The Isolation Lemma

TSS

24/11/21

Outline

- Statement, Proof, Background.
- Application to Perfect Matching.
- Impossibility of 'Derandomization'

Introduction

- For any family of subsets of a set, one can isolate a subset in the family by assigning random wts to elts.

Introduction

- For any family of subsets of a set, one can isolate a subset in the family by assigning random wts to elts.

Setting: E: finite set. wt fn. $w : E \to \mathbb{N}$ extended additively to subsets $M \subseteq E$ as $w(M) = \sum_{e \in M} w(e)$. w is isolating for $\mathcal{F} \subseteq 2^E$ if $\exists!$ min wt set in $\mathcal{F}$.

Lemma: E : finite, $|E| = m$; $\mathcal{Y} \subseteq 2^E$. Let $w : E \to [k]$ be a random wt fn (each $w(e) \in [k]$ picked uar).

Then, $\Pr[w \text{ is isolating for } \mathcal{Y}] \geq \left(1 - \frac{1}{k}\right)^m$

Lemma: E : finite, $|E| = m$; $\mathcal{F} \subseteq 2^E$. Let $w : E \to [k]$ be a random wt fn (each $w(e) \in [k]$ picked uar).

Then, $\Pr[w \text{ is isolating for } \mathcal{F}] \geq \left(1 - \frac{1}{k}\right)^m$

In particular, when $k = m$ or $k = 2m$

prob. of isolation is $\geq 1/e$ $\quad 1/\sqrt{e}$

Lemma: E : finite, $|E|=m$; $\mathcal{F} \subseteq 2^E$. Let $w : E \to [k]$ be a random wt fn (each $w(e) \in [k]$ picked uar).

Then, $\Pr[w \text{ is isolating for } \mathcal{F}] \geq \left(1 - \frac{1}{k}\right)^m$

Power: ① Note max wt $\leq mk$ but $|\mathcal{F}|$ could be exp. larger.

② No assumption about $\mathcal{F}$!

> Original Motivation: MVV '87 to give a randomized parallel algo. for matching

where E - set of edges, M: set of matchings

> Original Motivation: MVV '87 to give a randomized parallel algo. for matching

where E = set of edges, M: set of matchings

Lemma: ⟹ Assigning small rand wts to edges yields unique perfect matching of min wt.

> Original Motivation: MVV '87 to give a randomized parallel algo. for matching, where E = set of edges, M = set of matchings

Lemma: $\Rightarrow$ Assigning small rand wts to edges yields unique perfect matching of min wt.

will see in more detail but first, the proof!

Proof (Ta-Shma '15): Fix $\mathcal{Y} \subseteq 2^E$.

WLOG, no set in $\mathcal{Y}$ is contained in another.

Proof (Ta-Shma '15): Fix $\mathcal{Y} \subseteq 2^E$.

WLOG, no set in $\mathcal{Y}$ is contained in another.

Denote $W = \{w : [m] \to [k]\}$ & $W_{>1} = \{w : [m] \to \{2, \ldots, k\}\}$

Proof (Ta-Shma '15): Fix $\mathcal{Y} \subseteq 2^E$.

WLOG, no set in $\mathcal{Y}$ is contained in another.

Denote $W = \{w : [m] \to [k]\}$ & $W_{>1} = \{w : [m] \to \{2, \ldots, k\}\}$

Define $\phi : W_{>1} \to W$ as follows:

Given $w \in W_{>1}$, fix arb. set $S_0 \in \min_w(\mathcal{Y})$ and

define $w' = \phi(w)$ to be:

Proof (Ta-Shma '15): Fix $\mathcal{F} \subseteq 2^E$.

WLOG, no set in $\mathcal{F}$ is contained in another.

Denote $W = \{w : [m] \to [k]\}$ & $W_{>1} = \{w : [m] \to \{2, \ldots, k\}\}$

Define $\phi : W_{>1} \to W$ as follows:

Given $w \in W_{>1}$, fix arb. set $S_0 \in \min_w(\mathcal{F})$ and define $w' = \phi(w)$ to be:

$$w'(i) = \begin{cases} w(i) - 1, & \text{if } i \in S_0 \\ w(i), & \text{o/w} \end{cases}$$

Lm: ① If $w \in W_{>1}$, then $|\min_{\phi(w)}(\mathcal{Y})| = 1$. ($\phi(w)$ is isolating).

