Deterministic Algorithms for the Lovász Local Lemma[1]

Jonas Hübotter and Duri Janett
Advised by Yassir Akram

March 29, 2022

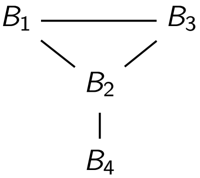
[1] David G Harris. "Deterministic algorithms for the Lovász local lemma: simpler, more general, and more parallel". In: *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM. 2022, pp. 1744–1779.

Setting

Distribution D over independent Σ-valued coordinates $X_1, \ldots, X_n$.
"Bad-events" $\mathcal{B} = \{B_1, \ldots, B_m\}$, each a boolean function of some subset of coordinates $\mathrm{var}(B_i) \subseteq \{X_1, \ldots, X_n\}$ with law p.

Example (3-SAT)

$$B_1 \doteq f_1(X_1, X_3, X_5)$$
$$B_2 \doteq f_2(X_2, X_3, X_6)$$
$$B_3 \doteq f_3(X_1, X_5, X_6)$$
$$B_4 \doteq f_4(X_2, X_4, X_7)$$

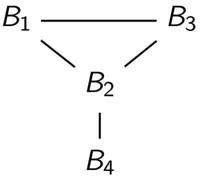


Setting

Distribution D over independent Σ-valued coordinates $X_1, \ldots, X_n$. "Bad-events" $\mathcal{B} = \{B_1, \ldots, B_m\}$, each a boolean function of some subset of coordinates $\mathrm{var}(B_i) \subseteq \{X_1, \ldots, X_n\}$ with law p.

Example (3-SAT)

$$B_1 \doteq f_1(X_1, X_3, X_5)$$
$$B_2 \doteq f_2(X_2, X_3, X_6)$$
$$B_3 \doteq f_3(X_1, X_5, X_6)$$
$$B_4 \doteq f_4(X_2, X_4, X_7)$$



Theorem ((Symmetric) Lovász Local Lemma)

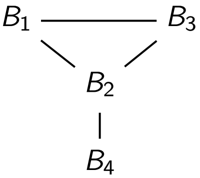
If for any i, $p(B_i) \leq p_{\max}$ and B_i affects at most d bad-events, then $ep_{\max}d \leq 1$ implies $Pr[\text{all } B_i \text{ avoided}] > 0$.

Setting

Distribution D over independent Σ-valued coordinates $X_1, \ldots, X_n$.
"Bad-events" $\mathcal{B} = \{B_1, \ldots, B_m\}$, each a boolean function of some subset of coordinates $\mathrm{var}(B_i) \subseteq \{X_1, \ldots, X_n\}$ with law p.

Example (3-SAT)

$$B_1 \doteq f_1(X_1, X_3, X_5)$$
$$B_2 \doteq f_2(X_2, X_3, X_6)$$
$$B_3 \doteq f_3(X_1, X_5, X_6)$$
$$B_4 \doteq f_4(X_2, X_4, X_7)$$



Theorem ((Symmetric) Lovász Local Lemma)

If for any i, $p(B_i) \leq p_{\max}$ and B_i affects at most d bad-events, then $ep_{\max}d \leq 1$ implies $Pr[\text{all } B_i \text{ avoided}] > 0$.

For k-SAT and $X_i \sim \mathrm{Unif}(\{0,1\})$, $p \equiv 2^{-k}$.
$\rightsquigarrow$ satisfiable if any variable appears in at most $2^k/ke$ clauses!

Applications

Example (k-Coloring)

Choose $C_v \sim \text{Unif}([k])$ independently.

Applications

Example (k-Coloring)

Choose $C_v \sim \text{Unif}([k])$ independently.
$B_{v,c} \doteq$ "$C_v = c$ and v has neighbor with color c".
$B_{v,c}$ affects $B_{v',c'}$ iff v and v' have distance ≤ 2

Applications

Example (k-Coloring)

Choose $C_v \sim \mathrm{Unif}([k])$ independently.
$B_{v,c} \doteq$ "$C_v = c$ and v has neighbor with color c".
$B_{v,c}$ affects $B_{v',c'}$ iff v and v' have distance $\leq 2 \rightsquigarrow d \leq k\Delta^2$.

Applications

Example (k-Coloring)

Choose $C_v \sim \text{Unif}([k])$ independently.
$B_{v,c} \doteq$ "$C_v = c$ and v has neighbor with color c".
$B_{v,c}$ affects $B_{v',c'}$ iff v and v' have distance $\leq 2 \rightsquigarrow d \leq k\Delta^2$.
$p(B_{v,c}) = \frac{1}{k}(\sum_{u \in N(v)} \frac{1}{k}) \leq \frac{\Delta}{k^2}$

Applications

Example (k-Coloring)

Choose $C_v \sim \text{Unif}([k])$ independently.
$B_{v,c} \doteq$ "$C_v = c$ and v has neighbor with color c".
$B_{v,c}$ affects $B_{v',c'}$ iff v and v' have distance $\leq 2 \rightsquigarrow d \leq k\Delta^2$.
$p(B_{v,c}) = \frac{1}{k}(\sum_{u \in N(v)} \frac{1}{k}) \leq \frac{\Delta}{k^2} \rightsquigarrow$ if $e\Delta^3 \leq k$, has k-coloring!

Applications

Example (k-Coloring)

Choose $C_v \sim \text{Unif}([k])$ independently.
$B_{v,c} \doteq$ "$C_v = c$ and v has neighbor with color c".
$B_{v,c}$ affects $B_{v',c'}$ iff v and v' have distance $\leq 2 \rightsquigarrow d \leq k\Delta^2$.
$p(B_{v,c}) = \frac{1}{k}(\sum_{u \in N(v)} \frac{1}{k}) \leq \frac{\Delta}{k^2} \rightsquigarrow$ if $e\Delta^3 \leq k$, has k-coloring!

More applications:

1. Defective coloring
2. Hypergraph coloring
3. Strong coloring
4. Non-repetitive coloring
5. Finding directed cycles of certain length (see exam, task 2 :))
6. Independent transversals

Applications

Example (k-Coloring)

Choose $C_v \sim \text{Unif}([k])$ independently.
$B_{v,c} \doteq$ "$C_v = c$ and v has neighbor with color c".
$B_{v,c}$ affects $B_{v',c'}$ iff v and v' have distance $\leq 2 \rightsquigarrow d \leq k\Delta^2$.
$p(B_{v,c}) = \frac{1}{k}(\sum_{u \in N(v)} \frac{1}{k}) \leq \frac{\Delta}{k^2} \rightsquigarrow$ if $e\Delta^3 \leq k$, has k-coloring!

More applications:

1. Defective coloring
2. Hypergraph coloring
3. Strong coloring
4. Non-repetitive coloring
5. Finding directed cycles of certain length (see exam, task 2 :))
6. Independent transversals

$\rightsquigarrow$ algorithmic versions of the Lovász Local Lemma yield automatic algorithms for these problems!

Prior Work

Algorithm: MT-Algorithm

Draw X from distribution D
while *some bad-event is true on* X **do**
  Select any true bad-event B
  For each $i \in \mathrm{var}(B)$, draw X_i from its distribution in D
end

[2] Robin A Moser and Gábor Tardos. "A constructive proof of the general Lovász local lemma". In: *Journal of the ACM (JACM)* 57.2 (2010), pp. 1–15.

Prior Work

Algorithm: MT-Algorithm

Draw X from distribution D
while *some bad-event is true on* X **do**
  Select any true bad-event B
  For each $i \in \mathrm{var}(B)$, draw X_i from its distribution in D
end

$\rightsquigarrow$ converges within expected polynomial time.[2]

[2] Robin A Moser and Gábor Tardos. "A constructive proof of the general Lovász local lemma". In: *Journal of the ACM (JACM)* 57.2 (2010), pp. 1–15.

Prior Work

Paper	Criterion	Det.?	Parallel?
[3]	asymmetric LLL	✗	(✓)
[3]	asymmetric LLL and $d \leq \mathcal{O}(1)$	✓	(✓)
[4]	symmetric LLL with ϵ-exponential slack	✓	(✓)
[5]	Shearer criterion with ϵ-slack	✗	✓
[5]	symmetric LLL with ϵ-exponential slack and atomic bad-events	✓	✓
[6]	symmetric LLL and bad-events depend on $\mathrm{polylog}(n)$ variables	✓	✓

(✓) : under more complex conditions

[3] Robin A Moser and Gábor Tardos. "A constructive proof of the general Lovász local lemma". In: *Journal of the ACM (JACM)* 57.2 (2010), pp. 1–15.

[4] Karthekeyan Chandrasekaran, Navin Goyal, and Bernhard Haeupler. "Deterministic algorithms for the Lovász local lemma". In: *SIAM Journal on Computing* 42.6 (2013), pp. 2132–2155.

