Improved bounds for randomly colouring simple hypergraphs

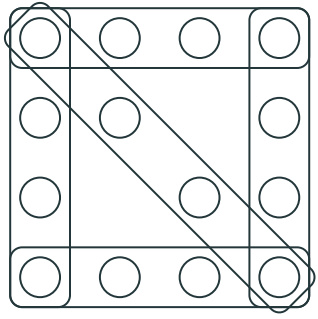
Weiming Feng Heng Guo **Jiaheng Wang**

University of Edinburgh

RANDOM 2022

Hypergraph (proper) colouring

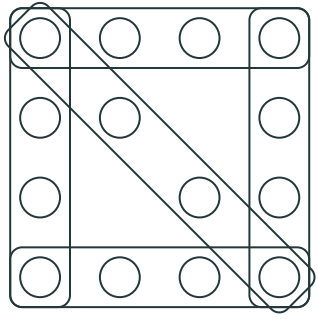
Classical combinatorial/computational problem!



Hypergraph (proper) colouring

Classical combinatorial/computational problem!

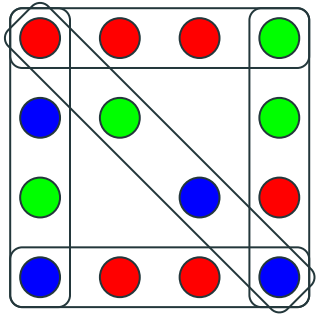
- Hypergraph $(V, \mathcal{E})$
  - Hyperedge $e \in \mathcal{E} : e \subseteq V$



Hypergraph (proper) colouring

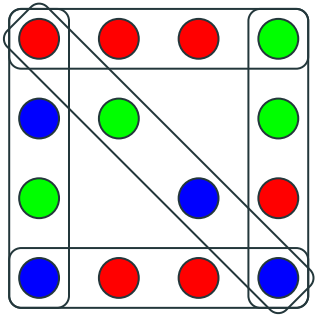
Classical combinatorial/computational problem!

- Hypergraph $(V, \mathcal{E})$
  - Hyperedge $e \in \mathcal{E} : e \subseteq V$
- Proper colouring
  - Forbidding monochromatic hyperedges



Computational problems

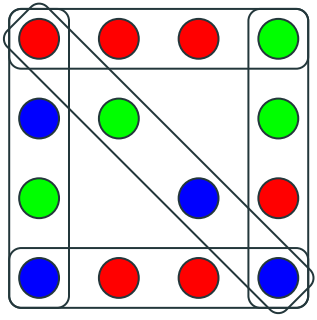
Three computational problems:



Computational problems

Three computational problems:

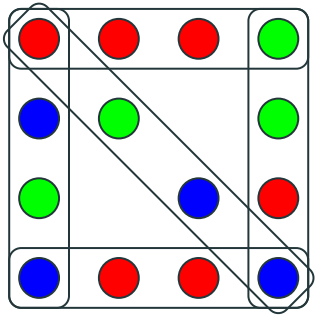
- Deciding
  - Decide if a colouring exists



Computational problems

Three computational problems:

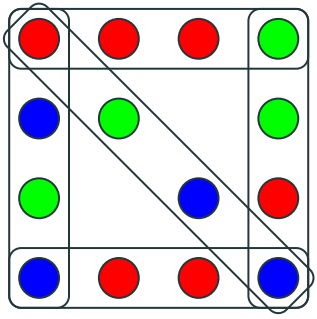
- Deciding
  - Decide if a colouring exists
- Searching
  - Construct a colouring



Computational problems

Three computational problems:

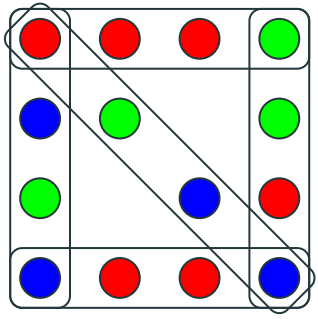
- Deciding
  - Decide if a colouring exists
- Searching
  - Construct a colouring
- Sampling / approximate counting
  - Output a uniform random colouring
  - Estimate the number of colourings



Computational problems

Three computational problems:

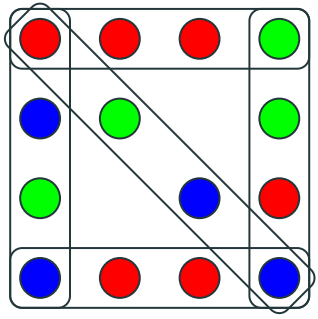
- Deciding
  - Decide if a colouring exists
- Searching
  - Construct a colouring
- Sampling / approximate counting
  - Output a uniform random colouring
  - Estimate the number of colourings
  - Self-reduction **[Jerrum-Vazirani-Vigoda'86]**



Computational problems

Three computational problems:

- Deciding
  - Decide if a colouring exists
- Searching
  - Construct a colouring
- Sampling / approximate counting
  - Output a uniform random colouring
  - Estimate the number of colourings
  - Self-reduction **[Jerrum-Vazirani-Vigoda'86]**

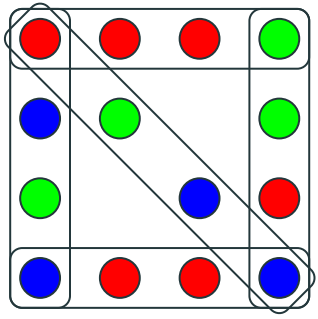


Deciding is $\mathbf{NP}$-hard in general (3-colourings on graphs).

Computational problems

Three computational problems:

- Deciding (hard)
  - Decide if a colouring exists
- Searching (hard)
  - Construct a colouring
- Sampling / approximate counting (hard)
  - Output a uniform random colouring
  - Estimate the number of colourings
  - Self-reduction **[Jerrum-Vazirani-Vigoda'86]**

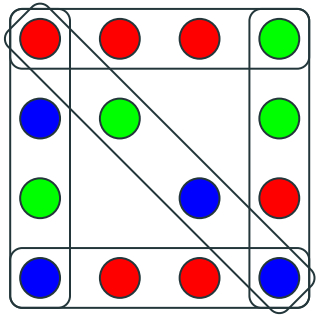


Deciding is $\mathbf{NP}$-hard in general (3-colourings on graphs).

Computational problems

Three computational problems:

- Deciding (hard)
  - Decide if a colouring exists
- Searching (hard)
  - Construct a colouring
- Sampling / approximate counting (hard)
  - Output a uniform random colouring
  - Estimate the number of colourings
  - Self-reduction **[Jerrum-Vazirani-Vigoda'86]**



Deciding is $\mathbf{NP}$-hard in general (3-colourings on graphs).

Posing restrictions to input instances?

(Symmetric) Lovász local lemma (LLL)

Treat hyperedges being monochromatic as (random) “bad” events...

(Symmetric) Lovász local lemma (LLL)

Treat hyperedges being monochromatic as (random) “bad” events...

Local lemma: weak dependency $\rightarrow$ non-zero probability of being good.

(Symmetric) Lovász local lemma (LLL)

Treat hyperedges being monochromatic as (random) “bad” events...

Local lemma: weak dependency $\rightarrow$ non-zero probability of being good.

- B_i: bad events with $\Pr[B_i] = p$.

(Symmetric) Lovász local lemma (LLL)

Treat hyperedges being monochromatic as (random) “bad” events...

Local lemma: weak dependency $\rightarrow$ non-zero probability of being good.

- B_i: bad events with $\Pr[B_i] = p$.
- Each depends on $\leq D$ other events.

(Symmetric) Lovász local lemma (LLL)

Treat hyperedges being monochromatic as (random) "bad" events...

Local lemma: weak dependency $\rightarrow$ non-zero probability of being good.

- B_i: bad events with $\Pr[B_i] = p$.
- Each depends on $\leq D$ other events.

Lemma (Symmetric Lovász local lemma [Erdős-Lovász'75])

If

$$\mathrm{e} \cdot p \cdot (D + 1) \leq 1,$$

then there is a non-zero probability that no bad event happens.

(Symmetric) Lovász local lemma (LLL)

Treat hyperedges being monochromatic as (random) "bad" events...

Local lemma: weak dependency $\rightarrow$ non-zero probability of being good.

- B_i: bad events with $\Pr[B_i] = p$. number of colours; size of hyperedges
- Each depends on $\leq D$ other events. maximum degree of vertices

Lemma (Symmetric Lovász local lemma [Erdős-Lovász'75])

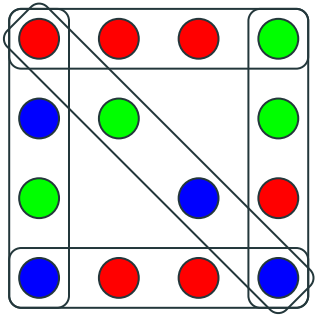
If

$$\mathrm{e} \cdot p \cdot (D+1) \leq 1,$$

then there is a non-zero probability that no bad event happens.