Lm: ① If $w \in W_{>1}$, then $|\min_{\phi(w)}(y)| = 1$. ($\phi(w)$ is isolating).

② ϕ is one-one from $W_{>1} \to W$

Clm: ① If $w \in W_{>1}$, then $|\min_{\phi(w)}(f)| = 1$. ($\phi(w)$ is isolating).

② ϕ is one-one from $W_{>1} \to W$

Combined, $\Pr_{w:[m]\to[k]}(w \text{ is isolating for } f) \geq \frac{|\phi(W_{>1})|}{|W|} = \frac{|W_{>1}|}{|W|} = \left(\frac{k-1}{k}\right)^m$

Lm: ① If $w \in W_{>1}$, then $|\min_{\phi(w)}(\mathcal{F})| = 1$. ($\phi(w)$ is isolating).

② ϕ is one-one from $W_{>1} \to W$

Combined, $\Pr_{w:[m]\to[k]}(w \text{ is isolating for } \mathcal{F}) \geq \frac{|\phi(W_{>1})|}{|W|} = \frac{|W_{>1}|}{|W|} = \left(\frac{k-1}{k}\right)^m$

Pf of ① $\forall S \in \mathcal{F}$, $w'(S) = w(S) - |S \cap S_0|$ $\therefore \forall S \neq S_0 \in \mathcal{F}$,

$w'(S_0) = w(S_0) - |S_0| \leq w(S) - |S_0| < w(S) - |S \cap S_0| = w'(S)$

Lm: ① If $w \in W_{>1}$, then $|\min_{\phi(w)}(\mathcal{F})| = 1$. ($\phi(w)$ is isolating).

② ϕ is one-one from $W_{>1} \to W$

Combined, $\Pr_{w:[m]\to[k]}(w \text{ is isolating for } \mathcal{F}) \geq \frac{|\phi(W_{>1})|}{|W|} = \frac{|W_{>1}|}{|W|} = \left(\frac{k-1}{k}\right)^m$

Pf of ① $\forall S \in \mathcal{F}$, $w'(S) = w(S) - |S \cap S_0|$ $\therefore \forall S \neq S_0 \in \mathcal{F}$,

$w'(S_0) = w(S_0) - |S_0| \leq w(S) - |S_0| < w(S) - |S \cap S_0| = w'(S)$

Pf of ② Given $w \in W_{>1}$, $\exists!\, S_0 \in \mathcal{F}$ with min wt under $\phi(w)$.

But from $\phi(w)$ & S_0, we can uniquely recover w.

Perfect Matchings

> $G(X, Y, E)$., m edges n vertices.

> Hopcroft-Karp $O(m\sqrt{n})$ algo. (deterministic).

> Randomized $O(n^w)$ algo.

Perfect Matchings

> $G(X, Y, E)$, m edges n vertices.

> Hopcroft-Karp $O(m\sqrt{n})$ algo. (deterministic).

> Randomized $O(n^w)$ algo.

Consider the modified bipartite adjacency matrix

$$Z = \begin{bmatrix} z_{1,1} & 0 & 0 & \cdots & z_{1n} \\ & & z_{ij} & 0 & \\ 0 & & & & z_{nn} \end{bmatrix}$$

$(i,j)^{th}$ entry is var z_{ij} if.

$(i,j) \in E$, 0 o/w.

Claim: $\det(Z) \not\equiv 0 \iff G$ has a PM.

Claim: $\det(Z) \not\equiv 0 \iff G$ has a PM.

Sketch: PM bijectively corresponds to a permutation. The monomial for a perm is 'alive' in $\det(Z)$ iff perm is PM.

Claim: $\det(Z) \not\equiv 0 \iff G$ has a PM.

Sketch: PM bijectively corresponds to a permutation.
The monomial for a perm is 'alive' in $\det(Z)$ iff perm is PM.

> this already gives a randomized $O(n^\omega)$ algo using Schwartz-Zippel:

Claim: $\det(Z) \not\equiv 0 \iff G$ has a PM.

Sketch: PM bijectively corresponds to a permutation. The monomial for a perm is 'alive' in $\det(Z)$ iff perm is PM.