[5] Bernhard Haeupler and David G Harris. "Parallel algorithms and concentration bounds for the Lovász local lemma via witness-DAGs". In: *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM. 2017, pp. 1170–1187.

[6] David G Harris. "Deterministic parallel algorithms for fooling polylogarithmic juntas and the Lovász local lemma". In: *ACM Transactions on Algorithms (TALG)* 14.4 (2018), pp. 1–24.

Contributions

1. *Deterministic algorithm* with a simpler & more general condition that is satisfied by *most* variants of the LLL.

Contributions

1. *Deterministic algorithm* with a simpler & more general condition that is satisfied by *most* variants of the LLL.
2. Faster *parallel algorithm* with simpler conditions.

Contributions

1. *Deterministic algorithm* with a simpler & more general condition that is satisfied by *most* variants of the LLL.
2. Faster *parallel algorithm* with simpler conditions.
3. We can ensure that the final distribution of the deterministic algorithm is not "far off" from the distribution at the end of the MT algorithm.

Plan

Alternative Characterization of MT Algorithm

Consider the resampling table R drawn according to distribution D:

	1	$\cdots$	t	$\cdots$
X_1	$*$	$*$	$*$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\cdots$
X_n	$*$	$*$	$*$	$\cdots$

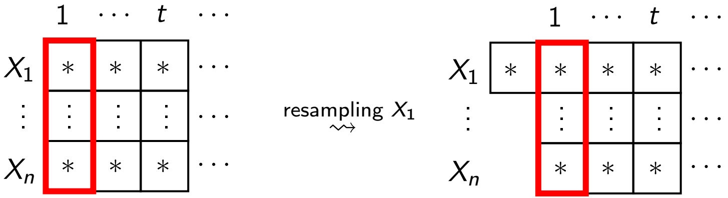
Alternative Characterization of MT Algorithm

Consider the resampling table R drawn according to distribution D:

	1	$\cdots$	t	$\cdots$
X_1	$*$	$*$	$*$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\cdots$
X_n	$*$	$*$	$*$	$\cdots$

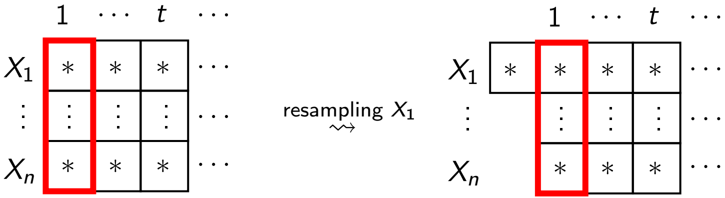
Alternative Characterization of MT Algorithm

Consider the resampling table R drawn according to distribution D:



Alternative Characterization of MT Algorithm

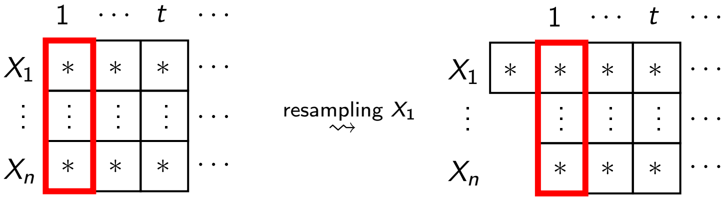
Consider the resampling table R drawn according to distribution D:



When resampling B_i, shift rows $\mathrm{var}(B_i)$ to left.

Alternative Characterization of MT Algorithm

Consider the resampling table R drawn according to distribution D:



When resampling B_i, shift rows $\mathrm{var}(B_i)$ to left.

$\rightsquigarrow$ MT algorithm deterministic with respect to resampling table!

Counting Resamples

Want to find an encoding of resamples such that we do not lose much information.

Counting Resamples

Want to find an encoding of resamples such that we do not lose much information.

Why may executions be long?

Given a resampling table R, a (partial) execution of the MT algorithm is described by the sequence of resampled bad-events.

Counting Resamples

Want to find an encoding of resamples such that we do not lose much information.

Why may executions be long?

Given a resampling table R, a (partial) execution of the MT algorithm is described by the sequence of resampled bad-events.

$$B_1, B_2, B_3 \qquad \mapsto$$

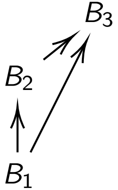
Counting Resamples

Want to find an encoding of resamples such that we do not lose much information.

Why may executions be long?

Given a resampling table R, a (partial) execution of the MT algorithm is described by the sequence of resampled bad-events.

$B_1, B_2, B_3 \quad \mapsto$



Witness DAG $\hat{G}$

$B_i \longrightarrow B_j$ iff $i < j$ and B_i affects B_j

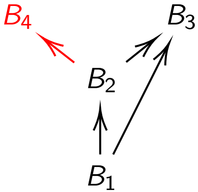
Counting Resamples

Want to find an encoding of resamples such that we do not lose much information.

Why may executions be long?

Given a resampling table R, a (partial) execution of the MT algorithm is described by the sequence of resampled bad-events.

$B_1, B_2, B_3, B_4 \quad \mapsto$



Witness DAG $\hat{G}$
$B_i \longrightarrow B_j$ iff $i < j$
and B_i affects B_j

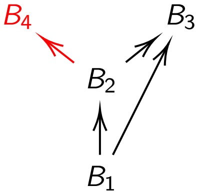
Counting Resamples

Want to find an encoding of resamples such that we do not lose much information.

Why may executions be long?

Given a resampling table R, a (partial) execution of the MT algorithm is described by the sequence of resampled bad-events.

$B_1, B_2, B_3, B_4 \quad \mapsto$



Witness DAG $\hat{G}$
$B_i \longrightarrow B_j$ iff $i < j$
and B_i affects B_j

$\rightsquigarrow$ $\hat{G}$ is always a DAG!

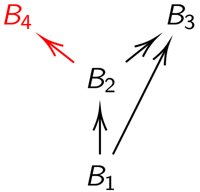
Counting Resamples

Want to find an encoding of resamples such that we do not lose much information.

Why may executions be long?

Given a resampling table R, a (partial) execution of the MT algorithm is described by the sequence of resampled bad-events.

$B_1, B_2, B_3, B_4 \quad \mapsto$

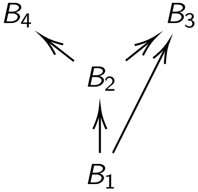


Witness DAG $\hat{G}$
$B_i \longrightarrow B_j$ iff $i < j$ and B_i affects B_j

$\rightsquigarrow$ $\hat{G}$ is always a DAG! But why are DAGs a good encoding?

Counting Resamples

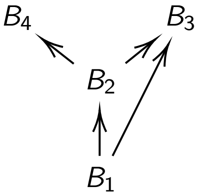
Witness DAGs encode the final configuration of the MT algorithm!



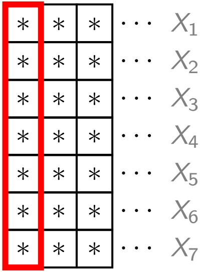
$\mathrm{var}(B_1) = \{X_1, X_3, X_5\}$
$\mathrm{var}(B_2) = \{X_2, X_3, X_6\}$
$\mathrm{var}(B_3) = \{X_1, X_5, X_6\}$
$\mathrm{var}(B_4) = \{X_2, X_4, X_7\}$

Counting Resamples

Witness DAGs encode the final configuration of the MT algorithm!



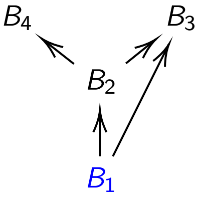
$\mathrm{var}(B_1) = \{X_1, X_3, X_5\}$
$\mathrm{var}(B_2) = \{X_2, X_3, X_6\}$
$\mathrm{var}(B_3) = \{X_1, X_5, X_6\}$
$\mathrm{var}(B_4) = \{X_2, X_4, X_7\}$



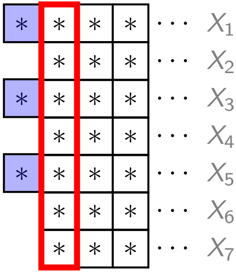
fixed resampling table R

Counting Resamples

Witness DAGs encode the final configuration of the MT algorithm!



$\mathrm{var}(B_1) = \{X_1, X_3, X_5\}$
$\mathrm{var}(B_2) = \{X_2, X_3, X_6\}$
$\mathrm{var}(B_3) = \{X_1, X_5, X_6\}$
$\mathrm{var}(B_4) = \{X_2, X_4, X_7\}$

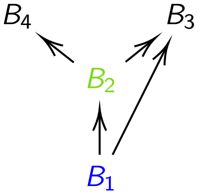


fixed resampling table R

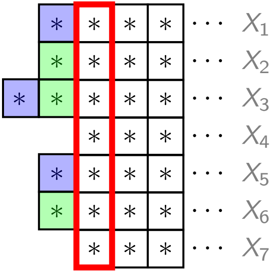
resamples: B_1

Counting Resamples

Witness DAGs encode the final configuration of the MT algorithm!



