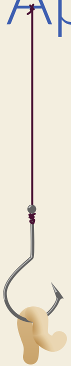
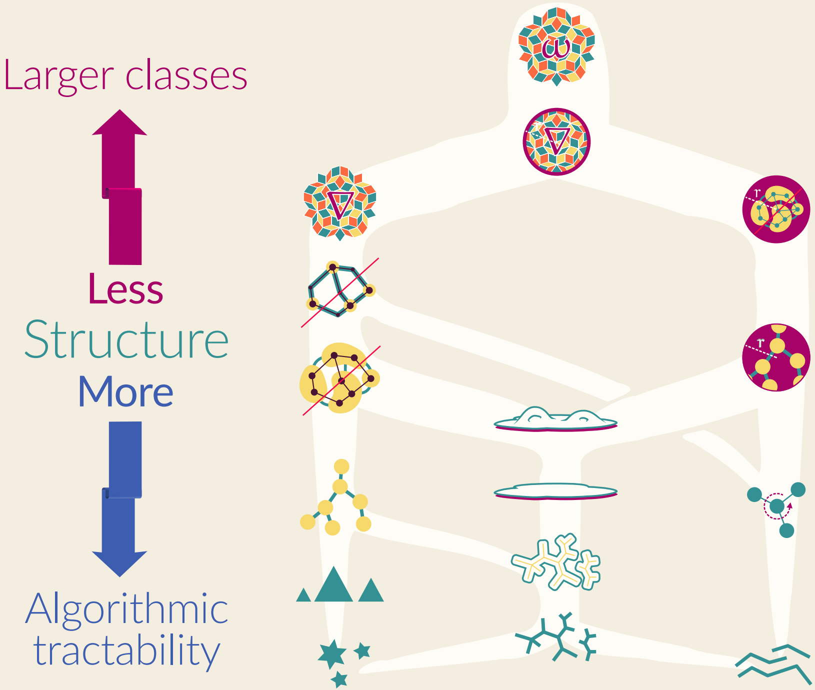
Complex networks & sparsity
Part III: Application



Felix Reidl
Blair D. Sullivan

DOCCOURSE '18


Larger classes
Less
Structure
More
Algorithmic tractability
ω
∇
r
r

Structural sparseness

A **graph measure** is an isomorphism-invariant function that maps graphs to $\mathbb{R}^+$.

e.g. density, average degree, clique number, degeneracy treewidth, etc.

A **parameterised graph measure** is a family of graph measures $(f_r)_{r\in\mathbb{N}_0}$.

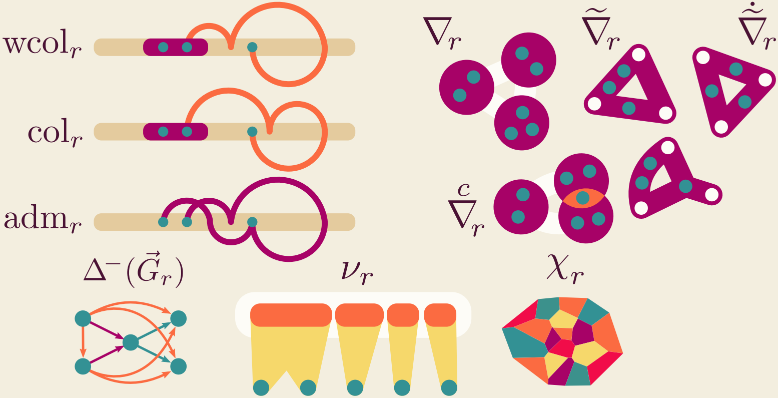
A graph class $\mathcal{G}$ is f_r**-bounded** if there exists g s.t.

$$f_r(\mathcal{G}) = \sup_{G\in\mathcal{G}} f_r(G) \leqslant g(r) \quad \text{for all } r.$$