> this already gives a randomized $O(n^\omega)$ algo using Schwartz-Zippel:

$\det(Z)$ is a deg n poly, so we can assign z_{ij} values from $\{1, \dots, n^2\}$ ind. & uar and

$\det \neq 0 \iff \det(Z) \not\equiv 0$ whp.

Parallel Algorithm

Thm [MVV '87] There is a parallel algorithm that finds a PM in $O(\log^2 n)$ time using $O(n^{3.5}m)$ processors.

Parallel Algorithm

Thm [MVV '87] There is a parallel algorithm that finds a PM in $O(\log^2 n)$ time using $O(n^{3.5}m)$ processors.

Why Parallel is even possible?

Parallel Algorithm

Thm [MVV '87] There is a parallel algorithm that finds a PM in $O(\log^2 n)$ time using $O(n^{3.5}m)$ processors.

Why Parallel is even possible?

Idea of self-reducibility: Pick any $(u,v) \in X \times Y$ and let $H = G \setminus (u,v)$. If H has a PM M', then $M' \cup \{u,v\}$ is a PM for G. If H has no PM, then (u,v) is part of no matching & can be removed.

With self-reducibility in mind, can we have an independent processor for each edge? which determine if that edge is part of a PM.

With self-reducibility in mind, can we have an independent processor for each edge? which determines if that edge is part of a PM.

But this is problematic! May end up outputting all edges!

With self-reducibility in mind, can we have an independent processor for each edge? which determine if that edge is part of a PM.

But this is problematic! May end up outputting all edges!

Idea: Isolate a PM! Let each processor figure out if its edge is part of a specific PM.

the Algorithm:

First assign random wts w_{ij} to edges (i,j). $w_{ij} \in [m]$ uar.

the Algorithm:

First assign random wts w_{ij} to edges (i,j). $w_{ij} \in [m]$ uar.

Then, in Z (the var matrix), plug in $z_{i,j} = 2^{w_{ij}}$.

Let D be resulting matrix.

the Algorithm:

First assign random wts w_{ij} to edges (i,j). $w_{ij} \in [m]$ uar.

Then, in Z (the var matrix), plug in $z_{i,j} = 2^{w_{ij}}$.

Let D be resulting matrix.

Obs: Let W_0 be the wt of min wt PM on G. Then,

the Algorithm:

First assign random wts w_{ij} to edges (i,j). $w_{ij} \in [m]$ uar.

Then, in Z (the var matrix), plug in $z_{i,j} = 2^{w_{ij}}$.

Let D be resulting matrix.

Obs: Let W_0 be the wt of min wt PM on G. Then,

$\rightarrow$ G has no PM $\Rightarrow \det D = 0$

the Algorithm:

First assign random wts w_{ij} to edges (i,j). $w_{ij} \in [m]$ uar.

Then, in Z (the var matrix), plug in $z_{i,j} = 2^{w_{ij}}$.

Let D be resulting matrix.

Obs: Let W_0 be the wt of a min wt PM on G. Then,

> G has no PM $\Rightarrow \det D = 0$

> G has a unique min wt PM $\Rightarrow \det D \neq 0$ & $2^{W_0} \| \det(D)$

the Algorithm:

First assign random wts w_{ij} to edges (i,j). $w_{ij} \in [m]$ uar.

Then, in Z (the var matrix), plug in $z_{ij} = 2^{w_{ij}}$.

Let D be resulting matrix.

Obs: Let W_0 be the wt of a min wt PM on G. Then,

- G has no PM $\Rightarrow \det D = 0$
- G has a unique min wt PM $\Rightarrow \det D \neq 0$ & $2^{W_0} \| \det(D)$
- G has more than one min-wt PM $\Rightarrow \det D = 0$ or $2^W \| \det D$ for $W > W_0$

Fact: Given an $n \times n$ matrix with values m-bit ints, its det can be computed in $O(\log^2 n)$ time using $O(n^{3.5} m)$ processors

Fact: If given an $n \times n$ matrix with values m-bit ints, its det can be computed in $O(\log^2 n)$ time using $O(n^{3.5} m)$ processors

Algo:

① Pick random wts $w_{ij} \in [m]$ ind. for edges of G.

Fact: Given an $n \times n$ matrix with values m-bit ints, its det can be computed in $O(\log^2 n)$ time using $O(n^{3.5} m)$ processors

Algo:

① Pick random wts $w_{ij} \in [m]$ ind. for edges of G.