LLL on hypergraph colourings

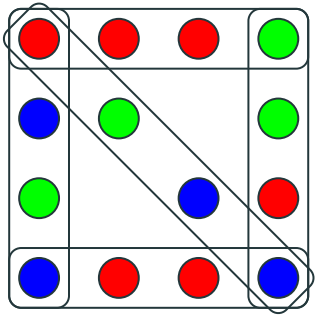
Parameters:



LLL on hypergraph colourings

Parameters:

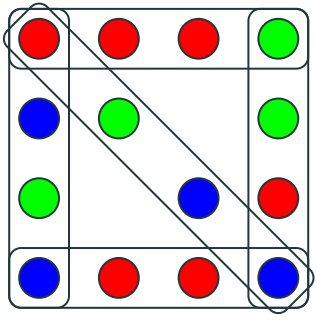
- q: Number of colours



LLL on hypergraph colourings

Parameters:

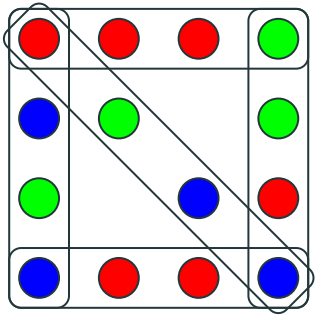
- q: Number of colours
- k: Size of hyperedges (k-uniform)



LLL on hypergraph colourings

Parameters:

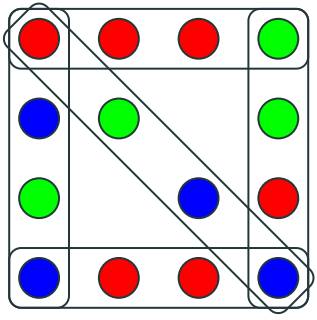
- q: Number of colours
- k: Size of hyperedges (k-uniform)
- Degree of a vertex: number of its incident hyperedges



LLL on hypergraph colourings

Parameters:

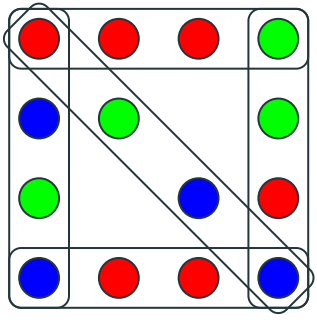
- q: Number of colours
- k: Size of hyperedges (k-uniform)
- Degree of a vertex: number of its incident hyperedges
- Δ: Maximum degree of vertices



LLL on hypergraph colourings

Parameters:

- q: Number of colours
- k: Size of hyperedges (k-uniform)
- Degree of a vertex: number of its incident hyperedges
- Δ: Maximum degree of vertices



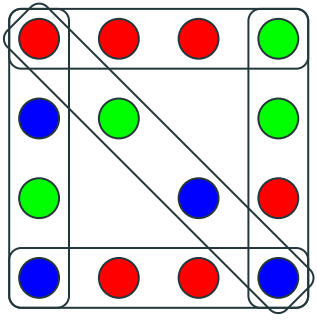
Apply LLL: a colouring exists if

$$\Delta \leq q^{k-1}/(\mathrm{e}k)$$

LLL on hypergraph colourings

Parameters:

- q: Number of colours
- k: Size of hyperedges (k-uniform)
- Degree of a vertex: number of its incident hyperedges
- Δ: Maximum degree of vertices



Apply LLL: a colouring exists if

$$\Delta \leq q^{k-1}/(\mathrm{e}k)$$

Computational problems: Lovász local lemma regime

Assuming LLL ($\Delta \leq q^{k-1}/(\mathrm{e}k)$) on input instances:

- Deciding
- Searching
- Sampling

Computational problems: Lovász local lemma regime

Assuming LLL ($\Delta \leq q^{k-1}/(\mathrm{e}k)$) on input instances:

- Deciding
  - Trivial! Just output YES
- Searching
- Sampling

Computational problems: Lovász local lemma regime

Assuming LLL ($\Delta \leq q^{k-1}/(\mathrm{e}k)$) on input instances:

- Deciding
  - Trivial! Just output YES
- Searching
  - Easy: Moser-Tardos algorithm **[Moser-Tardos'10]**
- Sampling

Computational problems: Lovász local lemma regime

Assuming LLL ($\Delta \leq q^{k-1}/(\mathrm{e}k)$) on input instances:

- Deciding
  - Trivial! Just output YES
- Searching
  - Easy: Moser-Tardos algorithm **[Moser-Tardos'10]**
  - Computational threshold (asymptotically) **[Gebauer-Szabó-Tardos'16]**
- Sampling

Computational problems: Lovász local lemma regime

Assuming LLL ($\Delta \leq q^{k-1}/(\mathrm{e}k)$) on input instances:

- Deciding
  - Trivial! Just output YES
- Searching
  - Easy: Moser-Tardos algorithm **[Moser-Tardos'10]**
  - Computational threshold (asymptotically) **[Gebauer-Szabó-Tardos'16]**
- Sampling
  - Moser-Tardos does not generate uniform colourings.

Computational problems: Lovász local lemma regime

Assuming LLL ($\Delta \leq q^{k-1}/(\mathrm{e}k)$) on input instances:

- Deciding
  - Trivial! Just output YES
- Searching
  - Easy: Moser-Tardos algorithm **[Moser-Tardos'10]**
  - Computational threshold (asymptotically) **[Gebauer-Szabó-Tardos'16]**
- Sampling
  - Moser-Tardos does not generate uniform colourings.
  - **NP**-hard, even when significantly below LLL threshold **[Galanis-Guo-W.'22]**

Computational problems: Lovász local lemma regime

Assuming LLL ($\Delta \leq q^{k-1}/(\mathrm{e}k)$) on input instances:

- Deciding
  - Trivial! Just output YES
- Searching
  - Easy: Moser-Tardos algorithm **[Moser-Tardos'10]**
  - Computational threshold (asymptotically) **[Gebauer-Szabó-Tardos'16]**
- Sampling
  - Moser-Tardos does not generate uniform colourings.
  - **NP**-hard, when $\Delta \geq 5 \cdot q^{k/2}$ and q is even **[Galanis-Guo-W.'22]**

Computational problems: ~~Lovász~~ local lemma regime

Assuming LLL ($\Delta \leq q^{k-1}/(\mathrm{e}k)$) on input instances:

- Deciding
  - Trivial! Just output YES
- Searching
  - Easy: Moser-Tardos algorithm **[Moser-Tardos'10]**
  - Computational threshold (asymptotically) **[Gebauer-Szabó-Tardos'16]**
- Sampling
  - Moser-Tardos does not generate uniform colourings.
  - **NP**-hard, when $\Delta \geq 5 \cdot q^{k/2}$ and q is even **[Galanis-Guo-W.'22]**

Computational problems: ~~Lovász~~ local lemma regime

Assuming LLL ($\Delta \leq q^{k-1}/(\mathrm{e}k)$) on input instances:

- Deciding
  - Trivial! Just output YES
- Searching
  - Easy: Moser-Tardos algorithm **[Moser-Tardos'10]**
  - Computational threshold (asymptotically) **[Gebauer-Szabó-Tardos'16]**
- Sampling
  - Moser-Tardos does not generate uniform colourings.
  - **NP**-hard, when $\Delta \geq 5 \cdot q^{k/2}$ and q is even **[Galanis-Guo-W.'22]**
  - Tractable, when $\Delta \lesssim q^{k/3}$ **[Jain-Pham-Vuong'21]** (perfect sampler **[He-Sun-Wu'21]**)

Computational problems: ~~Lovász~~ local lemma regime

Assuming LLL ($\Delta \leq q^{k-1}/(\mathrm{e}k)$) on input instances:

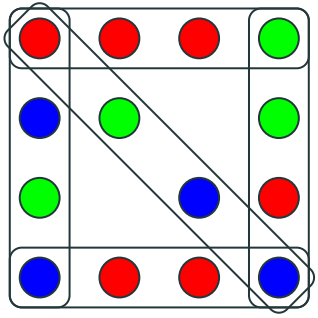
- Deciding
  - Trivial! Just output YES
- Searching
  - Easy: Moser-Tardos algorithm **[Moser-Tardos'10]**
  - Computational threshold (asymptotically) **[Gebauer-Szabó-Tardos'16]**
- Sampling
  - Moser-Tardos does not generate uniform colourings.
  - **NP**-hard, when $\Delta \geq 5 \cdot q^{k/2}$ and q is even **[Galanis-Guo-W.'22]**
  - Tractable, when $\Delta \lesssim q^{k/3}$ **[Jain-Pham-Vuong'21]** (perfect sampler **[He-Sun-Wu'21]**)

Open problem: computational threshold for sampling problem

Simple (aka. linear) hypergraphs

Simple hypergraph:

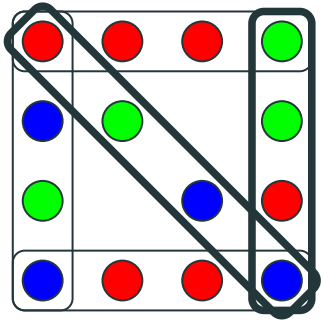
- Overlap of two hyperedges ≤ 1



Simple (aka. linear) hypergraphs

Simple hypergraph:

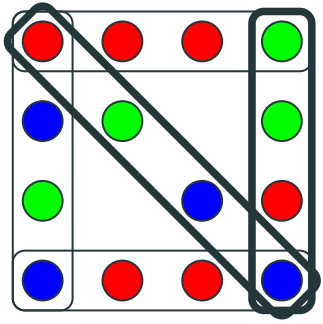
- Overlap of two hyperedges ≤ 1



Simple (aka. linear) hypergraphs

Simple hypergraph:

