A simple algorithm for sampling colourings of $G(n, d/n)$ up to Gibbs Uniqueness Threshold

Charis Efthymiou
efthymiou@gmail.com

Georgia Tech

Graphical Models, Statistical Inference, and Algorithms
IMA - University of Minnesota
Random colouring of a graph \( G \)

**Random Colouring Problem**

**input:** \( G = (V, E) \) and some integer \( k \)

**output:** A uniformly random \( k \)-colourings of \( G \).

**Gibbs Distribution**

**Remark**

The focus is on approximate random colouring algorithms

MCMC approach to the problem

random walk over the set of all \( k \)-colourings

the stationary distribution is the uniform one

show that the walk mixes in polynomial time

for general \( G \), we have polynomial mixing for any \( k > 11.6 \Delta \)

[Vigoda:'99]
Random colouring of a graph $G$

**Random Colouring Problem**

- **input:** $G = (V, E)$ and some integer $k$
- **output:** A *uniformly random* $k$-colourings of $G$. 

**Remark**
The focus is on approximate random colouring algorithms. The MCMC approach to the problem is based on a random walk over the set of all $k$-colourings. The stationary distribution is the uniform one. It has been shown that the walk mixes in polynomial time for general $G$, and for any $k > 11\Delta$ [Vigoda:'99].
Random colouring of a graph $G$

Random Colouring Problem

- input: $G = (V, E)$ and some integer $k$
- output: A uniformly random $k$-colourings of $G$. *Gibbs Distribution*

Remark

The focus is on approximate random colouring algorithms.

MCMC approach to the problem:

- random walk over the set of all $k$-colourings
- the stationary distribution is the uniform one
- show that the walk mixes in polynomial time for general $G$.

We have polynomial mixing for any $k > 11.6$ \([\text{Vigoda:'99}]\).
Random colouring of a graph $G$

Random Colouring Problem

- input: $G = (V, E)$ and some integer $k$
- output: A uniformly random $k$-colourings of $G$. Gibbs Distribution

Remark

The focus is on approximate random colouring algorithms
Random colouring of a graph $G$

**Random Colouring Problem**
- input: $G = (V, E)$ and some integer $k$
- output: A uniformly random $k$-colourings of $G$. *Gibbs Distribution*

**Remark**
The focus is on approximate random colouring algorithms

**MCMC approach to the problem**
Random colouring of a graph $G$

### Random Colouring Problem
- **input:** $G = (V, E)$ and some integer $k$
- **output:** A *uniformly random* $k$-colourings of $G$. *Gibbs Distribution*

### Remark
The focus is on *approximate* random colouring algorithms

### MCMC approach to the problem
- *random walk* over the set of all $k$-colourings
Random colouring of a graph \( G \)

**Random Colouring Problem**

- **input:** \( G = (V, E) \) and some integer \( k \)
- **output:** A uniformly random \( k \)-colourings of \( G \). \textit{Gibbs Distribution}

**Remark**

The focus is on approximate random colouring algorithms

**MCMC approach to the problem**

- **random walk** over the set of all \( k \)-colourings
- the **stationary distribution** is the uniform one
Random colouring of a graph $G$

Random Colouring Problem

- input: $G = (V, E)$ and some integer $k$
- output: A uniformly random $k$-colourings of $G$. Gibbs Distribution

Remark

The focus is on approximate random colouring algorithms

MCMC approach to the problem

- random walk over the set of all $k$-colourings
- the stationary distribution is the uniform one
- show that the walk mixes in polynomial time
Random colouring of a graph $G$

**Random Colouring Problem**
- input: $G = (V, E)$ and some integer $k$
- output: A uniformly random $k$-colourings of $G$. *Gibbs Distribution*

**Remark**
The focus is on approximate random colouring algorithms

**MCMC approach to the problem**
- random walk over the set of all $k$-colourings
- the stationary distribution is the uniform one
- show that the walk mixes in polynomial time
- for general $G$, we have polynomial mixing for any $k > \frac{11}{6} \Delta$
  
  [Vigoda:’99]
Average case scenario

Random graph $G(n, d/n)$ graph on $n$ vertices and each edge appears independently with probability $d/n$, where $d$ is fixed.

Why the problem is interesting...

Typically, the bounds on $k$ are expressed in terms of maximum degree in $G(n, d/n)$, the degrees fluctuate significantly...

Typically, the maximum degree is $\Theta(\log n \log \log n)$.

Typically, the "vast majority" of vertices are of degree in $(1 \pm c) d$.

The bounds on $k$ are expressed in terms of the expected degree.
Average case scenario

Random graph $G(n, d/n)$

- graph on $n$ vertices and each edge appears independently with probability $d/n$, where $d$ is fixed
Average case scenario

Random graph $G(n, d/n)$
- graph on $n$ vertices and each edge appears independently with probability $d/n$, where $d$ is fixed

Why the problem is interesting...

Average case scenario

Random graph $G(n, d/n)$
- graph on $n$ vertices and each edge appears independently with probability $d/n$, where $d$ is fixed

Why the problem is interesting...
- typically the bounds on $k$ are expressed in terms of maximum degree
Average case scenario

Random graph $G(n, d/n)$
- graph on $n$ vertices and each edge appears independently with probability $d/n$, where $d$ is fixed

Why the problem is interesting...
- typically the bounds on $k$ are expressed in terms of maximum degree
- in $G(n, d/n)$ the degrees fluctuate significantly...
Average case scenario

Random graph $G(n, d/n)$
- graph on $n$ vertices and each edge appears independently with probability $d/n$, where $d$ is fixed

Why the problem is interesting...
- typically the bounds on $k$ are expressed in terms of maximum degree
- in $G(n, d/n)$ the degrees fluctuate significantly...
  - typically, the maximum degree is $\Theta \left( \frac{\log n}{\log \log n} \right)$
Average case scenario

Random graph $G(n, d/n)$
- graph on $n$ vertices and each edge appears independently with probability $d/n$, where $d$ is fixed

Why the problem is interesting...
- typically the bounds on $k$ are expressed in terms of maximum degree
- in $G(n, d/n)$ the degrees fluctuate significantly...
  - typically, the maximum degree is $\Theta \left( \frac{\log n}{\log \log n} \right)$
  - typically, the “vast majority” of vertices are of degree in $(1 \pm c)d$
Average case scenario

Random graph $G(n, d/n)$
- graph on $n$ vertices and each edge appears independently with probability $d/n$, where $d$ is fixed

Why the problem is interesting...
- typically the bounds on $k$ are expressed in terms of maximum degree
- in $G(n, d/n)$ the degrees fluctuate significantly...
  - typically, the maximum degree is $\Theta\left(\frac{\log n}{\log \log n}\right)$
  - typically, the “vast majority” of vertices are of degree in $(1 \pm c)d$
- the bounds on $k$ are expressed in terms of the expected degree $d$
An overview

The algorithm it is NOT...