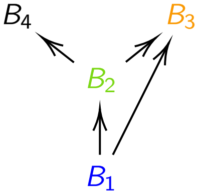
$\mathrm{var}(B_1) = \{X_1, X_3, X_5\}$
$\mathrm{var}(B_2) = \{X_2, X_3, X_6\}$
$\mathrm{var}(B_3) = \{X_1, X_5, X_6\}$
$\mathrm{var}(B_4) = \{X_2, X_4, X_7\}$

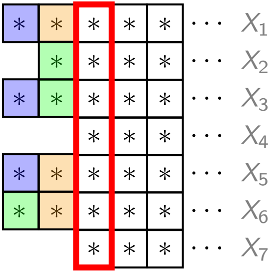


fixed resampling table R
resamples: B_1, B_2

Counting Resamples

Witness DAGs encode the final configuration of the MT algorithm!





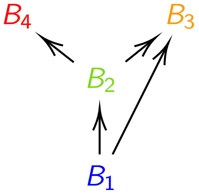
$\mathrm{var}(B_1) = \{X_1, X_3, X_5\}$
$\mathrm{var}(B_2) = \{X_2, X_3, X_6\}$
$\mathrm{var}(B_3) = \{X_1, X_5, X_6\}$
$\mathrm{var}(B_4) = \{X_2, X_4, X_7\}$

fixed resampling table R

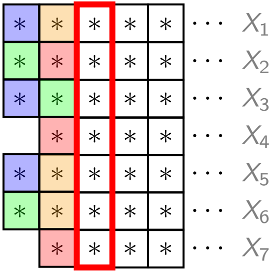
resamples: B_1, B_2, B_3

Counting Resamples

Witness DAGs encode the final configuration of the MT algorithm!



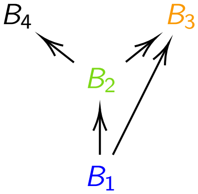
$\mathrm{var}(B_1) = \{X_1, X_3, X_5\}$
$\mathrm{var}(B_2) = \{X_2, X_3, X_6\}$
$\mathrm{var}(B_3) = \{X_1, X_5, X_6\}$
$\mathrm{var}(B_4) = \{X_2, X_4, X_7\}$



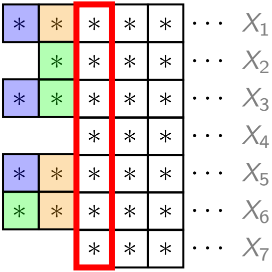
fixed resampling table R
resamples: B_1, B_2, B_3, B_4

Counting Resamples

Witness DAGs encode the final configuration of the MT algorithm!



$\mathrm{var}(B_1) = \{X_1, X_3, X_5\}$
$\mathrm{var}(B_2) = \{X_2, X_3, X_6\}$
$\mathrm{var}(B_3) = \{X_1, X_5, X_6\}$
$\mathrm{var}(B_4) = \{X_2, X_4, X_7\}$

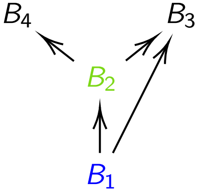


fixed resampling table R

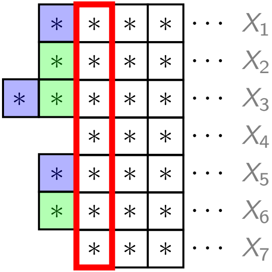
resamples: B_1, B_2, B_3

Counting Resamples

Witness DAGs encode the final configuration of the MT algorithm!



$\mathrm{var}(B_1) = \{X_1, X_3, X_5\}$
$\mathrm{var}(B_2) = \{X_2, X_3, X_6\}$
$\mathrm{var}(B_3) = \{X_1, X_5, X_6\}$
$\mathrm{var}(B_4) = \{X_2, X_4, X_7\}$

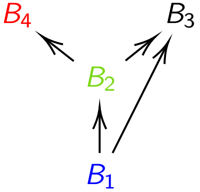


fixed resampling table R

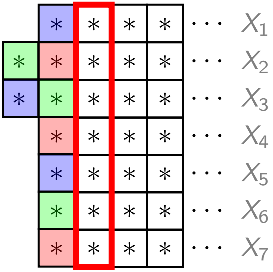
resamples: B_1, B_2

Counting Resamples

Witness DAGs encode the final configuration of the MT algorithm!



$\mathrm{var}(B_1) = \{X_1, X_3, X_5\}$
$\mathrm{var}(B_2) = \{X_2, X_3, X_6\}$
$\mathrm{var}(B_3) = \{X_1, X_5, X_6\}$
$\mathrm{var}(B_4) = \{X_2, X_4, X_7\}$

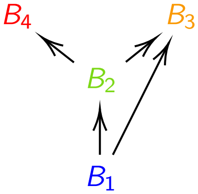


fixed resampling table R

resamples: B_1, B_2, B_4

Counting Resamples

Witness DAGs encode the final configuration of the MT algorithm!

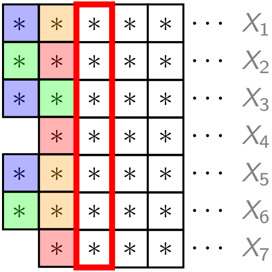


$\mathrm{var}(B_1) = \{X_1, X_3, X_5\}$

$\mathrm{var}(B_2) = \{X_2, X_3, X_6\}$

$\mathrm{var}(B_3) = \{X_1, X_5, X_6\}$

$\mathrm{var}(B_4) = \{X_2, X_4, X_7\}$

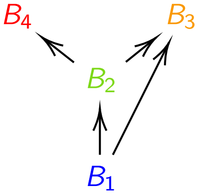


fixed resampling table R

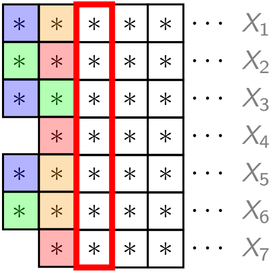
resamples: B_1, B_2, B_4, B_3

Counting Resamples

Witness DAGs encode the final configuration of the MT algorithm!



$\text{var}(B_1) = \{X_1, X_3, X_5\}$
$\text{var}(B_2) = \{X_2, X_3, X_6\}$
$\text{var}(B_3) = \{X_1, X_5, X_6\}$
$\text{var}(B_4) = \{X_2, X_4, X_7\}$

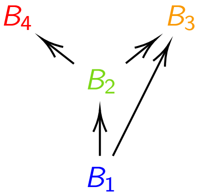


fixed resampling table R
resamples: B_1, B_2, B_4, B_3

$\rightsquigarrow$ may encode multiple executions, but *all* lead to the same final configuration!

Counting Resamples

Witness DAGs encode the final configuration of the MT algorithm!



$\mathrm{var}(B_1) = \{X_1, X_3, X_5\}$
$\mathrm{var}(B_2) = \{X_2, X_3, X_6\}$
$\mathrm{var}(B_3) = \{X_1, X_5, X_6\}$
$\mathrm{var}(B_4) = \{X_2, X_4, X_7\}$

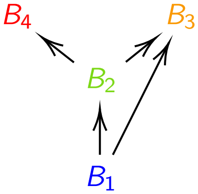
*	*	*	*	*	⋯	X_1
*	*	*	*	*	⋯	X_2
*	*	*	*	*	⋯	X_3
	*	*	*	*	⋯	X_4
*	*	*	*	*	⋯	X_5
*	*	*	*	*	⋯	X_6
	*	*	*	*	⋯	X_7

fixed resampling table R
resamples: B_1, B_2, B_4, B_3

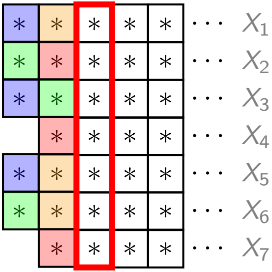
⇝ may encode multiple executions, but *all* lead to the same final configuration!
⇝ resampled bad-events depend on *disjoint* entries of R!

Counting Resamples

Witness DAGs encode the final configuration of the MT algorithm!



$\text{var}(B_1) = \{X_1, X_3, X_5\}$
$\text{var}(B_2) = \{X_2, X_3, X_6\}$
$\text{var}(B_3) = \{X_1, X_5, X_6\}$
$\text{var}(B_4) = \{X_2, X_4, X_7\}$

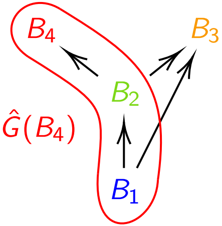


fixed resampling table R
resamples: B_1, B_2, B_4, B_3

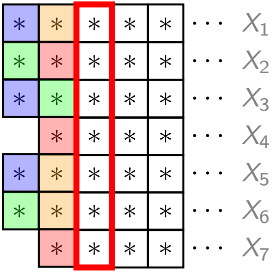
⇝ may encode multiple executions, but *all* lead to the same final configuration!
⇝ resampled bad-events depend on *disjoint* entries of R!
⇝ configuration at step t is drawn according to D!

