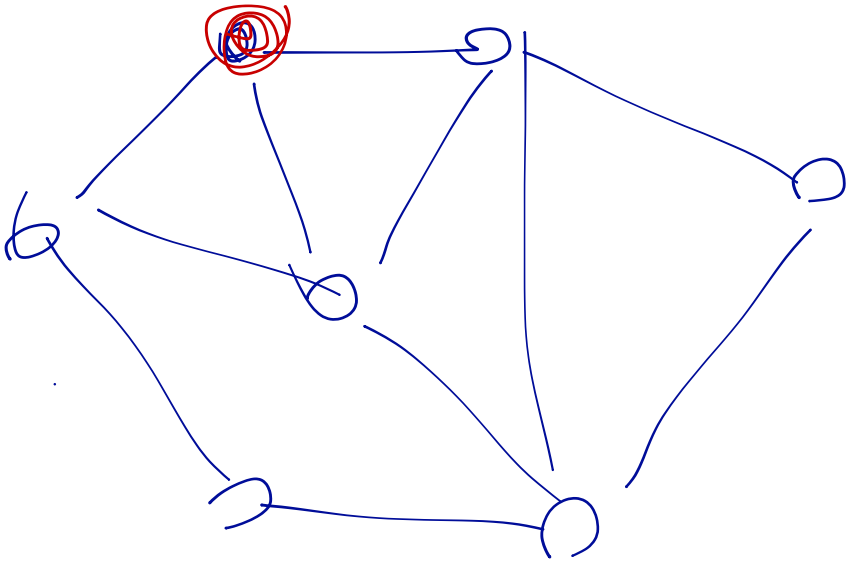
Distributed LLL

Saeed Amiri

Theorem 1.2. *Let $\mathcal{P}$ be a finite set of mutually independent random variables in a probability space. Let $\mathcal{A}$ be a finite set of events determined by these variables. If there exists an assignment of reals $x : \mathcal{A} \to (0, 1)$ such that*

$$\forall A \in \mathcal{A} \; : \; \Pr[A] \; \leq \; x(A) \prod_{B \in \Gamma_{\mathcal{A}}(A)} (1 - x(B)),$$

Assymetrics



$B \in \Gamma_{\mathcal{A}}(A)$

then there exists an assignment of values to the variables $\mathcal{P}$ not violating any of the events in $\mathcal{A}$. Moreover the randomized algorithm described above resamples an event $A \in \mathcal{A}$ at most an expected $x(A)/(1 - x(A))$ times before it finds such an evaluation. Thus the expected total number of resampling steps is at most $\sum_{A \in \mathcal{A}} \frac{x(A)}{1-x(A)}$.

What does it says? If probability of each bad event relates to its neighborhood, that is larger the neighborhood lower the probability, then it is possible to avoid all bad events

---

In particular for $P[A] \leq x(A) \prod_{B \in N(A) \text{ or } \Gamma(A)} (1 - x(B))$

we have a constructive algorithm to satisfy all events with runnintime $f\left(\frac{x(A)}{1-x(A)}\right)$

Alg is simple

---

function sequential_lll($\mathcal{P}, \mathcal{A}$)
  for all $P \in \mathcal{P}$ **do**
    $v_P \leftarrow$ a random evaluation of P;
  while $\exists A \in \mathcal{A} : A$ is violated when $(P = v_P : \forall P \in \mathcal{P})$ **do**
    pick an arbitrary violated event $A \in \mathcal{A}$;
    for all $P \in \text{vbl}(A)$ **do**
      $v_P \leftarrow$ a new random evaluation of P;
  return $(v_P)_{P \in \mathcal{P}}$;

We have to prove the algorithm terminates, more especifically we have to show each A will be sampled atmost $\frac{x(A)}{1-x(A)}$

Proof Strategy: Big idea

1- Construct an execution LOG for the alg

2- Construct a tree τ from LOG to keep track of each resampling (witness tree). Show that $P(\tau \text{ appears in LOG})$ is proportional to $\prod_{A \in \tau} Pr(A)$