- Markov Chain Monte Carlo, e.g. Glauber, Metropolis dynamics
- Heuristic from statistical physics, e.g. Belief Propagation
- Weitz-sampling algorithm

Simple conceptually

Analysis best guaranteed performance in terms of $k$

Sacrifice accuracy the output error depends only on the input $G$

Does not depend on the execution time
An overview

The algorithm

- It is NOT...
  - Markov Chain Monte Carlo, e.g. Glauber, Metropolis dynamics
  - Heuristic from statistical physics, e.g. Belief Propagation
  - Weitz-sampling algorithm

- Simple conceptually
- Analysis
- Best guaranteed performance in terms of $k$
- Sacrifice accuracy of the output error depends only on the input $G$
- Does not depend on the execution time
An overview

The algorithm

- it is NOT...
An overview

The algorithm

- it is NOT...
  - Markov Chain Monte Carlo, e.g. Glauber, Metropolis dynamics
An overview

The algorithm

- it is NOT...
  - Markov Chain Monte Carlo, e.g. Glauber, Metropolis dynamics
  - heuristic from statistical physics, e.g. Belief Propagation
An overview

The algorithm

- it is NOT...
  - Markov Chain Monte Carlo, e.g. Glauber, Metropolis dynamics
  - heuristic from statistical physics, e.g. Belief Propagation
  - Weitz-sampling algorithm
An overview

The algorithm

- it is NOT...
  - Markov Chain Monte Carlo, e.g. Glauber, Metropolis dynamics
  - heuristic from statistical physics, e.g. Belief Propagation
  - Weitz-sampling algorithm
- simple
An overview

The algorithm

- it is NOT...
  - Markov Chain Monte Carlo, e.g. Glauber, Metropolis dynamics
  - heuristic from statistical physics, e.g. Belief Propagation
  - Weitz-sampling algorithm
- simple
  - conceptually

Analysis best guaranteed performance in terms of $k$ sacrifice accuracy the output error depends only on the input $G$ and $k$ does not depend on the execution time
An overview

The algorithm

- it is NOT...
  - Markov Chain Monte Carlo, e.g. Glauber, Metropolis dynamics
  - heuristic from statistical physics, e.g. Belief Propagation
  - Weitz-sampling algorithm
- simple
  - conceptually
  - analysis
An overview

The algorithm

- it is NOT...
  - Markov Chain Monte Carlo, e.g. Glauber, Metropolis dynamics
  - heuristic from statistical physics, e.g. Belief Propagation
  - Weitz-sampling algorithm
- simple
  - conceptually
  - analysis
- best guaranteed performance in terms of $k$
An overview

The algorithm

- it is NOT...
  - Markov Chain Monte Carlo, e.g. Glauber, Metropolis dynamics
  - heuristic from statistical physics, e.g. Belief Propagation
  - Weitz-sampling algorithm

- simple
  - conceptually
  - analysis

- best guaranteed performance in terms of $k$

- sacrifice accuracy
An overview

The algorithm

- it is NOT...
  - Markov Chain Monte Carlo, e.g. Glauber, Metropolis dynamics
  - heuristic from statistical physics, e.g. Belief Propagation
  - Weitz-sampling algorithm
- simple
  - conceptually
  - analysis
- best guaranteed performance in terms of $k$
- sacrifice accuracy
  - the output error depends only on the input $G$ and $k$
The algorithm

- it is NOT...
  - Markov Chain Monte Carlo, e.g. Glauber, Metropolis dynamics
  - heuristic from statistical physics, e.g. Belief Propagation
  - Weitz-sampling algorithm

- simple
  - conceptually
  - analysis

- best guaranteed performance in terms of $k$

- sacrifice accuracy
  - the output error depends only on the input $G$ and $k$
  - ... does not depend on the execution time
Measure of comparison

There is a MCMC sampling \( k \)-colouring algorithm which has polynomial mixing for typical instances of \( G(n, d/n) \), for any \( k \geq 11^{2d} \). [Efthymiou:'14]

"Weitz-sampling" There is a FPAUS sampling \( k \)-colouring algorithm for typical instances of \( G(n, d/n) \), for any \( k > 3d \). [Yin, Zhang:'15]
MCMC sampling

There is a MCMC sampling $k$-colouring algorithm which has polynomial mixing for typical instances of $G(n, d/n)$, for any $k \geq \frac{11}{2} d$. [Efthymiou:'14]
Measure of comparison

**MCMC sampling**
There is a MCMC sampling $k$-colouring algorithm which has polynomial mixing for typical instances of $G(n, d/n)$, for any $k \geq \frac{11}{2} d$. [Efthymiou:'14]

**“Weitz-sampling”**
There is a FPAUS sampling $k$-colouring algorithm for typical instances of $G(n, d/n)$, for any $k > 3d$. [Yin, Zhang:'15]
A simple observation

\[ G \]

\[ u \quad v \]
A simple observation

Random colouring $G(n, d/n)$
A simple observation

A random colouring of $G$ can be seen as a random colouring of the simpler $G'$ conditional that $v, u$ receive different colours.
A simple observation

A random colouring of $G$ can be seen as a random colouring of the simpler $G'$ conditional that $v, u$ receive different colours.
A random colouring of $G$ can be seen as a random colouring of the simpler $G'$ conditional that $v, u$ receive different colours.
Suppose that ....

**UPDATE**

**input:** random $k$-colouring of $G$ and the vertices $v, u$.

**output:** random $k$-colouring of $G$, conditional $u, v$ are assigned different colours.

Be careful...

We cannot change the colours of the vertices arbitrarily.
Suppose that ....

**UPDATE**

input: random $k$-colouring of $G$ and the vertices $v, u$.

output: random $k$-colouring of $G$, conditional $u, v$ are assigned different colours.

Be careful...

We cannot change the colours of the vertices arbitrarily.

C.Efthymiou (GaTech)
Suppose that ....

**UPDATE**

**input:** random $k$-colouring of $G$ and the vertices $v, u$.

**output:** random $k$-colouring of $G$, conditional $u, v$ are assigned different colours.

Be careful...

We cannot change the colours of the vertices arbitrarily.

C.Efthymiou (GaTech)
Suppose that ....

**UPDATE**

**input:** random $k$-colouring of $G$ and the vertices $v, u$.

**output:** random $k$-colouring of $G$, conditional $u, v$ are assigned different colours.

Be careful...

