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

Charis Efthymiou
efthymiou@gmail.com

GeorgiaTech

Graphical Models, Statistical Inference, and Algorithms
IMA - University of Minnesota

Random colouring of a graph G

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.

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*

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. *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

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

An overview

The algorithm

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

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

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

Measure of comparison

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]

Measure of comparison

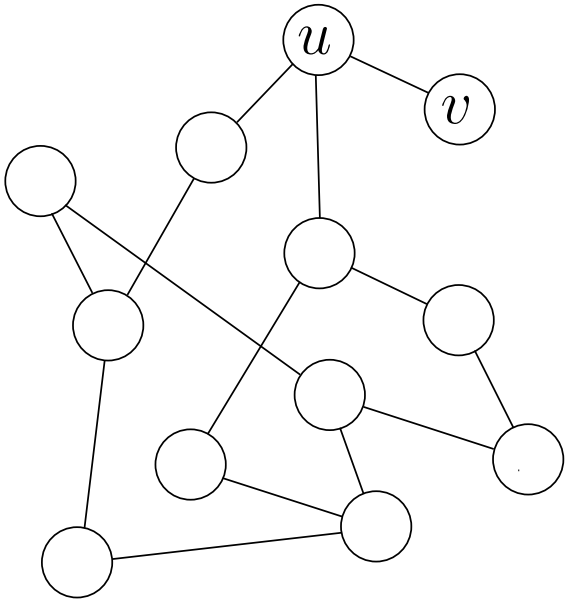
MCMC sampling

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"

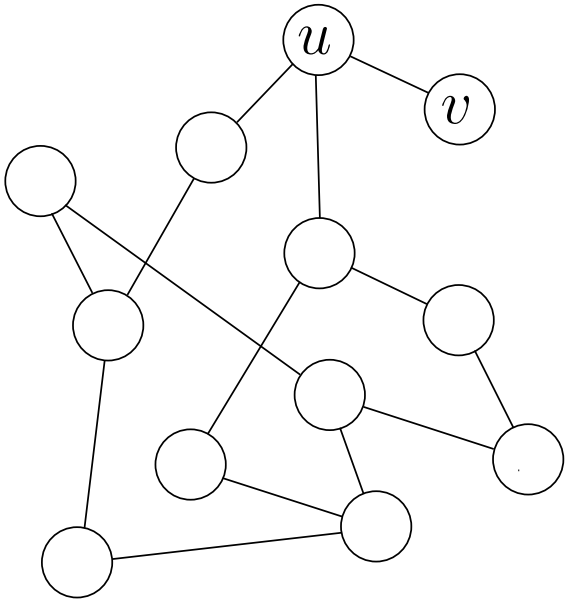
There is a FPAUS sampling k-colouring algorithm for typical instances of $G(n, d/n)$, for any $k > 3d$. [Yin, Zhang:'15]

A simple observation

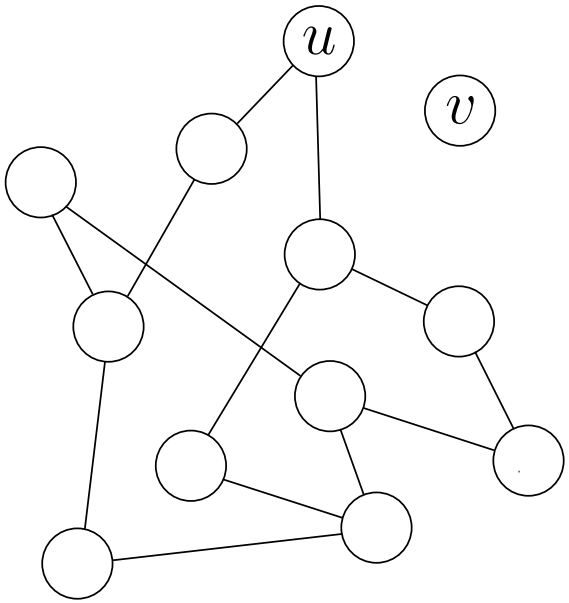


G

A simple observation

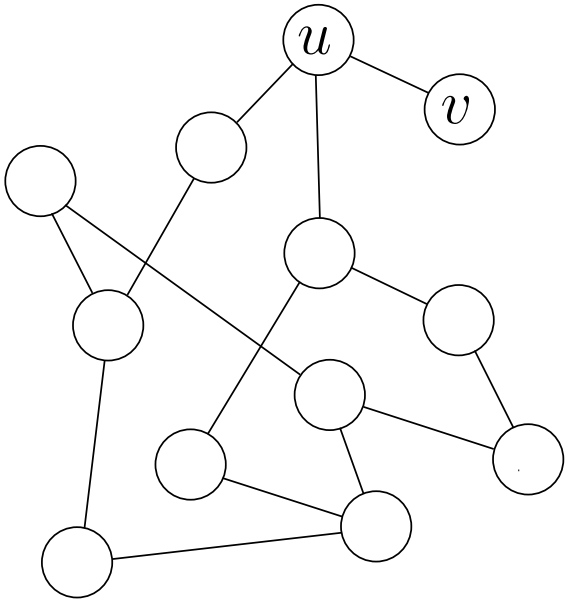


G

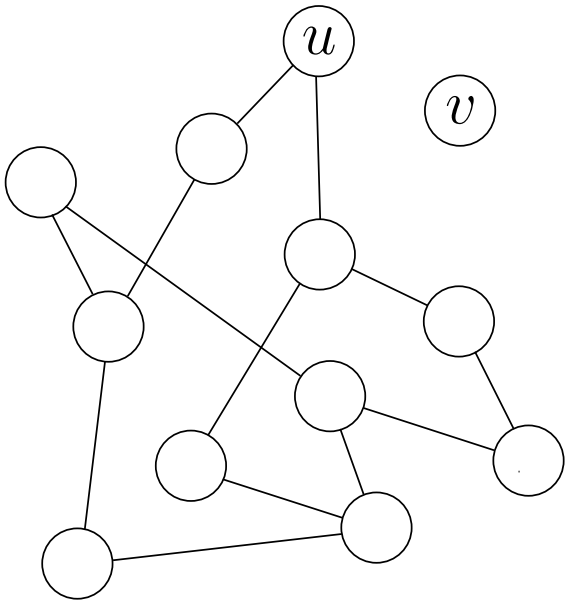


G'

A simple observation



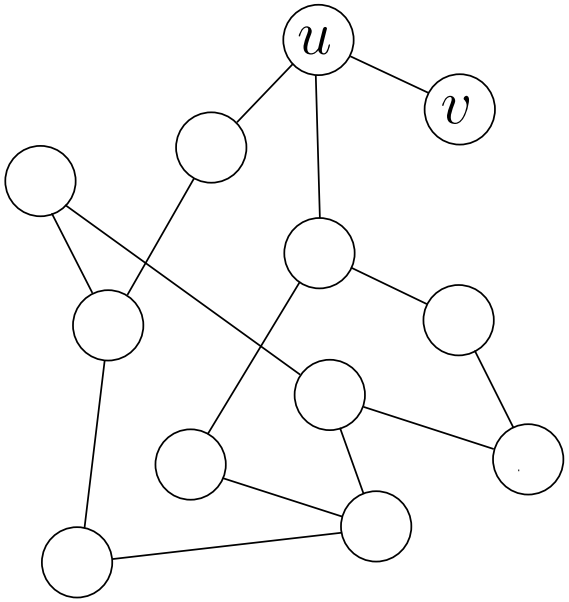
G



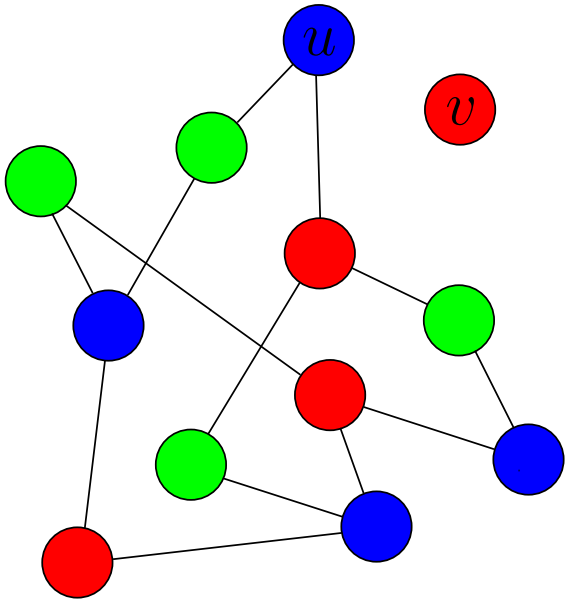
G'

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



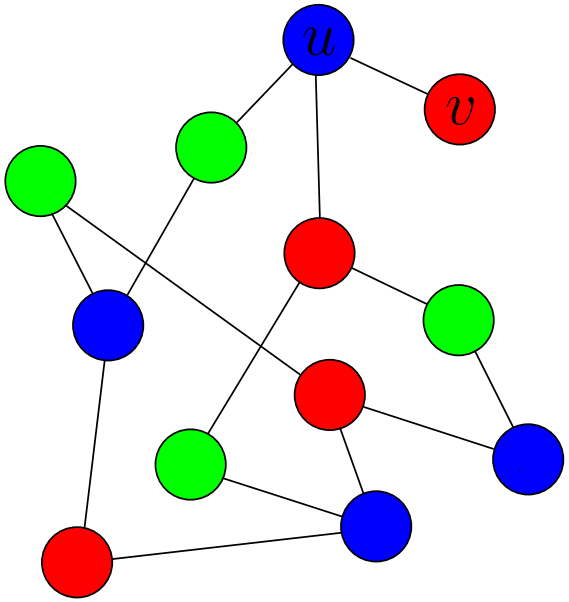
G



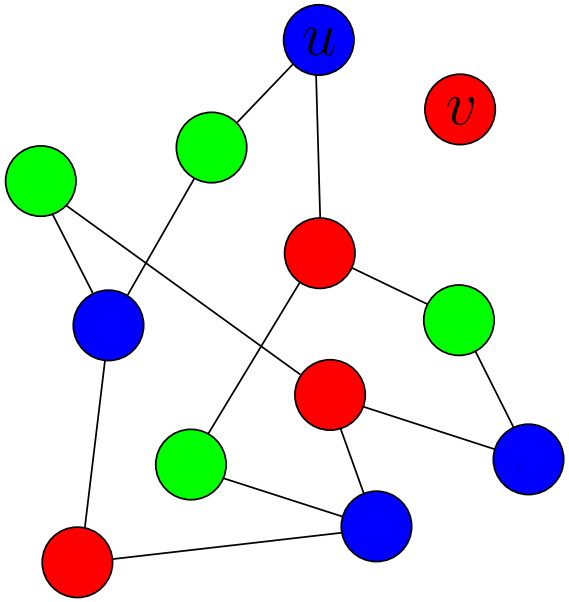
G'

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



G



G'

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.