3- Construct a stochastic process $\mathbb{P}$ to generate witness trees at random and show $\frac{1-x(A)}{x(A)} \prod_{B \in \tau(A)} x(B) < P(\tau \text{ appears in } \mathbb{P})$

4- from 2 and 3 : $E(\#A) \leq \sum Pr(\tau_A \text{ appears}) \leq \sum_{\tau \in \mathcal{T}_A} \prod_{B \in \tau} P[B] \leq \sum_{\tau \in \mathcal{T}_A} \frac{x(A)}{1-x(A)} P(\tau \text{ appears in } \mathbb{P}) \leq \frac{x(A)}{1-x(A)}$

Then we have the distributed version of the algorithm

function parallel_lll($\mathcal{P}, \mathcal{A}$)
  for all $P \in \mathcal{P}$ **do in parallel**
    $v_P \leftarrow$ a random evaluation of P;
  while $\exists A \in \mathcal{A} : A$ is violated when $(P = v_P : \forall P \in \mathcal{P})$ **do**
    $S \leftarrow$ a maximal independent set in the subgraph of $G_{\mathcal{A}}$ induced by all events which are violated when $(P = v_P : \forall P \in \mathcal{P})$, constructed in parallel;
    for all $P \in \bigcup_{A \in S} \mathrm{vbl}(A)$ **do in parallel** (1)
      $v_P \leftarrow$ a new random evaluation of P;
  return $(v_P)_{P \in \mathcal{P}}$;

(1)

And corresponding distributed theorem:

Theorem 1.3. *Let $\mathcal{P}$ be a finite set of mutually independent random variables in a probability space. Let $\mathcal{A}$ be a finite set of events determined by these variables. If $\varepsilon > 0$ and there exists an assignment of reals $x : \mathcal{A} \to (0, 1)$ such that*

$$\forall A \in \mathcal{A} \ : \ \Pr[A] \ \leq \ (1-\varepsilon)x(A) \prod_{B \in \Gamma_{\mathcal{A}}(A)} (1 - x(B)),$$

then the parallel version of our algorithm takes an expected $O(\frac{1}{\varepsilon} \log \sum_{A \in \mathcal{A}} \frac{x(A)}{1-x(A)})$ steps before it finds an evaluation violating no event in $\mathcal{A}$.

what's difference here?! we have $(1-\varepsilon)$ in probability evaluations it is not just about neighborhood.

How $(1-\varepsilon)$ plays a role: It helps us to bound the #times (1) happens: Probability it goes to depth k roughly speaking is $\leq (1-\varepsilon)^k$ so for $k \in O(\lg n)$. this won't happen w.h.p $(1-\varepsilon)^{\log n} = n^{-\varepsilon}$

We prove the previous claim, by showing that if distributed Alg. goes to that depth, then there is a witness tree τ of at least k vertice. But then with previous achievments we know probability of this happen $\leq \prod_{v \in \tau} (1-\varepsilon) P([v]) \leq (1-\varepsilon)^k$

($[v]$ label of v)

(talk why the running time is $O(\lg^2 n)$)

From now on we proceed as follows:

1) Define witness tree properly

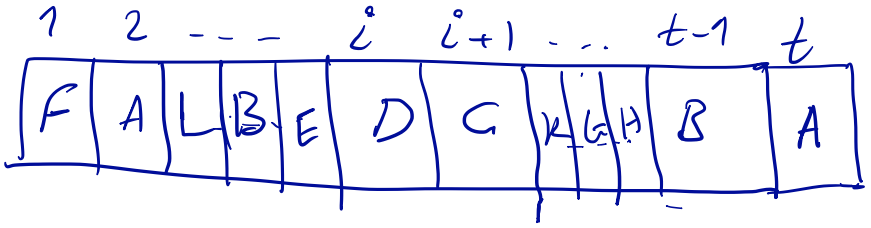
2) Prove $P(\tau \text{ in LOG}) \leq \prod_{v \in \tau} P([v])$

