Lovász Local Lemma

from probablistic to constructive

Yanheng Wang

Motivation

Quite a few combinatorics problems ask for existence of an object that satisfies constraint in the following form:

$$\overline{\bigvee_{B \in \mathcal{B}} B} = \bigwedge_{B \in \mathcal{B}} \overline{B}$$

where $\mathcal{B}$ is a family of "bad events", or undesired properties.

e.g. Colouring problem: Given a hypergraph $H = (V, E)$ and q colours, is there a colouring $\sigma : V \to [q]$ so that no edge $e \in E$ is monochromatic under σ?

- We are seeking for some $\sigma : V \to [q]$;
- For each $e \in E$, we create a bad event B_e, meaning that "$\sigma(e)$ is monochromatic." Let $\mathcal{B} := \{B_e \mid e \in E\}$;
- Now the constraint on σ can be expressed by $\bigwedge_{B \in \mathcal{B}} \overline{B}$.

The Lovász Local Lemma helps us identify *sufficient* conditions of the object's existence. For instance, we can apply it to Colouring and identify a condition that guarantees the existence of a proper colouring σ in H.

Remark. The condition it yields might not be necessary in most cases. Nonetheless, it demonstrates the power of *probablistic* argument, even if the original problem (e.g. Colouring) has nothing to do with randomness!

Basic Ideas

Goal: Prove existence of an object satisfying $\bigwedge_{B \in \mathcal{B}} \overline{B}$.

Idea: Impose a probability space; Prove that $\mathbb{P}(\bigcap_{B \in \mathcal{B}} \overline{B}) > 0$.

Clearly the idea implies the goal. But how can we achieve the idea? After all, the probability is typically difficult to calculate. Here are two possible solutions:

- Define some related random variables and bound their expectations (which is usually easier), then use the relation between expectation and probabilty.
- Compute $\mathbb{P}(B)$ for each $B \in \mathcal{B}$ (which is usually easy), and use them to bound $\mathbb{P}(\bigcap_{B \in \mathcal{B}} \overline{B})$. Lovász Local Lemma delves in the relation and makes it applicable *broadly*.

Compute $\mathbb{P}(B)$ for each $B \in \mathcal{B}$ (which is usually easy), and use them to bound $\mathbb{P}(\bigcap_{B\in\mathcal{B}} \overline{B})$. Lovász Local Lemma delves in the relation and makes it applicable *broadly*.

Now consider the following trivial scenario. Suppose we have computed $\mathbb{P}(B)$ for each individual $B \in \mathcal{B}$. In addition, we know the events in class $\mathcal{B}$ are *mutually independent*. Then, it is trivial to deduce

$$\mathbb{P}\left(\bigcap_{B\in\mathcal{B}} \overline{B}\right) = \prod_{B\in\mathcal{B}} (1 - \mathbb{P}(B)).$$

Next comes the question: what if the events in class $\mathcal{B}$ are *not* mutually independent?

Compute $\mathbb{P}(B)$ for each $B \in \mathcal{B}$ (which is usually easy), and use them to bound $\mathbb{P}(\bigcap_{B \in \mathcal{B}} \overline{B})$. Lovász Local Lemma delves in the relation and makes it applicable *broadly*.

Notation. Given a graph (V, E) and $v \in V$, we denote $\partial v := \{u \in V \mid (v, u) \in E\}$. We also denote $\partial^\bullet v := \partial v \cup \{v\}$.

Definition. Given a family of events, $\mathcal{B}$, we call the graph $G = (\mathcal{B}, \mathcal{D})$ its dependency graph if B and $\mathcal{B} \setminus \partial^\bullet B$ are mutually independent for all $B \in \mathcal{B}$.

Remark. Intuitively, $\mathcal{D}$ captures the dependency of events; adjacent events are possibly dependent while non-adjacent ones are independent.

Compute $\mathbb{P}(B)$ for each $B \in \mathcal{B}$ (which is usually easy), and use them to bound $\mathbb{P}(\bigcap_{B\in\mathcal{B}} \overline{B})$. Lovász Local Lemma delves in the relation and makes it applicable *broadly*.

