, interchange u_k and u_{k-1} , apply LLL-reduction to

$$u_1, \dots, u_{k-2}, u_k$$

and repeat this subprocedure.

In summary, the full algorithm is comprised of the Gauss subalgorithm together with strategic swaps. However, it is not at all obvious why this algorithm ever stops let alone why it only takes a feasible number of steps. We need some measure of “closeness to orthogonality” which improves after

each swap. It turns out that an appropriate quantity can be borrowed from the end of the previous section of these notes.

As a prerequisite, we require a two-sentence review of determinants. If A and B are two square real matrices then

$$\det(AB) = \det(A)\det(B).$$

Secondly, if T is an upper (or lower) triangular matrix then its determinant is the product of all of its diagonal entries.

Suppose $\mathbf{v} : v_1, \dots, v_r$ are linearly independent vectors in \mathbf{R}^n . Define

$$G(v_1, \dots, v_r) = \det(v_i \cdot v_j),$$

the determinant of the $r \times r$ matrix whose (i, j) -entry is the dot product of v_i and v_j . (This is $\det(B)$ for the dot product bilinear form B .) The rule for matrix multiplication implies that if $C(\mathbf{v})$ is the $r \times r$ matrix whose j^{th} column is v_j then

$$\det(C(\mathbf{v})^\top C(\mathbf{v})) = G(v_1, \dots, v_r).$$

Suppose we change \mathbf{v} to \mathbf{v}' by one application of the Replacement Principle for integer combinations. That is, for some t

$$v'_t = \pm v_t + \sum_{s \neq t} a_s v_s$$

and $v'_q = v_q$ for all q other than t . This change can be viewed as an iteration of even simpler moves,

$$v'_t = \pm v_t + b v_s \text{ for some } s \neq t$$

and $v'_q = v_q$ when $q \neq t$. It is not difficult to check that

$$C(\mathbf{v}') = C(\mathbf{v})E$$

where E is the $r \times r$ matrix that has ± 1 in the (t, t) -entry, 1 in all other diagonal entries, b in the (s, t) -position, and zeros elsewhere.

$$\begin{aligned} G(v'_1, v'_2, \dots, v'_r) &= \det \left[(C(\mathbf{v})E)^\top C(\mathbf{v})E \right] \\ &= \det \left[E^\top C(\mathbf{v})^\top C(\mathbf{v})E \right] \\ &= \det(E^\top) \det \left[C(\mathbf{v})^\top C(\mathbf{v}) \right] \det(E) \\ &= (\pm 1)^2 G(v_1, \dots, v_r). \end{aligned}$$

In other words, G is not affected when the list of vectors is changed via a sequence of legal lattice replacements.

Lemma 16.2 *Let $\mathbf{v} : v_1, \dots, v_r$ be a linearly independent list of vectors and let \mathbf{v}^* be its Gram-Schmidt orthogonalization. Then*

$$G(v_1, \dots, v_r) = \|v_1^*\|^2 \|v_2^*\|^2 \cdots \|v_r^*\|^2.$$

In particular, $G(v_1, \dots, v_r) > 0$.

Proof: Notice that $G(v_1, \dots, v_r) = G(v_1^*, \dots, v_r^*)$ because \mathbf{v}^* is obtained from \mathbf{v} by successive applications of the special replacement

$$v'_t = v_t + bv_s$$

for $b \in \mathbf{R}$. (The argument above that G is not affected by lattice replacements actually did not require that b be an integer!) Now \mathbf{v}^* is an orthogonal list so the matrix $(v_i^* \cdot v_j^*)$ is a diagonal matrix with

$$v_1^* \cdot v_1^*, \dots, v_r^* \cdot v_r^*$$

down the diagonal. ■

Again, assume $\mathbf{v} : v_1, \dots, v_r$ is a linearly independent list of vectors. Define

$$G^!(\mathbf{v}) = G(v_1)G(v_1, v_2)G(v_1, v_2, v_3) \cdots G(v_1, \dots, v_r).$$

$G^!$ does not change for any replacement move in the Gauss subalgorithm. But it does change for an adjacent swap. The question is how.

Suppose that $\mathbf{z} : z_1, \dots, z_r$ is obtained from $\mathbf{y} : y_1, \dots, y_r$ by transposing y_p and y_{p+1} . That is,

$$z_1 = y_1; \dots; z_{p-1} = y_{p-1}; z_p = y_{p+1}; z_{p+1} = y_p; z_{p+2} = y_{p+2}; \dots$$

The only factor of $G^!$ which changes is

$$G(y_1, \dots, y_p) \mapsto G(z_1, \dots, z_p).$$

(This is a long exercise which parallels the argument that G is not affected by an individual replacement. All other factors in $G^!$ have the same listed vectors except the relative order of two of them may be different. This corresponds to a matrix multiplication $C(\mathbf{v})E$ where, this time, E is the

identity matrix after its p^{th} and $(p+1)^{\text{st}}$ columns are transposed.) According to the lemma, the change in that single factor amounts to

$$\|y_1^*\|^2 \cdots \|y_p^*\|^2 \mapsto \|z_1^*\|^2 \cdots \|z_p^*\|^2 .$$

Thus

$$G^!(\mathbf{z}) = \frac{\|z_p^*\|^2}{\|y_p^*\|^2} G^!(\mathbf{y}).$$

We now compute z_p^* in terms of \mathbf{y}^* .

$$\begin{aligned} z_p^* &= z_p - \sum_{j=1}^{p-1} \mu_{pj}(\mathbf{z}) z_j^* \\ &= y_{p+1} - \sum_{j=1}^{p-1} \mu_{pj}(\mathbf{z}) y_j^* \\ &= y_{p+1} - \sum_{j=1}^{p-1} \mu_{p+1,j}(\mathbf{y}) y_j^* \\ &= y_{p+1}^* + \sum_{t=1}^p \mu_{p+1,t}(\mathbf{y}) y_t^* - \sum_{j=1}^{p-1} \mu_{p+1,j}(\mathbf{y}) y_j^*. \end{aligned}$$

After simplifying, we see that

$$z_p^* = y_{p+1}^* + \mu_{p+1,p}(\mathbf{y}) y_p^*.$$

It follows that

$$\|z_p^*\|^2 = \|y_{p+1}^*\|^2 + \mu_{p+1,p}^2(\mathbf{y}) \|y_p^*\|^2 .$$

Now the only reason we would ever swap in the algorithm is that

$$\|y_{p+1}^*\|^2 < \left(\frac{3}{4} - \mu_{p+1,p}^2\right) \|y_p^*\|^2 .$$

If we put the previous equality together with this inequality, we discover that swapping takes place when

$$\|z_p^*\|^2 < \frac{3}{4} \|y_p^*\|^2 .$$

In summary, we have shown that if \mathbf{z} is obtained from \mathbf{y} in the algorithm by performing an adjacent swap then

$$G^!(\mathbf{z}) < \frac{3}{4} G^!(\mathbf{y}).$$

The upshot is that the LLL-algorithm halts provided we know that G^l cannot become arbitrarily small after some sequence of replacements and swaps. In full generality, this is a consequence of a famous theorem due to Minkowski. However, for the special case of integer lattices, this is obvious: G^l is a nonzero (positive) integer so it cannot be smaller than 1.