The Lovász Local Lemma

Another way to argue that "nothing bad happens"

If $A_1 \ldots A_n$ are bad events

how do we know if there is positive probability that none occur?

usual way: Union bound

\[ \Pr[\bigcup A_i] \leq \sum \Pr[A_i] \]

no assumptions on $A_i$'s w.r.t. independence

if each $A_i$ occurs with prob $p$, then need $p < \frac{1}{n}$ to get anything interesting (i.e. sum $\leq 1$)

if $A_i$'s independent + "nontrivial":

\[ \Pr[\bigcup A_i] \leq 1 - \Pr[\bigcap \overline{A_i}] \]

\[ = 1 - \prod \Pr(\overline{A_i}) > 0 \]

What if $A_i$'s have "some" independence?

\[ \text{def } A \text{ "independent" of } B_1 \ldots B_k \text{ if } \forall J \subseteq [k] \]

\[ \Pr[ A \cap \bigcap_{j \in J} B_j ] = \Pr[A] \cdot \Pr[\bigcap_{j \in J} B_j] \quad J \neq \emptyset \]
def. $A_1 \ldots A_n$ events

$D \equiv (V; E)$ with $V = [n]$ is

"dependency digraph of $A_1 \ldots A_n"$

if each $A_i$ independent of all $A_j$ that don't
neighbor it in $D$ (i.e., all $A_j$ st. $(i, j) \notin E)$

---

**Lovász Local Lemma (Symmetric version)**

$A_1 \ldots A_n$ events st. $\Pr(A_i) \leq p \ \forall i$

with dependency digraph $D$ s.t. $D$ is of degree $\leq d$.

If $e(d + 1) \leq 1$ then

$$\Pr[\bigwedge_{i=1}^{n} \overline{A_i}] > 0$$

---

**Application:**

**Thm.** $S_1 \ldots S_m \subseteq S$, $|S_i| = l$, each $S_i$ intersects at most $d$ other $S_j$'s

before $m \leq 2^{l-1}$

now $m$ not restricted

if $e(d + 1) \leq 2^{l-1}$ then can $2$-color $S$ st. each $S_i$ not mono chromatic

**ie.** $H$ is a hypergraph with $m$ edges,

each containing $l$ nodes + each intersecting $\leq d$ other edges
\textbf{Prop.}\color{red}{^1}

color each elf of $S$ red/blue with prob $\frac{1}{2}$ iid.

$A_i$: event that $S_i$ monochromatic

$\Pr[A_i] = 2^{-(d-1)}$

$A_i$ ind of all $A_j$ st. $S_i \cap S_j = \emptyset$

depends on $\leq d$ other $A_j$

Since $e(p(d+1)) = e \frac{1}{2^{d-1}} (d+1) \leq 1$

\textit{LLL} $\Rightarrow$ exists a -coloring \hfill $\Box$

\textbf{Comparison:}

\begin{align*}
\# \text{ edges} &= m \\
\text{size of edge} &= d
\end{align*}

\begin{align*}
\# \text{ edges} &= m \\
\text{size of edge} &\geq d
\end{align*}

\begin{align*}
\text{each edge interacts} \\
\leq d \text{ others}
\end{align*}

\begin{align*}
d + 1 &\leq \frac{2^{d-1}}{e} \\
\text{no dependence on } m
\end{align*}

\textbf{A second application:}

Given CNF formula st. $d$ vars in each clause

$e \frac{(d+1)}{2^{d-1}} \leq 1$

there is a satisfying assignment
How do you find a solution?

Partial history:

Lovász 1975
non-constructive
(no fast algorithm to find soln)

Beck 1991
randomized algorithm
but for more restrictive conditions
on parameters

Moser 2009
negligible restrictions for SAT
" " " most problems

Moser Tardos

Then given $S_1, \ldots, S_m \subset S$

each $S_\ell$ intersects $\leq d$ other $S_\ell$‘s

if $e(d+1) \cdot C \leq 2^d$,
then can find 2-coloring of $S'$ s.t.
each $S_\ell$ not monochromatic
in time poly in $m/d$
Algorithm

- Color all els of $\mathcal{P}$ randomly (iid, uniform)

- While there is a monochromatic set $S_i$:
  - Pick arbitrary "violated" $S_i$
  - Randomly reassign colors to elements of $S_i$

Correctness: trivial ✓

Runtime: How many re-colorings? * see Q2

To analyze, define "witness tree" to explain why a certain event happened.

def. "log of execution" is a set of pairs $(i, s_i)(2, s_{i2})$...
where first entry is a "loop" number and second entry $s_{ij}$ is the set resampled at $j$th loop.

e.g. $(1, s_1) (2, s_2) (3, s_3) (4, s_4) (5, s_5) ...$
How many recolorings?

What independence properties do we have?

If $S_i \cap S_j = \emptyset$ then whether they are monochromatic is independent at all times.

If $S_i \cap S_j \neq \emptyset$ but, consider $\Pr[S_i \text{ 2-colored at time } t]$ and $\Pr[S_j \text{ 2-colored at time } u]$ such that there was a recoloring of $S_i \cap S_j$ at time $t < v < u$ then also independent!

Model as tree:

Recolorings of $S_i$ recolorings of connected component in this tree
Log: (1, B) (2, D) (3, A) (4, C) (5, E)

Example time 0

Time 1
- (1, B) resample edge

Time 2
- (2, D) resample D

Time 3
- (3, A) resample A

Time 4
- (4, C) resample C

Time 5
- resample E
A plan:

Show that for any "long" log, it is unlikely to happen.