Lovász Local Lemma. Fix $\mathcal{B}$ and its dependency graph. If there is an assignment $p : \mathcal{B} \to (0, 1)$ such that

$$\mathbb{P}(B) \leq p(B) \prod_{B'\in\partial B} (1 - p(B'))$$

for all $B \in \mathcal{B}$, then we have

$$\mathbb{P}\left(\bigcap_{B\in\mathcal{B}} \overline{B}\right) \geq \prod_{B\in\mathcal{B}} (1 - p(B)) > 0.$$

Remark. This lemma could bound $\mathbb{P}(\bigcap_{B\in\mathcal{B}} \overline{B})$ even if the events in the family were not mutually independent, thus lending itself broadly applicable. But please be noted that if $|\partial B|$ is too large (i.e. the dependency is too high), then the condition is likely to break.

An Inductive Proof

Lovász Local Lemma. Fix $\mathcal{B}$ and its dependency graph. If there is an assignment $p : \mathcal{B} \to (0, 1)$ such that

$$\mathbb{P}(B) \leq p(B) \prod_{B' \in \partial B} (1 - p(B'))$$

for all $B \in \mathcal{B}$, then we have

$$\mathbb{P}\left(\bigcap_{B \in \mathcal{B}} \overline{B} \right) \geq \prod_{B \in \mathcal{B}} (1 - p(B)) > 0.$$

We shall prove a stronger claim:

Claim. Under the same condition of the lemma, we have $\mathbb{P}(\bigcap_{B \in \mathcal{B}} \overline{B}) \geq (1 - p(B_\star)) \cdot \mathbb{P}(\bigcap_{B \in \mathcal{B} \setminus \{B_\star\}} \overline{B})$ for any $B_\star \in \mathcal{B}$.

Proof. By induction on $|\mathcal{B}|$.

$\underline{|\mathcal{B}| = 1.}$ LHS $= 1 - \mathbb{P}(B_\star)$; RHS $= 1 - p(B_\star)$;
By condition of the lemma, LHS $\geq$ RHS.

Claim. Under the same condition of the lemma, we have $\mathbb{P}(\bigcap_{B\in\mathcal{B}} \overline{B}) \geq (1 - p(B_\star)) \cdot \mathbb{P}(\bigcap_{B\in\mathcal{B}\setminus\{B_\star\}} \overline{B})$ for any $B_\star \in \mathcal{B}$.

$|\mathcal{B}| \geq 2$. To simplify notation, let $\mathcal{B}_\star := \mathcal{B} \setminus \{B_\star\}$. Then,

$$\mathbb{P}\left(\bigcap_{B\in\mathcal{B}} \overline{B}\right) = \mathbb{P}\left(\bigcap_{B\in\mathcal{B}_\star} \overline{B}\right) - \mathbb{P}\left(B_\star \cap \bigcap_{B\in\mathcal{B}_\star} \overline{B}\right).$$

$$\bullet \leq \mathbb{P}\left(B_\star \cap \bigcap_{B\notin\partial\bullet B_\star} \overline{B}\right) = \mathbb{P}(B_\star) \cdot \mathbb{P}\left(\bigcap_{B\notin\partial\bullet B_\star} \overline{B}\right)$$

Now let us bridge the gap between $\bullet$ and $\bullet$. This is easy: just incrementally add events in $\partial B_\star$ to $\bullet$. In each step, we use I.H. to obtain a relation ("*telescoping*"). Thus,

$$\bullet \leq \bullet \cdot \frac{1}{\prod_{B\in\partial B_\star}(1 - p(B))}$$

Therefore, by the condition of the lemma, we obtain $\bullet \leq \bullet \cdot p(B_\star)$. Plugging it into the first equation proves the claim. ■

Applcation to Colouring

Lovász Local Lemma. Fix $\mathcal{B}$ and its dependency graph. If there is an assignment $p : \mathcal{B} \to (0, 1)$ such that

$$\mathbb{P}(B) \leq p(B) \prod_{B' \in \partial B} (1 - p(B'))$$

for all $B \in \mathcal{B}$, then we have

$$\mathbb{P}\left(\bigcap_{B \in \mathcal{B}} \overline{B}\right) \geq \prod_{B \in \mathcal{B}} (1 - p(B)) > 0.$$

Probability space. We colour each vertex v randomly and independently. Formally, σ is a uniform random vector on the space $[q]^V$.