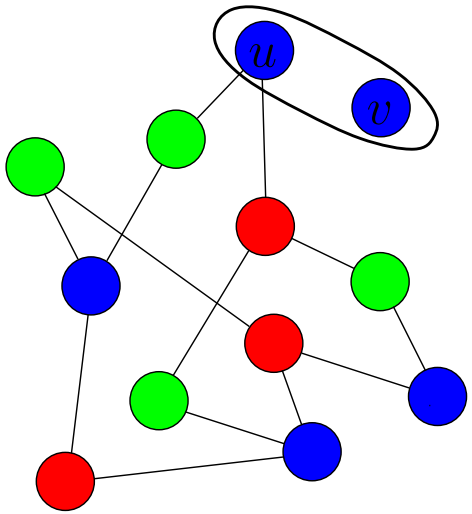
output: random k-colouring of G, conditional u, v are assigned different colours.

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.

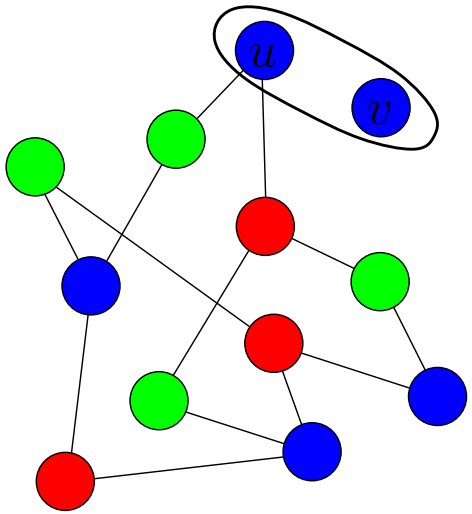


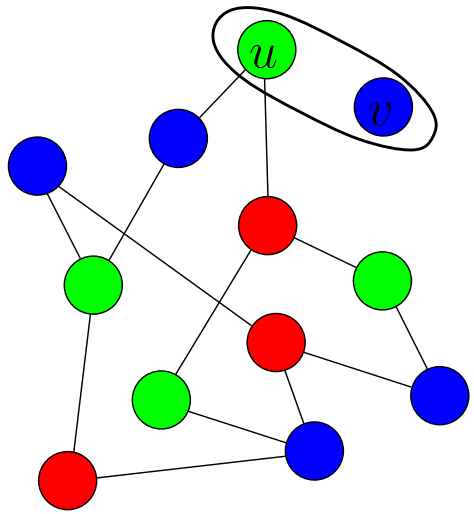
Suppose that ....

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.



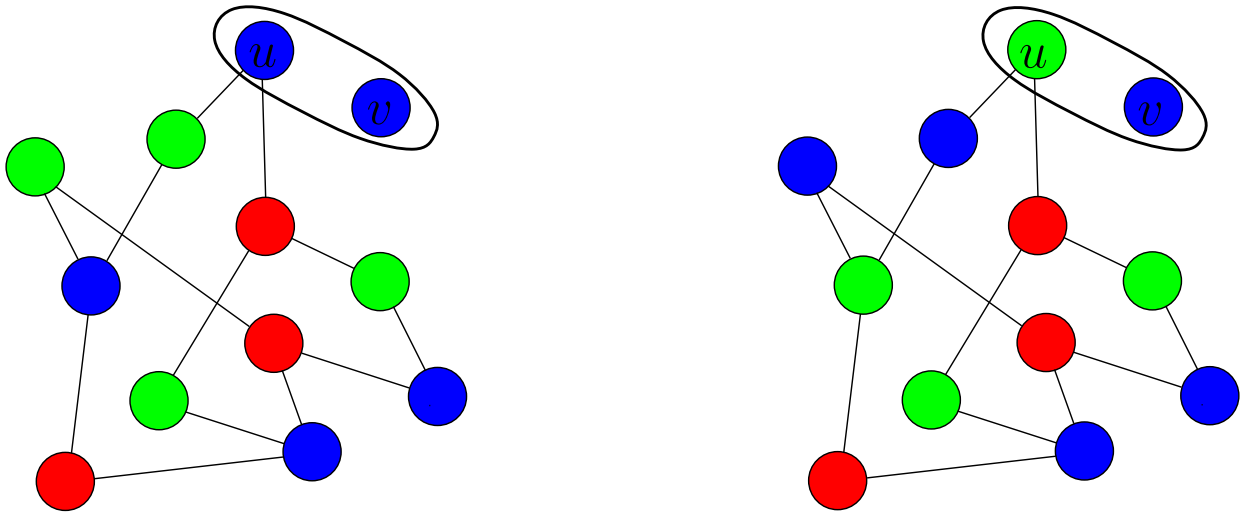


Suppose that ....

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

Use UPDATE for sampling colourings ...

The algorithm

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

Use UPDATE for sampling colourings ...

The algorithm

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

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

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 $\{v_i, u_i\}$

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"

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

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

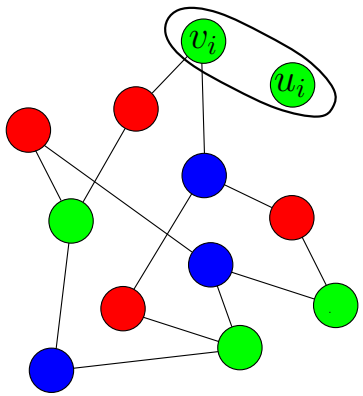
get G_i from G_{i+1} by deleting some edge $\{v_i, u_i\}$

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}



Use UPDATE for sampling colourings ...

The algorithm

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

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

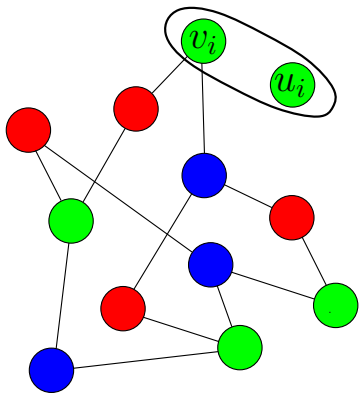
get G_i from G_{i+1} by deleting some edge $\{v_i, u_i\}$

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}

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 $G(n, d/n)$

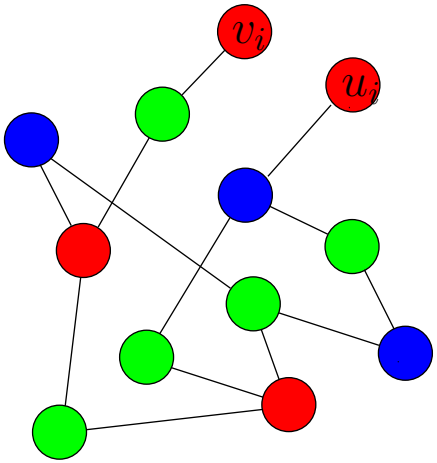
UPDATE

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

How does UPDATE look like for $G(n, d/n)$

UPDATE

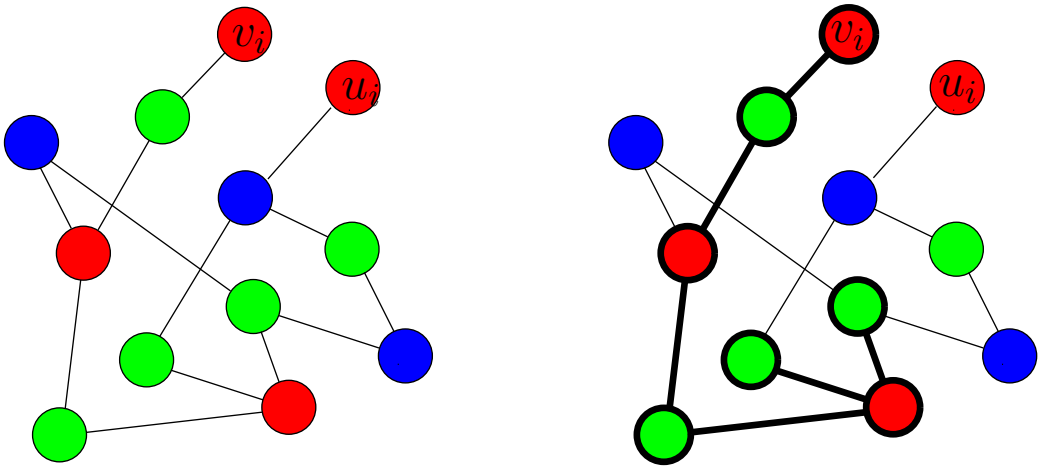
- **input**: G_i, σ, v_i u_i
- **if** $\sigma(v_i) \neq \sigma(u_i)$, **then** return σ
- **Otherwise**
  - q is chosen u.a.r. from $[k] \backslash \{\sigma_v\}$
  - **return** the q-switching of σ



How does UPDATE look like for $G(n, d/n)$

UPDATE

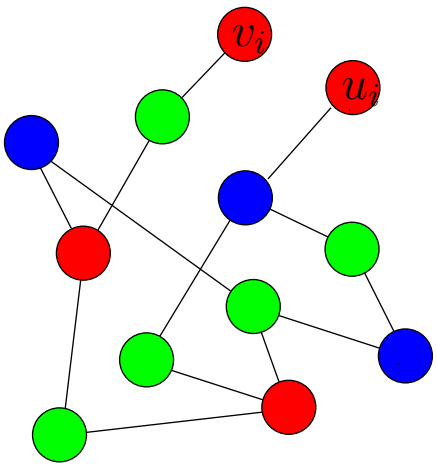
- **input**: G_i, σ, v_i u_i
- **if** $\sigma(v_i) \neq \sigma(u_i)$, **then** return σ
- **Otherwise**
  - q is chosen u.a.r. from $[k] \backslash \{\sigma_v\}$
  - **return** the q-switching of σ