We can not change the colours of the vertices arbitrarily.
Use UPDATE for sampling colourings ...

The algorithm

- Create $G_0, G_1, \ldots, G_r = G$ such that
- $G_i$ is obtained from $G_{i+1}$ by deleting some edge $\{v_i, u_i\}$
- $G_0$ is very "simple" colouring
- Randomly colour $G_0$ for $i = 0, \ldots, r-1$
- Apply UPDATE to the colouring of $G_i$ and get that of $G_{i+1}$

Output: The colouring of $G_r$
Use **UPDATE** for sampling colourings ...

The algorithm

**Input:** $G = (V, E) k$

Create $G_0, G_1, \ldots, G_r = G$ such that get $G_i$ from $G_{i+1}$ by deleting some edge $\{v_i, u_i\}$.

$G_0$ is very "simple"
color randomly $G_0$ for $i = 0, \ldots, r-1$
apply **UPDATE** to the colouring of $G_i$ and get that of $G_{i+1}$

**Output:** The colouring of $G_r$
Use UPDATE for sampling colourings ...

The algorithm

**Input:** $G = (V, E)$

create $G_0, G_1, \ldots, G_r = G$ such that

get $G_i$ from $G_{i+1}$ by deleting some edge $\{v_i, u_i\}$

$G_0$ is very "simple"

color randomly $G_0$ for $i = 0, \ldots, r - 1$

apply UPDATE to the colouring of $G_i$ and get that of $G_{i+1}$

**Output:** The colouring of $G_r$
Use \texttt{UPDATE} for sampling colourings ... 

The algorithm

\textbf{Input}: $G = (V, E)$ \textit{k}

create $G_0, G_1, \ldots, G_r = G$ such that

get $G_i$ from $G_{i+1}$ by deleting some edge $\{v_i, u_i\}$
The algorithm

**Input:** $G = (V, E)$

create $G_0, G_1, \ldots, G_r = G$ such that

- get $G_i$ from $G_{i+1}$ by deleting some edge $\{v_i, u_i\}$
- $G_0$ is very “simple”
Use **UPDATE** for sampling colourings ...

---

### The algorithm

**Input:** $G = (V, E) \ k$

- create $G_0, G_1, \ldots, G_r = G$ such that
  - get $G_i$ from $G_{i+1}$ by deleting some edge $\{v_i, u_i\}$
  - $G_0$ is very "simple"

- color randomly $G_0$
Use \textsc{UPDATE} for sampling colourings ... 

The algorithm

\textbf{Input:} \( G = (V, E) \) \( k \)

- create \( G_0, G_1, \ldots, G_r = G \) such that
  - get \( G_i \) from \( G_{i+1} \) by deleting some edge \( \{v_i, u_i\} \)
  - \( G_0 \) is very "simple"

- color randomly \( G_0 \)

- for \( i = 0, \ldots, r - 1 \)
Use UPDATE for sampling colourings ...

The algorithm

**Input:** $G = (V, E) \ k$

create $G_0, G_1, \ldots, G_r = G$ such that

get $G_i$ from $G_{i+1}$ by deleting some edge \{v_i, u_i\}

$G_0$ is very “simple”

color randomly $G_0$

for $i = 0, \ldots, r - 1$

apply UPDATE to the colouring of $G_i$ and get that of $G_{i+1}$
The algorithm

**Input:** $G = (V, E) \ k$

create $G_0, G_1, \ldots, G_r = G$ such that

get $G_i$ from $G_{i+1}$ by deleting some edge $\{v_i, u_i\}$

$G_0$ is very "simple"

color randomly $G_0$

for $i = 0, \ldots, r - 1$

apply UPDATE to the colouring of $G_i$ and get that of $G_{i+1}$
Use UPDATE for sampling colourings ...

The algorithm

**Input:** $G = (V, E) \ k$

- create $G_0, G_1, \ldots, G_r = G$ such that
  - get $G_i$ from $G_{i+1}$ by deleting some edge $\{v_i, u_i\}$
  - $G_0$ is very "simple"
- color randomly $G_0$
- for $i = 0, \ldots, r - 1$
  - apply UPDATE to the colouring of $G_i$ and get that of $G_{i+1}$
Use UPDATE for sampling colourings ...

The algorithm

**Input:** $G = (V, E) \ k$

- create $G_0, G_1, \ldots, G_r = G$ such that
  - get $G_i$ from $G_{i+1}$ by deleting some edge $\{v_i, u_i\}$
  - $G_0$ is very "simple"

- color randomly $G_0$

- for $i = 0, \ldots, r - 1$
  - apply UPDATE to the colouring of $G_i$ and get that of $G_{i+1}$

UPDATE
Use UPDATE for sampling colourings ...

The algorithm

**Input:** $G = (V, E)$ $k$

create $G_0, G_1, \ldots, G_r = G$ such that

get $G_i$ from $G_{i+1}$ by deleting some edge $\{v_i, u_i\}$

$G_0$ is very "simple"

color randomly $G_0$

for $i = 0, \ldots, r - 1$

apply UPDATE to the colouring of $G_i$ and get that of $G_{i+1}$
Use UPDATE for sampling colourings ...

The algorithm

**Input:** $G = (V, E) k$

create $G_0, G_1, \ldots, G_r = G$ such that

get $G_i$ from $G_{i+1}$ by deleting some edge $\{v_i, u_i\}$

$G_0$ is very “simple”

color randomly $G_0$

for $i = 0, \ldots, r - 1$

apply UPDATE to the colouring of $G_i$ and get that of $G_{i+1}$

**Output:** The colouring of $G_r$
How does UPDATE look like for $G(n, d/n)$?
How does UPDATE look like for $G(n, d/n)$

**UPDATE**

- **input**: $G_i, \sigma, v_i, u_i$
- **if** $\sigma(v_i) \neq \sigma(u_i)$, **then** return $\sigma$
- **Otherwise**
  - $q$ is chosen u.a.r. from $[k] \setminus \{\sigma_v\}$
  - **return** the $q$-switching of $\sigma$
How does UPDATE look like for $G(n, d/n)$

**UPDATE**

- **input:** $G_i$, $\sigma$, $v_i$ $u_i$
- **if** $\sigma(v_i) \neq \sigma(u_i)$, **then** return $\sigma$
- **Otherwise**
  - $q$ is chosen u.a.r. from $[k] \backslash \{\sigma_v\}$
  - **return** the $q$-switching of $\sigma$

**Diagram:**

A graph $G$ with nodes $v_i$ and $u_i$ connected by edges. The nodes are color-coded to represent the coloring process.
How does UPDATE look like for $G(n, d/n)$