Bad events. As before, we define B_e to be the event that $\sigma(e)$ is monochromatic. $\mathcal{B} := \{B_e \mid e \in E\}$.

Dependency graph. According to our probability space, $(B_e, B_{e'}) \in \mathcal{D} \iff e \cap e' \neq \emptyset$.

Lovász Local Lemma. Fix $\mathcal{B}$ and its dependency graph. If there is an assignment $p : \mathcal{B} \to (0, 1)$ such that

$$\mathbb{P}(B) \leq p(B) \prod_{B' \in \partial B} (1 - p(B'))$$

for all $B \in \mathcal{B}$, then we have

$$\mathbb{P}\left(\bigcap_{B \in \mathcal{B}} \overline{B} \right) \geq \prod_{B \in \mathcal{B}} (1 - p(B)) > 0.$$

Suppose that $\forall e \in E$ we have (1) $|e| = k$; (2) e intersects with at most d other edges. A direct calculation shows that $\mathbb{P}(B_e) = q \cdot q^{-k} = q^{1-k}$; in addition, $|\partial B_e| \leq d$.

We could pick p as follows: $p(B_e) := 1/(1 + d)$ for all $e \in E$. After some computation, we conclude that $\mathbb{P}(B) \leq p(B) \prod_{B' \in \partial B}(1 - p(B'))$ *whenever* $e(d + 1) \leq q^{k-1}$, where $e = 2.718....$ Hence by the lemma, under the condition $e(d + 1) \leq q^{k-1}$, there must be a proper colouring in H.

What Next?

It's remarkable to derive existential conditions via Lovász Local Lemma. But reassuring the existence is one thing, describing its appearance is another...
Can we construct the solution explicitly? Say, can we construct the proper colouring σ so long as $e(d+1) \leq q^{k-1}$?

The lemma itself is probablistic. It doesn't show us an obvious algorithmic way to construct such a solution. So, it turns out to be a major breakthrough when Moser and Tardos accomplished this in 2010 with very mild restriction.

Constructive Version of LLL

The orginal version of LLL doesn't include the concept of a "solution". So, before we get into the constructive counterpart, we must model a solution first.

Perhaps the most natural way is to introduce a set of variables, $\mathcal{X}$. Each variable takes value from its prescribed domain. A *solution* of the problem is merely an evaluation (or combination if you like) of all these variables that satisify the constraint $\bigwedge_{B \in \mathcal{B}} \overline{B}$.

Remark. The variables here are deterministic, since we are modelling a solution for a combinatorics problem where no randomness is involved. (e.g. Colouring, LP, SAT, and so on.) We do use randomness later, however, to *perform arguments*.

Now we shall make explicit the connections between variables and events.

Definition. Let $x \in \mathcal{X}$ and $B \in \mathcal{B}$. We say x underlies B if the change in value x could result in different outcomes of B. We denote by $\mathrm{vbl}(B)$ the set of variables that underlies B.

e.g. In the Colouring example, $\mathcal{X} = \{x_v \mid v \in V\}$. The domain for each x_v is $[q]$. Then $\mathrm{vbl}(B_e) = \{x_v \mid v \in e\}$, since the colours of remote vertices have nothing to do with the outcome of B_e.

From this point on, we introduce randomness. First, we *impose on* each $x \in \mathcal{X}$ a distribution; Second, we *require* that all variables in $\mathcal{X}$ are mutually independent. There's no restriction other than these two, so it's basically free to design the probability space (i.e. the detailed distributions).

Once the probability space is settled, we can readily port the definition of dependency graph to the current context.

