Test 1 Solutions

October 13, 2004

1. **Show by induction that every triangulation of polygon P of n vertices contains exactly $n-3$ diagonals.**.

   Proof:
   The proof is an induction on the number of vertices.
   Base case: $n = 3$ (a triangle) and $n - 3 = 0$ which is true because we cannot have a diagonal in a triangle.

   Induction hypothesis: Suppose that the theorem is true for all polygons on n vertices. We need to show that it is also true for every polygon of $n+1$ vertices. That is, show that any polygon of $n+1$ vertices has $(n+1)-3 = n-2$ diagonals.

   Let P be a triangulated polygon on $n+1$ vertices. We know that every triangulation has at least on ear. Let v_{i-1}, v_i, v_{i+1} be an ear of P (with v_i being the tip of the ear). We remove the vertex v_i and its incident edges from P and join $v_{i-1}v_{i+1}$. The resulting polygon $P\prime$ is a triangulated polygon on n vertices and according to our assumption, it has $n-3$ diagonals.

   Now, if we put back vertex v_i the edge $v_{i-1}v_{i+1}$ stops being an edge of the polygon P and becomes a *diagonal*. This means that we added one more diagonal to the triangulation of $P\prime$. Thus, the number of diagonals becomes $n-3+1 = n-2$, which is what we want.

2. **(a) Given an arrangement of n lines in the plane in general position (no 2 parallel, no 3 meeting at one point), show that the intersection of points can be 3-colored.**

   Proof:
   The proof is an induction on the number of **intersection points**.
   Base case: $n = 3 \Rightarrow$ we have 3 intersection points which can be colored with 3 different colors.

   Induction hypothesis: Let k be a number such that $0 < k < n^2$ (k is an integer). Suppose k of the intersection points can be 3-colored. We need to show that $k+1$ intersection points can also be 3-colored.

   Since the lines are in general position, then every line intersects every other line and hence, we have n^2 intersection points. Now, suppose we pick a random line l_i out of the n lines such that l_i divides the intersection points into two groups P_k and P_{n-k}, where P_k is a set of k intersection points (including the ones on l_i) and P_{n-k} is the remaining set of $n-k$ intersection points. Since $k < n^2$ then by our assumption we know that the k points of p_k can be 3-colored.

Now, add one more uncolored point v to P_k and show that if v can be colored properly, then we would have colored $k+1$ intersection points and the theorem is proved.
If v has no adjacent points which belong to P_k, then color it any color.
If v has 1 adjacent point in P_k, then color it with one of the two other colors.
If v has 2 adjacent point in P_k, then color it with the third available color.
(Note that v cannot have more than 2 adjacent vertices which belong to P_k)

Thus, the theorem is proved.

(b) Design and algorithm for 3-coloring the intersection points of the arrangement in (a). The worst-case complexity has to be $O(n^2 logn)$. Argue the correctness and the complexity of the algorithm.

Note: *In the test, this question asked you to provide a $O(n^2)$ time algorithm. This was supposed to be $O(n^2 logn)$. An algorithm that runs in time $O(n^2)$ exists but is not very easy. This is why I was easy on marking this question. So, below I will give a description of the $O(n^2 logn)$ algorithm first and then breifly explain how to do the $O(n^2)$ algorithm. But for next time keep in mind that a slow but correct algorithm is much better than a fast but wrong one.*

The algorithm runs as follows:
1. Sort the points by x-coordinate.
2. Take the first 3 points and color each with a different color.
3. Take the next point v in the sorted list. This point will have at most two already colored neighbors. Color v with the third available color.
4. Repeat step 3 until all the vertices are colored

Correctness:
The algorithm works mainly because we have sorted the points and are considering them in order. This ensures that whenever we attempt to color an intersection point v, v will have at most 2 already colored neighboring intersection points (idea is similar to that of the induction hypothesis). If we color the points in some random order without sorting, then we might run into situations where three of the neighbors of some point v are already using the three available colors, and we will be stuck. In these cases, redoing the coloring of these neighboring points will not work because we might need to recolor vertices exponentially many times.

Complexity:
The sorting in the first step takes $O(n^2 logn)$ time. We are later traversing and coloring all the points once. Since we have $O(n^2)$ points, then this will take $O(n^2)$ time. Thus the running time of the algorithm is $O(n^2 logn)$.

Now, if we wanted to construct a $O(n^2)$ algorithm for coloring the intersection points of the arrangement, we do the same thing as above, except that in the first step, we give assign a direction to each line and do a *topological sort* instead of a regular sort (for more information on topological sort, check out http://www.cs.fsu.edu/ cop4531/slideshow/chapter23/23-4.html). A

topological sort runs in $O(n^2)$ time. The rest is similar. Since the running time of our algorithm is bounded by the running time of the sort, then the total running time will be $O(n^2)$.

3. **Design a k-state Turing machine that reads a tape with n consecutive marks ($n > 5$) and erases all but the first two marks and the last two marks. k has to be at most 8.**

Most of you got this question right. Here I will describe a 5-state Turing machine

Current State	Current Symbol	Action	New State	Why?
1	X	R	2	Skip the first mark
2	X	R	3	Skip the second mark
3	X	-	2	Delete the rest of the marks
2	-	R	3	Once a mark is deleted, go to the next one
3	-	L	1	If we are on the one after the last mark, go left
1	-	X	4	If no marks, mark the last position
4	X	L	4	go left
4	-	X	0	mark the one before last position and end