6.842 Randomness and Computation	February 13, 2012

Lecture 2

Lecturer: Ronitt Rubinfeld — *Scribe: Hamsa Sridhar*

1 Review

Last class, we introduced the problem of hypergraph coloring. We found that we could guarantee the existence of a non-monochromatic 2-coloring for the l-element sets $S_1, ...S_m \subseteq S$ using the probabilistic method, given a bound on the number of sets $m < 2^{l-1}$. However, we also noted that we could likely find a weaker constraint if all the sets $S_1, ..., S_m$ were somewhat 'independent,' i.e. did not have too many points in common.

Definition: An event A is **independent** of $B_1, ...B_k$ if $\forall J \subseteq [K]$ such that $J \neq \emptyset$,

$$\Pr[A \cap \bigcap_{j \in J} B_j] = \Pr[A]\Pr[\bigcap_{j \in J} B_j].$$

With this notion of independence in mind, we could then define a dependency digraph:

Definition: Given the events $A_1, ...A_n$, $D = (V, E)$ with $V = [n]$ is a **dependency digraph** of $A_1, ...A_n$ if each A_i is independent of the set of A_j that don't neighbor it in D.

We now introduce the symmetric version of the **Lovász Local Lemma**, which states that:

Theorem: Given events $A_1, ...A_n$ such that $\Pr[A_i] \leq p$ $\forall i \in [n]$ with dependency digraph D such that D is of degree $\leq d$, if $ep(d+1) \leq 1$, then $\Pr[\bigcap_{i \in [n]} \bar{A}_i] > 0$.

A key point to note is that the Lovász Local Lemma only restricts the degree of D and not the number of vertices V. In other words, we are placing a lower bound on the independence of the 'bad events' A_i, rather than placing an upper bound on the number of A_i. Today, we will see this new restriction come into play for the hypergraph coloring problem. We will also begin constructing the Moser-Tardos algorithm for finding a solution to the symmetric case of the hypergraph coloring problem.

2 Hypergraph Coloring (revisited)

Theorem: Given l-element sets $S_1, ...S_m \subseteq S$ such that each S_i intersects $\leq d$ other S_j's, if $\frac{e(d+1)}{2^{l-1}} \leq 1$ then there exists a 2-coloring of S such that no S_i is monochromatic.

Proof: Color each element of S red or blue independently with probability 1/2. Define A_i as the event that S_i is monochromatic. Then, the $\Pr[A_i] = 2\frac{1}{2^l} = \frac{1}{2^{l-1}}$ $\forall i \in [m]$. Moreover, since each S_i intersects at most d other S_j's, A_i depends on at most d other A_j's and its dependency digraph D has degree $\leq d$. Now, we can directly apply the Lovász Local Lemma to conclude that there exists a non-monochromatic 2-coloring as desired. $\square$

Once again, we can compare the probabilistic method with the Lovász Local Lemma. Given a hypergraph of m edges with l vertices each, the probabilistic method guarantees a non-monochromatic coloring if we bound the number of edges, i.e. $m < 2^{l-1}$, and the Lovász Local Lemma guarantees a non-monochromatic coloring if we bound the degree, i.e. $d + 1 \leq 2^{l-1}/e$.

The Lovász Local Lemma is particularly powerful because it guarantees a solution for some sub-classes of NP-hard problems. For instance, we can prove that every Boolean formula ϕ in k-conjunctive normal form (k-CNF) is satisfiable if it has the property that every variable occurs in at most $2^{k-logk-log2}$ clauses of ϕ. (Recall that ϕ is a k-CNF if ϕ is a conjunction of clauses, where each clause contains at most k literals.)

2.1 Brief History

1975: Lovász Local Lemma proposed; non-constructive; bound $d \leq 2^{l-2}$
1991: Beck proposed constructive algorithm; bound $d \leq 2^{l/48}$
1991-2009: Many results
2009: Moser's constructive algorithm for the symmetric case; bound $d \leq 2^{l-O(1)}$

Later, Moser and Tardos extended Moser's constructive algorithm to most problems (non-symmetric cases).

3 Moser-Tardos Symmetric Algorithm

Algorithm: Start with a random coloring of S. While there is a monochromatic set, pick an arbitrary 'violated' S_i and randomly reassign colors to the elements of S_i.