② Compute W_0 ie. the wt of the unique min wt PM in G (by calculating the largest power of 2 dividing det D).

Fact: Given an $n \times n$ matrix with values m-bit ints, its det can be computed in $O(\log^2 n)$ time using $O(n^{3.5} m)$ processors

Algo:

① Pick random wts $w_{ij} \in [m]$ ind. for edges of G.

② Compute W_0 ie. the wt of the unique min wt PM in G (by calculating the largest power of 2 dividing det D).

③ If det $D = 0$, output "no PMs".

Fact: Given an $n \times n$ matrix with values m-bit ints, its det can be computed in $O(\log^2 n)$ time using $O(n^{3.5} m)$ processors

Algo:

① Pick random wts $w_{ij} \in [m]$ ind. for edges of G.

② Compute W_2 ie. the wt of the unique min wt PM in G (by calculating the largest power of 2 dividing det D).

③ If det $D = 0$, output "no PMs".

④ For each $(i,j) \in E$, do, in parallel:

Fact: Given an $n \times n$ matrix with values m-bit ints, its det can be computed in $O(\log^2 n)$ time using $O(n^{3.5} m)$ processors

Algo:

① Pick random wts $w_{ij} \in [m]$ ind. for edges of G.

② Compute W_2 ie. the wt of the unique min wt PM in G (by calculating the largest power of 2 dividing det D).

③ If det $D = 0$, output "no PMs".

④ For each $(i,j) \in E$, do, in parallel:

> evaluate det D_{ij}

Fact: Given an $n \times n$ matrix with values m-bit ints, its det can be computed in $O(\log^2 n)$ time using $O(n^{3.5} m)$ processors

Algo:

① Pick random wts $w_{ij} \in [m]$ ind. for edges of G.

② Compute W_0 ie. the wt of the unique min wt PM in G (by calculating the largest power of 2 dividing det D).

③ If det $D = 0$, output "no PMs".

④ For each $(i,j) \in E$, do, in parallel:

> evaluate det D_{ij}

> If det $D_{ij} = 0$, do nothing

Fact: Given an $n \times n$ matrix with values m-bit ints, its det can be computed in $O(\log^2 n)$ time using $O(n^{3.5} m)$ processors

Algo:

① Pick random wts $w_{ij} \in [m]$ ind. for edges of G.

② Compute W, ie. the wt of the unique min wt PM in G (by calculating the largest power of 2 dividing det D).

③ If $\det D = 0$, output "no PMs".

④ For each $(i,j) \in E$, do, in parallel:

> evaluate $\det D_{ij}$

> If $\det D_{ij} = 0$, do nothing

> else, find w_{ij} s.t. $2^{w_{ij}} \| \det D_{ij}$

Fact: Given an $n \times n$ matrix with values m-bit ints, its det can be computed in $O(\log^2 n)$ time using $O(n^{3.5} m)$ processors

Algo:

① Pick random wts $w_{ij} \in [m]$ ind. for edges of G.

② Compute W_0 ie. the wt of the unique min wt PM in G (by calculating the largest power of 2 dividing det D).

③ If det $D = 0$, output "no PMs".

④ For each $(i,j) \in E$, do, in parallel:

> evaluate det D_{ij}

> If det $D_{ij} = 0$, do nothing

> else, find w_{ij} s.t. $2^{w_{ij}} \| \det D_{ij}$

> If $W_{ij} + w_{ij} = W_0$, output (i,j)

Fact: Given an $n \times n$ matrix with values m-bit ints, its det can be computed in $O(\log^2 n)$ time using $O(n^{3.5} m)$ processors

Algo:

① Pick random wts $w_{ij} \in [m]$ ind. for edges of G.

② Compute W_0 ie. the wt of the min wt PM in G (by calculating the largest power of 2 dividing det D).

③ If $\det D = 0$, output "no PMs".

④ For each $(i,j) \in E$, do, in parallel:

- > evaluate $\det D_{ij}$
- > If $\det D_{ij} = 0$, do nothing
- > else, find w_{ij} s.t. $2^{w_{ij}} \| \det D_{ij}$
- > If $W_{ij} + w_{ij} = W_0$, output (i,j)
- > else, do nothing.