Definition. Fix a set of (random) variables $\mathcal{X}$ that conform to the requirements above, and a family of events $\mathcal{B}$. We call the graph $G = (\mathcal{B}, \mathcal{D})$ a dependency graph, where $\mathcal{D} := \{(B, B') \in \binom{\mathcal{B}}{2} \mid \mathrm{vbl}(B) \cap \mathrm{vbl}(B') \neq \emptyset\}$.

LLL, Constructive Version. Fix $\mathcal{X}$, $\mathcal{B}$ and its dependency graph. If there is an assignment $p : \mathcal{B} \to (0, 1)$ such that

$$\mathbb{P}(B) \leq p(B) \prod_{B' \in \partial B} (1 - p(B'))$$

for all $B \in \mathcal{B}$, then the following algorithm produces a solution in expected time $O\left(\sum_{B \in \mathcal{B}} \frac{p(B)}{1-p(B)}\right)$:

foreach $x \in \mathcal{X}$ **do**
  sample x independently according to its distribution
while $\exists B \in \mathcal{B} : B[X]$ is true **do**
  take such a B arbitrarily
  foreach $x \in \mathrm{vbl}(B)$ **do**
    sample x independently according to its distribution
return the current evaluation of all $x \in \mathcal{X}$

Remark. The time is charged in steps of *samplings*. Thus it's the user's responsibilty to provide an efficient sampling subroutine.

```
foreach x ∈ 𝒳 do
    sample x independently according to its distribution
while ∃B ∈ ℬ : B[X] is true do
    take such a B arbitrarily
    foreach x ∈ vbl(B) do
        sample x independently according to its distribution
return the current evaluation of all x ∈ 𝒳
```

Observe that the algorithm always produces a legal solution whenever it terminates. The hard part is: Why does it terminate (and even terminates quickly)? We will analyse its behaviour in the rest of the slides.

Remark. The lemma says that the algorithm terminates quickly *in expectation*. This is strong enough for proving its termination under all consequences: If there were some possitive probability that it doesn't terminate, then the expectation would have diverged.

For each $x \in \mathcal{X}$, define an infinite i.i.d. random sequence $(x^{(t)})$ that has the same distribution as x. So we have $|\mathcal{X}|$ sequences in total, all of which are independent. They serve as the random source to drive our algorithm.

x_1	$x_1^{(1)}, x_1^{(2)}, x_1^{(3)}, \ldots$
x_2	$x_2^{(1)}, x_2^{(2)}, x_2^{(3)}, \ldots$
$\vdots$	
x_n	$x_n^{(1)}, x_n^{(2)}, x_n^{(3)}, \ldots$

Every time the algorithm decides to sample $x \in \mathcal{X}$, it goes to the random sequence $(x^{(t)})$ and retrieve the next fresh variable.

Remark. We won't do this in a practical implementation, of course. The notion of infinite random source is for analysis purpose only. It is clearly equivalent to generating random variables on demand.

So long as the values of the random source are determined, the trajectory of the algorithm is uniquely determined; Put it in another way, the trajectory of the algorithm is a random variable depends on the random source only.

Definition. The *log* of the algorithm is a list $L \in \mathcal{B}^* \cup \mathcal{B}^\infty$ that drops down the chosen event B in each round of execution. It could be finite or infinite, depending on whether or not the algorithm terminates. As noted above, L is a random list depends solely on the random source.

e.g. $L = (B_3, B_2, B_5, B_2)$ means that the algorithm considers B_3, B_2, B_5 and B_2 in its first four rounds of execution, and terminates afterwards.

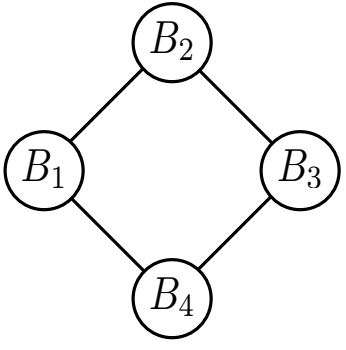
Definition. We define N_B to be the number of occurrences of B in L. Ultimately, we will bound $\mathbb{E}(N_B)$ for all $B \in \mathcal{B}$, then the analysis would be complete.