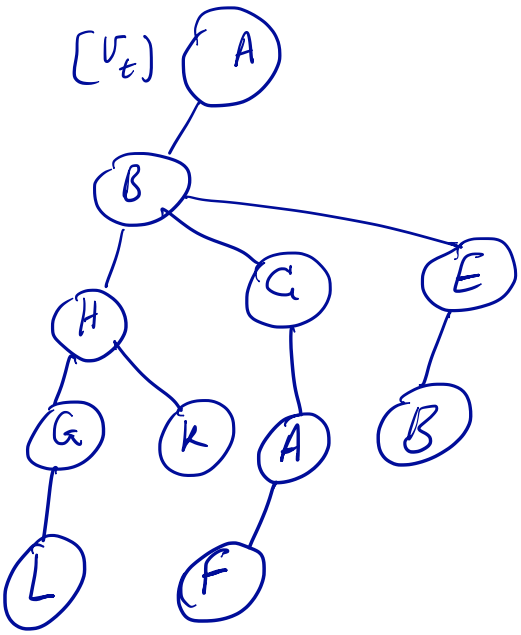
3) Define the stocastic process to generate a random witness tree.

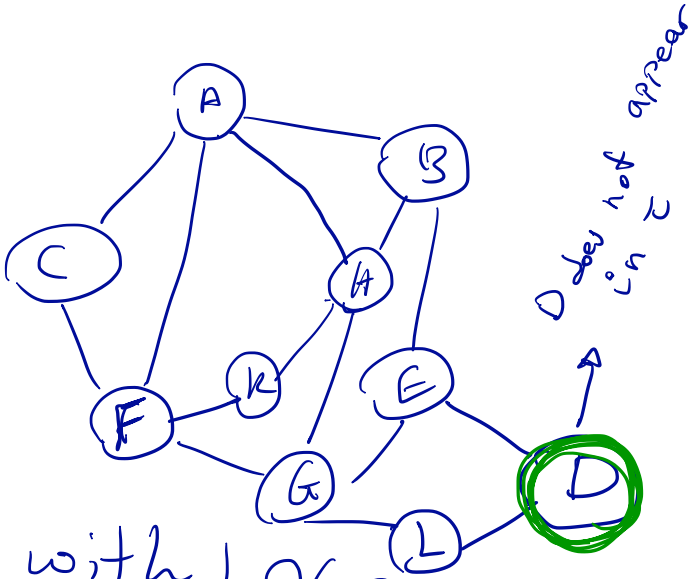
4) from 2 and 3 Conclude each $A \in \mathcal{A}$ appears in expectation $\frac{x(A)}{1-x(A)}$ times in the LOG

5) Prove in distributed version after kth call the depth of tree is $k-1$

witness tree τ at time t:







Start with $LOG[t]$ (paper calls it $C(t)$) make it root and construct $\tau^{(t)}$.

in each step after that construct τ^i from τ^{i+1} by adding $C(i)$ to farmost $v \in \tau^{i+1}$ (from root) where $C(i) \in \Gamma(v)$. If there is no such v, ignore it

Proper witness tree: τ is proper if for $v \in \tau$, $u, w \in$ children(v) label$(u) \neq$ label(v)

Lemma 2.1. *Let τ be a fixed witness tree and C the (random) log produced by the algorithm.*

(i) *If τ occurs in C, then τ is proper.*

(ii) *The probability that τ appears in C is at most $\prod_{v \in V(\tau)} \Pr[[v]]$.*

τ is proper because for $v \in \tau$, children of v are independent (ask why)

