CIS 677: Homework 1

Rohan Shah

Problem 1.

For the sets S_1 and S_2 to correspond to the two sides of the minimum cut in the graph G a few properties must hold. The heaviest weight edge in the minimum weight spanning tree T that is removed by the algorithm must be an edge from the minimum cut. Proof: It must be that at least one edge in T is an edge from the min-cut, otherwise T would be not be connected which is a contradiction. If the edge removed from T is not an edge from the minimum cut then the endpoints of the min-cut edge both lie in either S_1 or S_2 which means S_1 and S_2 do not correspond to the two sides of the min-cut. By the same logic, there must be exactly one edge from the min-cut in T. Therefore, for S_1 and S_2 to correspond to the two sides of the min-cut, Kruskal's algorithm must select an edge from the min-cut for the first time in its final step, i.e. as the $(n-1)th$ edge in T. Since we assign the weights $w(e)$ to the edges uniformly at random, we are then choosing edges to be in T uniformly at random. As a slight modification to Kruskal's algorithm, each time an edge is selected to be in T we can remove any edegs adjacent to vertices we just connected who now have both their endpoints in the connected components since it can never be selected to be in T in the future. We are looking for the probability that an edge in the min-cut survives (is not chosen to be in T) until the final $(n-1)th$ step i.e. that it survives the first $n-2$ edge selections. This analysis is exactly the same as the one for the original contraction algorithm and further these two algorithms are essentially the same where when we pick a minimum weight edge to be in T and remove edges that now only connect endpoints in the connected components, it is the same as choosing an edge at random and contracting its vertices as in the contraction algorithm. Thus since our analysis for the probability of success is the same as the contraction algorithm the probability that S_1 and S_2 correspond to the two sides of the minimum cut is $\Omega(\frac{1}{n^2})$.

Problem 2.

If $NP \subseteq BPP$ then there is exists a BPP algorithm for SAT. Without loss of generality we say this algorithm returns a correct answer with probability at least $1 - \frac{1}{n^2}$. We can convert this BPP algorithm to an RP algorithm for SAT in the following way. Choose a variable uniformly at random and assign its value as true. Then reduce the SAT problem, and run the BPP algorithm for this smaller SAT problem. If the BPP algorithm returns saying that this smaller problem is satisfiable then leave the variable assigned as true otherwise assign it the value false and re-reduce the original SAT problem. Repeat this procedure until all the variables have been assigned a value. Finally, verify that the assignment we have obtained satisfies the original SAT problem. If it does then return true (it is satisfiable) otherwise return false (it is not satisfiable). In the case where the SAT problem is not satisfiable we always return the correct answer since we verify our assignment at the end (which in this case where the problem is not satisifable would always yield false) i.e. the probability we return that the problem is satisfiable when in fact it is not, is 0, which is what we need for an RP algorithm. On the other hand, in the case where the problem is satisfiable, we return the correct answer if and only if we assigned each variable in a way such that we end up with a satisfying assignment. This is equivalent to the BPP algorithm returning the correct answer in each iteration. Since the BPP algorithm returns the correct answer with probability at least $1 - \frac{1}{n^2}$ and we ran it for n variables, the probability that it returned the correct answer every time is $(1 - \frac{1}{n^2})^n \geq \frac{1}{2}$. So with probability greater than $\frac{1}{2}$ our RP algorithm returns the correct answer when the SAT problem is in fact satisfiable, which is what is required for an RP algorithm. Therefore $NP \subseteq RP$.