Counting Resamples

Witness DAGs encode the final configuration of the MT algorithm!



$$\mathrm{var}(B_1) = \{X_1, X_3, X_5\}$$
$$\mathrm{var}(B_2) = \{X_2, X_3, X_6\}$$
$$\mathrm{var}(B_3) = \{X_1, X_5, X_6\}$$
$$\mathrm{var}(B_4) = \{X_2, X_4, X_7\}$$



fixed resampling table R

resamples: B_1, B_2, B_4, B_3

⇝ may encode multiple executions, but *all* lead to the same final configuration!

⇝ resampled bad-events depend on *disjoint* entries of R!

⇝ configuration at step t is drawn according to D!

Analyzing the MT Algorithm

Are *all* witness DAGs used as an encoding of a resample?

Analyzing the MT Algorithm

Are *all* witness DAGs used as an encoding of a resample?
No! $\rightsquigarrow$ we can improve our counting!

Analyzing the MT Algorithm

Are *all* witness DAGs used as an encoding of a resample?
No! $\rightsquigarrow$ we can improve our counting!

- $\hat{G}(B_i)$ always has a single sink (set denoted $\mathcal{G}$)

Analyzing the MT Algorithm

Are *all* witness DAGs used as an encoding of a resample?
No! $\rightsquigarrow$ we can improve our counting!

- $\hat{G}(B_i)$ always has a single sink (set denoted $\mathcal{G}$)
- If we fix a resampling table R, do we need to consider all single-sink witness DAGs G?

Analyzing the MT Algorithm

Are *all* witness DAGs used as an encoding of a resample?
No! $\rightsquigarrow$ we can improve our counting!

- $\hat{G}(B_i)$ always has a single sink (set denoted $\mathcal{G}$)
- If we fix a resampling table R, do we need to consider all single-sink witness DAGs G?
  $\rightsquigarrow$ No!

Analyzing the MT Algorithm

Are *all* witness DAGs used as an encoding of a resample?
No! $\rightsquigarrow$ we can improve our counting!

- $\hat{G}(B_i)$ always has a single sink (set denoted $\mathcal{G}$)
- If we fix a resampling table R, do we need to consider all single-sink witness DAGs G?
  $\rightsquigarrow$ No! G & R must be compatible (set denoted $\mathcal{G}[R]$)

Analyzing the MT Algorithm

Are *all* witness DAGs used as an encoding of a resample?
No! $\rightsquigarrow$ we can improve our counting!

- $\hat{G}(B_i)$ always has a single sink (set denoted $\mathcal{G}$)
- If we fix a resampling table R, do we need to consider all single-sink witness DAGs G?
  $\rightsquigarrow$ No! G & R must be compatible (set denoted $\mathcal{G}[R]$)

  Note: $Pr_{R \sim D}[G \text{ \& } R \text{ compatible}]$

Analyzing the MT Algorithm

Are *all* witness DAGs used as an encoding of a resample?
No! $\rightsquigarrow$ we can improve our counting!

- $\hat{G}(B_i)$ always has a single sink (set denoted $\mathcal{G}$)
- If we fix a resampling table R, do we need to consider all single-sink witness DAGs G?
  $\rightsquigarrow$ No! G & R must be compatible (set denoted $\mathcal{G}[R]$)

  Note: $Pr_{R \sim D}[G \text{ \& } R \text{ compatible}] = \prod_{B \in G} p(B)$

Analyzing the MT Algorithm

Are *all* witness DAGs used as an encoding of a resample?
No! $\rightsquigarrow$ we can improve our counting!

- $\hat{G}(B_i)$ always has a single sink (set denoted $\mathcal{G}$)
- If we fix a resampling table R, do we need to consider all single-sink witness DAGs G?
  $\rightsquigarrow$ No! G & R must be compatible (set denoted $\mathcal{G}[R]$)

  Note: $Pr_{R \sim D}[G \ \& \ R \text{ compatible}] = \prod_{B \in G} p(B) \doteq w_p(G)$.

Analyzing the MT Algorithm

Are *all* witness DAGs used as an encoding of a resample?
No! $\rightsquigarrow$ we can improve our counting!

- $\hat{G}(B_i)$ always has a single sink (set denoted $\mathcal{G}$)
- If we fix a resampling table R, do we need to consider all single-sink witness DAGs G?
  $\rightsquigarrow$ No! G & R must be compatible (set denoted $\mathcal{G}[R]$)

  Note: $Pr_{R \sim D}[G \text{ \& } R \text{ compatible}] = \prod_{B \in G} p(B) \doteq w_p(G)$.

$\rightsquigarrow$ for fixed resampling table R, at most $|\mathcal{G}[R]|$ resamplings

Analyzing the MT Algorithm

Are *all* witness DAGs used as an encoding of a resample?
No! $\rightsquigarrow$ we can improve our counting!

- $\hat{G}(B_i)$ always has a single sink (set denoted $\mathcal{G}$)
- If we fix a resampling table R, do we need to consider all single-sink witness DAGs G?
  $\rightsquigarrow$ No! G & R must be compatible (set denoted $\mathcal{G}[R]$)

  Note: $Pr_{R \sim D}[G \ \& \ R \text{ compatible}] = \prod_{B \in G} p(B) \doteq w_p(G)$.

$\rightsquigarrow$ for fixed resampling table R, at most $|\mathcal{G}[R]|$ resamplings

$\mathbb{E}|\mathcal{G}[R]|$

Analyzing the MT Algorithm

Are *all* witness DAGs used as an encoding of a resample?
No! $\rightsquigarrow$ we can improve our counting!

- $\hat{G}(B_i)$ always has a single sink (set denoted $\mathcal{G}$)
- If we fix a resampling table R, do we need to consider all single-sink witness DAGs G?
  $\rightsquigarrow$ No! G & R must be compatible (set denoted $\mathcal{G}[R]$)

  Note: $Pr_{R \sim D}[G \text{ \& } R \text{ compatible}] = \prod_{B \in G} p(B) \doteq w_p(G)$.

$\rightsquigarrow$ for fixed resampling table R, at most $|\mathcal{G}[R]|$ resamplings

$$\mathbb{E}|\mathcal{G}[R]| = \sum_{G \in \mathcal{G}} Pr[G \text{ \& } R \text{ compatible}] = \sum_{G \in \mathcal{G}} w_p(G)$$

Analyzing the MT Algorithm

Are *all* witness DAGs used as an encoding of a resample?
No! $\rightsquigarrow$ we can improve our counting!

- $\hat{G}(B_i)$ always has a single sink (set denoted $\mathcal{G}$)
- If we fix a resampling table R, do we need to consider all single-sink witness DAGs G?
  $\rightsquigarrow$ No! G & R must be compatible (set denoted $\mathcal{G}[R]$)

  Note: $Pr_{R \sim D}[G \;\&\; R \text{ compatible}] = \prod_{B \in G} p(B) \doteq w_p(G)$.

$\rightsquigarrow$ for fixed resampling table R, at most $|\mathcal{G}[R]|$ resamplings

$$\mathbb{E}|\mathcal{G}[R]| = \sum_{G \in \mathcal{G}} Pr[G \;\&\; R \text{ compatible}] = \sum_{G \in \mathcal{G}} w_p(G) \doteq w_p(\mathcal{G})$$

Analyzing the MT Algorithm

Are *all* witness DAGs used as an encoding of a resample?
No! $\rightsquigarrow$ we can improve our counting!

- $\hat{G}(B_i)$ always has a single sink (set denoted $\mathcal{G}$)
- If we fix a resampling table R, do we need to consider all single-sink witness DAGs G?
  $\rightsquigarrow$ No! G & R must be compatible (set denoted $\mathcal{G}[R]$)

  Note: $Pr_{R \sim D}[G \text{ \& } R \text{ compatible}] = \prod_{B \in G} p(B) \doteq w_p(G)$.

$\rightsquigarrow$ for fixed resampling table R, at most $|\mathcal{G}[R]|$ resamplings

$$\mathbb{E}|\mathcal{G}[R]| = \sum_{G \in \mathcal{G}} Pr[G \text{ \& } R \text{ compatible}] = \sum_{G \in \mathcal{G}} w_p(G) \doteq \underbrace{w_p(\mathcal{G}) < \infty.}_{\text{Shearer Criterion}}$$

Plan

Likely & Unlikely Resamples

Want to find resampling table R such that $|\mathcal{G}[R]|$ is polynomial.