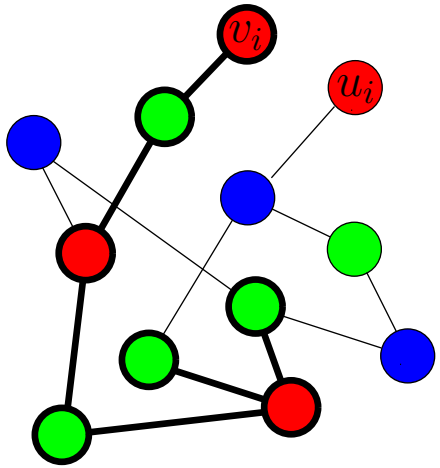


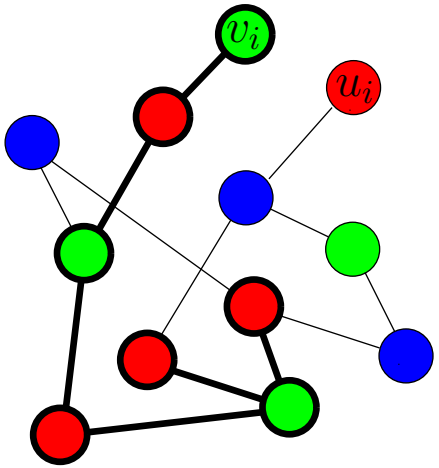
How does UPDATE look like for $G(n, d/n)$