**UPDATE**

- **input:** $G_i, \sigma, v_i, u_i$
- **if** $\sigma(v_i) \neq \sigma(u_i)$, **then** return $\sigma$
- **Otherwise**
  - $q$ is chosen u.a.r. from $[k] \setminus \{\sigma_v\}$
  - **return** the $q$-switching of $\sigma$

![Diagram of G(n, d/n)]
How does \textbf{UPDATE} look like for \( G(n, d/n) \)

\textbf{UPDATE}

- **input:** \( G_i, \sigma, v_i, u_i \)
- **if** \( \sigma(v_i) \neq \sigma(u_i) \), **then** return \( \sigma \)
- **Otherwise**
  - \( q \) is chosen u.a.r. from \( [k] \setminus \{\sigma_v\} \)
  - **return** the \( q \)-switching of \( \sigma \)
... there is no panacea
... there is no panacea
... there is no panacea
... there is no panacea
... there is no panacea

Pathological Colouring

Every $k$-colouring which specifies a 2-coloured path between $v_i$ and $u_i$
... there is no panacea

Pathological Colouring

Every $k$-colouring which specifies a 2-coloured path between $v_i$ and $u_i$

Remark

The existence of pathological colourings makes UPDATE an approximation algorithm
Pathological Colouring

Every $k$-colouring which specifies a 2-coloured path between $v_i$ and $u_i$

Consequently...

The random colouring is an approximation one algorithm
... there is no panacea

Pathological Colouring

Every $k$-colouring which specifies a 2-coloured path between $v_i$ and $u_i$

Remark

The algorithm turns out to be accurate because the pathological colouring are relatively rare for the range of $k$ we consider
Result - The algorithm

Input:
- \( G(n, d/n) \)
- \( k \)

Create \( G_0, G_1, \ldots, G_r \) such that:
- \( G_i := \text{delete u.a.r. an edge } \{v_i, u_i\} \) of \( G_{i+1} \) which does not belong to a cycle of length \( \log n < 10 \log d \).
- Colour randomly \( G_0 \)

For \( i = 0, \ldots, r-1 \), apply \text{UPDATE} to the colouring of \( G_i \) to get that of \( G_{i+1} \).

Output: the colouring of \( G_r \)

Remark: Typically, each component of \( G_0 \) is either trivial or an isolated cycle.
The algorithm

\[ G(n, \frac{d}{n}) \]

\[ \text{create } G_0, G_1, \ldots, G_r \text{ such that } \]

\[ G_i \leftarrow \text{delete u.a.r. an edge } \{v_i, u_i\} \text{ of } G_{i+1} \text{ which does not belong to a cycle of length } < \log n + 10 \log d. \]

\[ \text{colour randomly } G_0 \text{ for } i = 0, \ldots, r-1 \text{ apply UPDATE to the colouring of } G_i \text{ to get that of } G_{i+1}. \]

Output: the colouring of \( G_r \)

Remark

Typically, each component of \( G_0 \) is either trivial or an isolated cycle.
The algorithm

**input:** $G(n, d/n), k$

- Create $G_0, G_1, ..., G_r$ such that $G_i :=$ delete u.a.r. an edge $\{v_i, u_i\}$ of $G_{i+1}$ which does not belong to a cycle of length $< \log n / 10 \log d$.

- Colour randomly $G_0$ for $i = 0, ..., r-1$, apply UPDATE to the colouring of $G_i$ to get that of $G_{i+1}$.

**Output:** the colouring of $G_r$

**Remark**
Typically, each component of $G_0$ is either trivial or an isolated cycle.
The algorithm

**input:** $G(n, d/n), k$

create $G_0, G_1, \ldots, G_r$ such that

1. 

Remark

Typically, each component of $G_0$ is either trivial or an isolated cycle.
The algorithm

**input:** $G(n, d/n), k$

create $G_0, G_1, \ldots, G_r$ such that

$G_i :=$ delete u.a.r. an edge $\{v_i, u_i\}$ of $G_{i+1}$ which does not belong to a cycle of length $< \frac{\log n}{10 \log d}$. 

**Output:** the colouring of $G_r$.

**Remark**

Typically, each component of $G_0$ is either trivial or an isolated cycle.
The algorithm

**input:** $G(n, d/n), k$

create $G_0, G_1, \ldots, G_r$ such that $G_i := $ delete u.a.r. an edge $\{v_i, u_i\}$ of $G_{i+1}$ which does not belong to a cycle of length $< \frac{\log n}{10 \log d}$.

**Remark**

Typically, each component of $G_0$ is either trivial or an isolated cycle.
The algorithm

**input:** $G(n, d/n), k$

create $G_0, G_1, \ldots, G_r$ such that

$G_i :=$ delete u.a.r. an edge $\{v_i, u_i\}$ of $G_{i+1}$ which does not belong to a cycle of length $< \frac{\log n}{10 \log d}$.

colour randomly $G_0$

**Remark**

Typically, each component of $G_0$ is either trivial or an isolated cycle.
Result - The algorithm

The algorithm

**input:** $G(n, d/n), k$

create $G_0, G_1, \ldots, G_r$ such that

$G_i :=$ delete u.a.r. an edge $\{v_i, u_i\}$ of $G_{i+1}$ which does not belong to a cycle of length $< \frac{\log n}{10 \log d}$.

colour randomly $G_0$

for $i = 0, \ldots, r - 1$

Output: the colouring of $G_r$

Remark

Typically, each component of $G_0$ is either trivial or an isolated cycle.
The algorithm

**input:** \( G(n, d/n), k \)

create \( G_0, G_1, \ldots, G_r \) such that

\[ G_i := \text{delete u.a.r. an edge } \{v_i, u_i\} \text{ of } G_{i+1} \text{ which does not belong to a cycle of length } < \frac{\log n}{10 \log d}. \]

colour randomly \( G_0 \)

for \( i = 0, \ldots, r - 1 \)

apply UPDATE to the colouring of \( G_i \) to get that of \( G_{i+1} \)

**Remark**

Typically, each component of \( G_0 \) is either trivial or an isolated cycle.
The algorithm

**input:** $G(n, d/n), k$
- create $G_0, G_1, \ldots, G_r$ such that
  - $G_i :=$ delete u.a.r. an edge $\{v_i, u_i\}$ of $G_{i+1}$ which does not belong to a cycle of length $< \frac{\log n}{10 \log d}$.
- colour randomly $G_0$
- for $i = 0, \ldots, r - 1$
  - apply UPDATE to the colouring of $G_i$ to get that of $G_{i+1}$

**Output:** the colouring of $G_r$

**Remark**

Typically, each component of $G_0$ is either trivial or an isolated cycle.
Theorem

Take $k = (1 + \epsilon)d$ and assume the input graph is $G(n, d/n)$. Let $\mu, \mu'$ be the Gibbs distribution of the $k$-colourings of the input graph and the distribution of the output of the algorithm, respectively. With probability $1 - O(n^{-\gamma})$ over $G(n, d/n)$ it holds that

$$\|\mu - \mu'\|_{TV} = O(n^{-\gamma})$$
Some remarks about the error

Definition

Given $G_i$, $v_i$, $u_i$ and $k$, we let

- $X_i$ a random colouring of $G_i$
- $Y_{i+1} = \text{Update}(X_i, v_i, u_i)$
- $\mu_i, \nu_i$ are the distribution of $X_i$ and $Y_i$, respectively.