Likely & Unlikely Resamples

Want to find resampling table R such that $|\mathcal{G}[R]|$ is polynomial.
But, $|\mathcal{G}| = \infty$!

Likely & Unlikely Resamples

Want to find resampling table R such that $|\mathcal{G}[R]|$ is polynomial.
But, $|\mathcal{G}| = \infty$!

Example $w_p(G) = 1/2$. B_1

Likely & Unlikely Resamples

Want to find resampling table R such that $|\mathcal{G}[R]|$ is polynomial.
But, $|\mathcal{G}| = \infty$!

Example $w_p(G) = 1/4$. $B_1 \twoheadrightarrow B_2$

Likely & Unlikely Resamples

Want to find resampling table R such that $|\mathcal{G}[R]|$ is polynomial.
But, $|\mathcal{G}| = \infty$!

Example $w_p(G) = 1/8$. $B_1 \twoheadrightarrow B_2 \twoheadrightarrow B_3$

Likely & Unlikely Resamples

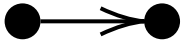
Want to find resampling table R such that $|\mathcal{G}[R]|$ is polynomial.
But, $|\mathcal{G}| = \infty$!

Example $w_p(G) = 1/8$. $B_1 \rightarrow B_2 \rightarrow B_3$

Likely & Unlikely Resamples

Want to find resampling table R such that $|\mathcal{G}[R]|$ is polynomial.
But, $|\mathcal{G}| = \infty$!

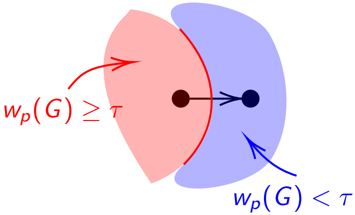
Example $w_p(G) = 1/8$. $B_1 \rightarrow B_2 \rightarrow B_3$



Likely & Unlikely Resamples

Want to find resampling table R such that $|\mathcal{G}[R]|$ is polynomial.
But, $|\mathcal{G}| = \infty$!

Example $w_p(G) = 1/8$. $B_1 \twoheadrightarrow B_2 \twoheadrightarrow B_3$



Likely & Unlikely Resamples

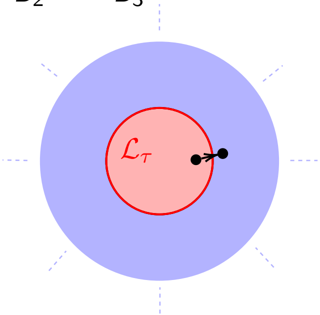
Want to find resampling table R such that $|\mathcal{G}[R]|$ is polynomial.
But, $|\mathcal{G}| = \infty$!

Example $w_p(G) = 1/8$.

$B_1 \twoheadrightarrow B_2 \twoheadrightarrow B_3$

For a threshold $\tau \in [0, 1]$,

- let $\mathcal{L}_\tau \subseteq \mathcal{G}$ be the set of likely witness DAGs, $w_p(G) \geq \tau$



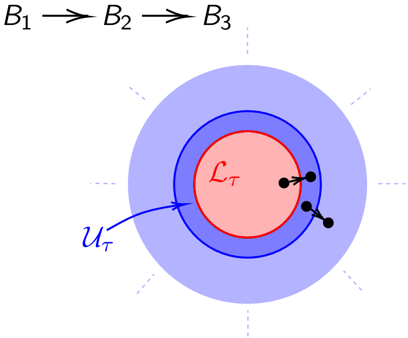
Likely & Unlikely Resamples

Want to find resampling table R such that $|\mathcal{G}[R]|$ is polynomial.
But, $|\mathcal{G}| = \infty$!

Example $w_p(G) = 1/8$.

For a threshold $\tau \in [0, 1]$,

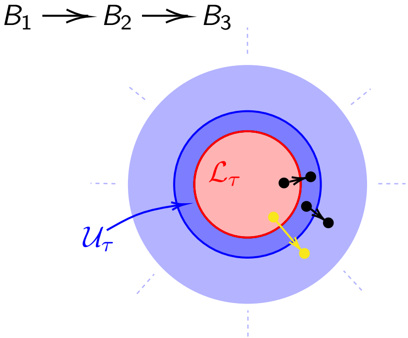
- let $\mathcal{L}_\tau \subseteq \mathcal{G}$ be the set of likely witness DAGs, $w_p(G) \geq \tau$;
- let $\mathcal{U}_\tau \subseteq \mathcal{G}$ be the set of (most likely) unlikely witness DAGs, $w_p(G) < \tau$ such that all strict prefixes are likely.



Likely & Unlikely Resamples

Want to find resampling table R such that $|\mathcal{G}[R]|$ is polynomial.
But, $|\mathcal{G}| = \infty$!

Example $w_p(G) = 1/8$.



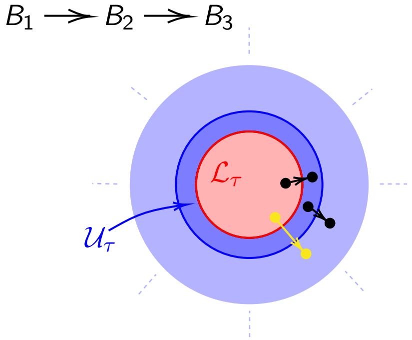
For a threshold $\tau \in [0, 1]$,

- let $\mathcal{L}_\tau \subseteq \mathcal{G}$ be the set of likely witness DAGs, $w_p(G) \geq \tau$;
- let $\mathcal{U}_\tau \subseteq \mathcal{G}$ be the set of (most likely) unlikely witness DAGs, $w_p(G) < \tau$ such that all strict prefixes are likely.

Likely & Unlikely Resamples

Want to find resampling table R such that $|\mathcal{G}[R]|$ is polynomial.
But, $|\mathcal{G}| = \infty$!

Example $w_p(G) = 1/8$.



For a threshold $\tau \in [0, 1]$,

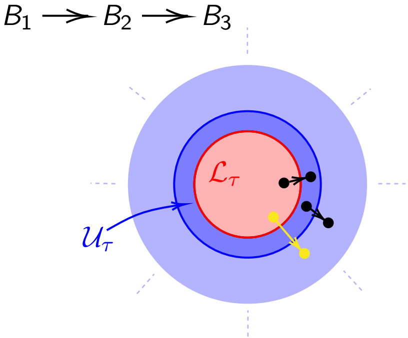
- let $\mathcal{L}_\tau \subseteq \mathcal{C}$ be the set of likely witness DAGs, $w_p(G) \geq \tau$;
- let $\mathcal{U}_\tau \subseteq \mathcal{C}$ be the set of (most likely) unlikely witness DAGs, $w_p(G) < \tau$ such that all strict prefixes are likely.

Need to consider all witness DAGs attainable by removing the sink of some single-sink witness DAG (set denoted $\mathcal{C}$).

Likely & Unlikely Resamples

Want to find resampling table R such that $|\mathcal{G}[R]|$ is polynomial.
But, $|\mathcal{G}| = \infty$!

Example $w_p(G) = 1/8$.



For a threshold $\tau \in [0, 1]$,

- let $\mathcal{L}_\tau \subseteq \mathcal{C}$ be the set of likely witness DAGs, $w_p(G) \geq \tau$;
- let $\mathcal{U}_\tau \subseteq \mathcal{C}$ be the set of (most likely) unlikely witness DAGs, $w_p(G) < \tau$ such that all strict prefixes are likely.

Need to consider all witness DAGs attainable by removing the sink of some single-sink witness DAG (set denoted $\mathcal{C}$).
$\rightsquigarrow$ fixing resampling table R, if $\mathcal{U}_\tau[R] = \emptyset$, then $\mathcal{C}[R] \subseteq \mathcal{L}_\tau[R]$.

Finding Resampling Table avoiding $\mathcal{U}_\tau$

Using the method of conditional expectation, we find R such that

$$|\mathcal{U}_\tau[R]| \leq \mathbb{E}_{R\sim D}|\mathcal{U}_\tau[R]|$$

Finding Resampling Table avoiding $\mathcal{U}_\tau$

Using the method of conditional expectation, we find R such that

$$|\mathcal{U}_\tau[R]| \leq \mathbb{E}_{R\sim D}|\mathcal{U}_\tau[R]| = w_p(\mathcal{U}_\tau).$$

Finding Resampling Table avoiding $\mathcal{U}_\tau$

Using the method of conditional expectation, we find R such that

$$|\mathcal{U}_\tau[R]| \leq \mathbb{E}_{R\sim D}|\mathcal{U}_\tau[R]| = w_p(\mathcal{U}_\tau).$$

$\rightsquigarrow$ if we choose τ such that $w_p(\mathcal{U}_\tau) < 1$,
then $\mathcal{U}_\tau[R] = \emptyset$ and $\mathcal{G}[R] \subseteq \mathcal{L}_\tau[R]$.