UPDATE

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

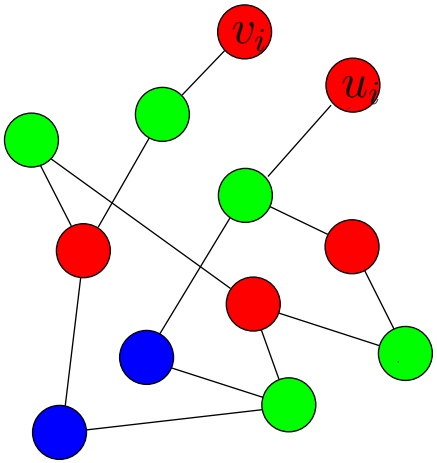




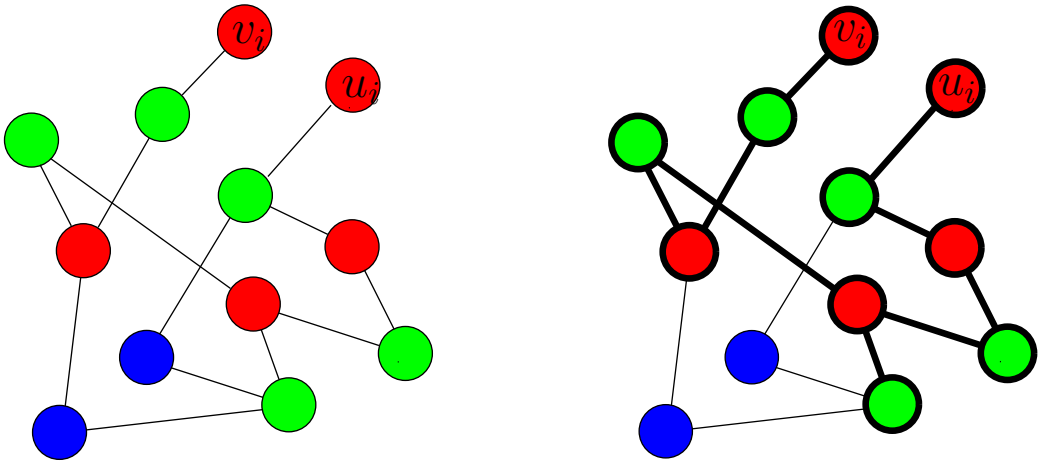


... there is no panacea

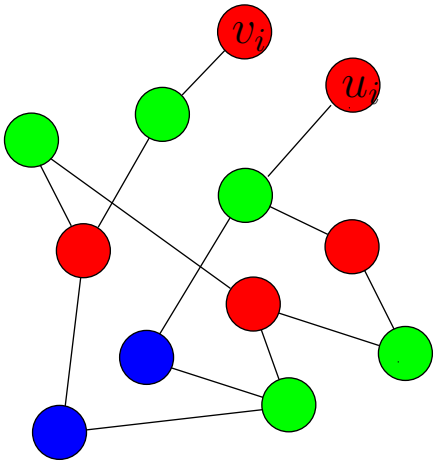
... there is no panacea

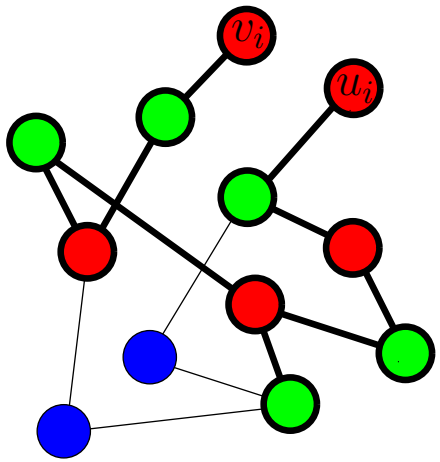


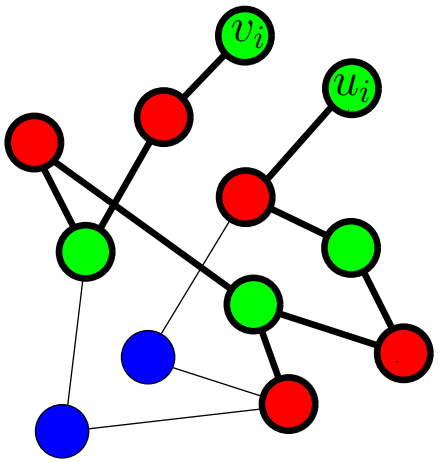
... there is no panacea



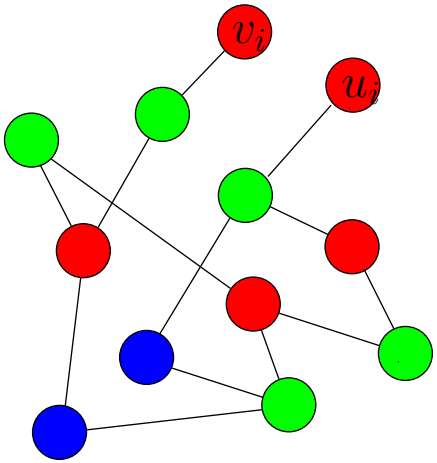
... there is no panacea

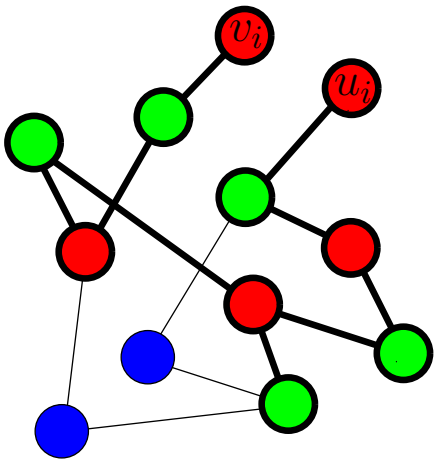


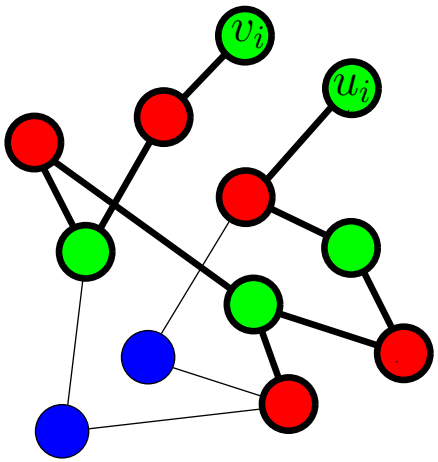




... there is no panacea



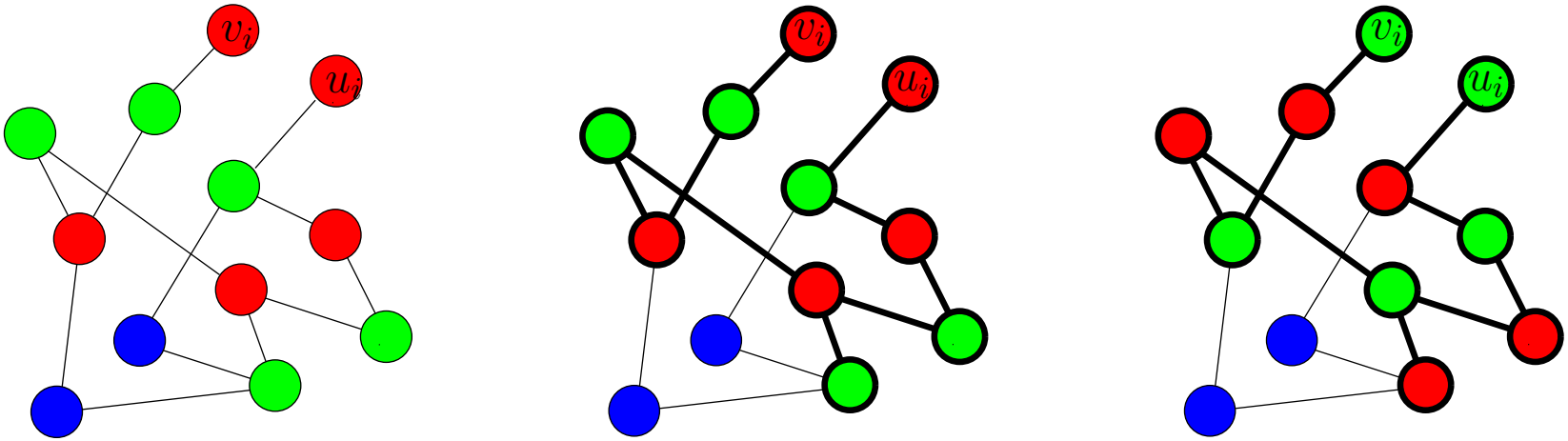




Pathological Colouring

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

... there is no panacea



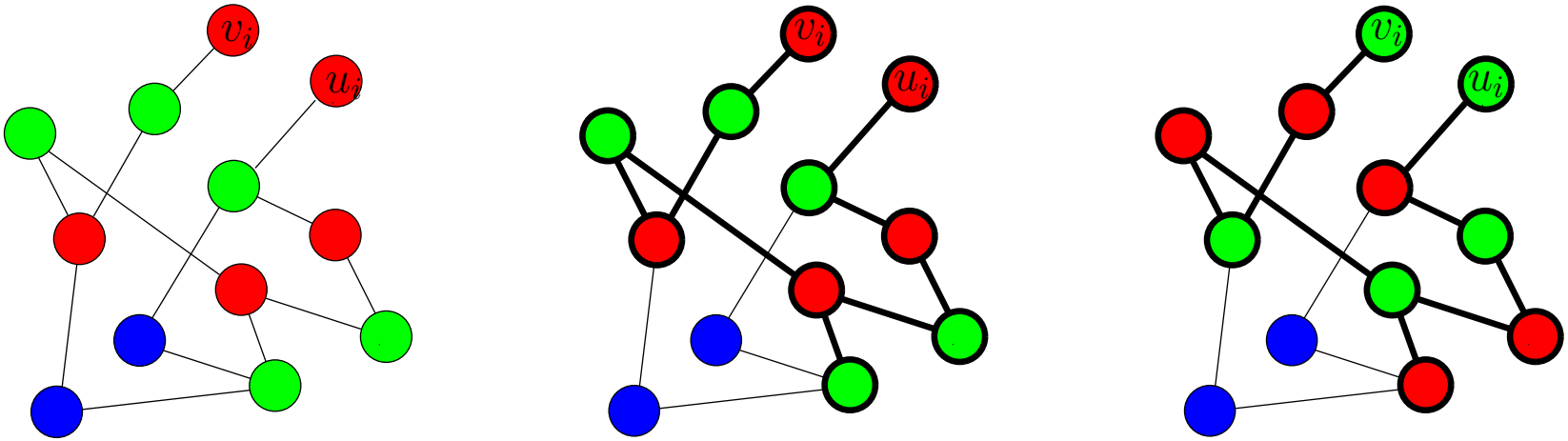
Pathological Colouring

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

... there is no panacea



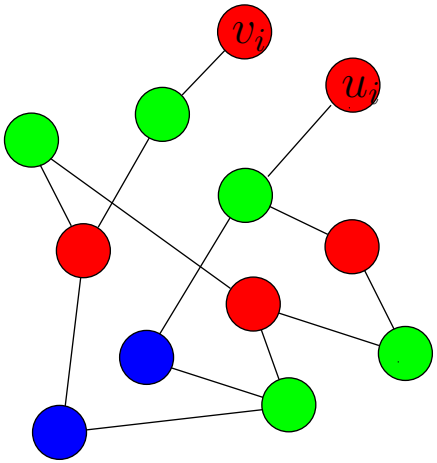
Pathological Colouring

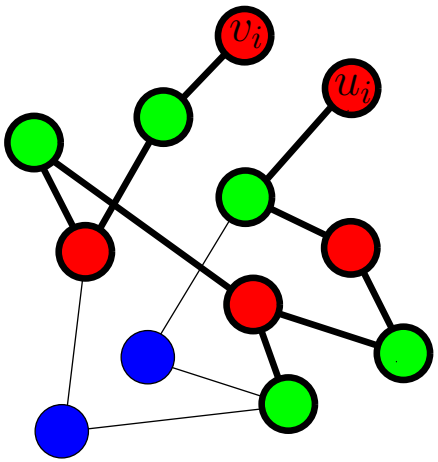
Every k-colouring which specifies a 2-coloured path between v_i and u_i

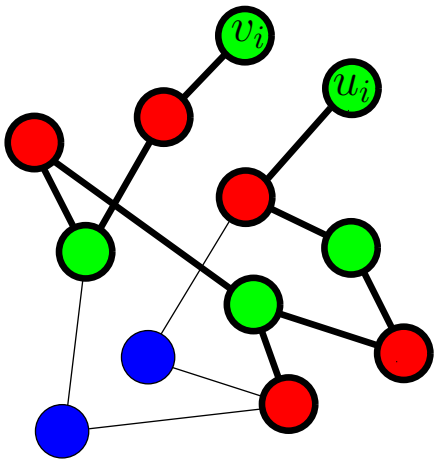
Concequently...

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

Result -The algorithm

The algorithm

Result -The algorithm

The algorithm

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

Result -The algorithm

The algorithm

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

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

.

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 $< \frac{\log n}{10 \log d}$.

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 $< \frac{\log n}{10 \log d}$.

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

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

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

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.

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

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.

Result

Theorem

Take $k = (1+\epsilon)d$ and assume the input graph is $G(n, d/n)$. Let μ, μ' 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
- $Y_{i+1} = \texttt{Update}(X_i, v_i, u_i)$
- μ_i, ν_i are the distribution of X_i and Y_i, respectively.

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
- $Y_{i+1} = \texttt{Update}(X_i, v_i, u_i)$
- μ_i, ν_i are the distribution of X_i and Y_i, respectively.

Theorem

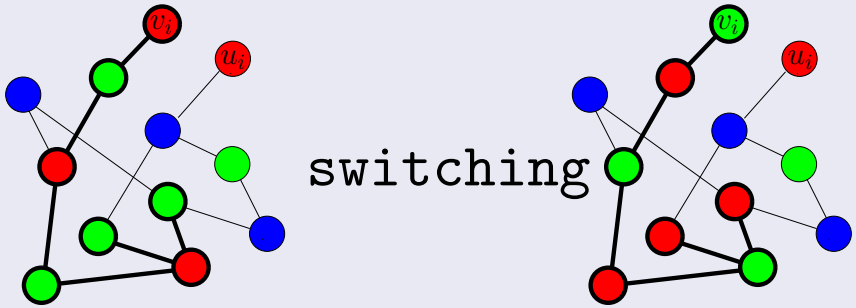
Let μ, μ' be the Gibbs distribution of the colourings of input graph and the distribution of the output of the algorithm, respectively. It holds that

$$||\mu - \mu'||_{TV} \leq \sum_{i=1}^{r} ||\mu_i - \nu_i||_{TV}$$

switching as a map

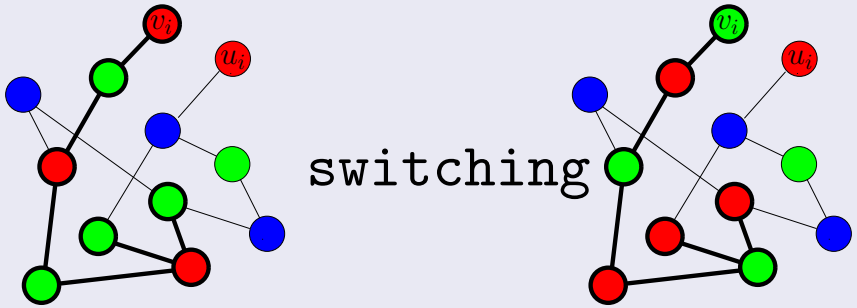
switching as a map

"switching from red to green"



switching as a map

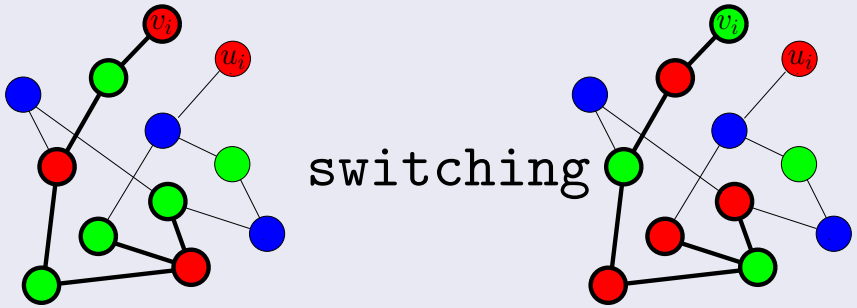
"switching from red to green"



Remarks

switching as a map

"switching from red to green"

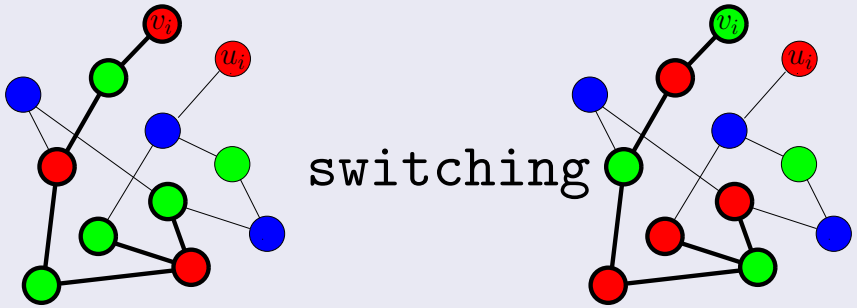


Remarks

- we want to implement a mapping from Ω_{rr} to Ω_{gr}

switching as a map

"switching from red to green"

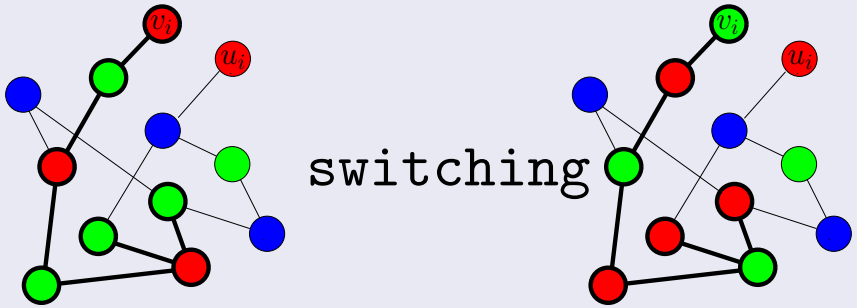


Remarks

- we want to implement a mapping from Ω_{rr} to Ω_{gr}
- we want the mapping to be as "close" to a bijection as possible

switching as a map

"switching from red to green"