Theorem

Let $\mu$, $\mu'$ be the Gibbs distribution of the colourings of input graph and the distribution of the output of the algorithm, respectively. It holds that

$$||\mu - \mu'||_{TV} \leq r \sum_{i=1}^{k} ||\mu_i - \nu_i||_{TV}$$
Some remarks about the error

**Definition**

Given \( G_i, v_i, u_i \) and \( k \), we let

- \( X_i \) a random colouring of \( G_i \)
- \( Y_{i+1} = \text{Update}(X_i, v_i, u_i) \)
- \( \mu_i, \nu_i \) are the distribution of \( X_i \) and \( Y_i \), respectively.

**Theorem**

Let \( \mu, \mu' \) be the Gibbs distribution of the colourings of input graph and the distribution of the output of the algorithm, respectively. It holds that

\[
\|\mu - \mu'\|_{TV} \leq \sum_{i=1}^{r} \|\mu_i - \nu_i\|_{TV}
\]
switching as a map

we want to implement a mapping from $\Omega^{rr}$ to $\Omega^{gr}$

we want the mapping to be as "close" to a bijection as possible

For a bijection $h: S_1 \rightarrow S_2$, if $X$ is uniformly random in $S_1$, then $h(X)$ is uniformly random in $S_2$

switching is a kind of "approximate bijection"

it fails only on the "pathological colourings"
switching as a map

“switching from red to green”

Remarks
we want to implement a mapping from $\Omega^{rr}$ to $\Omega^{gr}$
we want the mapping to be as “close” to a bijection as possible
For a bijection $h: S_1 \rightarrow S_2$,
if $X$ is uniformly random in $S_1$,
then $h(X)$ is uniformly random in $S_2$
switching is a kind of “approximate bijection”
it fails only one the “pathological colourings”
switching as a map

“switching from red to green”

Remarks
“switching from red to green”

Remarks

- we want to implement a mapping from $\Omega_{rr}$ to $\Omega_{gr}$
switching as a map

“switching from red to green”

Remarks
- we want to implement a **mapping** from $\Omega_{rr}$ to $\Omega_{gr}$
- we want the mapping to be as “close” to a **bijection** as possible
switching as a map

“switching from red to green”

Remarks

- we want to implement a mapping from $\Omega_{rr}$ to $\Omega_{gr}$
- we want the mapping to be as “close” to a bijection as possible
  - For a bijection $h : S_1 \rightarrow S_2$, 
we want to implement a mapping from $\Omega_{rr}$ to $\Omega_{gr}$

we want the mapping to be as “close” to a bijection as possible

- For a bijection $h : S_1 \rightarrow S_2$,

  if $X$ is uniformly random in $S_1$, then $h(X)$ is uniformly random in $S_2$
we want to implement a mapping from $\Omega_{rr}$ to $\Omega_{gr}$

we want the mapping to be as “close” to a bijection as possible

For a bijection $h : S_1 \rightarrow S_2$,

if $X$ is uniformly random in $S_1$, then $h(X)$ is uniformly random in $S_2$

switching is a kind of “approximate bijection”
switching as a map

“switching from red to green”

Remarks

- we want to implement a mapping from $\Omega_{rr}$ to $\Omega_{gr}$
- we want the mapping to be as “close” to a bijection as possible
  - For a bijection $h : S_1 \rightarrow S_2$,
    
    if $X$ is uniformly random in $S_1$, then $h(X)$ is uniformly random in $S_2$

- switching is a kind of “approximate bijection”
  - it fails only one the “pathological colourings”
The effect of pathological colouring

Gi, vi, ui X_i is a uniformly random colouring of G_i

Y_i + 1 = Update (X_i, v_i, u_i)

Paths of disagreements in X_i

For c, q ∈ [k] such that c ≠ q, let ϱ_i(c, q) be the expected number of paths from v_i to u_i coloured with (c, q), in X_i

C. Efthymiou (GaTech)
The effect of pathological colouring

... reminder

- $G_i, v_i, u_i$
- $X_i$ is a uniformly random a coloring of $G_i$
- $Y_{i+1} = \text{Update}(X_i, v_i, u_i)$

$$\|\mu_i - \nu_i\|_{TV} \leq ?$$
The effect of pathological colouring

... reminder

- $G_i, v_i, u_i$
- $X_i$ is a uniformly random a coloring of $G_i$
- $Y_{i+1} = \text{Update}(X_i, v_i, u_i)$

$$\|\mu_i - \nu_i\|_{TV} \leq ?$$

Paths of disagreements in $X_i$

For $c, q \in [k]$ such that $c \neq q$, let $\varrho_i(c, q)$ be the expected number of paths from $v_i$ to $u_i$ coloured with $(c, q)$, in $X_i$
The effect of pathological colouring

... reminder

- $G_i, v_i, u_i$
- $X_i$ is a uniformly random a coloring of $G_i$
- $Y_{i+1} = \text{Update}(X_i, v_i, u_i)$

$$\|\mu_i - \nu_i\|_{TV} \leq \Theta(1) \max_{c,q} \{\varrho_{i-1}(c,q)\}$$

Paths of disagreements in $X_i$

For $c, q \in [k]$ such that $c \neq q$, let $\varrho_i(c, q)$ be the expected number of paths from $v_i$ to $u_i$ coloured with $(c, q)$, in $X_i$
Upper bounding $\varrho_i(c, q)$
Upper bounding $\varrho_i(c, q)$

... since we are dealing with random graphs!
Upper bounding $\varrho_i(c, q)$

... since we are dealing with random graphs!

- focus on $E[\varrho_i(c, q)]$
Upper bounding $\varrho_i(c, q)$

... since we are dealing with random graphs!

- focus on $\mathbb{E}[\varrho_i(c, q)]$
  - take an instance of $G_i$
Upper bounding $\varrho_i(c, q)$

... since we are dealing with random graphs!

- focus on $\mathbb{E}[\varrho_i(c, q)]$
  - take an instance of $G_i$
  - take $X_i$ a random colouring of $G_i$
Upper bounding $\varrho_i(c, q)$

... since we are dealing with random graphs!