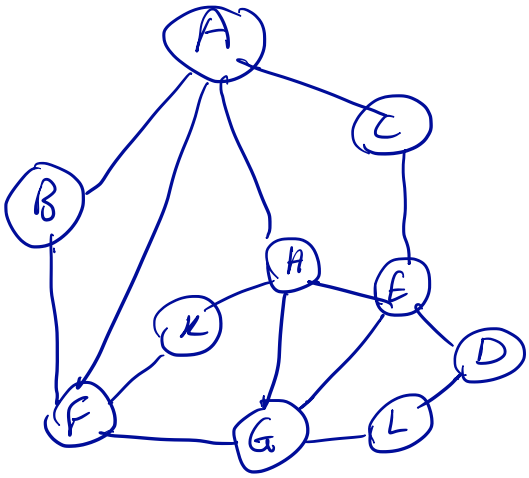
Consider the following procedure that we call τ-check: In an order of decreasing depth (e.g., reversed breadth first search order) visit the vertices of τ and for a vertex v take a random evaluation of the variables in vbl($[v]$) (according to their distribution, independent of possible earlier evaluations) and check if the resulting evaluation violates $[v]$. We say that the τ-check passes if all events were violated when checked.

τ check passes w.p. $\prod_{v \in \tau} P([v])$, τ appears with Probability $\leq P(\tau$ check passes$)$, because whenever τ appears τ check passes if it runs on a exact probability distribution

How to relate two probability distribution, needs some more arguments, we leave it as an exercise.

Creating a witness tree with root A:

Start with A as root, let v be the last vertex added to τ in step i. In step i+1 add vertex



$v' \in T^+(v)$ ($T^+(v)$ = closed Neighborhood of v) with probability $x([v'])$ or skip it with Prob: $1-x([v'])$

Define $x'(B) = x(B) \prod_{C \in T(B)} (1 - x(C))$

Lemma 3.1. *Let τ a fixed proper witness tree with its root vertex labelled A. The probability p_τ that the Galton-Watson process described above yields exactly the tree τ is*

$$p_\tau = \frac{1 - x(A)}{x(A)} \prod_{v \in V(\tau)} x'([v]).$$

By the above Lemma(homework) and prev lemma we conclude Thm 1.2:

Let $\mathcal{T}_A$ denote the set of all proper witness trees having the root labelled A. We have

$$\mathbb{E}(N_A) = \sum_{\tau \in \mathcal{T}_A} \Pr[\tau \text{ appears in the log } C] \leq \sum_{\tau \in \mathcal{T}_A} \prod_{v \in V(\tau)} \Pr[[v]] \leq \sum_{\tau \in \mathcal{T}_A} \prod_{v \in V(\tau)} x'([v]),$$

where the first inequality follows from Lemma 2.1, while the second follows from the assumption in Theorem 1.2. We further have