Finding Resampling Table avoiding $\mathcal{U}_\tau$

Using the method of conditional expectation, we find R such that

$$|\mathcal{U}_\tau[R]| \le \mathbb{E}_{R\sim D}|\mathcal{U}_\tau[R]| = w_p(\mathcal{U}_\tau).$$

$\rightsquigarrow$ if we choose τ such that $w_p(\mathcal{U}_\tau) < 1$,
then $\mathcal{U}_\tau[R] = \emptyset$ and $\mathcal{G}[R] \subseteq \mathcal{L}_\tau[R]$.

What is the effect of changing τ?

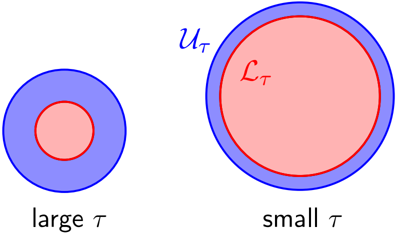
Finding Resampling Table avoiding $\mathcal{U}_\tau$

Using the method of conditional expectation, we find R such that

$$|\mathcal{U}_\tau[R]| \leq \mathbb{E}_{R\sim D}|\mathcal{U}_\tau[R]| = w_p(\mathcal{U}_\tau).$$

$\rightsquigarrow$ if we choose τ such that $w_p(\mathcal{U}_\tau) < 1$,
then $\mathcal{U}_\tau[R] = \emptyset$ and $\mathcal{G}[R] \subseteq \mathcal{L}_\tau[R]$.

What is the effect of changing τ?



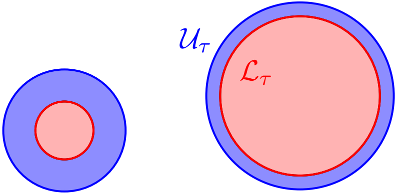
Finding Resampling Table avoiding $\mathcal{U}_\tau$

Using the method of conditional expectation, we find R such that

$$|\mathcal{U}_\tau[R]| \leq \mathbb{E}_{R\sim D}|\mathcal{U}_\tau[R]| = w_p(\mathcal{U}_\tau).$$

$\rightsquigarrow$ if we choose τ such that $w_p(\mathcal{U}_\tau) < 1$,
then $\mathcal{U}_\tau[R] = \emptyset$ and $\mathcal{G}[R] \subseteq \mathcal{L}_\tau[R]$.

What is the effect of changing τ?



large τ | small τ

small $|\mathcal{L}_\tau|$ and $|\mathcal{U}_\tau|$,
but large $w_p(\mathcal{U}_\tau)$

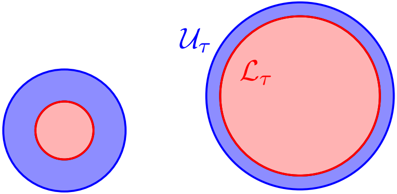
Finding Resampling Table avoiding $\mathcal{U}_\tau$

Using the method of conditional expectation, we find R such that

$$|\mathcal{U}_\tau[R]| \leq \mathbb{E}_{R\sim D}|\mathcal{U}_\tau[R]| = w_p(\mathcal{U}_\tau).$$

$\rightsquigarrow$ if we choose τ such that $w_p(\mathcal{U}_\tau) < 1$,
then $\mathcal{U}_\tau[R] = \emptyset$ and $\mathcal{G}[R] \subseteq \mathcal{L}_\tau[R]$.

What is the effect of changing τ?



large τ	small τ
small $\lvert\mathcal{L}_\tau\rvert$ and $\lvert\mathcal{U}_\tau\rvert$, but large $w_p(\mathcal{U}_\tau)$	large $\lvert\mathcal{L}_\tau\rvert$ and $\lvert\mathcal{U}_\tau\rvert$, but small $w_p(\mathcal{U}_\tau)$ if $w_p(\mathcal{C}) < \infty$

Choosing the Threshold

Can we choose τ such that $w_p(\mathcal{U}_\tau) < 1$ and $\mathcal{U}_\tau$ and $\mathcal{L}_\tau$ are of polynomial size?

Choosing the Threshold

Can we choose τ such that $w_p(\mathcal{U}_\tau) < 1$ and $\mathcal{U}_\tau$ and $\mathcal{L}_\tau$ are of polynomial size?

What is the largest τ guaranteeing $w_p(\mathcal{U}_\tau) < 1$?

Choosing the Threshold

Can we choose τ such that $w_p(\mathcal{U}_\tau) < 1$ and $\mathcal{U}_\tau$ and $\mathcal{L}_\tau$ are of polynomial size?
What is the largest τ guaranteeing $w_p(\mathcal{U}_\tau) < 1$?

$$w_p(G) = w_{p^{1-\epsilon}}(G)^{\frac{1}{1-\epsilon}}$$

Choosing the Threshold

Can we choose τ such that $w_p(\mathcal{U}_\tau) < 1$ and $\mathcal{U}_\tau$ and $\mathcal{L}_\tau$ are of polynomial size?
What is the largest τ guaranteeing $w_p(\mathcal{U}_\tau) < 1$?

$$w_p(G) = w_{p^{1-\epsilon}}(G)^{\frac{1}{1-\epsilon}} = w_{p^{1-\epsilon}}(G)^{1+\epsilon'} = w_{p^{1-\epsilon}}(G)^{\epsilon'} w_{p^{1-\epsilon}}(G).$$

Choosing the Threshold

Can we choose τ such that $w_p(\mathcal{U}_\tau) < 1$ and $\mathcal{U}_\tau$ and $\mathcal{L}_\tau$ are of polynomial size?
What is the largest τ guaranteeing $w_p(\mathcal{U}_\tau) < 1$?

$$w_p(G) = w_{p^{1-\epsilon}}(G)^{\frac{1}{1-\epsilon}} = w_{p^{1-\epsilon}}(G)^{1+\epsilon'} = \underbrace{w_{p^{1-\epsilon}}(G)}_{<\tau}{}^{\epsilon'} w_{p^{1-\epsilon}}(G).$$
$$\rightsquigarrow \; w_p(\mathcal{U}_\tau) < \tau^{\epsilon'} w_{p^{1-\epsilon}}(\mathcal{U}_\tau).$$

Choosing the Threshold

Can we choose τ such that $w_p(\mathcal{U}_\tau) < 1$ and $\mathcal{U}_\tau$ and $\mathcal{L}_\tau$ are of polynomial size?
What is the largest τ guaranteeing $w_p(\mathcal{U}_\tau) < 1$?

$$w_p(G) = w_{p^{1-\epsilon}}(G)^{\frac{1}{1-\epsilon}} = w_{p^{1-\epsilon}}(G)^{1+\epsilon'} = \underbrace{w_{p^{1-\epsilon}}(G)}_{<\tau}{}^{\epsilon'} w_{p^{1-\epsilon}}(G).$$

$\rightsquigarrow$ $w_p(\mathcal{U}_\tau) < \tau^{\epsilon'} w_{p^{1-\epsilon}}(\mathcal{U}_\tau)$.

$\rightsquigarrow$ for $\tau \leq \tau_{\max}$, we have $w_p(\mathcal{U}_\tau) < 1$.

Choosing the Threshold

Can we choose τ such that $w_p(\mathcal{U}_\tau) < 1$ and $\mathcal{U}_\tau$ and $\mathcal{L}_\tau$ are of polynomial size?
What is the largest τ guaranteeing $w_p(\mathcal{U}_\tau) < 1$?

$$w_p(G) = w_{p^{1-\epsilon}}(G)^{\frac{1}{1-\epsilon}} = w_{p^{1-\epsilon}}(G)^{1+\epsilon'} = \underbrace{w_{p^{1-\epsilon}}(G)}_{<\tau}{}^{\epsilon'} w_{p^{1-\epsilon}}(G).$$

$\rightsquigarrow$ $w_p(\mathcal{U}_\tau) < \tau^{\epsilon'} w_{p^{1-\epsilon}}(\mathcal{U}_\tau)$.
$\rightsquigarrow$ for $\tau \leq \tau_{\max}$, we have $w_p(\mathcal{U}_\tau) < 1$.

How do we compute τ?