Bounded expansion

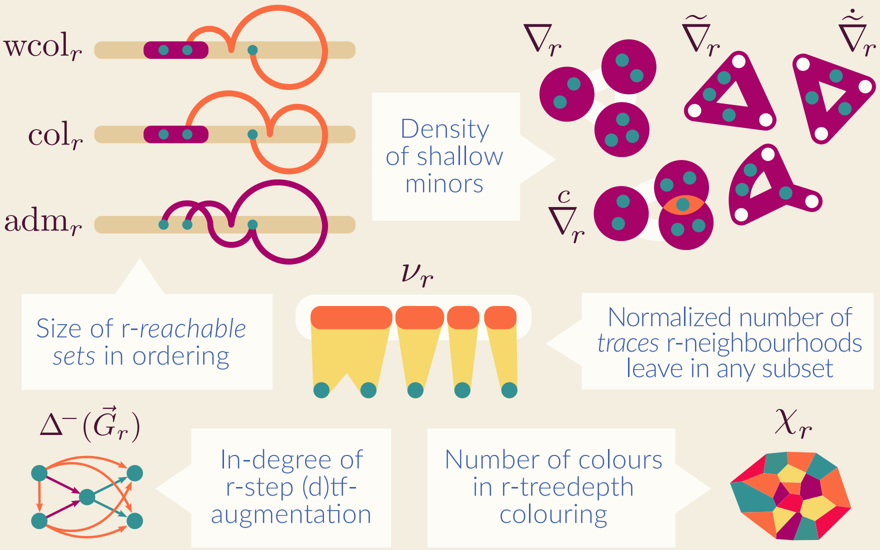
Jarik & Patrice:

Many notions of f_r-boundedness are equivalent!



Nešetřil J, Ossona de Mendez P. **Sparsity**.
Algorithms and Combinatorics. 2012;28.

Bounded expansion



Nešetřil J, Ossona de Mendez P. **Sparsity**.
Algorithms and Combinatorics. 2012;28.

Close-to-Closeness Centralities

$$C(v)$$

Closeness	$\left(\sum_{u \in G} \mathrm{dist}(u,v)\right)^{-1}$
Harmonic	$\sum_{u \in G} \mathrm{dist}(u,v)^{-1}$
Lin's index	$\dfrac{\lvert\{u \mid \mathrm{dist}(u,v) < \infty\}\rvert^2}{\sum_{\mathrm{dist}(u,v)<\infty} \mathrm{dist}(u,v)}$

Close-to-Closeness Centralities

$C(v)$

Closeness	$\left(\sum_{u \in G} \mathrm{dist}(u,v) \right)^{-1}$
Harmonic	$\sum_{u \in G} \mathrm{dist}(u,v)^{-1}$
Lin's index	$\dfrac{\lvert \{u \mid \mathrm{dist}(u,v) < \infty\} \rvert^2}{\sum_{\mathrm{dist}(u,v)<\infty} \mathrm{dist}(u,v)}$

All three measures can be computed quickly if we know $|N^d(v)|$ for $1 \leqslant d \leqslant \mathrm{rad}(G)$.

Close-to-Closeness Centralities

$C(v)$

Closeness	$\left(\sum_{u \in G} \operatorname{dist}(u,v)\right)^{-1}$
Harmonic	$\sum_{u \in G} \operatorname{dist}(u,v)^{-1}$
Lin's index	$\dfrac{\lvert\{u \mid \operatorname{dist}(u,v) < \infty\}\rvert^2}{\sum_{\operatorname{dist}(u,v)<\infty} \operatorname{dist}(u,v)}$

All three measures can be computed quickly if we know $|N^d(v)|$ for $1 \leqslant d \leqslant \operatorname{rad}(G)$.

Can we compute this quickly in sparse graphs?

Close-to-Closeness Centralities

	$C(v)$	r-Local version
Closeness	$\left(\sum_{u \in G} \mathrm{dist}(u,v)\right)^{-1}$	$\left(\sum_{u \in N^r[v]} \mathrm{dist}(v,u)\right)^{-1}$
Harmonic	$\sum_{u \in G} \mathrm{dist}(u,v)^{-1}$	$\sum_{u \in N^r[v]} \mathrm{dist}(v,u)^{-1}$
Lin's index	$\dfrac{\lvert\{u \mid \mathrm{dist}(u,v) < \infty\}\rvert^2}{\sum_{\mathrm{dist}(u,v)<\infty} \mathrm{dist}(u,v)}$	$\dfrac{\lvert N^r[v]\rvert^2}{\sum_{u \in N^r[v]} \mathrm{dist}(v,u)}$

All three measures can be computed quickly if we know $|N^d(v)|$ for $1 \leqslant d \leqslant r$.