A key concept in the analysis is defined below.

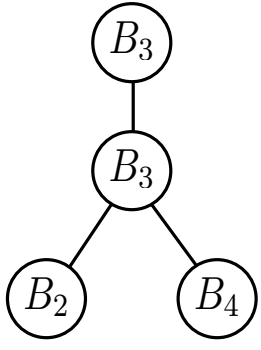
Definition. A rooted tree is called a *witness tree* if

1. Each vertex v is associated with a label $\sharp v \in \mathcal{B}$;
2. For any vertex v, its children have distinct labels;
3. For any vertex v and any of its child u, $(\sharp v, \sharp u) \in \mathcal{D}$.

e.g.



Dependency Graph



A Witness Tree

This definition seems somewhat technical. But it becomes natural later.

Definition. A rooted tree is called a *witness tree* if

1. Each vertex v is associated with a label $\sharp v \in \mathcal{B}$;
2. For any vertex v, its children have distinct labels;
3. For any vertex v and any of its child u, $(\sharp u, \sharp v) \in \mathcal{D}$.

Let L be the log of the algorithm. We construct a sequence of witness trees (T_t). The procedure for constructing T_t is

```
the root of T_t is labelled L[t]
for i = t − 1, ..., 1 do
    if ∃v ∈ T_t : (L[i], ♯v) ∈ D then
        v := the deepest one with such property
        attach to v a new child u, with ♯u := L[i]
```

Lemma. Suppose vertices $u, v \in T_t$ have depths d_u and d_v, respectively. If $d_u \geq d_v$, then either $(\sharp u, \sharp v) \notin \mathcal{D}$, or u is attached later than v. ■

Corollary. If $d_u = d_v$, then $(\sharp u, \sharp v) \notin \mathcal{D}$. In specific, $\sharp u \neq \sharp v$. Therefore, requirement 2 is satisfied and T_t is indeed a witness tree. ■

Definition. A rooted tree is called a *witness tree* if

1. Each vertex v is associated with a label $\sharp v \in \mathcal{B}$;
2. For any vertex v, its children have distinct labels;
3. For any vertex v and any of its child u, $(\sharp v, \sharp u) \in \mathcal{D}$.

Lemma. Any witness tree $T_\star$ cannot appear twice in (T_t).
Proof. Suppose it appears twice, say at time $t_1 < t_2$. Then according to the procedure $L[t_1] = L[t_2]$, hence T_{t_2} must be strictly bigger than T_{t_1}, a contradiction. ■

Corollary. Let $\mathcal{T}_B$ be the collection of all witness trees with root label B, then $N_B = \sum_{T_\star \in \mathcal{T}_B} \mathbf{1}[T_\star \text{ occurs in } (T_t)]$.
This directly implies a way of bounding $\mathbb{E}(N_B)$:

$$\mathbb{E}(N_B) = \sum_{T_\star \in \mathcal{T}_B} \mathbb{P}(T_\star \text{ occurs in } (T_t)) \qquad \blacksquare$$

Theorem. Fix a witness tree $T_\star$, then $\mathbb{P}(T_\star \text{ occurs in } (T_t))$ is at most $\prod_{v \in T_\star} \mathbb{P}(\sharp v)$.

Proof. If $T_\star$ occurs in (T_t), say at time t, let us read information from the structure of $T_\star = T_t$. We first list the vertices of $T_\star$ in depth-decreasing order: $v_1, v_2, \ldots, v_n$ with $d_1 \geq d_2 \geq \cdots \geq d_n = 0$. Define i_j to be the iteration v_j was added to T_t; be alert that it's a random variable.
For each v_j we know $\sharp v_j = L[i_j]$, so the event $\sharp v_j$ happened at time i_j in the algorithm, whose probability is $\mathbb{P}(\sharp v_j)$. But the algorithm immediately resamples all $x \in \text{vbl}(\sharp v_j)$ by fresh randomness, which effectively cuts off the connections between the current event $\sharp v_j$ and all events $\sharp v_{j+1}, \ldots, \sharp v_n$. (Recall that $d_j \geq d_k$ imples either $i_j < i_k$ or $\sharp v_j$ has nothing to do with $\sharp v_k$.) Therefore, when we crawl along the vertices in order, we get the overall probability $\prod_{j=1}^{n} \mathbb{P}(\sharp v_j)$, proving the theorem. ■