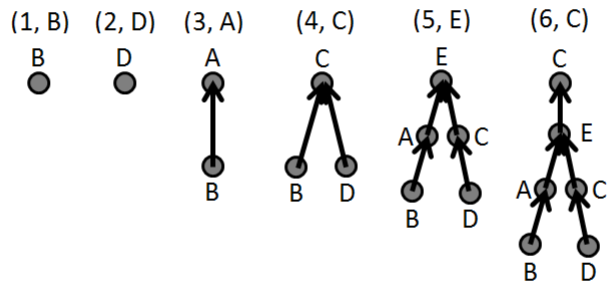
Now, to see why this algorithm works, we define the following:

Definition: The **log of execution** is a set of pairs $(1, S_{i1}), (2, S_{i2}), ...$ where the first entry of each pair is the 'loop number' and the second entry is the name of the the resampled set.

Definition: The **witness tree for step** j ($j \geq 0$) is constructed as follows -

1. The root vertex is labeled by S_{ij}.
2. Go backwards through the log for $t = j$ down to $t = 1$. If S_{it} intersects with any S_{ij} in the witness tree, add S_{it} to the witness tree by making it point to a node S_{ij} with which it intersects that is of maximum depth (picking arbitrarily if there is more than one choice at the maximum depth).

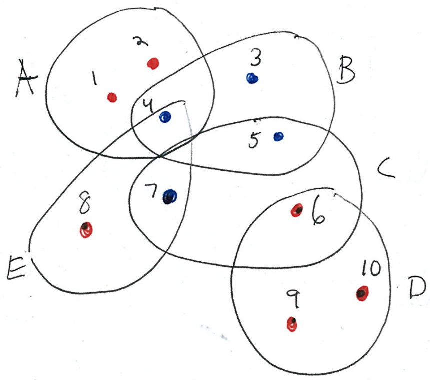
See the example from the class handout (next page). The corresponding log of execution and witness trees are given below:



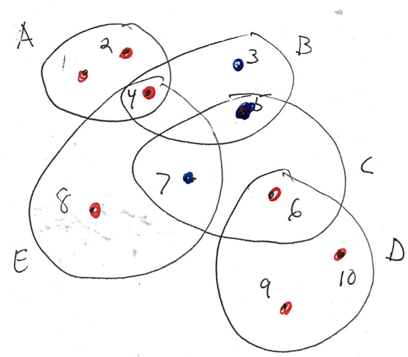
Note that the number of re-samplings is given by the number of trees. Next class, we will bound the number of possible trees and the probability of getting a large tree using indicator variables. This will show that the probability of not getting a non-monochromatic 2-coloring after a large number of re-samplings is extremely small, giving us the desired result.

Log: (1, B) (2, D) (3, A) (4, C) (5, E)

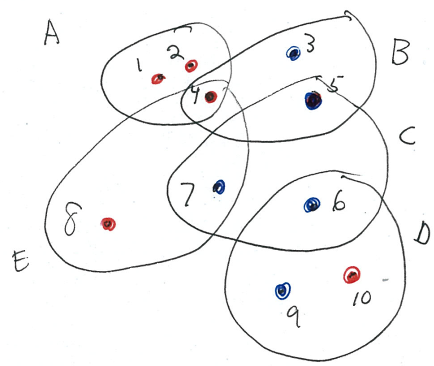
example time 0



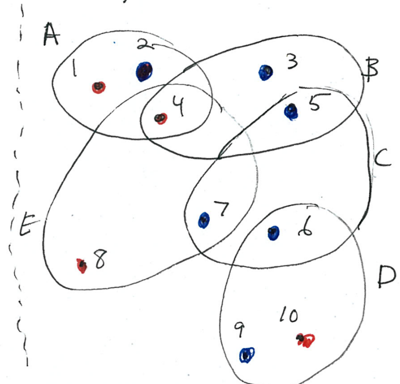
time 1
(1, B) resample edge B



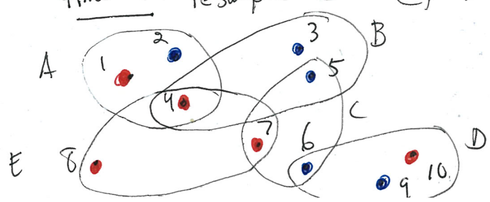
time 2
(2, D) resample D



time 3
(3, A) resample A



time 4 resample C (4, C)



time 5 resample E (5, E)