Can we compute this quickly in sparse graphs?

Counting neighbourhood sizes

For all these centrality measures, we need to compute the size of *distance r-neighbourhoods* around each vertex.

$C(v)$	r-Local version
$\left(\sum_{u \in G} \operatorname{dist}(u,v)\right)^{-1}$	$\left(\sum_{u \in N^r[v]} \operatorname{dist}(v,u)\right)^{-1}$
$\sum_{u \in G} \operatorname{dist}(u,v)^{-1}$	$\sum_{u \in N^r[v]} \operatorname{dist}(v,u)^{-1}$
$\dfrac{\lvert\{u \mid \operatorname{dist}(u,v) < \infty\}\rvert^2}{\sum_{\operatorname{dist}(u,v)<\infty} \operatorname{dist}(u,v)}$	$\dfrac{\lvert N^r[v]\rvert^2}{\sum_{u \in N^r[v]} \operatorname{dist}(v,u)}$

Counting neighbourhood sizes

For all these centrality measures, we need to compute the size of *distance r-neighbourhoods* around each vertex.

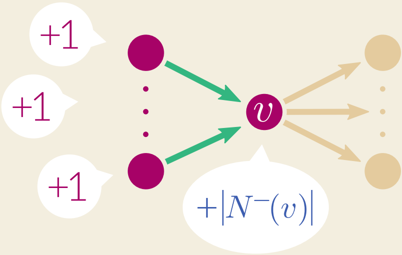
This needs quadratic time in general! Can we do better in sparse graphs?

$C(v)$	r-Local version
$\left(\sum_{u \in G} \mathrm{dist}(u,v)\right)^{-1}$	$\left(\sum_{u \in N^r[v]} \mathrm{dist}(v,u)\right)^{-1}$
$\sum_{u \in G} \mathrm{dist}(u,v)^{-1}$	$\sum_{u \in N^r[v]} \mathrm{dist}(v,u)^{-1}$
$\dfrac{\lvert\{u \mid \mathrm{dist}(u,v) < \infty\}\rvert^2}{\sum_{\mathrm{dist}(u,v)<\infty} \mathrm{dist}(u,v)}$	$\dfrac{\lvert N^r[v]\rvert^2}{\sum_{u \in N^r[v]} \mathrm{dist}(v,u)}$

Warm-up: Counting with degeneracy

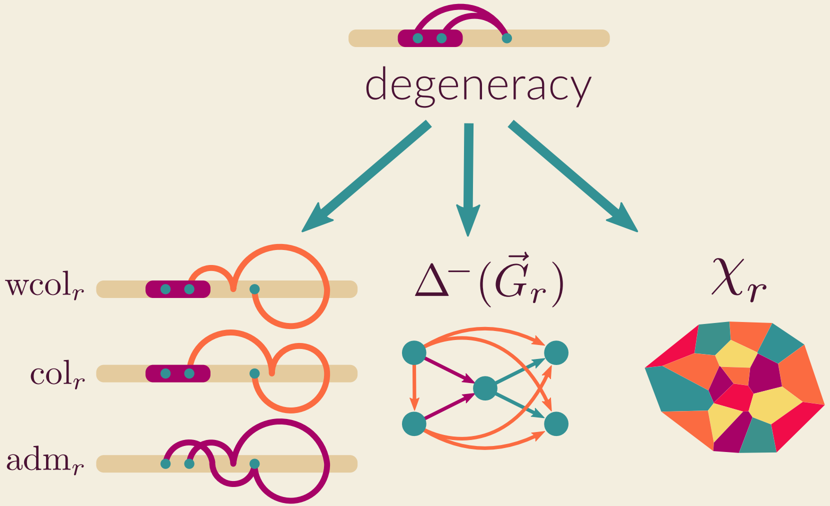
Let G be (d-1)-degenerate.