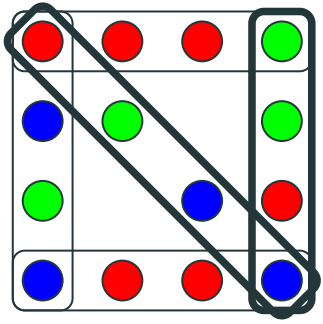
- Overlap of two hyperedges ≤ 1
- Chromatic number by LLL: $\chi(H) \leq C\Delta^{\frac{1}{k-1}}$



Simple (aka. linear) hypergraphs

Simple hypergraph:

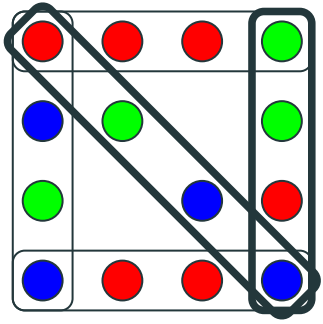
- Overlap of two hyperedges ≤ 1
- Chromatic number by LLL: $\chi(H) \leq C\Delta^{\frac{1}{k-1}}$
- Refined: $\chi(H) \leq C_k \left(\frac{\Delta}{\log \Delta}\right)^{\frac{1}{k-1}}$ **[Frieze-Mubayi'13]**



Simple (aka. linear) hypergraphs

Simple hypergraph:

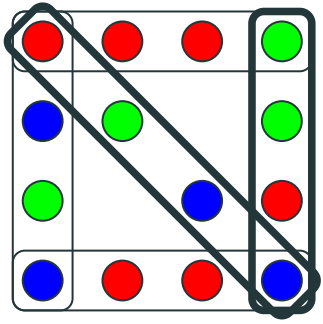
- Overlap of two hyperedges ≤ 1
- LLL: $\Delta \leq q^{k-1}/(\mathrm{e}k)$ guarantees a solution
- Refined: $\Delta \leq C_k k q^{k-1} \log q$ **[Frieze-Mubayi'13]**



Simple (aka. linear) hypergraphs

Simple hypergraph:

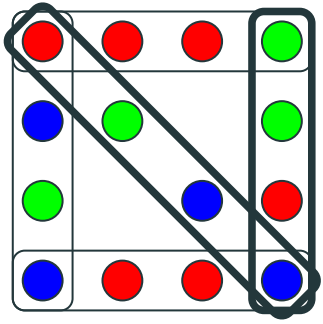
- Overlap of two hyperedges ≤ 1
- LLL: $\Delta \leq q^{k-1}/(\mathrm{e}k)$ guarantees a solution
- Refined: $\Delta \leq C_k k q^{k-1} \log q$ **[Frieze-Mubayi'13]**
- Searching is **NP**-hard when $\Delta \geq 2kq^k \ln q + 2q$
- Sampling is **NP**-hard when $\Delta \geq Ckq^{k-1} \ln q$ **[Galanis-Guo-W.'22]**



Simple (aka. linear) hypergraphs

Simple hypergraph:

- Overlap of two hyperedges ≤ 1
- LLL: $\Delta \leq q^{k-1}/(\mathrm{e}k)$ guarantees a solution
- Refined: $\Delta \leq C_k k q^{k-1} \log q$ **[Frieze-Mubayi'13]**
- Searching is **NP**-hard when $\Delta \geq 2kq^k \ln q + 2q$
- Sampling is **NP**-hard when $\Delta \geq Ckq^{k-1} \ln q$ **[Galanis-Guo-W.'22]**

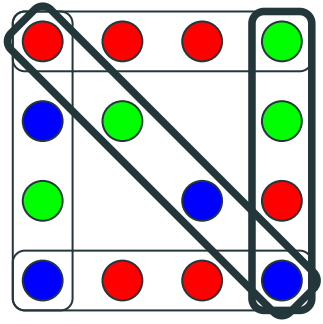


$\Delta \geq 5 \cdot q^{k/2}$ hardness for sampling requires overlap $= k/2$.

Simple (aka. linear) hypergraphs

Simple hypergraph:

- Overlap of two hyperedges ≤ 1
- LLL: $\Delta \leq q^{k-1}/(\mathrm{e}k)$ guarantees a solution
- Refined: $\Delta \leq C_k k q^{k-1} \log q$ **[Frieze-Mubayi'13]**
- Searching is **NP**-hard when $\Delta \geq 2kq^k \ln q + 2q$
- Sampling is **NP**-hard when $\Delta \geq Ckq^{k-1} \ln q$ **[Galanis-Guo-W.'22]**



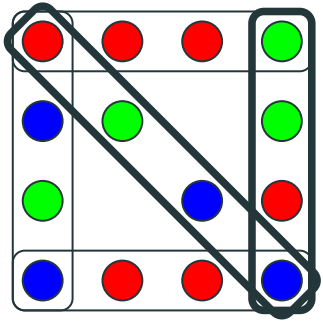
$\Delta \geq 5 \cdot q^{k/2}$ hardness for sampling requires overlap $= k/2$.

Better sampler for simple hypergraphs (than the $\Delta \lesssim q^{k/3}$ one)?

Simple (aka. linear) hypergraphs

Simple hypergraph:

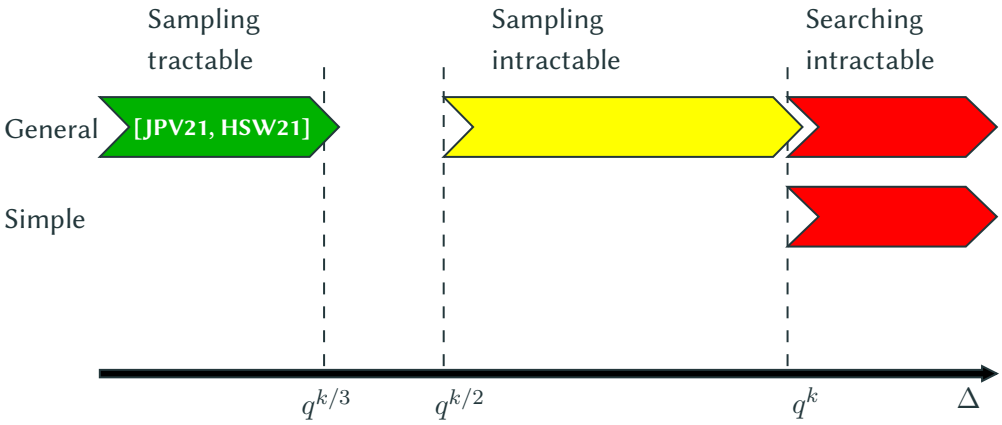
- Overlap of two hyperedges ≤ 1
- LLL: $\Delta \leq q^{k-1}/(\mathrm{e}k)$ guarantees a solution
- Refined: $\Delta \leq C_k k q^{k-1} \log q$ **[Frieze-Mubayi'13]**
- Searching is **NP**-hard when $\Delta \geq 2kq^k \ln q + 2q$
- Sampling is **NP**-hard when $\Delta \geq Ckq^{k-1} \ln q$ **[Galanis-Guo-W.'22]**



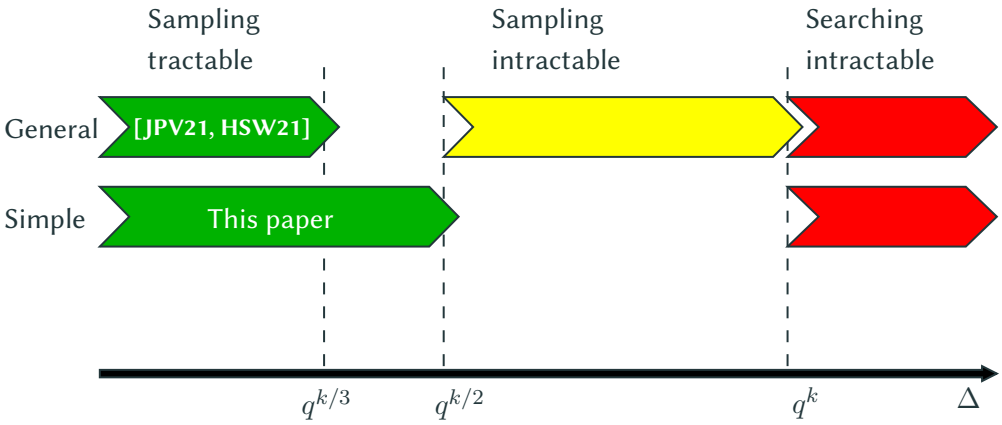
$\Delta \geq 5 \cdot q^{k/2}$ hardness for sampling requires overlap $= k/2$.

Better sampler for simple hypergraphs (than the $\Delta \lesssim q^{k/3}$ one)? **Yes**.