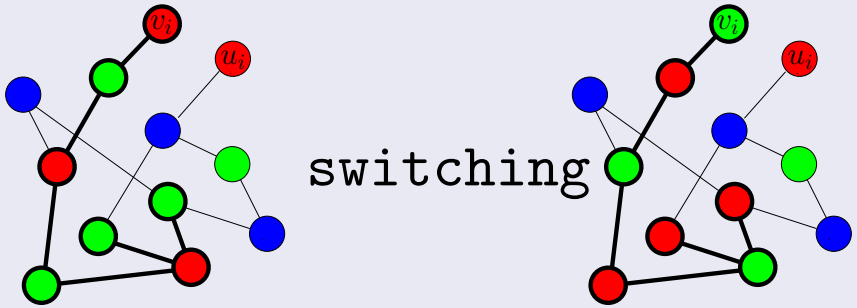


Remarks

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

switching as a map

"switching from red to green"



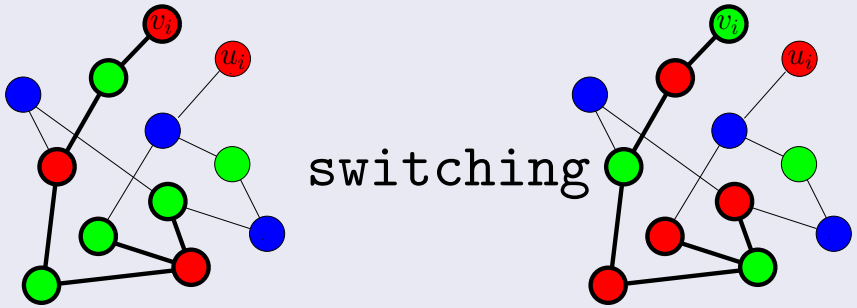
Remarks

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

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

switching as a map

"switching from red to green"



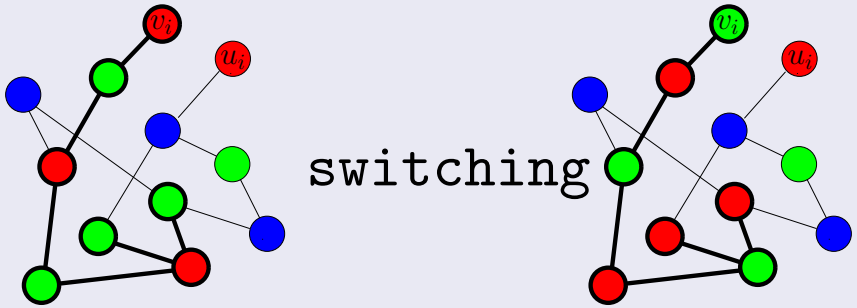
Remarks

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

    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 Ω_{rr} to Ω_{gr}
- we want the mapping to be as "close" to a bijection as possible
  - For a bijection $h : S_1 \to S_2$,

    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

The effect of pathological colouring

... reminder

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

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

The effect of pathological colouring

... reminder

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

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

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
- $Y_{i+1} = \texttt{Update}(X_i, v_i, u_i)$

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

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 $\mathbb{E}[\varrho_i(c, q)]$

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 $\mathbb{E}[\varrho_i(c, q)]$

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 $\mathbb{E}[\varrho_i(c, q)]$

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

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 $\mathbb{E}[\varrho_i(c, q)]$

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

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 $\mathbb{E}[\varrho_i(c, q)]$

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

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 $\mathbb{E}[\varrho_i(c, q)]$

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

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 I

The probability of a 2 coloured path I

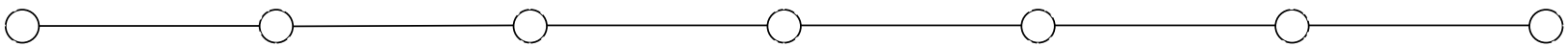
Graph first

The probability of a 2 coloured path I

Graph first

The probability of a 2 coloured path I

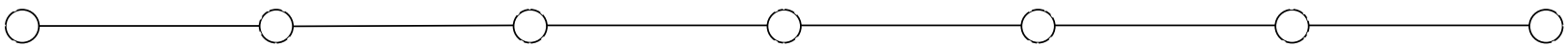
Graph first



The probability of a 2 coloured path I

Graph first

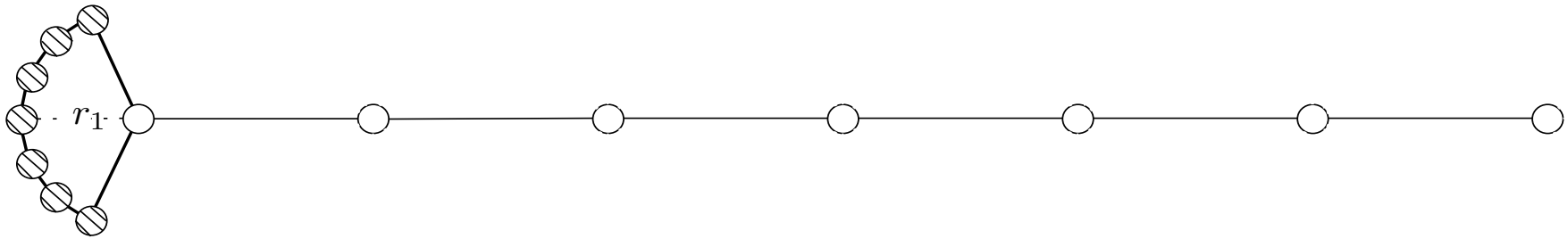
- reveal a neighborhood around each vertex in a BFS manner



The probability of a 2 coloured path I

Graph first

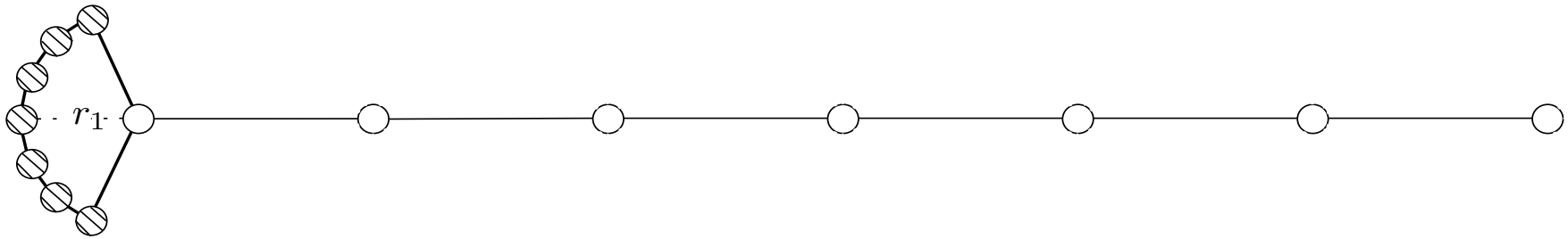
- reveal a neighborhood around each vertex in a BFS manner



The probability of a 2 coloured path I

Graph first

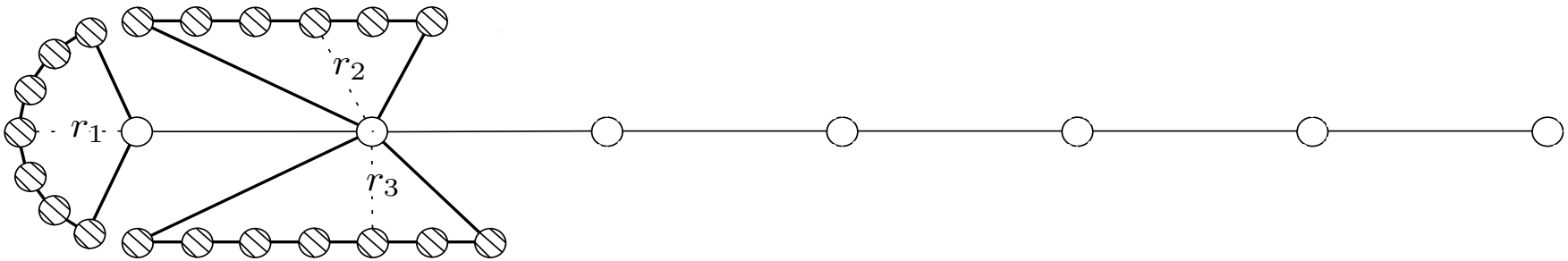
- reveal a neighborhood around each vertex in a BFS manner
  - constant number of neighbours
  - everything is within radius r'



The probability of a 2 coloured path I

Graph first

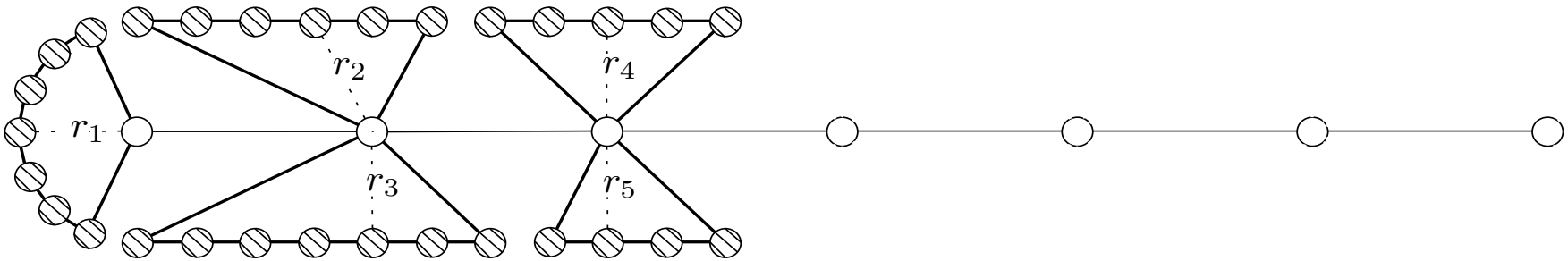
- reveal a neighborhood around each vertex in a BFS manner
  - constant number of neighbours
  - everything is within radius r'