- focus on $\mathbb{E}[\varrho_i(c, q)]$
  - take an instance of $G_i$
  - take $X_i$ a random colouring of $G_i$
  - count the paths from $v_i$ to $u_i$ which are coloured with $c, q$, in $X_i$
Upper bounding $\varrho_i(c, q)$

... since we are dealing with random graphs!

- focus on $\mathbb{E}[\varrho_i(c, q)]$
  - take an instance of $G_i$
  - take $X_i$ a random colouring of $G_i$
  - count the paths from $v_i$ to $u_i$ which are coloured with $c, q$, in $X_i$
- show that $\mathbb{E}[\varrho_i(c, q)]$ is sufficiently small.
Upper bounding $\mathbb{E}[\rho_i(c, q)]$
Upper bounding $\mathbb{E}[\varrho_i(c, q)]$

**Linearity of the expectation**
- consider a permutation of, say, $l$ vertices with $v_i$ first and $u_i$ last
- find the probability the vertices in the permutation form a path coloured $c, q$ in $X_i$
Upper bounding $\mathbb{E}[\rho_i(c, q)]$

**Linearity of the expectation**
- consider a permutation of, say, $l$ vertices with $v_i$ first and $u_i$ last
- find the probability the vertices in the permutation form a path coloured $c, q$ in $X_i$

**The probability of a path to be 2 coloured**
Upper bounding $\mathbb{E}[\varrho_i(c, q)]$

**Linearity of the expectation**

- consider a permutation of, say, $l$ vertices with $v_i$ first and $u_i$ last
- find the probability the vertices in the permutation form a path coloured $c, q$ in $X_i$

**The probability of a path to be 2 coloured**

- highly non trivial to compute the probability exactly
Upper bounding $E[\varphi_i(c, q)]$

### Linearity of the expectation
- Consider a permutation of, say, $l$ vertices with $v_i$ first and $u_i$ last.
- Find the probability the vertices in the permutation form a path coloured $c, q$ in $X_i$.

### The probability of a path to be 2 coloured
- Highly non trivial to compute the probability exactly.
  - Structure of $G_i$ too complex.
Upper bounding $\mathbb{E}[\varrho_i(c, q)]$

### Linearity of the expectation
- consider a permutation of, say, $l$ vertices with $v_i$ first and $u_i$ last
- find the probability the vertices in the permutation form a path coloured $c, q$ in $X_i$

### The probability of a path to be 2 coloured
- highly non trivial to compute the probability exactly
  - structure of $G_i$ too complex
  - Gibbs distribution even more complex
Upper bounding $\mathbb{E}[\rho_i(c, q)]$

**Linearity of the expectation**
- consider a permutation of, say, $l$ vertices with $v_i$ first and $u_i$ last
- find the probability the vertices in the permutation form a path
  coloured $c, q$ in $X_i$

**The probability of a path to be 2 coloured**
- highly non trivial to compute the probability exactly
  - structure of $G_i$ too complex
  - Gibbs distribution even more complex
- compute sufficiently good approximations
Upper bounding $\mathbb{E}[\varrho_i(c, q)]$

**Linearity of the expectation**
- consider a permutation of, say, $l$ vertices with $v_i$ first and $u_i$ last
- find the probability the vertices in the permutation form a path coloured $c, q$ in $X_i$

**The probability of a path to be 2 coloured**
- highly non trivial to compute the probability exactly
  - structure of $G_i$ too complex
  - Gibbs distribution even more complex
- compute sufficiently good approximations
  - reveal a small neighborhoods around the vertices
Upper bounding $\mathbb{E}[\rho_i(c, q)]$

**Linearity of the expectation**
- consider a permutation of, say, $l$ vertices with $v_i$ first and $u_i$ last
- find the probability the vertices in the permutation form a path coloured $c, q$ in $X_i$

**The probability of a path to be 2 coloured**
- highly non trivial to compute the probability exactly
  - structure of $G_i$ too complex
  - Gibbs distribution even more complex
- compute sufficiently good approximations
  - reveal a small neighborhoods around the vertices
  - make some kind of “worst-case” assumption about what is beyond
Upper bounding $\mathbb{E}[\varrho_i(c, q)]$

**Linearity of the expectation**
- consider a permutation of, say, $l$ vertices with $v_i$ first and $u_i$ last
- find the probability the vertices in the permutation form a path coloured $c, q$ in $X_i$

**The probability of a path to be 2 coloured**
- highly non trivial to compute the probability exactly
  - structure of $G_i$ too complex
  - Gibbs distribution even more complex
- compute sufficiently good approximations
  - reveal a small neighborhoods around the vertices
  - make some kind of “worst-case” assumption about what is beyond
  - estimate the probability based on what we have revealed and the worst-case assumptions
The probability of a 2 coloured path

Graph first reveal a neighborhood around each vertex in a BFS manner
constant number of neighbors
everything is within radius $r$
“most of the times”
the neighborhood is a tree of height at most $r'$
maximum degree $< (1 + \epsilon/2)d$
the neighborhood does not intersect with others
The probability of a 2 coloured path

Graph first
The probability of a 2 coloured path I

Graph first
The probability of a 2 coloured path

Graph first
The probability of a 2 coloured path

Graph first

- reveal a neighborhood around each vertex in a BFS manner
The probability of a 2 coloured path

Graph first

- reveal a neighborhood around each vertex in a BFS manner
Graph first

- reveal a neighborhood around each vertex in a BFS manner
  - constant number of neighbours
  - everything is within radius $r'$

\[ r_1 \]
\[ r_2 \]
\[ r_3 \]
\[ r_4 \]
\[ r_5 \]
\[ r_6 \]
\[ r_7 \]
\[ r_8 \]
\[ r_9 \]
\[ r_{10} \]
\[ r_{11} \]
\[ r_{12} \]
The probability of a 2 coloured path I

Graph first

- reveal a neighborhood around each vertex in a BFS manner
  - constant number of neighbours
  - everything is within radius $r'$

[Diagram showing a graph with radii $r_1$, $r_2$, $r_3$, etc.]
The probability of a 2 coloured path

Graph first

- reveal a neighborhood around each vertex in a BFS manner
  - constant number of neighbours
  - everything is within radius $r'$

![Graph diagram]

$r_1 \ r_2 \ r_3 \ r_4 \ r_5 \ r_6 \ r_7 \ r_8 \ r_9 \ r_{10} \ r_{11} \ r_{12}$
The probability of a 2 coloured path

Graph first

- reveal a neighborhood around each vertex in a BFS manner
  - constant number of neighbours
  - everything is within radius $r'$