Our result


Sampling
tractable
Sampling
intractable
Searching
intractable
General
[JPV21, HSW21]
Simple
$q^{k/3}$
$q^{k/2}$
q^k
Δ

Our result

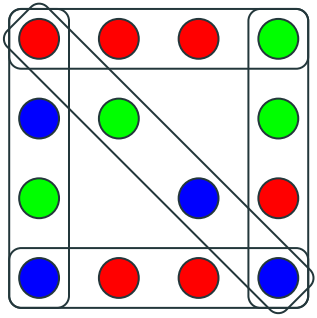


Theorem

There exists an algorithm such that, for any $\delta > 0$, given a k-uniform Δ-degree hypergraph as an input, the algorithm outputs an almost uniform random q-colouring, if $k \geq 20\left(1 + \frac{1}{\delta}\right)$ and $\Delta \leq 0.1^k q^{k/2-(k\delta+1/\delta)}$. The running time is $\tilde{O}(\mathrm{poly}(\Delta k) \cdot n^{1.01})$.

Glauber dynamics

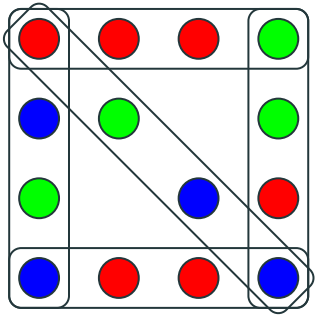
Natural approach: Glauber dynamics



Glauber dynamics

Natural approach: Glauber dynamics

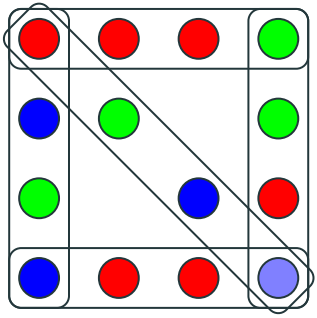
- Uniformly choose one vertex
- Update its value according to its marginal



Glauber dynamics

Natural approach: Glauber dynamics

- Uniformly choose one vertex ⇐
- Update its value according to its marginal

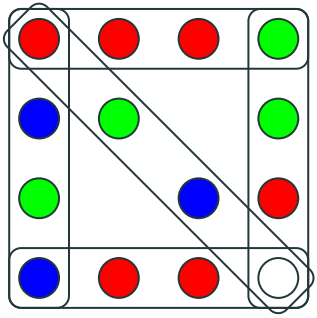


Glauber dynamics

Natural approach: Glauber dynamics

- Uniformly choose one vertex
- Update its value according to its marginal $\Leftarrow$

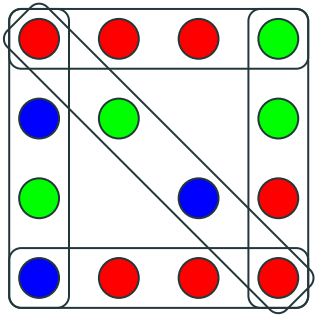
$$\Pr[\bullet] = \frac{1}{3}, \Pr[\bullet] = \frac{1}{3}, \Pr[\bullet] = \frac{1}{3}$$



Glauber dynamics

Natural approach: Glauber dynamics

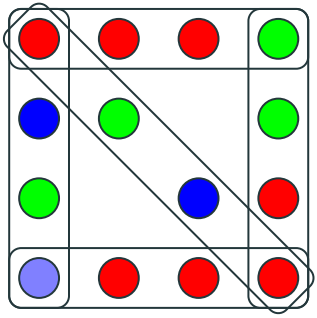
- Uniformly choose one vertex
- Update its value according to its marginal $\Leftarrow$



Glauber dynamics

Natural approach: Glauber dynamics

- Uniformly choose one vertex ⇐
- Update its value according to its marginal

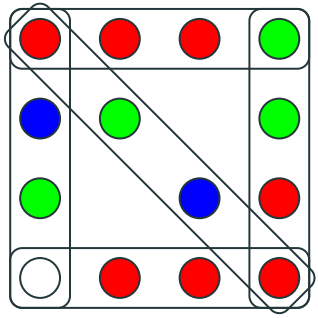


Glauber dynamics

Natural approach: Glauber dynamics

- Uniformly choose one vertex
- Update its value according to its marginal $\Leftarrow$

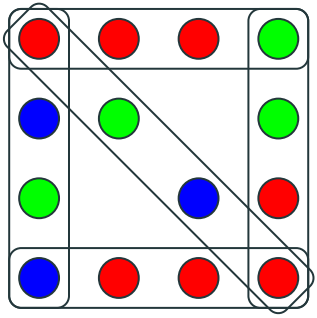
$$\Pr[\bullet] = 0, \Pr[\bullet] = \frac{1}{2}, \Pr[\bullet] = \frac{1}{2}$$



Glauber dynamics

Natural approach: Glauber dynamics

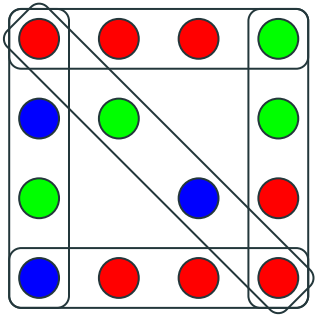
- Uniformly choose one vertex
- Update its value according to its marginal $\Leftarrow$



Glauber dynamics

Natural approach: Glauber dynamics

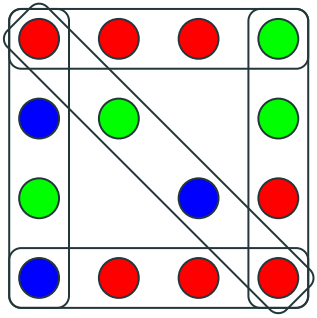
- Uniformly choose one vertex
- Update its value according to its marginal



Glauber dynamics

Natural approach: Glauber dynamics

- Uniformly choose one vertex
- Update its value according to its marginal

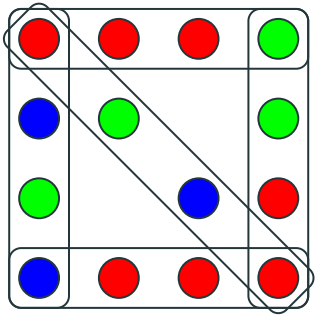


Stationary distribution is uniform (the correct distribution).

Glauber dynamics

Natural approach: Glauber dynamics

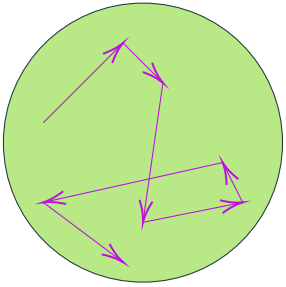
- Uniformly choose one vertex
- Update its value according to its marginal



Stationary distribution is uniform (the correct distribution).

Does this chain *mix rapidly* (i.e., converge to stationary distribution quickly)?

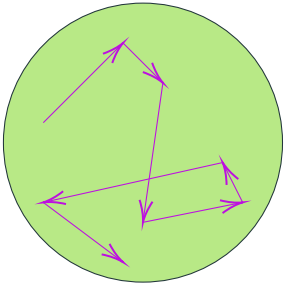
Dystopia of MCMC?



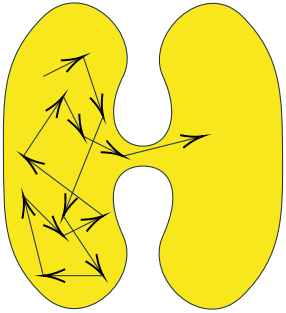
Well connected

Fast mixing

Dystopia of MCMC?

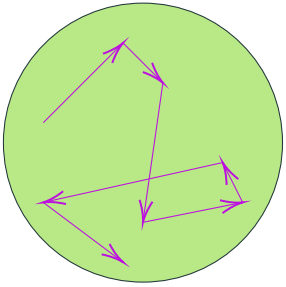


Well connected
Fast mixing

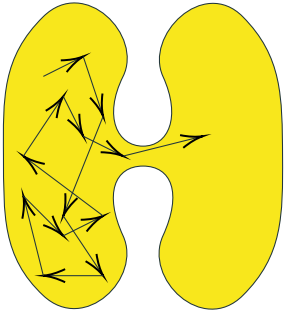


Poorly connected
Slow mixing

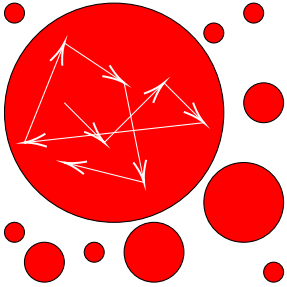
Dystopia of MCMC?



Well connected
Fast mixing

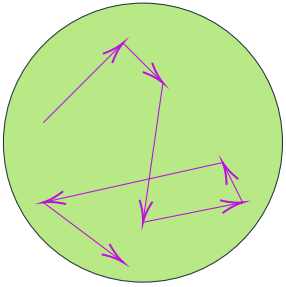


Poorly connected
Slow mixing

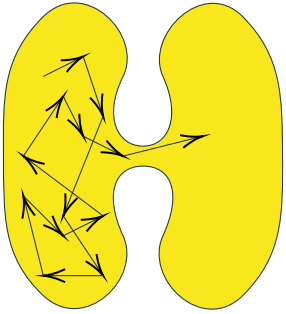


Not connected
Not mixing

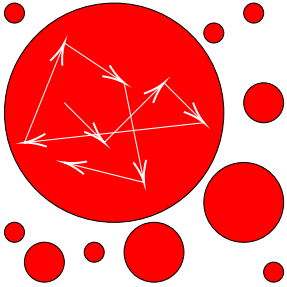
Dystopia of MCMC?