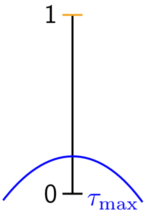
Choosing the Threshold

Can we choose τ such that $w_p(\mathcal{U}_\tau) < 1$ and $\mathcal{U}_\tau$ and $\mathcal{L}_\tau$ are of polynomial size?
What is the largest τ guaranteeing $w_p(\mathcal{U}_\tau) < 1$?

$$w_p(G) = w_{p^{1-\epsilon}}(G)^{\frac{1}{1-\epsilon}} = w_{p^{1-\epsilon}}(G)^{1+\epsilon'} = \underbrace{w_{p^{1-\epsilon}}(G)}_{<\tau}{}^{\epsilon'} w_{p^{1-\epsilon}}(G).$$

$\rightsquigarrow$ $w_p(\mathcal{U}_\tau) < \tau^{\epsilon'} w_{p^{1-\epsilon}}(\mathcal{U}_\tau)$.
$\rightsquigarrow$ for $\tau \leq \tau_{\max}$, we have $w_p(\mathcal{U}_\tau) < 1$.

How do we compute τ?
Use exponential backoff!
Example $\tau = 2^0 = 1$.



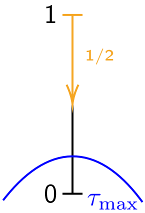
Choosing the Threshold

Can we choose τ such that $w_p(\mathcal{U}_\tau) < 1$ and $\mathcal{U}_\tau$ and $\mathcal{L}_\tau$ are of polynomial size?
What is the largest τ guaranteeing $w_p(\mathcal{U}_\tau) < 1$?

$$w_p(G) = w_{p^{1-\epsilon}}(G)^{\frac{1}{1-\epsilon}} = w_{p^{1-\epsilon}}(G)^{1+\epsilon'} = \underbrace{w_{p^{1-\epsilon}}(G)}_{<\tau}{}^{\epsilon'} w_{p^{1-\epsilon}}(G).$$

$\rightsquigarrow$ $w_p(\mathcal{U}_\tau) < \tau^{\epsilon'} w_{p^{1-\epsilon}}(\mathcal{U}_\tau)$.
$\rightsquigarrow$ for $\tau \leq \tau_{\max}$, we have $w_p(\mathcal{U}_\tau) < 1$.

How do we compute τ?
Use exponential backoff!
Example $\tau = 2^{-1} = 1/2$.



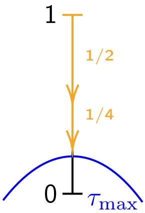
Choosing the Threshold

Can we choose τ such that $w_p(\mathcal{U}_\tau) < 1$ and $\mathcal{U}_\tau$ and $\mathcal{L}_\tau$ are of polynomial size?
What is the largest τ guaranteeing $w_p(\mathcal{U}_\tau) < 1$?

$$w_p(G) = w_{p^{1-\epsilon}}(G)^{\frac{1}{1-\epsilon}} = w_{p^{1-\epsilon}}(G)^{1+\epsilon'} = \underbrace{w_{p^{1-\epsilon}}(G)}_{<\tau}{}^{\epsilon'} w_{p^{1-\epsilon}}(G).$$

$\rightsquigarrow$ $w_p(\mathcal{U}_\tau) < \tau^{\epsilon'} w_{p^{1-\epsilon}}(\mathcal{U}_\tau)$.
$\rightsquigarrow$ for $\tau \leq \tau_{\max}$, we have $w_p(\mathcal{U}_\tau) < 1$.

How do we compute τ?
Use exponential backoff!
Example $\tau = 2^{-2} = 1/4$.



Choosing the Threshold

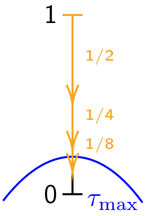
Can we choose τ such that $w_p(\mathcal{U}_\tau) < 1$ and $\mathcal{U}_\tau$ and $\mathcal{L}_\tau$ are of polynomial size?
What is the largest τ guaranteeing $w_p(\mathcal{U}_\tau) < 1$?

$$w_p(G) = w_{p^{1-\epsilon}}(G)^{\frac{1}{1-\epsilon}} = w_{p^{1-\epsilon}}(G)^{1+\epsilon'} = \underbrace{w_{p^{1-\epsilon}}(G)}_{<\tau}{}^{\epsilon'} w_{p^{1-\epsilon}}(G).$$

$\rightsquigarrow$ $w_p(\mathcal{U}_\tau) < \tau^{\epsilon'} w_{p^{1-\epsilon}}(\mathcal{U}_\tau)$.

$\rightsquigarrow$ for $\tau \leq \tau_{\max}$, we have $w_p(\mathcal{U}_\tau) < 1$.

How do we compute τ?
Use exponential backoff!
Example $\tau = 2^{-3} = {}^1\!/_8$.



Choosing the Threshold

Can we choose τ such that $w_p(\mathcal{U}_\tau) < 1$ and $\mathcal{U}_\tau$ and $\mathcal{L}_\tau$ are of polynomial size?
What is the largest τ guaranteeing $w_p(\mathcal{U}_\tau) < 1$?

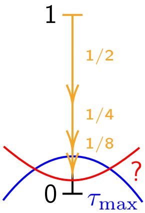
$$w_p(G) = w_{p^{1-\epsilon}}(G)^{\frac{1}{1-\epsilon}} = w_{p^{1-\epsilon}}(G)^{1+\epsilon'} = \underbrace{w_{p^{1-\epsilon}}(G)}_{<\tau}{}^{\epsilon'} w_{p^{1-\epsilon}}(G).$$

$\rightsquigarrow$ $w_p(\mathcal{U}_\tau) < \tau^{\epsilon'} w_{p^{1-\epsilon}}(\mathcal{U}_\tau)$.

$\rightsquigarrow$ for $\tau \leq \tau_{\max}$, we have $w_p(\mathcal{U}_\tau) < 1$.

How do we compute τ?
Use exponential backoff!
Example $\tau = 2^{-3} = 1/8$.

Are $\mathcal{U}_\tau$ and $\mathcal{L}_\tau$ of polynomial size?



Is $\mathcal{U}_\tau \cup \mathcal{L}_\tau$ of polynomial size?

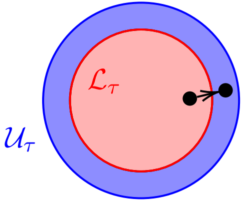
Need to bound # of multi-sink witness DAGs
and # of single-sink witness DAGs.

Is $\mathcal{U}_\tau \cup \mathcal{L}_\tau$ of polynomial size?

Need to bound # of multi-sink witness DAGs
and # of single-sink witness DAGs.

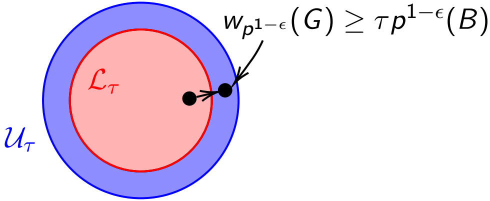
Is $\mathcal{U}_\tau \cup \mathcal{L}_\tau$ of polynomial size?

Need to bound # of multi-sink witness DAGs
and <u># of single-sink witness DAGs</u>.



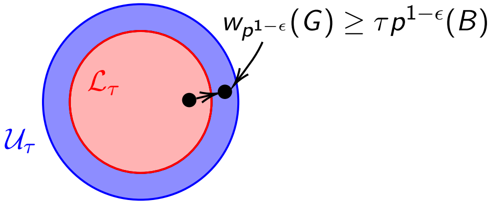
Is $\mathcal{U}_\tau \cup \mathcal{L}_\tau$ of polynomial size?

Need to bound # of multi-sink witness DAGs
and # of single-sink witness DAGs.



Is $\mathcal{U}_\tau \cup \mathcal{L}_\tau$ of polynomial size?

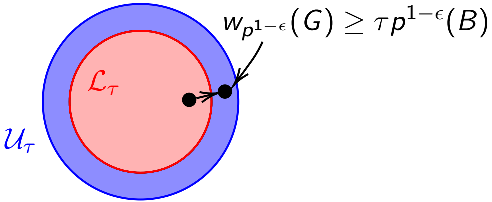
Need to bound # of multi-sink witness DAGs
and # of single-sink witness DAGs.



$\rightsquigarrow \frac{w_{p^{1-\epsilon}}(G)}{\tau p^{1-\epsilon}(B)} \geq 1.$

Is $\mathcal{U}_\tau \cup \mathcal{L}_\tau$ of polynomial size?

Need to bound # of multi-sink witness DAGs
and # of single-sink witness DAGs.

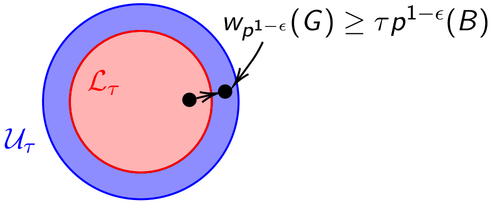


$\rightsquigarrow$ $\frac{w_{p^{1-\epsilon}}(G)}{\tau p^{1-\epsilon}(B)} \geq 1.$

$\rightsquigarrow$ $|\mathcal{U}_\tau \cup \mathcal{L}_\tau| \leq \sum_{B \in \mathcal{B}} \frac{w_{p^{1-\epsilon}}(\mathcal{G}_B)}{\tau p^{1-\epsilon}(B)}$

Is $\mathcal{U}_\tau \cup \mathcal{L}_\tau$ of polynomial size?