![Graph Diagram]

$r_1, r_2, r_3, r_4, r_5, r_6, r_7, r_8, r_9, r_{10}, r_{11}, r_{12}$
The probability of a 2 coloured path 1

Graph first

- reveal a neighborhood around each vertex in a BFS manner
  - constant number of neighbours
  - everything is within radius $r'$
The probability of a 2 coloured path 1

**Graph first**
- reveal a neighborhood around each vertex in a BFS manner
  - **constant** number of neighbours
  - everything is within **radius** \( r' \)

\[
\begin{align*}
r_1 & \quad r_2 \\
r_3 & \quad r_4 \\
r_5 & \quad r_6 \\
r_7 & \quad r_8 \\
r_9 & \quad r_{10} \\
r_{11} & \quad r_{12}
\end{align*}
\]
The probability of a 2 coloured path I

**Graph first**

- reveal a neighborhood around each vertex in a BFS manner
  - **constant** number of neighbours
  - everything is within **radius** $r'$
The probability of a 2 coloured path

Graph first

- reveal a neighborhood around each vertex in a BFS manner
  - constant number of neighbours
  - everything is within radius $r'$
The probability of a 2 coloured path

Graph first

- reveal a neighborhood around each vertex in a BFS manner
  - constant number of neighbours
  - everything is within radius $r'$
- “most of the times”
  - the neighborhood is a tree of height at most $r'$
  - maximum degree $< (1 + \epsilon/2)d$
  - the neighborhood does not intersect with others
The probability of a 2 coloured path II

The random colouring part

Ideally we consider a convex combination of boundary conditions. Instead, we consider a worst case boundary condition. The probability of each vertex to take on the "appropriate" colour mainly depends on its "immediate neighbourhood."
The random colouring part

ideally we consider a convex combination of boundary conditions

instead we consider a worst case boundary condition

the probability of each vertex to take on the "appropriate" colour

mainly depends on its "immediate neighbourhood"
The probability of a 2-coloured path II

The random colouring part

Ideally, we consider a convex combination of boundary conditions. Instead, we consider a worst case boundary condition. The probability of each vertex to take on the "appropriate" colour mainly depends on its "immediate neighbourhood".
The random colouring part

- ideally we consider a **convex combination** of boundary conditions
The probability of a 2 coloured path II

The random colouring part

- ideally we consider a **convex combination** of boundary conditions
- instead we consider a **worst case** boundary condition

---

C.Efthymiou (GaTech)
The random colouring part

- ideally we consider a **convex combination** of boundary conditions
- instead we consider a **worst case** boundary condition
The probability of a 2 coloured path II

The random colouring part

- ideally we consider a **convex combination** of boundary conditions
- instead we consider a **worst case** boundary condition
- the probability of each vertex to take on the “appropriate” colour mainly depends on its “immediate neighbourhood”
The random colouring part

- Ideally we consider a **convex combination** of boundary conditions.
- Instead we consider a **worst case** boundary condition.
- The probability of each vertex to take on the “appropriate” colour mainly depends on its “immediate neighbourhood”.
Structure around the path

Good Vs Bad Neighbourhoods

Good

tree of height at most $r^\prime$

maximum degree $\leq (1 + \epsilon/2)d$
does not intersect with other neighbourhoods

Bad

... everything that is not Good
Structure around the path

Good Vs Bad Neighbourhoods

Good
- tree of height at most $r'$
- maximum degree $\leq (1 + \epsilon/2)d$
- does not intersect with other neighbourhoods

Bad
- everything that is not Good
Structure around the path

Good Vs Bad Neighbourhoods

Good
- tree of height at most $r'$
- maximum degree $\leq (1 + \epsilon/2)d$
- does not intersect with other neighbourhoods

Bad
- everything that is not Good

\[
\begin{align*}
\text{r}_1 & \quad \text{r}_2 \\
\text{r}_3 & \quad \text{r}_4 \\
\text{r}_5 & \quad \text{r}_6 \\
\text{r}_7 & \quad \text{r}_8 \\
\text{r}_9 & \quad \text{r}_{10} \\
\text{r}_{11} & \quad \text{r}_{12}
\end{align*}
\]
Structure around the path

Good Vs Bad Neighbourhoods

- Good

- Bad
Structure around the path

**Good Vs Bad Neighbourhoods**

- **Good**
  - tree of height at most $r'$
  - maximum degree $\leq (1 + \epsilon/2)d$
  - does not intersect with other neighbourhoods

- **Bad**
Structure around the path

Good Vs Bad Neighbourhoods

- **Good**
  - tree of height at most \( r' \)
  - maximum degree \( \leq (1 + \epsilon/2)d \)
  - does not intersect with other neighbourhoods

- **Bad** ... everything that is not Good
Effect of neighbour’s structure

If the neighbourhood is "Good", then for $k = (1 + \epsilon)d$ we have

$$\Pr[v \text{ is coloured } q | \text{ colouring of boundary}] = 1^{k}(1 + f\epsilon, r')$$

where $f\epsilon, r' \to 0$ as $r'$ grows.

If the neighbourhood is "Bad", then

$$\Pr[v \text{ is coloured } q | \text{ colouring of boundary}] \leq 1^{r_1 r_2 r_3 r_4 r_5 r_6 r_7 r_8 r_9 r_{10} r_{11} r_{12}}$$
Effect of neighbour’s structure

Cases

If the neighbourhood is "Good", then for $k = (1 + \epsilon)d$ we have

$$\Pr[v \text{ is coloured } q \mid \text{ colouring of boundary}] = 1$$

where $\epsilon, r' \to 0$ as $r'$ grows.

If the neighbourhood is "Bad", then

$$\Pr[v \text{ is coloured } q \mid \text{ colouring of boundary}] \leq 1$$
Effect of neighbour’s structure

Cases

- if the neighbourhood is “Good”, then for $k = (1 + \epsilon)d$ we have

  \[
  \Pr[\nu \text{ is coloured } q | \text{colouring of boundry}] = \frac{1}{k} (1 + f_{\epsilon, r'})
  \]

  where $f_{\epsilon, r'} \to 0$ as $r'$ grows.

- if the neighbourhood is “Bad”, then

  \[
  \Pr[\nu \text{ is coloured } q | \text{colouring of boundry}] \leq \frac{1}{r_1 r_2 r_3 r_4 r_5 r_6 r_7 r_8 r_9 r_{10} r_{11} r_{12}}
  \]
Effect of neighbour’s structure

Cases

- if the neighbourhood is “Good”, then for \( k = (1 + \epsilon)d \) we have

\[
\Pr[\nu \text{ is coloured } q | \text{colouring of boundry}] = \frac{1}{k} (1 + f_{\epsilon,r'})
\]

where \( f_{\epsilon,r'} \to 0 \) as \( r' \) grows.