Well connected
Fast mixing

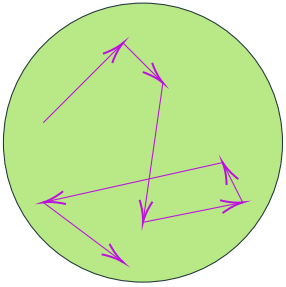


Poorly connected
Slow mixing

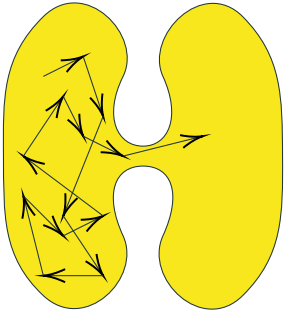


Not connected
Not mixing

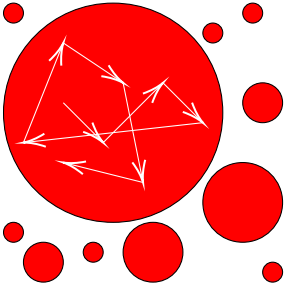
Dystopia of MCMC?



Well connected
Fast mixing



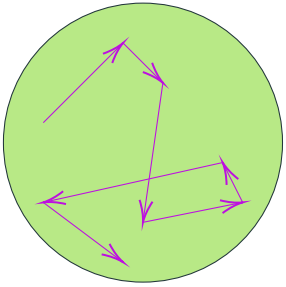
Poorly connected
Slow mixing



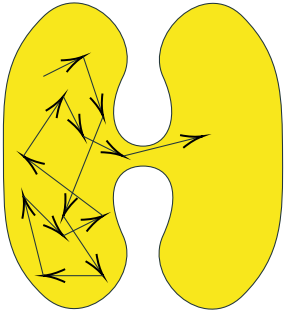
Not connected
Not mixing

Not that bad if there is a giant component — start from a random configuration

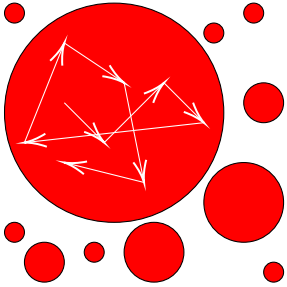
Dystopia of MCMC?



Well connected
Fast mixing



Poorly connected
Slow mixing

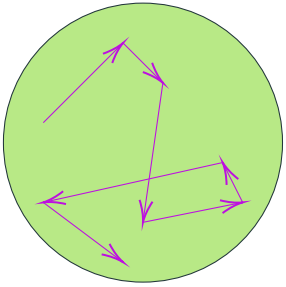


Not connected
Not mixing

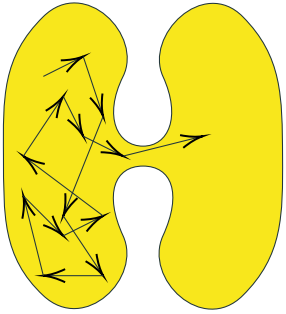
Not that bad if there is a giant component — start from a random configuration

- Simple hypergraph with $q \geq \max\{\Theta_k(\log n), \Theta_k(\Delta^{\frac{1}{k-1}})\}$ **[Frieze-Anastos'17]**

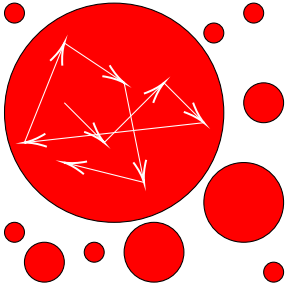
Dystopia of MCMC?



Well connected
Fast mixing



Poorly connected
Slow mixing

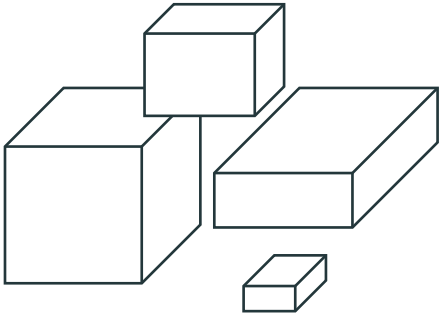


Not connected
Not mixing

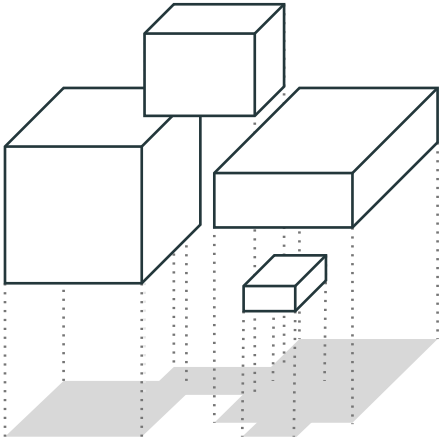
Not that bad if there is a giant component — start from a random configuration

- Simple hypergraph with $q \geq \max\{\Theta_k(\log n), \Theta_k(\Delta^{\frac{1}{k-1}})\}$ **[Frieze-Anastos'17]**
- Constant number of colours?

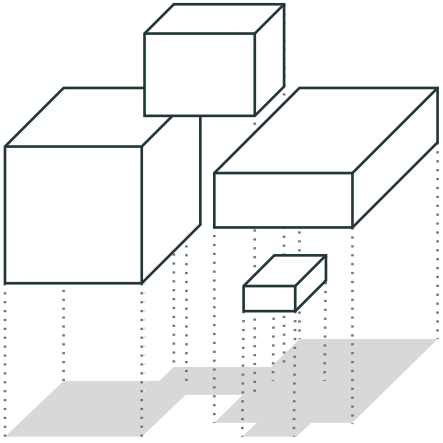
Markov chain projection [Feng-Guo-Yin-Zhang'21]



Markov chain projection [Feng-Guo-Yin-Zhang'21]

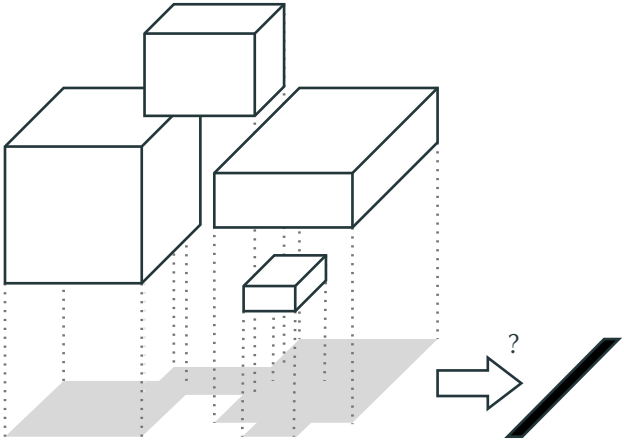


Markov chain projection [Feng-Guo-Yin-Zhang'21]



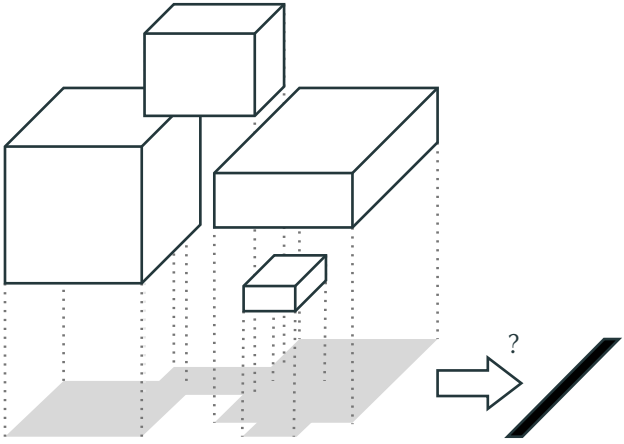
- Disconnected / poorly connected $\xrightarrow{\text{projection}}$ well connected

Markov chain projection [Feng-Guo-Yin-Zhang'21]



- Disconnected / poorly connected $\xrightarrow{\text{projection}}$ well connected

Markov chain projection [Feng-Guo-Yin-Zhang'21]



- Disconnected / poorly connected $\xrightarrow{\text{projection}}$ well connected
- Harder to simulate transition / recover a sample

The algorithm

Algorithm:

- Run Glauber dynamics on the projected distribution to get a projected sample $\boldsymbol{Y}$

The algorithm

Algorithm:

- Run Glauber dynamics on the projected distribution to get a projected sample $\boldsymbol{Y}$
- Sample a proper colouring $\boldsymbol{X}$ conditioned on its projection being $\boldsymbol{Y}$

The algorithm

Algorithm:

- Run Glauber dynamics on the projected distribution to get a projected sample $\boldsymbol{Y}$
- Sample a proper colouring $\boldsymbol{X}$ conditioned on its projection being $\boldsymbol{Y}$

Things to handle:

- Choose a proper projection

The algorithm

Algorithm:

- Run Glauber dynamics on the projected distribution to get a projected sample $\boldsymbol{Y}$
- Sample a proper colouring $\boldsymbol{X}$ conditioned on its projection being $\boldsymbol{Y}$

Things to handle:

- Choose a proper projection
- Analyse the mixing time of projected chain

The algorithm

Algorithm:

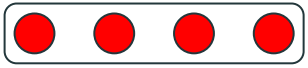
- Run Glauber dynamics on the projected distribution to get a projected sample $\boldsymbol{Y}$
- Sample a proper colouring $\boldsymbol{X}$ conditioned on its projection being $\boldsymbol{Y}$

Things to handle:

- Choose a proper projection
- Analyse the mixing time of projected chain
- Simulate transition and get the final sample

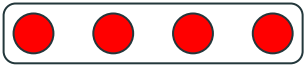
Bucketing [Feng-He-Yin'21]

- Disconnection arises from hard constraints:

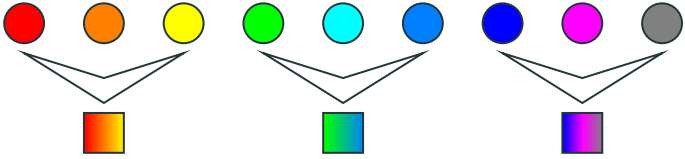


Bucketing [Feng-He-Yin'21]

- Disconnection arises from hard constraints:

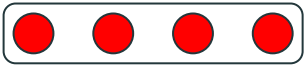


- Projection by bucketing:

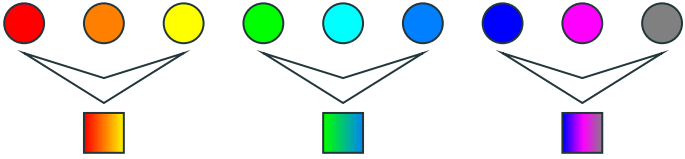


Bucketing [Feng-He-Yin'21]

- Disconnection arises from hard constraints:



- Projection by bucketing:

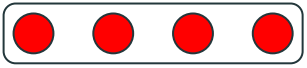


- The constraint is soft (ensured by LLL), i.e.,

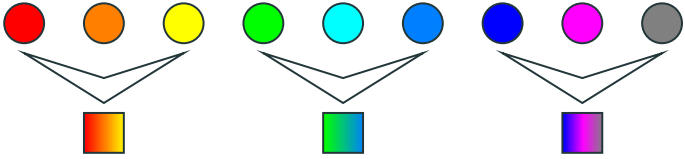
$$\Pr\left[\ \right] > 0$$

Bucketing [Feng-He-Yin'21]

- Disconnection arises from hard constraints:



- Projection by bucketing:



- The constraint is soft (ensured by LLL), i.e.,

$$\Pr\Big[\ \Big] > 0$$

despite that

$$\Pr\Big[\ \Big] < \Pr\Big[\ \Big] = \Pr\Big[\ \Big]$$

The algorithm

Things to handle:

- Choose a proper projection
  - Bucketing
- Analyse the mixing time of projected chain
- Simulate transition and get the final sample

The algorithm

Things to handle:

- Choose a proper projection
  - Bucketing
- Analyse the mixing time of projected chain
- Simulate transition and get the final sample

Current $\Delta \lesssim q^{k/3}$ barrier: trade-off between mixing and implementation.

The algorithm

Things to handle:

- Choose a proper projection
  - Bucketing
- Analyse the mixing time of projected chain
- Simulate transition and get the final sample

Current $\Delta \lesssim q^{k/3}$ barrier: trade-off between mixing and implementation.

Improve both to get $\Delta \lesssim q^{k/2}$ on simple hypergraphs.

The algorithm

Things to handle:

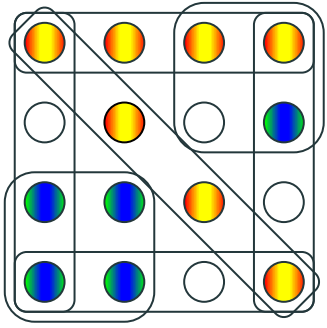
- Choose a proper projection
  - Bucketing
- Analyse the mixing time of projected chain
- Simulate transition and get the final sample

Current $\Delta \lesssim q^{k/3}$ barrier: trade-off between mixing and implementation.

Improve both to get $\Delta \lesssim q^{k/2}$ on simple hypergraphs.

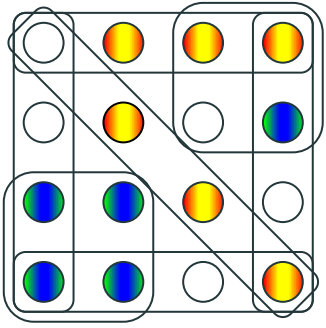
- We focus on implementation in this talk.

Fast implementation



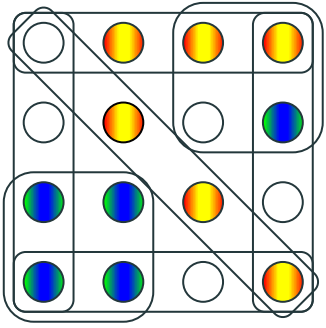
Fast implementation

- How can we know the correct projected distribution?



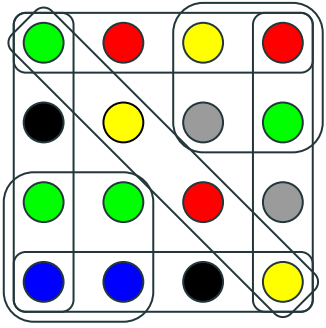
Fast implementation

- How can we know the correct projected distribution?
  - Inverse the projection independently!



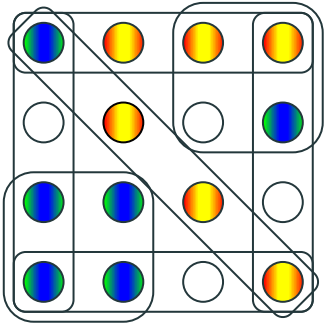
Fast implementation

- How can we know the correct projected distribution?
  - Inverse the projection independently!



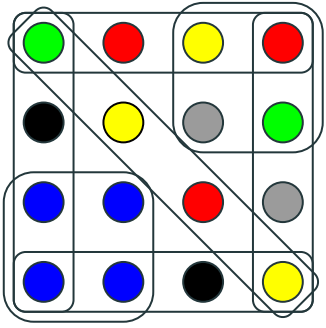
Fast implementation

- How can we know the correct projected distribution?
  - Inverse the projection independently!



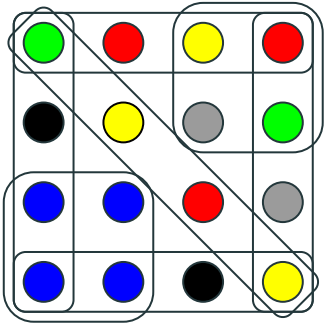
Fast implementation

- How can we know the correct projected distribution?
  - Inverse the projection independently!
- What if the inversion sample is not proper?



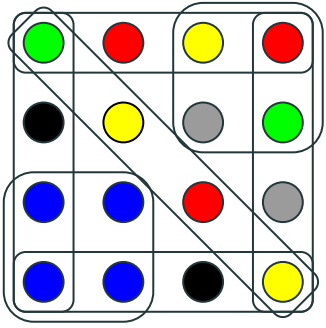
Fast implementation

- How can we know the correct projected distribution?
  - Inverse the projection independently!
- What if the inversion sample is not proper?
  - Repeat again! (rejection sampling)



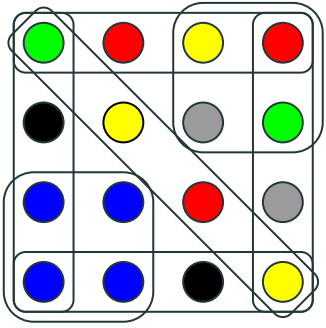
Fast implementation

- How can we know the correct projected distribution?
  - Inverse the projection independently!
- What if the inversion sample is not proper?
  - Repeat again! (rejection sampling)
- Expected number of trials?



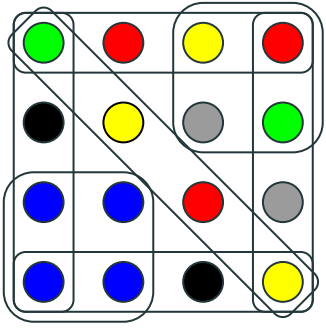
Fast implementation

- How can we know the correct projected distribution?
  - Inverse the projection independently!
- What if the inversion sample is not proper?
  - Repeat again! (rejection sampling)
- Expected number of trials?
  - Each hyperedge fails with constant probability.



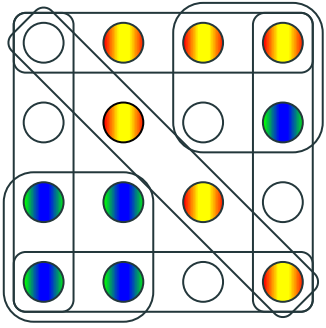
Fast implementation

- How can we know the correct projected distribution?
  - Inverse the projection independently!
- What if the inversion sample is not proper?
  - Repeat again! (rejection sampling)
- Expected number of trials?
  - Each hyperedge fails with constant probability.
  - Expect $c^{\Theta(n)}$ rounds of resampling!