(unique)

Notes:

> We showed Bipartite Matching $\in RNC^2$

Notes:

> We showed Bipartite Matching $\in RNC^2$

> Possible to extend to general graphs [MVV '87]

Notes:

- We showed Bipartite Matching $\in RNC^2$
- Possible to extend to general graphs [MV '87]
- Breakthrough in 2016, Bipartite PM $\in$ Quasi-NC ($n^{O(\log n)}$ processors, poly log time) [FGT 16]

Notes:

- We showed Bipartite Matching $\in RNC^2$
- Possible to extend to general graphs [MV '87]
- Breakthrough in 2016, Bipartite PM $\in$ Quasi-NC ($n^{O(\log n)}$ processors, poly log time) [FGT 16]
- Breakthrough in 2017, General PM $\in$ Quasi-NC [ST '17]

Notes:

- We showed Bipartite Matching $\in RNC^2$
- Possible to extend to general graphs [MV '87]
- Breakthrough in 2016, Bipartite PM $\in$ Quasi-NC ($n^{O(\log n)}$ processors, poly log time) [FGT 16]
- Breakthrough in 2017, General PM $\in$ Quasi-NC [ST '17]
- Both proceed by a certain 'derandomization' of the isolation lemma for certain specific set families

What does 'derandomization' mean?

What does 'derandomization' mean?

We picked $w(e) \in [k]$ uar for each $e \in E$.

What does 'derandomization' mean?

We picked $w(e) \in [k]$ uar for each $e \in E$.

One choice needs $\log k$ random bits, m choices need $m \cdot \log k$ random bits

What does 'derandomization' mean?

We picked $w(e) \in [k]$ uar for each $e \in E$.

One choice needs $\log k$ random bits, m choices need $m \cdot \log k$ random bits

Ie, our distribution was over $2^{m \log k}$ = exponentially many wt fns.

What does 'derandomization' mean?

We picked $w(e) \in [k]$ uar for each $e \in E$.

One choice needs $\log k$ random bits, m choices need $m \cdot \log k$ random bits

Ie, our distribution was over $2^{m \log k}$ = exponentially many wt fns.

Can we make this polynomial? Ie, reduce random bits to $O(\log m)$.

Derandomizing the Isolation lemma would lead to dramatic consequences both in the fields of deterministic algorithm design (such as matching $\in$ NC, efficient PIT, linear matroid intersection etc) & complexity theory. (such as NL = UL, NP = UP etc).

Alas, fully derandomizing Iso. lemma for arbitrary set families is impossible [CRS'95, AIO]

Alas, fully derandomizing Iso. lemma for arbitrary set families is impossible [CRS'95, AvM]

Thm: E: finite, $|E| = m$. Let $\mathcal{W}$ be an isolating collection of Nwt fns $E \to [k]$ (i.e. $\forall$ families $\mathcal{Y} \subseteq 2^E$, $\exists w \in \mathcal{W}$ s.t $\mathcal{Y}$ has a unique min wt set wrt w). Then,

$$N(mk+1) \geq 2^m.$$

Alas, fully derandomizing Iso. lemma for arbitrary set families is impossible [CRS'95, AvM]

Thm: $\mathcal{F}$: finite, $|E| = m$. Let $\mathcal{W}$ be an isolating collection of Nwt fns $E \to [k]$ (i.e. $\forall$ families $\mathcal{Y} \subseteq 2^E$, $\exists w \in \mathcal{W}$ s.t $\mathcal{Y}$ has a unique min wt set wrt w). Then,

$$N(mk+1) \geq 2^m.$$

I.e, both 'small wts' & 'small amt of randomness' is impossible!

Before pf, observe:

Before pf, observe:

Derandomizing possible with large wts:

Before pf, observe:

Derandomizing possible with large wts:

Assign $w(e_i) = 2^{i-1}$. Then every set $M \subseteq E$ has a distinct wt. (every int. has unique binary rep).

Before pf, observe:

Derandomizing possible with large wts:

Assign $w(e_i) = 2^{i-1}$. Then every set $M \subseteq E$ has a distinct wt. (every int. has unique binary rep).