The probability of a 2 coloured path I

Graph first

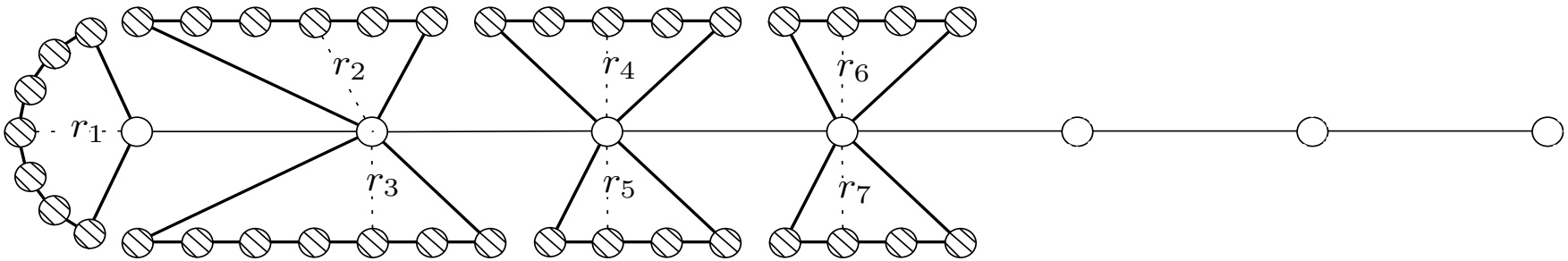
- reveal a neighborhood around each vertex in a BFS manner
  - constant number of neighbours
  - everything is within radius r'



The probability of a 2 coloured path I

Graph first

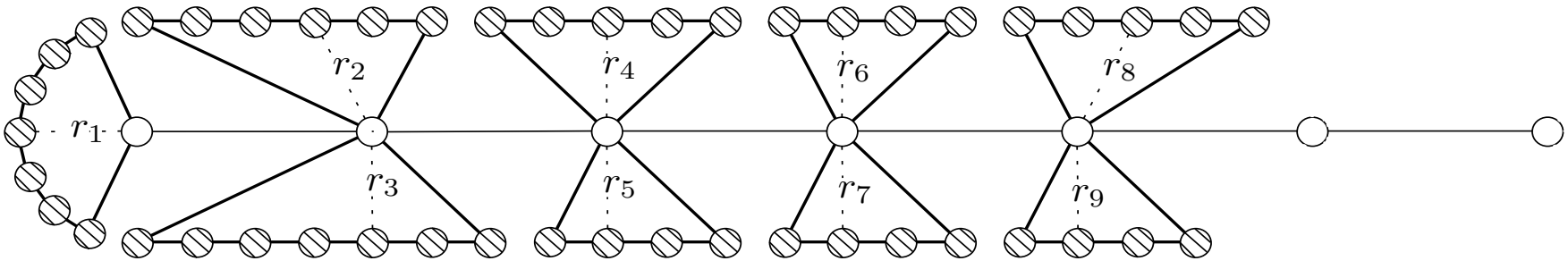
- reveal a neighborhood around each vertex in a BFS manner
  - constant number of neighbours
  - everything is within radius r'



The probability of a 2 coloured path I

Graph first

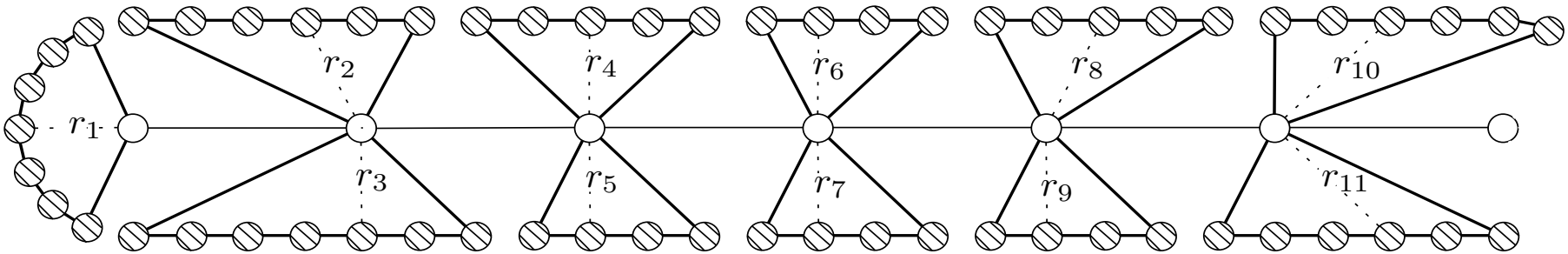
- reveal a neighborhood around each vertex in a BFS manner
  - constant number of neighbours
  - everything is within radius r'



The probability of a 2 coloured path I

Graph first

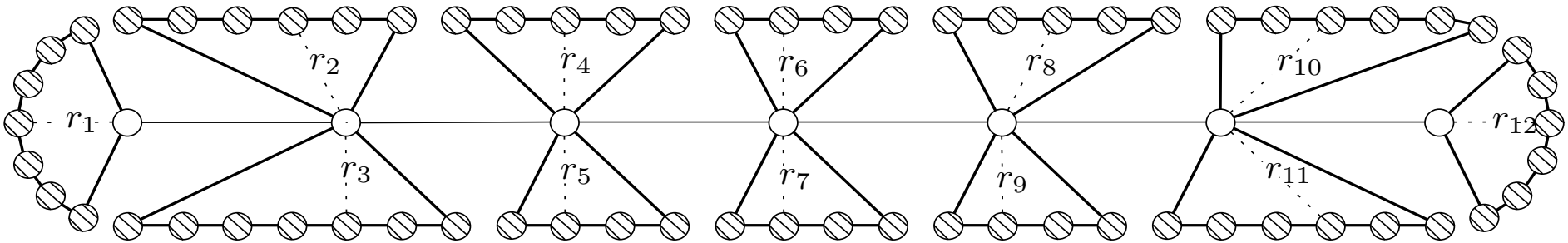
- reveal a neighborhood around each vertex in a BFS manner
  - constant number of neighbours
  - everything is within radius r'



The probability of a 2 coloured path I

Graph first

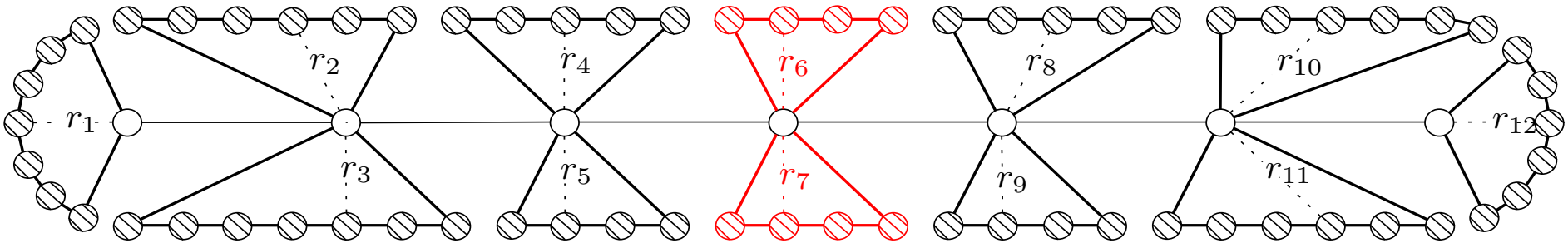
- reveal a neighborhood around each vertex in a BFS manner
  - constant number of neighbours
  - everything is within radius r'



The probability of a 2 coloured path I

Graph first

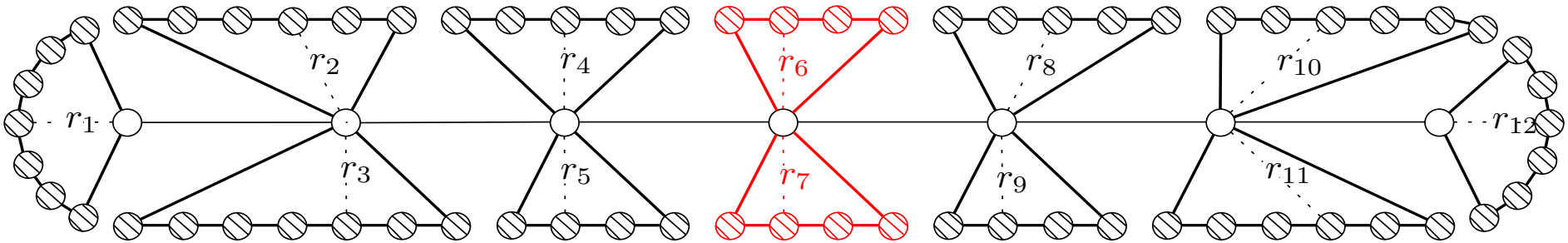
- reveal a neighborhood around each vertex in a BFS manner
  - constant number of neighbours
  - everything is within radius r'