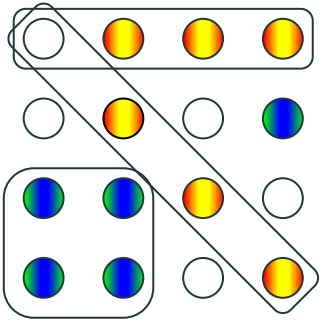
Fast implementation

- How can we know the correct projected distribution?
  - Inverse the projection independently!
- What if the inversion sample is not proper?
  - Repeat again! (rejection sampling)
- Expected number of trials?
  - Each hyperedge fails with constant probability.
  - Expect $c^{\Theta(n)}$ rounds of resampling!
- Satisfied (by bucketing) hyperedges affect nothing.



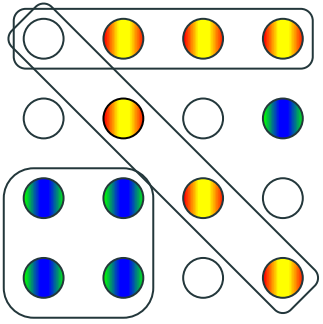
Fast implementation

- How can we know the correct projected distribution?
  - Inverse the projection independently!
- What if the inversion sample is not proper?
  - Repeat again! (rejection sampling)
- Expected number of trials?
  - Each hyperedge fails with constant probability.
  - Expect $c^{\Theta(n)}$ rounds of resampling!
- Satisfied (by bucketing) hyperedges affect nothing.



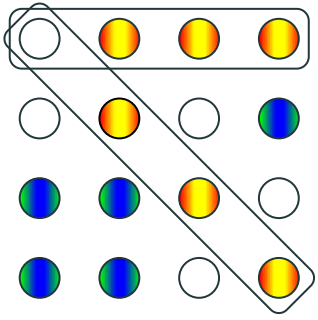
Fast implementation

- How can we know the correct projected distribution?
  - Inverse the projection independently!
- What if the inversion sample is not proper?
  - Repeat again! (rejection sampling)
- Expected number of trials?
  - Each hyperedge fails with constant probability.
  - Expect $c^{\Theta(n)}$ rounds of resampling!
- Satisfied (by bucketing) hyperedges affect nothing.
- Disconnected components affect nothing.



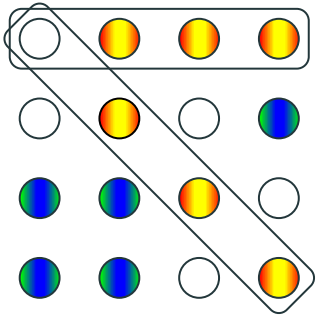
Fast implementation

- How can we know the correct projected distribution?
  - Inverse the projection independently!
- What if the inversion sample is not proper?
  - Repeat again! (rejection sampling)
- Expected number of trials?
  - Each hyperedge fails with constant probability.
  - Expect $c^{\Theta(n)}$ rounds of resampling!
- Satisfied (by bucketing) hyperedges affect nothing.
- Disconnected components affect nothing.



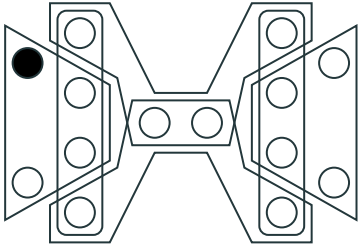
Fast implementation

- How can we know the correct projected distribution?
  - Inverse the projection independently!
- What if the inversion sample is not proper?
  - Repeat again! (rejection sampling)
- Expected number of trials?
  - Each hyperedge fails with constant probability.
  - Expect $c^{\Theta(n)}$ rounds of resampling!
- Satisfied (by bucketing) hyperedges affect nothing.
- Disconnected components affect nothing.
- We are done if number of hyperedges is $O(\log n)$ w.h.p.



Connected component

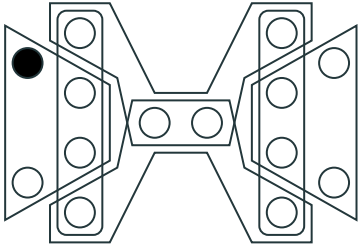
Union bound over all possible size-α(#edges) components:



Connected component

Union bound over all possible size-α(#edges) components:

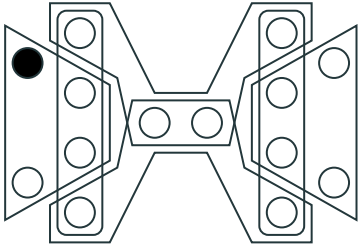
- Probability that a size-α component fails?



Connected component

Union bound over all possible size-α(#edges) components:

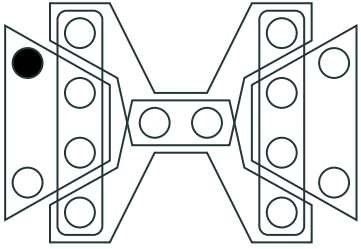
- Probability that a size-α component fails?
  - Impossible to argue exactly.



Connected component

Union bound over all possible size-α(#edges) components:

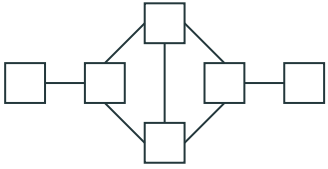
- Probability that a size-α component fails?
  - Impossible to argue exactly.
- Independent hyperedges $\rightarrow$ probability upper bound



Connected component

Union bound over all possible size-α(#edges) components:

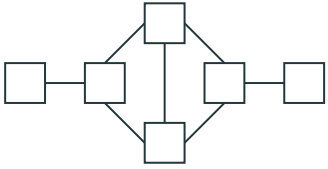
- Probability that a size-α component fails?
  - Impossible to argue exactly.
- Independent hyperedges $\rightarrow$ probability upper bound
- Working on line graph L:



Connected component

Union bound over all possible size-α(#edges) components:

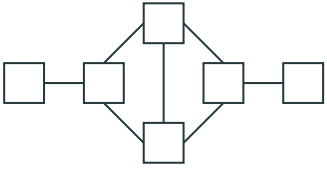
- Probability that a size-α component fails?
  - Impossible to argue exactly.
- Independent hyperedges $\rightarrow$ probability upper bound
- Working on line graph L:
- 2-tree **[Alon'91]** T of L:



Connected component

Union bound over all possible size-α(#edges) components:

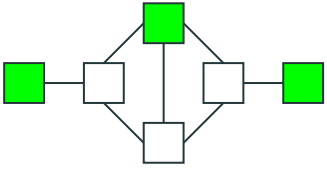
- Probability that a size-α component fails?
  - Impossible to argue exactly.
- Independent hyperedges $\rightarrow$ probability upper bound
- Working on line graph L:
- 2-tree **[Alon'91]** T of L:
  - Independent set
  - Connected on L^2



Connected component

Union bound over all possible size-α(#edges) components:

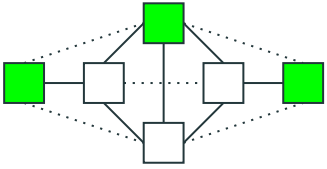
- Probability that a size-α component fails?
  - Impossible to argue exactly.
- Independent hyperedges $\rightarrow$ probability upper bound
- Working on line graph L:
- 2-tree **[Alon'91]** T of L:
  - Independent set
  - Connected on L^2



Connected component

Union bound over all possible size-α(#edges) components:

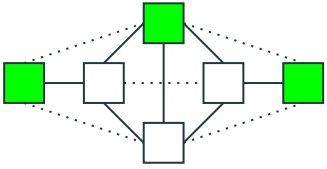
- Probability that a size-α component fails?
  - Impossible to argue exactly.
- Independent hyperedges $\rightarrow$ probability upper bound
- Working on line graph L:
- 2-tree **[Alon'91]** T of L:
  - Independent set
  - Connected on L^2



Connected component

Union bound over all possible size-α(#edges) components:

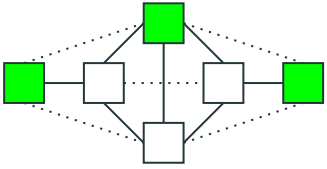
- Probability that a size-α component fails?
  - Impossible to argue exactly.
- Independent hyperedges $\rightarrow$ probability upper bound
- Working on line graph L:
- 2-tree **[Alon'91]** T of L:
  - Independent set
  - Connected on L^2
- Any size-α component has a size-$\alpha/(k\Delta)$ 2-tree.



Connected component

Union bound over all possible size-α(#edges) components:

- Probability that a size-α component fails?
  - Impossible to argue exactly.
- Independent hyperedges $\rightarrow$ probability upper bound
- Working on line graph L:
- 2-tree **[Alon'91]** T of L:
  - Independent set
  - Connected on L^2
- Any size-α component has a size-$\alpha/(k\Delta)$ 2-tree.
- Union bound over all 2-trees instead.



Do 2-trees suffice?

Assuming bucketing into $\sqrt{q}$ buckets.

$$\sum_{\ell} \Pr[\text{size-}\ell \text{ 2-tree exists}] < 1.$$

Do 2-trees suffice?

Assuming bucketing into $\sqrt{q}$ buckets.

$$\Pr[\text{size-}\ell \text{ 2-tree exists}] \lesssim 2^{-\ell}$$

Do 2-trees suffice?

Assuming bucketing into $\sqrt{q}$ buckets.

- Union bound over all 2-trees.

$$\text{Number of 2-trees} \times \Pr[\text{A size-}\ell\text{ 2-tree survives}] \lesssim 2^{-\ell}$$

Do 2-trees suffice?