Derandomizing possible with large collections:

Let $\mathcal{W} = \{w_M\}_{M \subseteq E}$ where $w_M(e) = \begin{cases} 1 & \text{if } e \in M \\ 0 & \text{o/w} \end{cases}$ be the char. wt fns. Then $\mathcal{W}$ isolates any set family $\mathcal{F}$.

Pf of Impossibility of derandomization: (Polynomial Method).

Let $\mathcal{H}$ be a collection of N wt fns $E \to [k]$.

Pf of Impossibility of derandomization: (Polynomial Method).

Let $\mathcal{W}$ be a collection of N wt fns $E \to [k]$.

Let $p(z)$ be a multilinear poly in $z = (z_1, \ldots, z_m)$.

ie. each monomial is a product $\alpha_M \prod_{i \in M} z_i$.

Let Y_p be the collection corresponding to non-zero coeffs.

Pf of Impossibility of derandomization: (Polynomial Method).

Let $\mathcal{H}$ be a collection of N wt fns $E \to [k]$.

Let $p(z)$ be a multilinear poly in $z = (z_1, \ldots, z_m)$.

i.e. each monomial is a product $\alpha_M \prod_{i \in M} z_i$.

Let $\mathcal{H}_p$ be the collection corresponding to non-zero coeffs.

Then, $$p(z) = \sum_{M \in \mathcal{H}_p} \alpha_M \prod_{i \in M} z_i$$

By assumption, $\exists\, w \in W$ s.t. it isolates y_p & let $M^* \in \mathcal{Y}_p$ be the unique min wt set in $\mathcal{Y}_p$.

By assumption, $\exists\, w \in \mathbb{W}$ s.t. it isolates $\mathcal{Y}_p$ & let $M^* \in \mathcal{Y}_p$ be the unique min wt set in $\mathcal{Y}_p$.

Consider the substitution:

$$x_j \longmapsto t^{w(j)}.$$

By assumption, $\exists\, w \in W$ s.t. it isolates y_p & let $M^* \in y_p$ be the unique min wt set in y_p.

Consider the substitution:

$$z_j \longmapsto t^{w(j)}.$$

So, a monomial $\prod_{j \in M} z_j \longmapsto t^{w(M)}$.

By assumption, $\exists\, w \in \mathbb{W}$ s.t. it isolates $\mathcal{Y}_p$ & let $M^* \in \mathcal{Y}_p$ be the unique min wt set in $\mathcal{Y}_p$.

Consider the substitution:

$$z_j \longmapsto t^{w(j)}.$$

So, a monomial $\prod_{j \in M} z_j \longmapsto t^{w(M)}$.

Let $q(t)$ be the single var poly obtained from $p(z)$.

Clm: $q(t) \not\equiv 0$

By assumption, $\exists\, w \in \mathbb{W}$ s.t. it isolates $\mathcal{Y}_p$ & let $M^* \in \mathcal{Y}_p$ be the unique min wt set in $\mathcal{Y}_p$.

Consider the substitution:

$$z_j \longmapsto t^{w(j)}.$$

So, a monomial $\prod_{j \in M} z_j \longmapsto t^{w(M)}$.

Let $q(t)$ be the single var poly obtained from $p(z)$.

Clm: $q(t) \not\equiv 0$

Sketch: $t^{w(M^*)}$ cannot get cancelled out.

2°. $q(t)$ has degree at most mk and is therefore non-zero for at least one of the points in $\{1, \ldots, mk+1\}$.

2°. $q(t)$ has degree at most mk and is therefore non-zero for at least one of the points in $\{1, \ldots, mk+1\}$.

$\Rightarrow$ any multilinear poly $p(z)$ is non-zero on at least one of the points on

$$T = \left\{ \left(t^{w(1)}, t^{w(2)}, \ldots, t^{w(m)} \right) \,\middle|\, w \in [d], t \in [mk+1] \right\}$$

$$|T| = N(mk+1).$$

But, a std linear alg argument shows that for any set $H \subsetneq \mathbb{R}^m$ of size $< 2^m$, there is a non-zero multilinear poly $p(z_1, \ldots, z_m)$ which is zero on H.

But, a std linear alg argument shows that for any set $H \subsetneq \mathbb{R}^m$ of size $< 2^m$, there is a non-zero multilinear poly $p(z_1, \ldots, z_m)$ which is zero on H.

We conclude

$$N(mk+1) \geq 2^m.$$