Need to bound # of multi-sink witness DAGs
and # of single-sink witness DAGs.

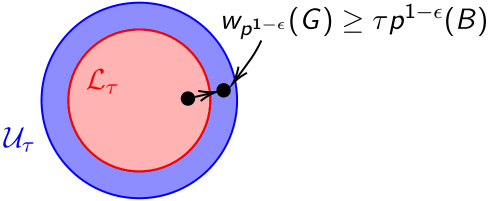


$\rightsquigarrow$ $\frac{w_{p^{1-\epsilon}}(G)}{\tau p^{1-\epsilon}(B)} \geq 1$.

$\rightsquigarrow$ $|\mathcal{U}_\tau \cup \mathcal{L}_\tau| \leq \sum_{B \in \mathcal{B}} \frac{w_{p^{1-\epsilon}}(\mathcal{G}_B)}{\tau p^{1-\epsilon}(B)} \doteq \frac{W_\epsilon}{\tau}$,
where W_ϵ is the work parameter.

Is $\mathcal{U}_\tau \cup \mathcal{L}_\tau$ of polynomial size?

Need to bound # of multi-sink witness DAGs
and # of single-sink witness DAGs.



$\rightsquigarrow$ $\frac{w_{p^{1-\epsilon}}(G)}{\tau p^{1-\epsilon}(B)} \geq 1.$

$\rightsquigarrow$ $|\mathcal{U}_\tau \cup \mathcal{L}_\tau| \leq \sum_{B \in \mathcal{B}} \frac{w_{p^{1-\epsilon}}(\mathcal{G}_B)}{\tau p^{1-\epsilon}(B)} \doteq \frac{W_\epsilon}{\tau}$,
where W_ϵ is the work parameter.

W_ϵ is polynomial under common LLL conditions!

The Algorithm

Algorithm: Deterministic MT-Algorithm

The Algorithm

Algorithm: Deterministic MT-Algorithm

Using exponential backoff, select "large" τ such that $w_p(\mathcal{U}_\tau) < 1$
Using method of conditional expectations, find resampling table R avoiding $\mathcal{U}_\tau$
Run the deterministic MT algorithm on R

We have seen that the final step takes at most $|\mathcal{G}[R]| \leq |\mathcal{L}_\tau[R]|$ iterations!

Limitations

This algorithm does not cover some scenarios:

- superpolynomial $|\mathcal{B}|$ and $|\Sigma|$

Limitations

This algorithm does not cover some scenarios:

- superpolynomial $|\mathcal{B}|$ and $|\Sigma|$
- non-variable probability spaces

Limitations

This algorithm does not cover some scenarios:

- superpolynomial $|\mathcal{B}|$ and $|\Sigma|$
- non-variable probability spaces
- does not cover lopsidependency

Thanks for your attention! Questions?

Computing the Resampling Table

Can R be computed efficiently?

Computing the Resampling Table

Can R be computed efficiently?

Observe: The MT algorithm uses at most as many columns as the size of the largest witness DAG in $\mathcal{L}_\tau$

Computing the Resampling Table

Can R be computed efficiently?

Observe: The MT algorithm uses at most as many columns as the size of the largest witness DAG in $\mathcal{L}_\tau$, which is at most $|\mathcal{L}_\tau|$.

Computing the Resampling Table

Can R be computed efficiently?

Observe: The MT algorithm uses at most as many columns as the size of the largest witness DAG in $\mathcal{L}_\tau$, which is at most $|\mathcal{L}_\tau|$.

For each cell of R, choose one of $|\Sigma|$ values to minimize the conditional probability of G & R being compatible for each $G \in \mathcal{U}_\tau$.

Computing the Resampling Table

Can R be computed efficiently?

Observe: The MT algorithm uses at most as many columns as the size of the largest witness DAG in $\mathcal{L}_\tau$, which is at most $|\mathcal{L}_\tau|$.

For each cell of R, choose one of $|\Sigma|$ values to minimize the conditional probability of G & R being compatible for each $G \in \mathcal{U}_\tau$.

$\rightsquigarrow \mathcal{O}(n|\mathcal{L}_\tau| \cdot |\Sigma| \cdot |\mathcal{L}_\tau| T \cdot |\mathcal{U}_\tau|)$

Computing the Resampling Table

Can R be computed efficiently?

<u>Observe</u>: The MT algorithm uses at most as many columns as the size of the largest witness DAG in $\mathcal{L}_\tau$, which is at most $|\mathcal{L}_\tau|$.

For each cell of R, choose one of $|\Sigma|$ values to minimize the conditional probability of G & R being compatible for each $G \in \mathcal{U}_\tau$.

$\rightsquigarrow$ $\mathcal{O}(n|\mathcal{L}_\tau| \cdot |\Sigma| \cdot |\mathcal{L}_\tau| T \cdot |\mathcal{U}_\tau|)$, where T is the runtime of computing conditional probabilities of bad-events given a partial resampling table.

Computing the Resampling Table

Can R be computed efficiently?

<u>Observe</u>: The MT algorithm uses at most as many columns as the size of the largest witness DAG in $\mathcal{L}_\tau$, which is at most $|\mathcal{L}_\tau|$.

For each cell of R, choose one of $|\Sigma|$ values to minimize the conditional probability of G & R being compatible for each $G \in \mathcal{U}_\tau$.

$\rightsquigarrow$ $\mathcal{O}(n|\mathcal{L}_\tau| \cdot |\Sigma| \cdot |\mathcal{L}_\tau| T \cdot |\mathcal{U}_\tau|)$, where T is the runtime of computing conditional probabilities of bad-events given a partial resampling table.
Also need to generate $\mathcal{U}_\tau$, which can be done in $\mathrm{poly}(|\mathcal{U}_\tau|)$ time.

Polynomial Bound of ${w_{p^{1-\epsilon}}(\mathcal{G}_B)}/{p^{1-\epsilon}(B)}$

$$\mu^{(h)}(I) = w(\{G \mid \text{sink } I, \text{ max. depth } h\}) \rightsquigarrow \mu(B) = w(\mathcal{G}_B).$$

Polynomial Bound of $w_{p^{1-\epsilon}}(\mathcal{G}_B) / p^{1-\epsilon}(B)$

$$\mu^{(h)}(I) = w(\{G \mid \text{sink } I, \text{ max. depth } h\}) \rightsquigarrow \mu(B) = w(\mathcal{G}_B).$$

We have,

1. $\mu^{(h+1)}(I) = p(I) \sum_{J \in \mathsf{Stab}(I)} \mu^{(h)}(J)$

Polynomial Bound of $w_{p^{1-\epsilon}}(\mathcal{G}_B) / p^{1-\epsilon}(B)$

$$\mu^{(h)}(I) = w(\{G \mid \text{sink } I, \text{ max. depth } h\}) \rightsquigarrow \mu(B) = w(\mathcal{G}_B).$$

We have,

1. $\mu^{(h+1)}(I) = p(I) \sum_{J \in \mathsf{Stab}(I)} \mu^{(h)}(J)$
2. $\mu^{(h)}(I) \leq \prod_{B \in I} \mu^{(h)}(B)$ if $\mu(B) \doteq ep(B)$

Polynomial Bound of $w_{p^{1-\epsilon}}(\mathcal{G}_B)/p^{1-\epsilon}(B)$

$$\mu^{(h)}(I) = w(\{G \mid \text{sink } I, \text{ max. depth } h\}) \rightsquigarrow \mu(B) = w(\mathcal{G}_B).$$

We have,

1. $\mu^{(h+1)}(I) = p(I) \sum_{J \in \mathsf{Stab}(I)} \mu^{(h)}(J)$
2. $\mu^{(h)}(I) \leq \prod_{B \in I} \mu^{(h)}(B)$ if $\mu(B) \doteq ep(B)$

$$\begin{aligned}\mu^{(h+1)}(B) &= p(B) \sum_{J \in \mathsf{Stab}(B)} \mu^{(h)}(J) \\ &\leq p(B) \sum_{J \subseteq \bar{\Gamma}(B)} \prod_{B' \in J} \mu^{(h)}(B')\end{aligned}$$

$$\begin{aligned}\sum_{J \subseteq \bar{\Gamma}(B)} \prod_{B' \in J} ep(B') &\leq \sum_{J \subseteq \bar{\Gamma}(B)} (ep_{\max})^{|J|} \leq \sum_{k=0}^{d} \binom{d}{k} (ep_{\max})^k \\ &= (1 + ep_{\max})^d \leq \exp(\underbrace{ep_{\max} d}_{\leq 1}) \leq e.\end{aligned}$$