The probability of a 2 coloured path I

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



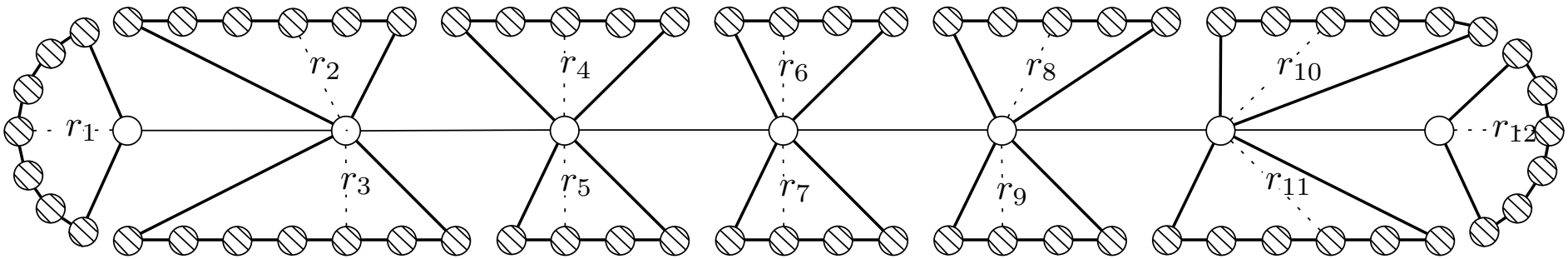
The probability of a 2 coloured path II

The probability of a 2 coloured path II

The random colouring part

The probability of a 2 coloured path II

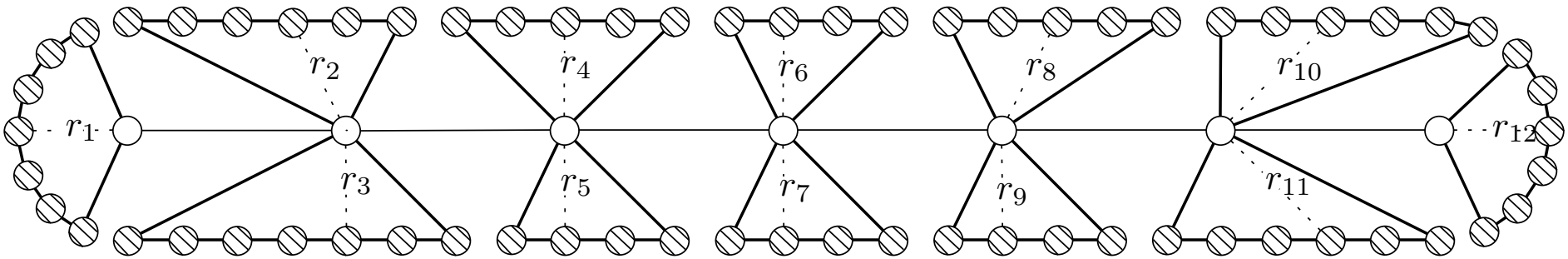
The random colouring part



The probability of a 2 coloured path II

The random colouring part

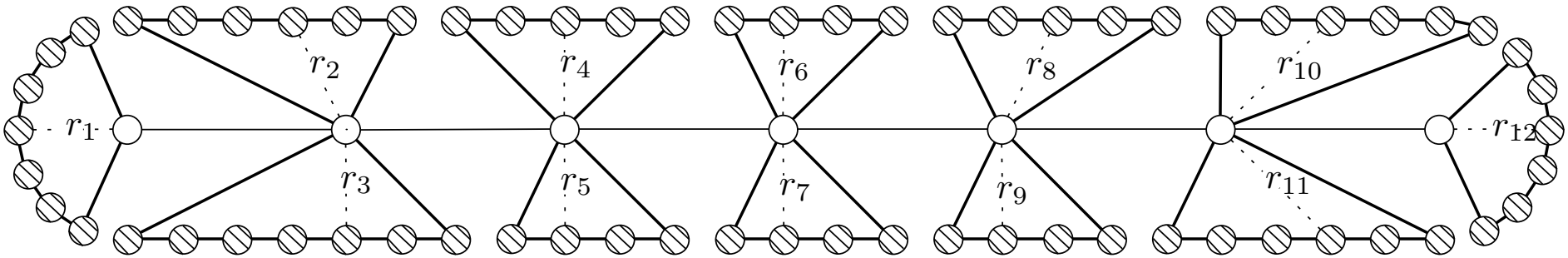
- ideally we consider a convex combination of boundary conditions



The probability of a 2 coloured path II

The random colouring part

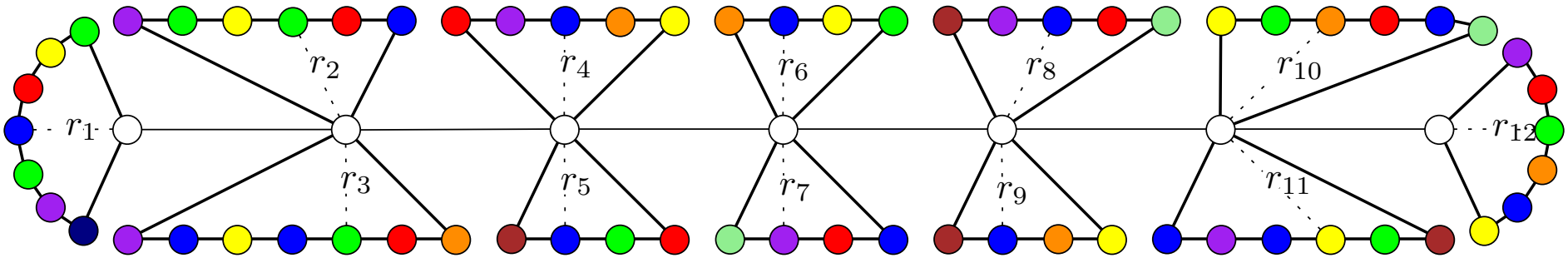
- ideally we consider a convex combination of boundary conditions
- instead we consider a worst case boundary condition



The probability of a 2 coloured path II

The random colouring part

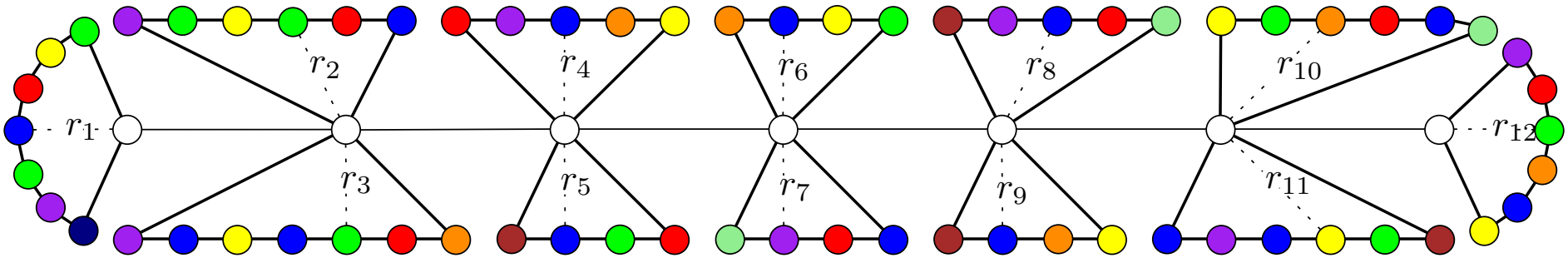
- ideally we consider a convex combination of boundary conditions
- instead we consider a worst case boundary condition



The probability of a 2 coloured path II

The random colouring part

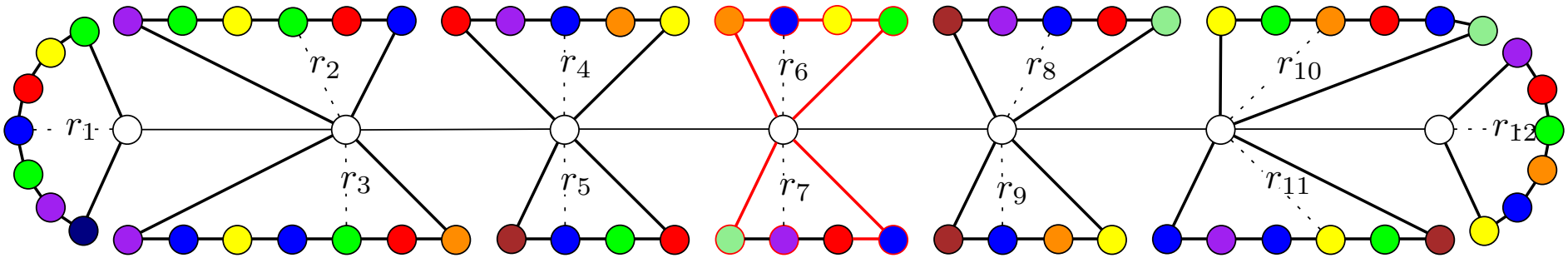
- ideally we consider a convex combination of boundary conditions
- instead we consider a worst case boundary condition
- the probability of each vertex to take on the "appropriate" colour mainly depends on its "immediate neighbourhood"



The probability of a 2 coloured path II

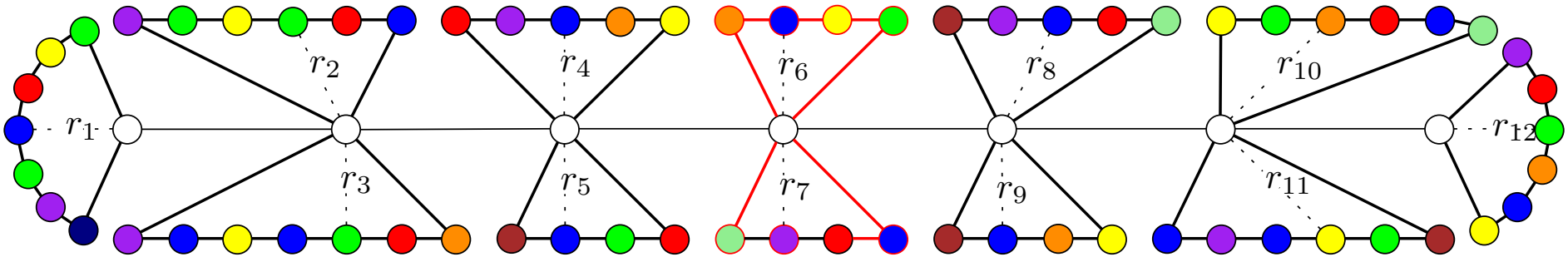
The random colouring part

- 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"