Assuming bucketing into $\sqrt{q}$ buckets.

- Union bound over all 2-trees.
- Local uniformity (ensured by LLL).

$$\text{Number of 2-trees} \times (\sqrt{q})^{(1-k)\ell} \lesssim 2^{-\ell}$$

Do 2-trees suffice?

Assuming bucketing into $\sqrt{q}$ buckets.

- Union bound over all 2-trees.
- Local uniformity (ensured by LLL).
- 2-tree counting argument:

Lemma (Corollary of [Borgs-Chayes-Kahn-Lovász'13])

Let G be a graph with maximum degree D and v is a vertex. Then the number of 2-trees in G of size ℓ containing v is at most $(eD^2)^{\ell-1}/2$.

$$(e(k\Delta)^2)^{\ell-1} \times (\sqrt{q})^{(1-k)\ell} \lesssim 2^{-\ell}$$

Do 2-trees suffice?

Assuming bucketing into $\sqrt{q}$ buckets.

- Union bound over all 2-trees.
- Local uniformity (ensured by LLL).
- 2-tree counting argument:

Lemma (Corollary of [Borgs-Chayes-Kahn-Lovász'13])

Let G be a graph with maximum degree D and v is a vertex. Then the number of 2-trees in G of size ℓ containing v is at most $(eD^2)^{\ell-1}/2$.

$$\Delta \lesssim q^{k/4}$$

Do 2-trees suffice?

Assuming bucketing into $\sqrt{q}$ buckets.

- Union bound over all 2-trees.
- Local uniformity (ensured by LLL).
- 2-tree counting argument:

Lemma (Corollary of [Borgs-Chayes-Kahn-Lovász'13])

Let G be a graph with maximum degree D and v is a vertex. Then the number of 2-trees in G of size ℓ containing v is at most $(eD^2)^{\ell-1}/2$.

$$\Delta \lesssim q^{k/4}$$

Best we can do using 2-trees: $\Delta \lesssim q^{k/3}$ [Jain-Pham-Vuong'21].

Do 2-trees suffice?

Assuming bucketing into $\sqrt{q}$ buckets.

- Union bound over all 2-trees.
- Local uniformity (ensured by LLL).
- 2-tree counting argument:

Lemma (Corollary of [Borgs-Chayes-Kahn-Lovász'13])

Let G be a graph with maximum degree D and v is a vertex. Then the number of 2-trees in G of size ℓ containing v is at most $(eD^2)^{\ell-1}/2$.

$$\Delta \lesssim q^{k/4}$$

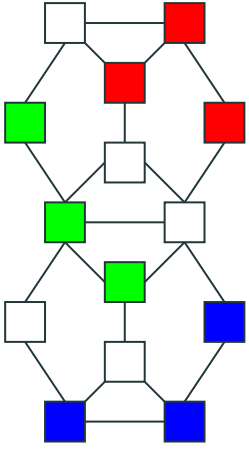
Best we can do using 2-trees: $\Delta \lesssim q^{k/3}$ [Jain-Pham-Vuong'21].

- $q^{2/3}$ buckets and trade-off with mixing.

2-block-trees

Idea: utilising small overlaps!

- Single vertex in 2-tree $\rightarrow$ size-θ component (block)

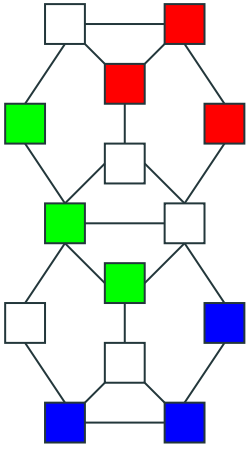


2-block-trees

Idea: utilising small overlaps!

- Single vertex in 2-tree $\rightarrow$ size-θ component (block)
- Probability of each block:

$$\approx (\sqrt{q})^{-\theta(k-\theta)}.$$



2-block-trees

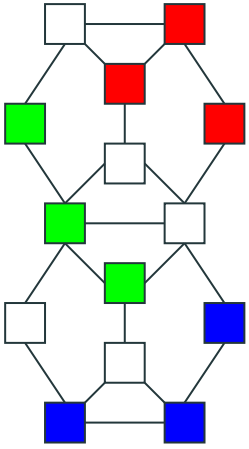
Idea: utilising small overlaps!

- Single vertex in 2-tree $\to$ size-θ component (block)
- Probability of each block:

$$\approx (\sqrt{q})^{-\theta(k-\theta)}.$$

- Number of 2-block-trees:

$$\approx (\theta \mathrm{e}^{\theta} D^{\theta+1})^{\ell}.$$



2-block-trees

Idea: utilising small overlaps!

- Single vertex in 2-tree $\rightarrow$ size-θ component (block)
- Probability of each block:

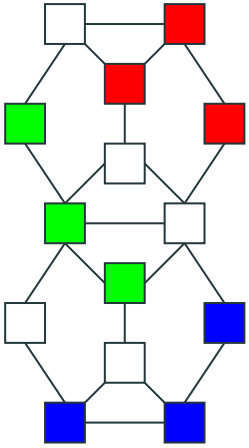
$$\approx (\sqrt{q})^{-\theta(k-\theta)}.$$

- Number of 2-block-trees:

$$\approx (\theta \mathrm{e}^{\theta} D^{\theta+1})^{\ell}.$$

Comparing with 2-trees ($\theta = 1$):

$$\approx (\mathrm{e}D^2)^{\ell}.$$



2-block-trees

Idea: utilising small overlaps!

- Single vertex in 2-tree $\to$ size-θ component (block)
- Probability of each block:

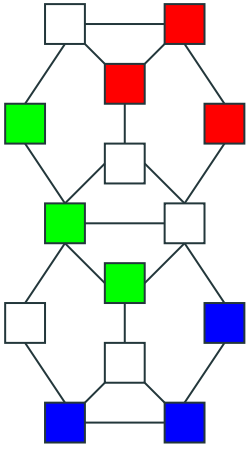
$$\approx (\sqrt{q})^{-\theta(k-\theta)}.$$

- Number of 2-block-trees:

$$\approx (\theta \mathrm{e}^{\theta} D^{\theta+1})^{\ell}.$$

Comparing with 2-trees ($\theta = 1$):

$$\approx (\mathrm{e}D^2)^{\ell}.$$



Requires:

$$\Delta \lesssim q^{\frac{k}{2+O(1/\theta)}}$$

Future directions

Establish computational threshold for sampling hypergraph colourings.

Future directions

Establish computational threshold for sampling hypergraph colourings.

- Overcoming disconnectivity issue:
  - Block dynamics, instead of updating only one vertex? **[Chen-Liu-Vigoda'21]**

Future directions

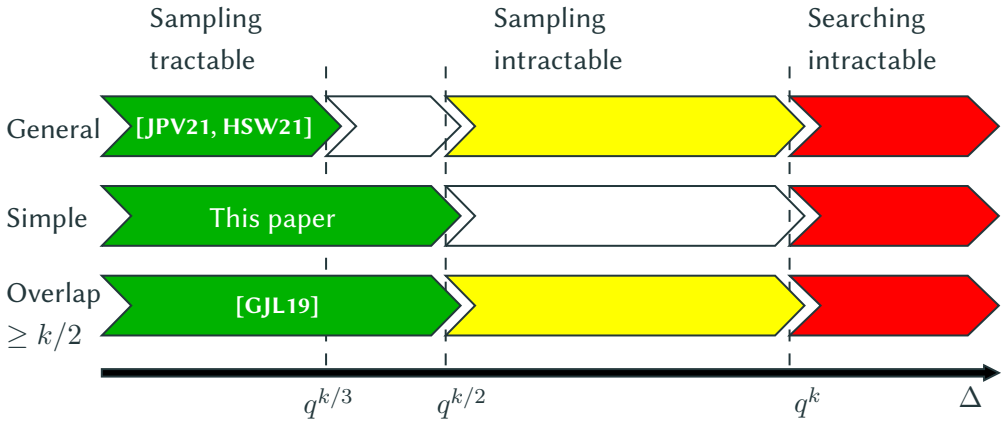
Establish computational threshold for sampling hypergraph colourings.

- Overcoming disconnectivity issue:
  - Block dynamics, instead of updating only one vertex? **[Chen-Liu-Vigoda'21]**
- New methods: Recursive sampler **[Anand-Jerrum'22]**?
  - Applications under LLL setting **[He-Wang-Yin'22, He-Wu-Yang'22]**.
  - Better condition?

Future directions

Establish computational threshold for sampling hypergraph colourings.

- Overcoming disconnectivity issue:
  - Block dynamics, instead of updating only one vertex? **[Chen-Liu-Vigoda'21]**
- New methods: Recursive sampler **[Anand-Jerrum'22]**?
  - Applications under LLL setting **[He-Wang-Yin'22, He-Wu-Yang'22]**.
  - Better condition?
- Utilising overlap information?
  - Partial rejection sampling **[Guo-Jerrum-Liu'19]** gives transition at $\Delta \approx q^{k/2}$ when overlaps are large.


Sampling tractable
Sampling intractable
Searching intractable
General
[JPV21, HSW21]
Simple
This paper
Overlap $\geq k/2$
[GJL19]
$q^{k/3}$
$q^{k/2}$
q^k
Δ

Thank you!

arXiv: 2202.05554