- if the neighbourhood is “Bad”, then

\[
\Pr[\nu \text{ is coloured } q | \text{colouring of boundry}] \leq 1
\]
Then we get that ...

**Corollary 1**

For \( k = (1 + \epsilon)d \) and any \( 0 \leq i \leq r \), it holds that

\[
E[\varrho_i] \leq n - (1 + \gamma),
\]

for \( \gamma = \gamma(\epsilon, d) > 0 \).

**Corollary 2**

For \( k = (1 + \epsilon)d \) and input graph \( G(n, d/n) \) the following is true: Let \( \mu, \mu' \) be the Gibbs distribution of the \( k \)-colourings of \( G(n, d/n) \) and the distribution of the output of the algorithm, respectively, then

\[
E[|\mu - \mu'|_{TV}] \leq O(n - \gamma).
\]
Then we get that ...

**Corollary 1**

For $k = (1 + \epsilon)d$ and any $0 \leq i \leq r$, it holds that

$$\mathbb{E} [\varrho_i] \leq n^{-(1+\gamma)},$$

for $\gamma = \gamma(\epsilon, d) > 0$. 

**Corollary 2**

For $k = (1 + \epsilon)d$ and input graph $G(n, d/n)$ the following is true: Let $\mu, \mu'$ be the Gibbs distribution of the $k$-colourings of $G(n, d/n)$ and the distribution of the output of the algorithm, respectively, then

$$\mathbb{E} ||\mu - \mu'||_{TV} \leq O(n^{-\gamma}).$$
Then we get that ...

**Corollary 1**

For $k = (1 + \epsilon)d$ and any $0 \leq i \leq r$, it holds that

$$\mathbb{E} [\varrho_i] \leq n^{-(1+\gamma)},$$

for $\gamma = \gamma(\epsilon, d) > 0$.

**Corollary 2**

For $k = (1 + \epsilon)d$ and input graph $G(n, d/n)$ the following is true: Let $\mu, \mu'$ be the Gibbs distribution of the $k$-colourings of $G(n, d/n)$ and the distribution of the output of the algorithm, respectively, then

$$\mathbb{E} \| \mu - \mu' \|_{TV} \leq O(n^{-\gamma}).$$
Conclusions

We presented a simple algorithm for random $k$-colouring $G(n, d/n)$ where $k = (1 + \epsilon) d$ use less colours than any other algorithm. The distribution of the colouring is asymptotically the uniform one. Is there any improvement?

The lower bound for $k$ is expected to be $2\chi(G(n, d/n))$, a factor $\ln d$ away from the conjectured bound. There is a phase transition when $k < d$.

The set of $k$-colourings of $G(n, d/n)$ "looks different"... We argue on both the statistical properties of $G(n, d/n)$ and its random colourings.

UPDATE: In its current form, the algorithm is not expected to work for $k < d$. 

C. Efthymiou (GaTech)
we presented a simple algorithm for random $k$-colouring $G(n, d/n)$ where $k = (1 + \epsilon)d$
we presented a simple algorithm for random $k$-colouring $G(n, d/n)$ where $k = (1 + \epsilon)d$

- less colours than any other algorithm
Conclusions

- We presented a simple algorithm for random $k$-colouring $G(n, d/n)$ where $k = (1 + \epsilon)d$
  - Less colours than any other algorithm
  - The distribution of the colouring is asymptotically the uniform one
Conclusions

- we presented a simple algorithm for random $k$-colouring $G(n, d/n)$ where $k = (1 + \epsilon)d$
  - less colours than any other algorithm
  - the distribution of the colouring is asymptotically the uniform one
- is there any improvement?
Conclusions

- We presented a simple algorithm for random $k$-colouring $G(n, d/n)$ where $k = (1 + \epsilon)d$.
  - Less colours than any other algorithm.
  - The distribution of the colouring is asymptotically the uniform one.
- Is there any improvement?
  - The lower bound for $k$ is expected to be $2\chi(G(n, d/n))$.
Conclusions

- We presented a simple algorithm for random $k$-colouring $G(n, d/n)$ where $k = (1 + \epsilon)d$
  - less colours than any other algorithm
  - the distribution of the colouring is asymptotically the uniform one

- Is there any improvement?
  - the lower bound for $k$ is expected to be $2\chi(G(n, d/n))$
  - a factor $\ln d$ away from the conjectured bound
Conclusions

- we presented a simple algorithm for random $k$-colouring $G(n, d/n)$ where $k = (1 + \epsilon)d$
  - less colours than any other algorithm
  - the distribution of the colouring is asymptotically the uniform one
- is there any improvement?
  - the lower bound for $k$ is expected to be $2\chi(G(n, d/n))$
  - a factor $\ln d$ away from the conjectured bound
  - there is a phase transition when $k < d$
we presented a simple algorithm for random $k$-colouring $G(n, d/n)$ where $k = (1 + \epsilon)d$

- less colours than any other algorithm
- the distribution of the colouring is asymptotically the uniform one

is there any improvement?

- the lower bound for $k$ is expected to be $2\chi(G(n, d/n))$
- a factor $\ln d$ away from the conjectured bound
- there is a phase transition when $k < d$
  - the set of $k$-colourings of $G(n, d/n)$ “looks different” ...
we presented a simple algorithm for random $k$-colouring $G(n, d/n)$ where $k = (1 + \epsilon)d$

- less colours than any other algorithm
- the distribution of the colouring is asymptotically the uniform one

is there any improvement?

- the lower bound for $k$ is expected to be $2\chi(G(n, d/n))$
- a factor $\ln d$ away from the conjectured bound
- there is a phase transition when $k < d$
  - the set of $k$-colourings of $G(n, d/n)$ “looks different” ...
  - argue on both the statistical properties of $G(n, d/n)$ and its random colourings
Conclusions

- We presented a simple algorithm for random \( k \)-colouring \( G(n, d/n) \) where \( k = (1 + \epsilon)d \)
  - Less colours than any other algorithm
  - The distribution of the colouring is asymptotically the uniform one

- Is there any improvement?
  - The lower bound for \( k \) is expected to be \( 2\chi(G(n, d/n)) \)
  - A factor \( \ln d \) away from the conjectured bound
  - There is a phase transition when \( k < d \)
    - The set of \( k \)-colourings of \( G(n, d/n) \) “looks different” ...
    - Argue on both the statistical properties of \( G(n, d/n) \) and its random colourings

- UPDATE, in its current form, is not expected to work for \( k < d \)
Thank You!!!