1. Compute orientation $\vec{G}$ with $\Delta^-(\vec{G}) \leqslant d$ in linear time.
2. Initialize counter $C[v] = 0$ for all $v \in G$.
3. For every $v \in G$, increment $C[v]$ *and* $C[u]$ for every in-neighbour $u \in N^-(v)$.



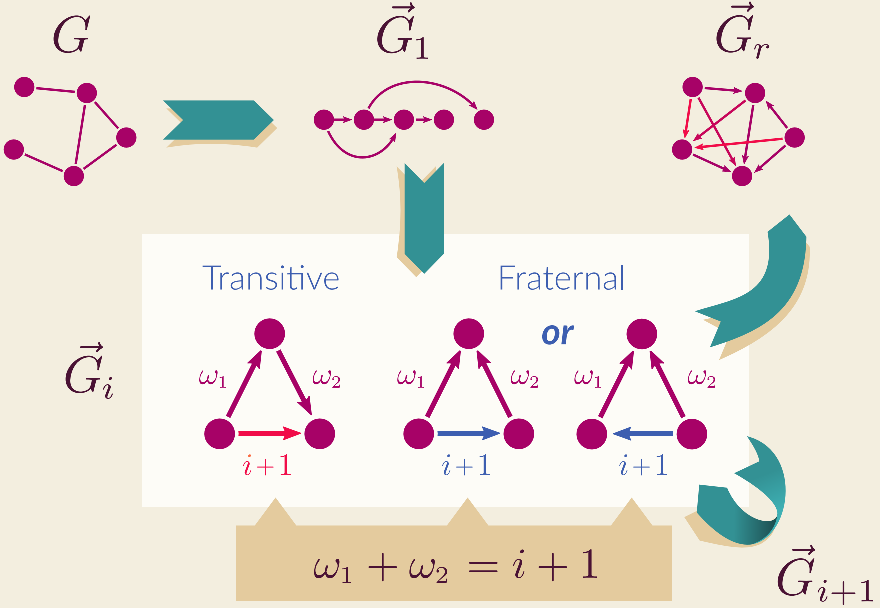
Generalizing degeneracy

‘Lifting’ degeneracy

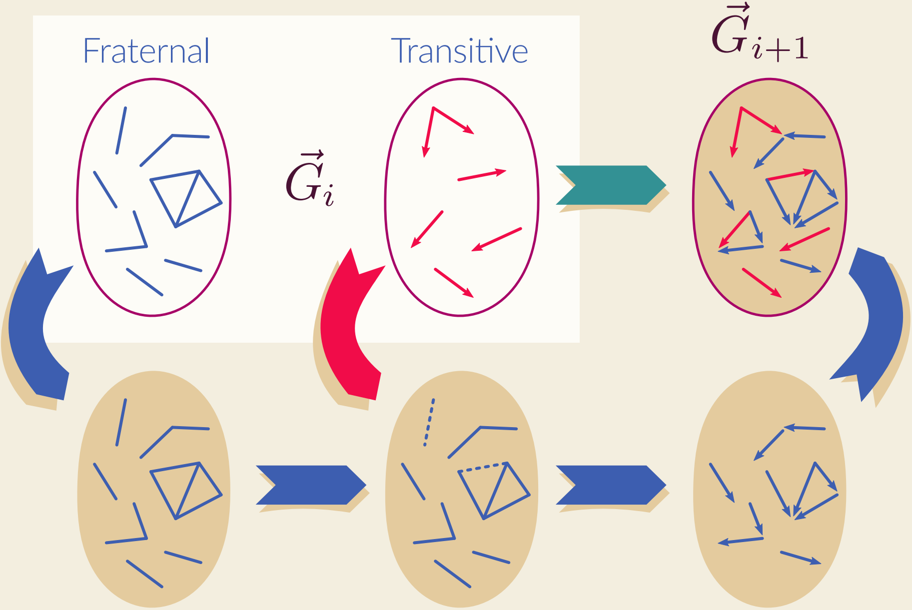


Pick your poison

dtf-augmentations

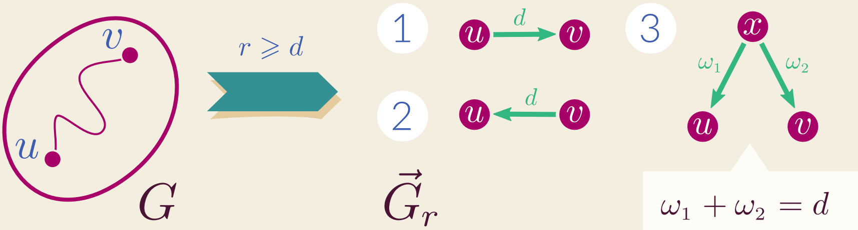


The details



Distances under dtf-augmentations

Let u and v be at distance d in G :



Pairs at distance at most r in the original graph have distance at most two in the r[th] augmentation.

