Spot it! Writeup

Cross Product of Two Points

Same points: [1,2,1]x[2,4,2] = [2\*2-1\*4, 1\*2-1\*2, 1\*4-2\*2] = [0,0,0]

Different points: [1,2,1]x[2,1,2] = [(2\*2-1\*1), (1\*2-1\*2), (1\*1 - 2\*2)] = [1, 0, -3]

Recipe for *generate\_all\_points*

Inputs:

* *mod*, which is a prime number that we restrict our finite field to Zmod

1. Create an empty sequence called *allpts*, which will store all the points generated
2. Use three nested iterations through the range [0, *mod*), which is also all possible coordinate values. For each iteration, the number from the first iteration will represent the x coordinate, the one from the second will represent the y coordinate and the one from third would represent the z coordinate. Add the x, y, and z coordinates as an ordered triple to *allpts*.
3. Pop the first element (which is the ordered triple (0,0,0)) of the sequence *allpts*.
4. Use two nested for loops. The first loop iterates through all the points and the element that is currently iterated over is assigned as *pt1*. The second loop iterates through the rest of the sequence *allpts*, which does not including the elements before *pt1* and *pt1* itself; each element in this loop is referred to as *pt2*.
5. With equivalent function, input the ordered triplets *pt1* and *pt2* as well as *mod* to check if the two points are the same point. If *pt1* and *pt2* are same, then remove *pt2* from *allpts*.
6. Return the *allpts*.

Recipe for *create\_cards*

Inputs:

* *mod*, which is a prime number that we restrict our finite field to Zmod
* *lines* and *points*, both of which are sequences that contains tuples consisting of three integers in the range of [0,mod)

1. Create an empty sequence *cards* that will store all possible cards
2. Iterate through *lines*. The element that is currently iterated over is assigned to the variable *line*.
3. Within the iteration from step 2, create another empty sequence called *card*.
4. Within the same iteration from step 2, iterate through *points*, where each individual element is referred to as *point.*
5. In the for loop from step 4, call the function incident that takes in *point*, *line*, and *mod*. If there is an incident from *point* and *line* given the *mod*, then add the *point to* the sequence *card*.
6. Outside of the iteration from step 4 but inside the iteration from step 2, add *card* to the sequence *cards*.
7. Return *cards.*

Discussion: If you use a prime modulus of 5, you will generate a valid deck of 31 "Spot it!" cards, each with 6 images on them. If you use a prime modules of 7, you will generate a valid deck of 57 "Spot it!" cards, each with 8 images on them.

Suppose you wanted to create a valid deck of 40 "Spot it!" cards. Is this possible? If not, why not? If so, how would you go about doing so?

We can see a pattern between the prime modulus, the number of images, and the cards. For a projective plane of order n, there will be n+1 images, n2 + n + 1 cards (eg. for prime modulo of 2, there are at least 2+1 = 3 images and 22+2+1= 7 cards). With a deck of 40 cards, the prime modulus will not be a prime number or even an integer, so a deck of 40 cards is not possible.