Then

$$\Pr[\text{any long log occurs}] \leq (\# \text{ long logs}) \cdot (\text{max prob of a long log})$$

union bound

but need to be a bit more elaborate, may be show:

$$\Pr[\text{a log longer than size } t_0]$$

$$\leq \sum_{b > t_0} (\# \text{ logs of length } b) \cdot (\text{Prob of log of length } b)$$

still too many of these to do naively

Plan here:

Focus on point of view of each set $$S_i$$

- how labellings can evolve
  - not too many ways due to locality
  - each big one has low probability
def: "witness tree for step $j$" ($j \geq 0$)

is constructed as follows

- root vertex labelled by $S_{ij}$

- go backwards thru $\log^3$

- do for step $j$, $j-1$, $j-2$, ...

- if edge relabeled at current step $t$
  shares any nodes with edges already
  in witness tree,

any $S_{ij}$ can be added many
times to witness tree

$\Rightarrow$

add $S_{it}$ to witness tree
by making it point to
arbitrary node on witness tree
which is at max distance
from root

In our example:

- witness tree for time 1:
  
  witness tree for time 2:
  
  witness tree for time 3:
  
  time 4:
  
  time 5:
  
  time 6:
How do we bound probability of specific witness tree $\Upsilon$ in a run?

To analyze prob of tree $\Upsilon$, upper bound via "$\Upsilon$-check" procedure

i.e. ensure that:

- prob $\Upsilon$ occurs as a witness tree
- $\leq$ prob $\Upsilon$-check passes

**Def. $\Upsilon$-check procedure:**

- Visit nodes of $\Upsilon$ in reverse BFS order (max depth first)
- take random evaluations of vars in current set
- check that set is monochromatic (violated)
- pass if all checks are violated

<table>
<thead>
<tr>
<th>Vars</th>
<th>resampling</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Initial settings</th>
</tr>
</thead>
</table>

**Important Point**

Prob of violation $= 2^{-|E|} \equiv p$
Observe:

- If 2 sets at same level in tree,
  cant intersect! (by construction)
  \[ \Rightarrow \text{independent} \quad \text{(i.e., order of coin tosses doesn't matter)} \]

- If 2 sets at different levels,
  will resample & get totally new bits

\[ \Rightarrow \text{before later set} \Rightarrow \text{independent} \]

Note that we consider reverse BFS.

\[ \Pr [ \gamma \text{-check passes}] \leq \rho^{|\Gamma|} \]
\[ = \left(2^{-(d-1)} \right)^{|\Gamma|} \]

How to use the $\gamma$-check?

1) Prob of getting tree somewhere in log
   \[ \leq \text{prob of } \gamma \text{-check passing} \]

2) No tree occurs twice in log.
   (has to have previous tree as subtree!)

3) So, expected length of log
   \[ = \text{expected } \# \text{ of distinct trees in log} \]

\[ \gamma \text{ generality } \# 1 \]

\[ \gamma \text{ trees consistent with by: We are bounding prob of any of them} \]
\[ (\text{i.e., sum}) \]

\[ \gamma \text{ generality } \# 2 \]
\[ \gamma \text{ some of distinct trees can't even happen in our input, we are not a lot unirandom, more } \gamma \text{ happen} \]
Expected # of resamplings

\[ E[\# \text{ resamples}] = \sum_{\text{roots } T \text{ without } i} \sum_{\text{T rooted at } i} E[X_T] \text{ in execution of an algorithm} \]

where \( X_T = \begin{cases} 1 & \text{if } T \text{ occurs in} \\ 0 & \text{o.w.} \end{cases} \)

\[ = \sum_{\text{roots } T} \sum_{T \text{ rooted at } i} E[X_T] \]

\[ = m \sum_{s=1}^{\infty} \sum_{|T| = s} E[X_T] \]

\[ \leq m \sum_{s=1}^{\infty} \left( \frac{sd}{s-1} \right) p^s \]

\[ \leq m \sum_{s=1}^{\infty} \left( (d+1)e \right)^s p^s \]

since \( p < \frac{1-\epsilon}{e(d+1)} \) this is geometric sum \( \Theta(1) \)

\[ \text{if } p < \frac{1}{2e(d+1)} \text{ then goes down exponentially } \]

\[ \therefore E[\text{runtime}] = E[\# \text{ resamples}] \times \text{ time per resample} \]

is \( \text{ poly } (m, \ell, d) \)
Why?
How many labelled rooted trees of size $s$?

describe via Eularian tour (left→right):
write 1 if go down
0 if skip child
(2 for "pop up" is redundant)
then, each node contributes $d$ bits
String is $\leq sd$ characters with $s-1$ 1's
$\leq \left(\frac{sd}{s-1}\right)$ such strings
$\leq \left(\frac{ed}{s-1}\right)^{s-1} \propto (ed)^{s-1}$ by Stirling's approx

Example

$A$  $d$ children

$0, 1000, 1000$

$B$

$s = 3$

$\Rightarrow$

$1000, 1000, 0$

$\text{skip A's kids}$

$\text{skip B's kids}$

$\text{go to B}$

$\text{go to C}$

$\text{skip C's kids}$

$\text{skip B's kids}$

$\text{skip A's kids}$