CMPUT 675: Randomized Algorithms	**Fall 2005**

Lecture 11: Oct 13

Lecturer: Mohammad R. Salavatipour — *Scribe: Tim Furtak*

11.1 Algorithmic version of LLL

Today we are going to see how to turn the LLL into an algorithm.

Suppose we have a set $\mathcal{F} = \{f_1, \ldots, f_n\}$ of independent random trials, each taking values in some domain of cardinality γ. Suppose we also have "bad" events $E_1, \ldots, E_m$, such that each E_i is determined by the values of the trials in $F_i \subseteq \mathcal{F}$. We say E_i intersects E_j iff $F_i \cap F_j \neq \emptyset$. Let d be the maximum number of other E_j's that any E_i intersects. Let $\omega = \max_i |F_i|$. We can typically assume that $\omega \leq d$ (but not always). Also, we assume that we can compute the *conditional* probability of every event E_i *fast*, if we have determined the outcomes of some of the trials in it. More formally we can compute $\Pr^*[E_i] = \Pr^*[E_i | f_{i_1} = \omega_{i_1}, \ldots, f_{i_k} = \omega_{i_k}]$ for any set of outcomes $\omega_{i_1}, \ldots, \omega_{i_k}$. Let us denote this probability by $\Pr^*(E_i)$. Then:

Theorem 11.1 (Molloy and Reed [1]) *If* $\Pr(E_i) \leq p$, *for* $1 \leq i \leq m$, *and* $pd^9 < O(1)$, *then we can find an assignment for all the* f_i*'s in randomized poly*$(m, \gamma^{\omega d \log\log m})$ *time such that* $\bigcap_{i=1}^m \overline{E_i}$ *holds.*

Remark: in the case of k-uniform hypergraph 2-coloring, we have $\gamma = 2$ and $t_1 \in O(k)$, $\omega \in O(k) \ll d$ where $d \approx 2^{\alpha k}$ with $\alpha \approx 1/50$.

Proof: We have three phases. In our first phase, we carry out the trials $f_1, f_2, \ldots$ sequentially. After each f_i we compute $\Pr^*(E_j)$, for each j such that $f_i \in F_j$. If $\Pr^*(E_j) > p^{\frac{2}{3}}$ we call E_j *dangerous*, undo f_i and freeze all the other trials in F_j and don't touch them until the next phase. So after the first phase has finished, we have a setting of some of the random trials in $\mathcal{F}$, such that:

$$\overset{*}{\Pr}(E_i) \leq p^{\frac{2}{3}}, \qquad 1 \leq i \leq m.$$

It can be easily verified, using the simple case of the Local Lemma, that there exists a setting for the remaining trials of $\mathcal{F}$, such that $\bigwedge_{1 \leq i \leq m} \overline{E_i}$. So there is an extension of this partial solution to a feasible solution. Note that we can easily deduce that the probability that each event E_i becomes dangerous is at most $p^{\frac{1}{3}}$.

Let G be the dependency graph of the problem, i.e. the graph whose vertices correspond to bad events E_i and two vertices E_i and E_j are adjacent iff $F_i \cap F_j \neq \emptyset$. Let us call a set of vertices C of G a $(1,2)$-tree if C is the set of vertices of a connected subgraph in the square of G, where the square of a graph is the graph in which two vertices are adjacent if they are at distance at most 2 in the original graph. We call a $(1,2)$-tree dangerous if all of its vertices correspond to dangerous events.

Observation: No event E_i intersects two events which belong to two different *maximal* dangerous $(1,2)$-trees, because otherwise these two events would be at distance 2 of each other and so these $(1,2)$-trees would have been merged.

Therefore we can deal with the frozen trials in each maximal dangerous $(1,2)$-tree independently.

Lemma 11.2 *With probability at least* $\frac{1}{2}$, *there is no dangerous* $(1,2)$*-tree of size greater than* $d \log 2m$.

Proof: Call $T \subseteq G$ a $(2,3)$-tree if the $E_i \in T$ are such that their mutual distances in G are at least 2, $(E_i, E_j) \in T$ if the distance of E_i and E_j in G is either 2 or 3, and the resulting graph is connected. It is easy to see that for each dangerous $(1,2)$-tree of size $u(d+1)$, there is a dangerous $(2,3)$-tree of size u, because from every set of $d+1$ vertices in a $(1,2)$-tree, consisting a vertex and its neighbors in G, we can select one vertex. It can be proved [2] that the number of $(2,3)$-trees of size u is at most $(ed^3)^u$. Also the vertices of any such tree correspond to independent vertices of G. So the probability of each being dangerous is independent from the others. Thus the probability that all the events of a $(2,3)$-tree of size u become dangerous is at most $(p^{\frac{1}{3}})^u$. So the expected number of these trees is at most $m(ed^3p^{\frac{1}{3}})^u$ which is less than $\frac{1}{2}$ for $u = \log 2m$. ■

Now, if $p^{\frac{2}{3}} d \log 2m \leq \frac{1}{2}$ then a random setting will avoid all the corresponding E_i's with probability at least $\frac{1}{2}$. Otherwise we repeat the first phase until there is no $(1,2)$-tree of size greater than $d \log 2m$. The expected number of times we have to do this is constant. Now in the second phase, we apply the same method to the frozen trials of each dangerous $(1,2)$-tree independently. Using similar arguments we find disjoint components, each having size at most $O(d(\log d + \log\log m))$. So the number of frozen trials f_i in each is at most $O(\omega d(\log d + \log\log m)) \in O(d^2(\log d + \log\log m))$. Note that in this case $p^{\frac{2}{3}} d \log 2m > \frac{1}{2}$ and so $d^2 \leq O((\log m)^{\frac{2}{5}})$. Therefore the number of trials in each component will be at most $O((\log m)^{\frac{2}{5}} \log\log m)$. So we can do an exhaustive search over all possible outcomes of each f_i in each component and this takes $O(\gamma^{(\log m)^{\frac{2}{5}} \log\log m})$. For the cases that $\gamma \in O(\log^c(m+n))$, for some constant c, this will give a polynomial time algorithm. ■

References

[1] M. Molloy and B. Reed, Further algorithmic aspects of the Local Lemma, *STOC 1998*.

[2] D. Knuth, The Art of Computer Programming, Vol I, *Addision Wesley* (1998).