B.E. & dtf-augmentations

There exist two (horrible) polynomials P and Q such that:

$$\chi_r(G) \leqslant P\big(\tilde{\nabla}_{(2\log r)^r}(G)\big)$$

$$\Delta^-(\vec{G}_r) \leqslant Q\big(\tilde{\nabla}_r(G)\Delta^-(\vec{G}_1)\big)$$

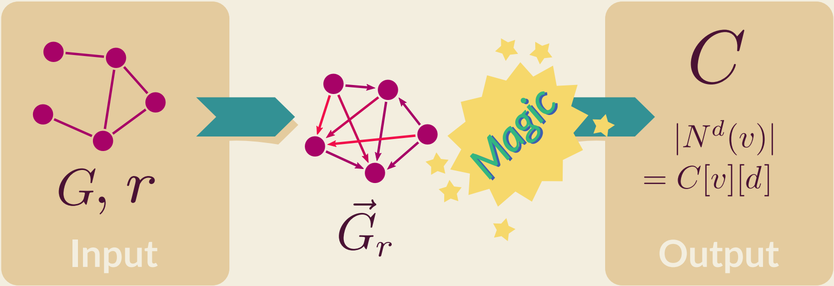
A graph class has bounded expansion iff it is $\Delta^-(\vec{G}_r)$-bounded.

We can compute dtf-augmenations in linear time (in bounded expansion classes)

Algorithm

Degeneracy to dtf-augmentations

Thm. Given a graph G and an integer r, we can compute the size of $|N^d(v)|$ for all $v \in G$ and $1 \leqslant d \leqslant r$ in total time $O(2^{\Delta^-(\vec{G}_r)}n)$.



Counting using dtf-augmentations

We compute the size of the r[th] nbhds:

1. Compute dtf-augm. $\vec{G}_r$ with small $\Delta^-(\vec{G}_r)$ in linear time.

Counting using dtf-augmentations

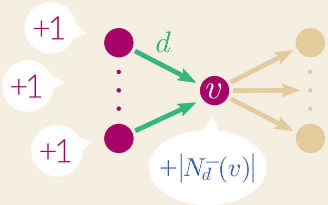
We compute the size of the r[th] nbhds:

1. Compute dtf-augm. $\vec{G}_r$ with small $\Delta^-(\vec{G}_r)$ in linear time.
2. Initialize counter $C[v][d] = 0$ for all $v \in G$ and $d \leqslant r$.

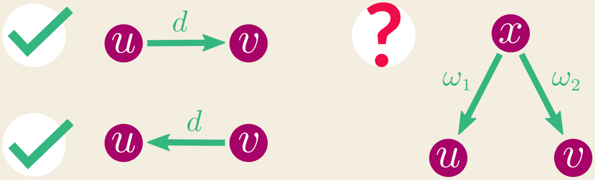
Counting using dtf-augmentations

We compute the size of the r^th nbhds:

1. Compute dtf-augm. $\vec{G}_r$ with small $\Delta^-(\vec{G}_r)$ in linear time.
2. Initialize counter $C[v][d] = 0$ for all $v \in G$ and $d \leqslant r$.
3. For every $v \in G$, increment $C[v][d]$ *and* $C[u][d]$ for every in-neighbour $u \in N^-_d(v)$.