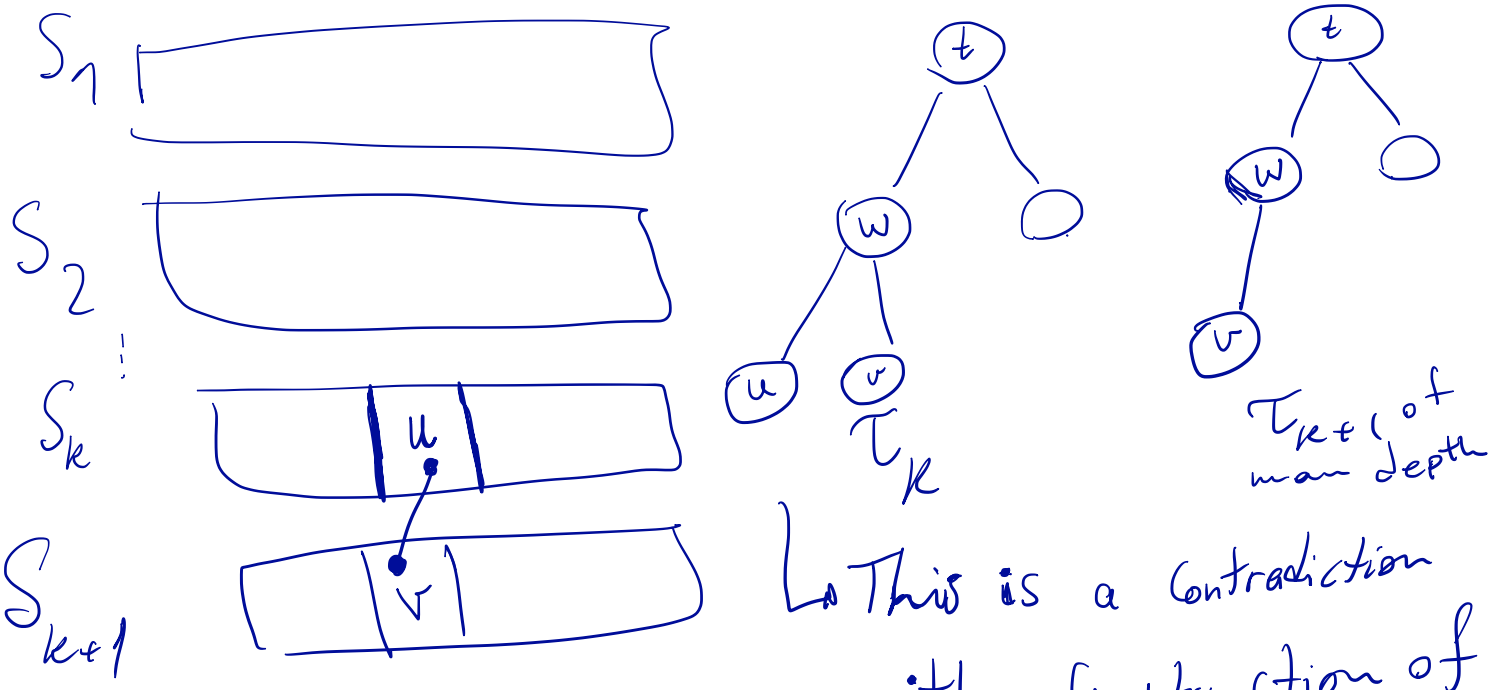
$$\mathbb{E}(N_A) \leq \sum_{\tau \in \mathcal{T}_A} \prod_{v \in V(\tau)} x'([v]) = \frac{x(A)}{1 - x(A)} \sum_{\tau \in \mathcal{T}_A} p_\tau \leq \frac{x(A)}{1 - x(A)},$$

where the equality comes from Lemma 3.1, while the last inequality follows from the fact that the Galton-Watson process produces exactly one tree at a time (not necessarily one from $\mathcal{T}_A$ since it might also grow infinite). This concludes the proof of Theorem 1.2. □

To analyze parallel algorithm, first we simulate it by sequential means: order vertices appearing in jth call arbitrarily as S_j:

Lemma 4.1. *If $t \in S_j$, then the depth of $\tau_C(t)$ is $j-1$.*

Proof. Let t_k be the first number in the segment S_k and let $\tau_k = \tau_C^{(t_k)}(t)$ for $k \le j$. As the events resampled in the j-th parallel step are independent, the root is the only vertex of τ_j. For $k < j$ we obtain τ_k from τ_{k+1} by attaching some vertices corresponding to the k-th parallel step of the algorithm. As these vertices have independent labels they can only add one to the depth. To see that they do add one to the depth consider a vertex v of τ_{k+1} of maximal depth. This vertex corresponds to a resampling of the event $[v]$ some time after step k of the parallel algorithm. If τ_k has no vertex with higher depth than v, then from the parallel step k to the resampling corresponding to v no event from $\Gamma^+([v])$ was resampled. But this implies that $[v]$ was already violated at parallel step k and we did not select a maximal independent set of violated events there for resampling. The contradiction shows that the depth of τ_k is indeed one more than that of τ_{k+1}. To finish the proof notice that $\tau_C(t) = \tau_1$. □



↳ This is a contradiction with construction of τ: we add u to the farmost verten of τ_{k+1} from root

Homework: using above, Prove the recursion depth is at most $\lg n$ w.h.p.

Using Luby's independent set algorithm, total #rounds will be $\lg^2 n$