Problem. We use informal descriptions when we argue that $\sharp x_{j+1}, \ldots, \sharp x_n$ are independent of the current event $\sharp x_j$. Please formalise the argument by rephrasing it in the language of random source. (*Hint: At the time i_k, what random variables drove the event $\sharp x_k$ to happen?*)

Problem. Try to simplify the proof by listing the vertices in reverse order they were attached to T_t.

Now that we know

$$\mathbb{E}(N_B) \leq \sum_{T_\star \in \mathcal{T}_B} \prod_{v \in T_\star} \mathbb{P}(\sharp v)$$

How can we bound it? We do so by introducing a *conceptual* random process.

the root of the tree is labelled B
$d := 0$
repeat
  foreach v at depth d **do**
    foreach $B' \in \partial(\sharp v)$ **do**
      with probability $p(B')$, attach to v a child u, with $\sharp u := B'$
  $d := d + 1$
until there is no vertex at depth d;

Keep in mind that the probability space here is a brand new one. The process strictly follows the definition of a witness tree, so it's of course closely related with $\mathcal{T}_B$.

the root of the tree is labelled B
$d := 0$
repeat
  foreach v at depth d **do**
    foreach $B' \in \partial^\bullet(\sharp v)$ **do**
      with probability $p(B')$, attach to v a child u, with $\sharp u := B'$
  $d := d + 1$
until there is no vertex at depth d;

Observations.

- If it terminates, the process generates a witness tree in $\mathcal{T}_B$.
- Any $T_\star \in \mathcal{T}_B$ has some probability of being generated, in specific, the probability is

$$q(T_\star) := \frac{1 - p(B)}{p(B)} \prod_{v \in T_\star} p(\sharp v) \prod_{B' \in \partial(\sharp v)} (1 - p(B'))$$

By direct calculation

$$\geq \frac{1 - p(B)}{p(B)} \prod_{v \in T_\star} \mathbb{P}(\sharp v)$$

By assumption of LLL

$$q(T_\star) \geq \frac{1-p(B)}{p(B)} \prod_{v \in T_\star} \mathbb{P}(\sharp v)$$

$$\mathbb{E}(N_B) \leq \sum_{T_\star \in \mathcal{T}_B} \prod_{v \in T_\star} \mathbb{P}(\sharp v)$$

Therefore,

$$\begin{aligned}
\mathbb{E}(N_B) &\leq \sum_{T_\star \in \mathcal{T}_B} \frac{p(B)}{1-p(B)} q(T_\star) \\
&= \frac{p(B)}{1-p(B)} \sum_{T_\star \in \mathcal{T}_B} q(T_\star) \\
&\leq \frac{p(B)}{1-p(B)}
\end{aligned}$$

Remark. The key technique here is to relate the product term with a probability that resides in an entirely different probability space.

Witness Tree, Revisited

Definition. A rooted tree is called a *witness tree* if

1. Each vertex v is associated with a label $\sharp v \in \mathcal{B}$;
2. For any vertex v, its children have distinct labels;
3. For any vertex v and any of its child u, $(\sharp v, \sharp u) \in \mathcal{D}$.

Let's review the definition of a witness tree. Requirements 2 and 3 are indeed *designed* to make the probability q in the conceptual process has a factor $p(\cdot)\prod_{B'\in\partial\cdot}(1-p(B'))$, so that we could use the assumption of LLL to bridge the two probability space.

One might ask why we work on (T_t) instead of on L directly. Indeed, it's easy to bound the probability $\mathbb{P}(L = L_\star)$ for any $L_\star$. However, it's hard to generate a legal $L_\star$. A naïve approach of generating all sequences in $\mathcal{B}^* \cup \mathcal{B}^\infty$ doesn't work because legal $L_\star$'s embed very sparsely.