Counting using dtf-augmentations



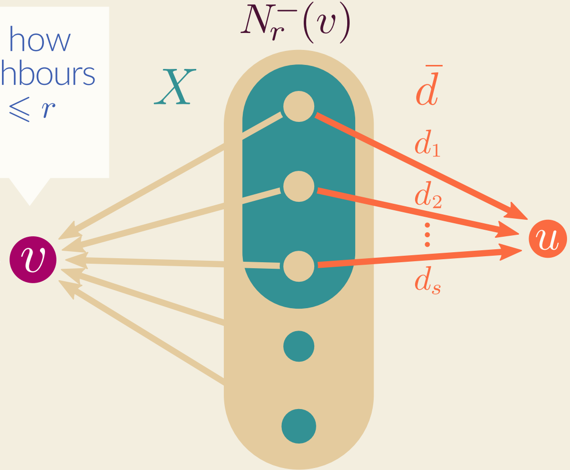
The counting so far takes care of the first two cases, but what about the *indirect* neighbours?

This is where the algorithm becomes **interesting**.

Counting using dtf-augmentations

v needs to know how many indirect neighbours at distance $2 \leqslant d \leqslant r$ there are.

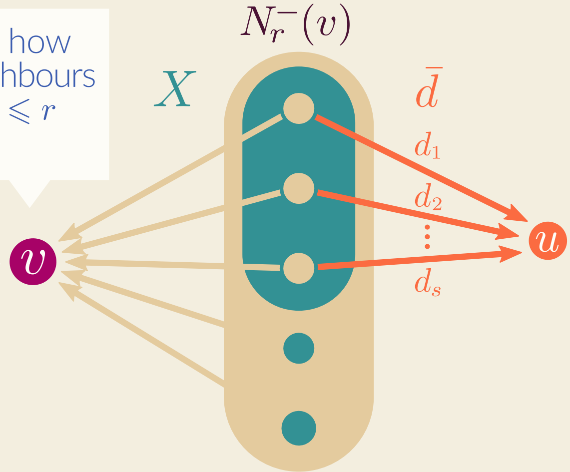
Indirect neighbours connect via $N_r^-(v)$.



Counting using dtf-augmentations

v needs to know how many indirect neighbours at distance $2 \leqslant d \leqslant r$ there are.

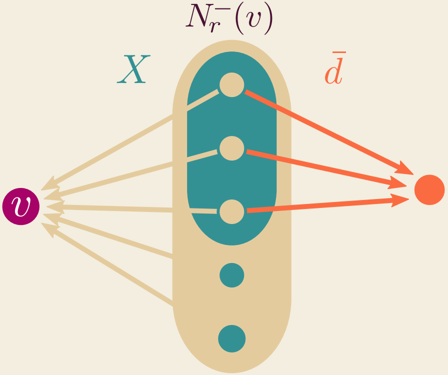
Indirect neighbours connect via $N_r^-(v)$.



We compute the distance between v,u as follows:

$$\mathrm{dist}(u, v) = \min\big(\mathrm{dist}(v, X) + \mathrm{dist}(u, X)\big)$$

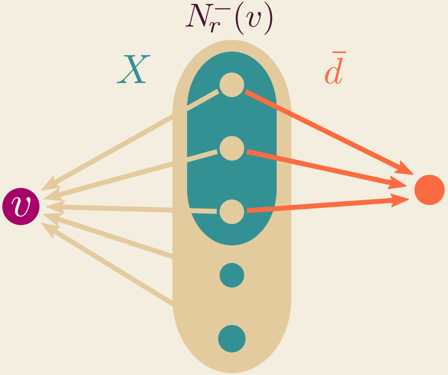
Counting using dtf-augmentations



We need to compute for every set $X \subseteq N_r^-(v)$ and every possible dist.-vector $\bar{d} \in [r]^{|X|}$ the number of vertices u such that:

1. $N_r^-(u) \cap N_r^-(v) = X$
2. $\mathrm{dist}(u, X) = \bar{d}$

Counting using dtf-augmentations



We need to compute for every set $X \subseteq N_r^-(v)$ and every possible dist.-vector $\bar{d} \in [r]^{|X|}$ the number of vertices u such that:

1. $N_r^-(u) \cap N_r^-(v) = X$
2. $\mathrm{dist}(u, X) = \bar{d}$

Let us call this number $c(v, X, \bar{d})$. Our first goal is to compute it for every vertex.