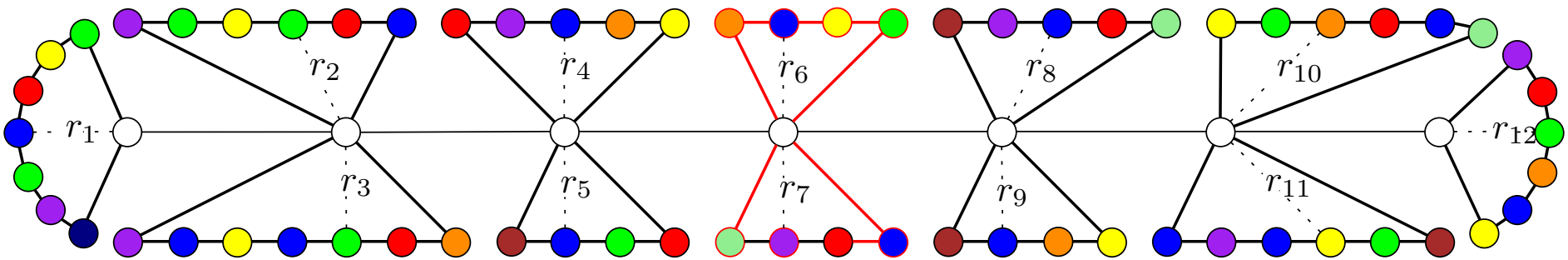
Structure around the path

Structure around the path



Structure around the path

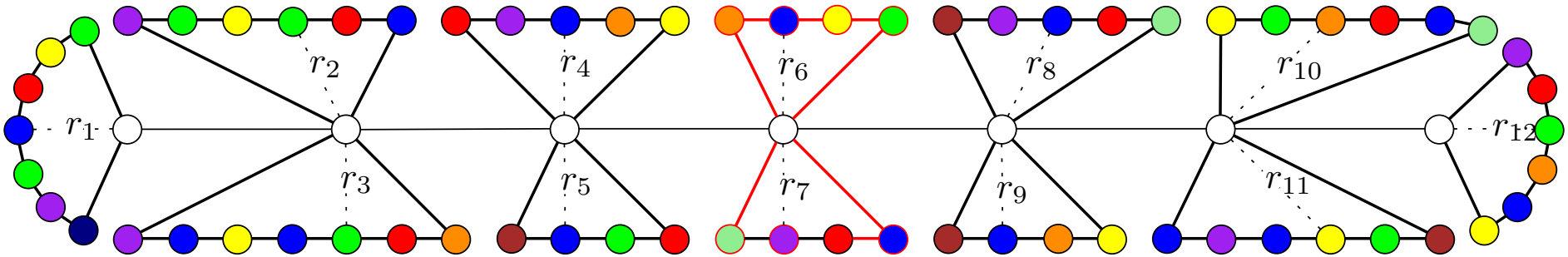
Good Vs Bad Neighbourhoods



Structure around the path

Good Vs Bad Neighbourhoods

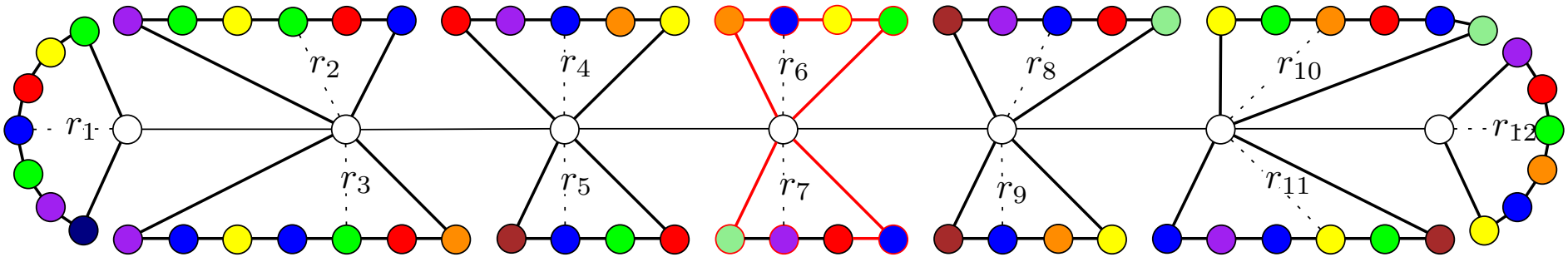
- Good
- Bad



Structure around the path

Good Vs Bad Neighbourhoods

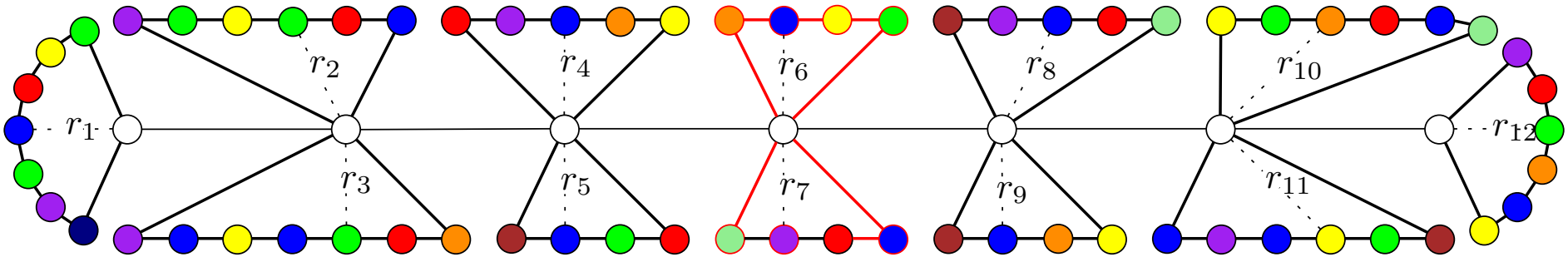
- Good
  - tree of height at most r'
  - maximum degree $\leq (1 + \epsilon/2)d$
  - does not intersect with other neighbourhoods
- Bad



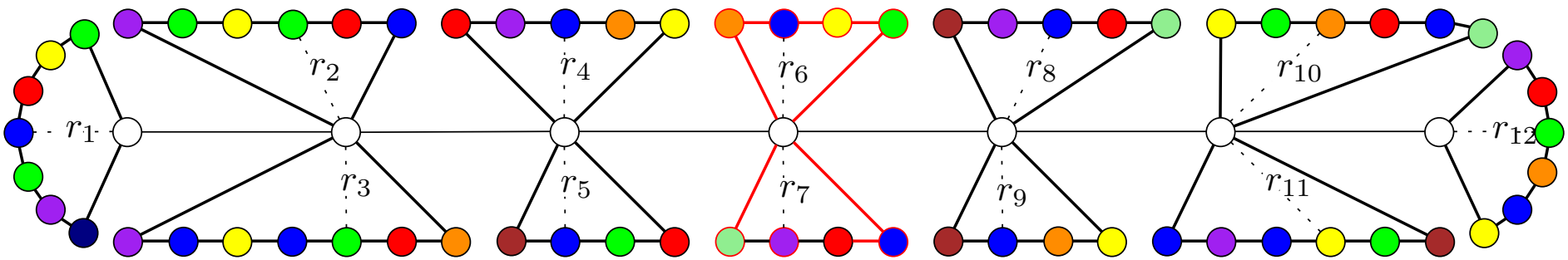
Structure around the path

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

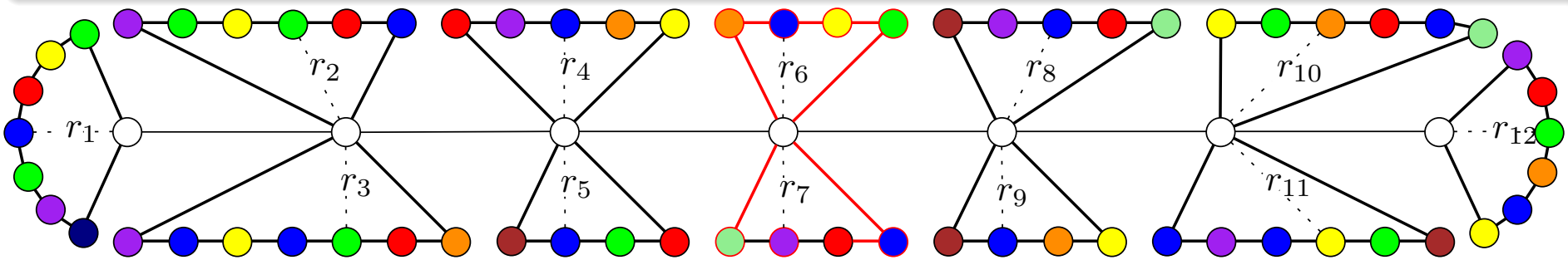


Effect of neighbour's structure



Effect of neighbour's structure

Cases



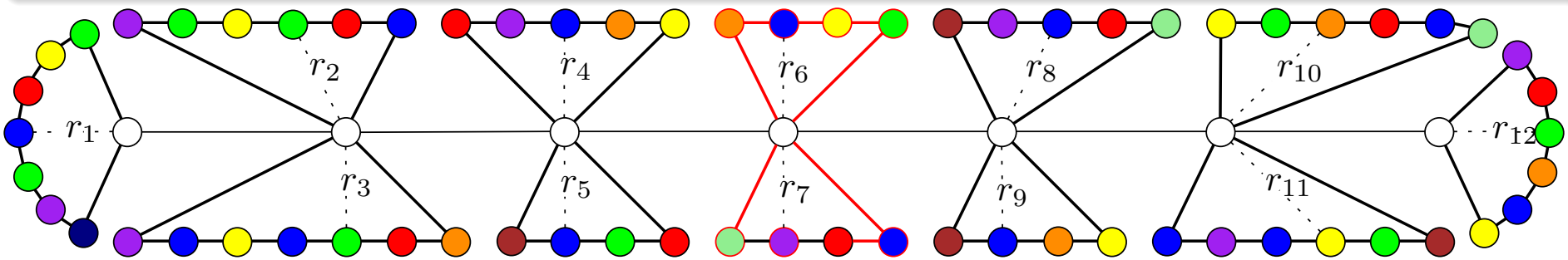
Effect of neighbour's structure

Cases

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

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

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



Effect of neighbour's structure

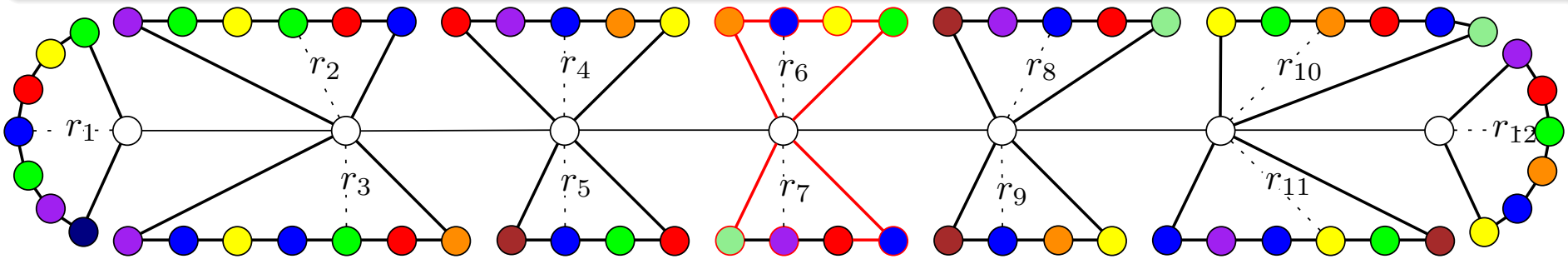
Cases

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

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

  where $f_{\epsilon,r'} \to 0$ as r' grows.
- if the neighbourhood is "Bad", then

$$\Pr[v \text{ is coloured } q | \text{colouring of boundry}] \leq 1$$



Then we get that ...

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

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 μ, μ' 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

Conclusions

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

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

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 $k < d$

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 ig k-colourings of $G(n, d/n)$ "looks different" ...

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 ig 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 ig 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!!!