A data structure for $c(v, X, \bar{d})$

1. For every $v \in \vec{G}_r$, $X \subseteq N_r^-(v)$ and $\bar{d} \in [r]^{|X|}$, initialize $R[X][\bar{d}] = 0$.

A data structure for $c(v, X, \bar{d})$

1. For every $v \in \vec{G}_r$, $X \subseteq N_r^-(v)$ and $\bar{d} \in [r]^{|X|}$, initialize $R[X][\bar{d}] = 0$.

2. For every $v \in \vec{G}_r$, $X \subseteq N_r^-(v)$, increment
$$R[X][\mathrm{dist}(v, X)]$$
by one.

A data structure for $c(v, X, \bar{d})$

1. For every $v \in \vec{G}_r$, $X \subseteq N_r^-(v)$ and $\bar{d} \in [r]^{|X|}$, initialize $R[X][\bar{d}] = 0$.

2. For every $v \in \vec{G}_r$, $X \subseteq N_r^-(v)$, increment
$$R[X][\mathrm{dist}(v, X)]$$
by one.

Claim.

$$c(v, X, \bar{d}) = \sum_{X \subseteq Y \subseteq N_r^-(v)} (-1)^{|Y \setminus X|} \sum_{\bar{d}' : \bar{d}'|_X = \bar{d}} R[Y][\bar{d}'].$$

A data structure for $c(v, X, \bar{d})$

Claim.

$$c\left(v, X, \bar{d}\right) = \sum_{X \subseteq Y \subseteq N_r^-(v)} (-1)^{|Y \setminus X|} \sum_{\bar{d}' : \bar{d}'|_X = \bar{d}} R[Y][\bar{d}'].$$

Case 1.

Assume that u satisfies $\begin{cases} N_r^-(u) \cap N_r^-(v) = X \\ \operatorname{dist}(u, X) = d. \end{cases}$

A data structure for $c(v, X, \bar{d})$

Claim.

$$c\left(v, X, \bar{d}\right) = \sum_{X \subseteq Y \subseteq N_r^-(v)} (-1)^{|Y \setminus X|} \sum_{\bar{d}' : \bar{d}'|_X = \bar{d}} R[Y][\bar{d}'].$$

Case 1.

Assume that u satisfies $\begin{cases} N_r^-(u) \cap N_r^-(v) = X \\ \mathrm{dist}(u, X) = d. \end{cases}$

Then the above sum counts it exactly once, namely when $Y = X$ and $\bar{d}' = \bar{d}$, since it only contributes to $R[X][\bar{d}\,]$.

A data structure for $c(v, X, \bar{d})$

Claim.

$$c\left(v, X, \bar{d}\right) = \sum_{X \subseteq Y \subseteq N_r^-(v)} (-1)^{|Y \setminus X|} \sum_{\bar{d}' : \bar{d}'|_X = \bar{d}} R[Y][\bar{d}'].$$

Case 2.

Assume that u satisfies $\mathrm{dist}(u, X) \neq d$.

A data structure for $c(v, X, \bar{d})$

Claim.

$$c\left(v, X, \bar{d}\right) = \sum_{X \subseteq Y \subseteq N_r^-(v)} (-1)^{|Y \setminus X|} \sum_{\bar{d}' : \bar{d}'|_X = \bar{d}} R[Y][\bar{d}'].$$

Case 2.

Assume that u satisfies $\operatorname{dist}(u, X) \neq d$.

Then the above sum does not count it.

A data structure for $c(v, X, \bar{d})$

Claim.

$$c\left(v, X, \bar{d}\right) = \sum_{X \subseteq Y \subseteq N_r^-(v)} (-1)^{|Y \setminus X|} \sum_{\bar{d}' : \bar{d}'|_X = \bar{d}} R[Y][\bar{d}'].$$

Case 3.

Assume that u satisfies $\mathrm{dist}(u, X) = d$ but $N_r^-(u) \cap N_r^-(v) = Z$ where $X \subsetneq Z \subseteq N_r^-(v)$.

A data structure for $c(v, X, \bar{d})$

Claim.

$$c\left(v, X, \bar{d}\right) = \sum_{X \subseteq Y \subseteq N_r^-(v)} (-1)^{|Y \setminus X|} \sum_{\bar{d}' : \bar{d}'|_X = \bar{d}} R[Y][\bar{d}'].$$

Case 3.
Assume that u satisfies $\operatorname{dist}(u, X) = d$ but $N_r^-(u) \cap N_r^-(v) = Z$ where $X \subsetneq Z \subseteq N_r^-(v)$.
Then u contributes to the following terms:

$$\sum_{X \subseteq Y \subseteq Z} (-1)^{|Y \setminus X|} R[Y][\operatorname{dist}(u, Z)|_Y]$$

A data structure for $c(v, X, \bar{d})$

Claim.

$$c\left(v, X, \bar{d}\right) = \sum_{X \subseteq Y \subseteq N_r^-(v)} (-1)^{|Y \setminus X|} \sum_{\bar{d}' : \bar{d}'|_X = \bar{d}} R[Y][\bar{d}'].$$

Case 3.

Therefore the contribution of u cancels out!

$$\sum_{X \subseteq Y \subseteq Z} (-1)^{|Y \setminus X|} R[Y][\mathrm{dist}(u, Z)|_Y]$$

$$\sum_{X \subseteq Y \subseteq Z} (-1)^{|Y \setminus X|} = \sum_{0 \leqslant k \leqslant |Z \setminus X|} (-1)^k \binom{|Z \setminus X|}{k} = 0$$

Counting using dtf-augmentations

Given $c(v, \bullet, \bullet)$ we can now count the number of indirect neighbours of v. For every subset $X \subseteq N_r^-(v)$ and distance-vector $\bar{d} \in [r]^{|X|}$, apply the update:

$$C[v][\min(\bar{d} + \operatorname{dist}(v, X))] \mathrel{+}= c\,(v, X, \bar{d})$$

Counting using dtf-augmentations

Given $c(v, \bullet, \bullet)$ we can now count the number of indirect neighbours of v. For every subset $X \subseteq N_r^-(v)$ and distance-vector $\bar{d} \in [r]^{|X|}$, apply the update:

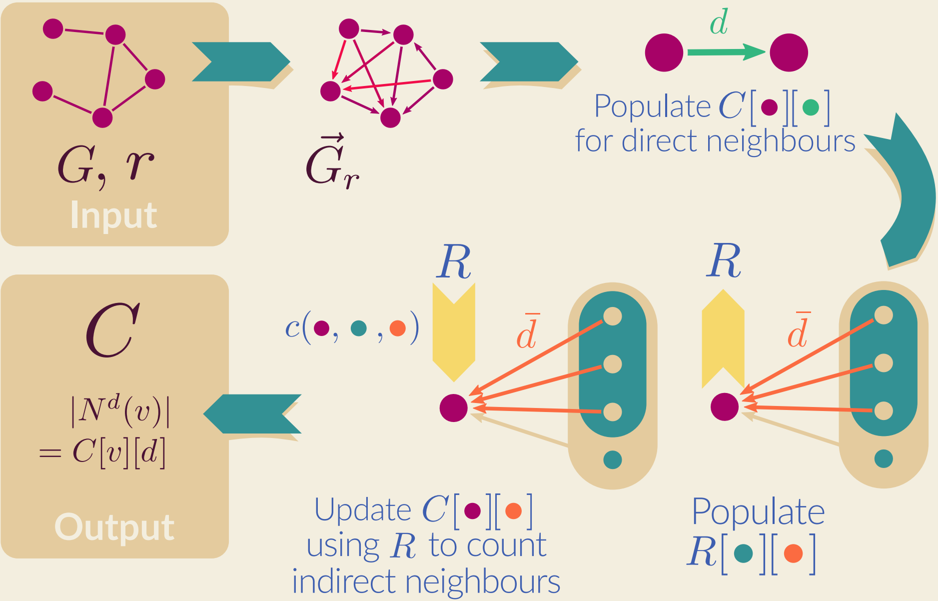
$$C[v][\min(\bar{d} + \mathrm{dist}(v, X))] \;\mathrel{+}= \; c\,(v, X, \bar{d})$$

Since the above counts v as a neighbour of itself, we apply the following correction:

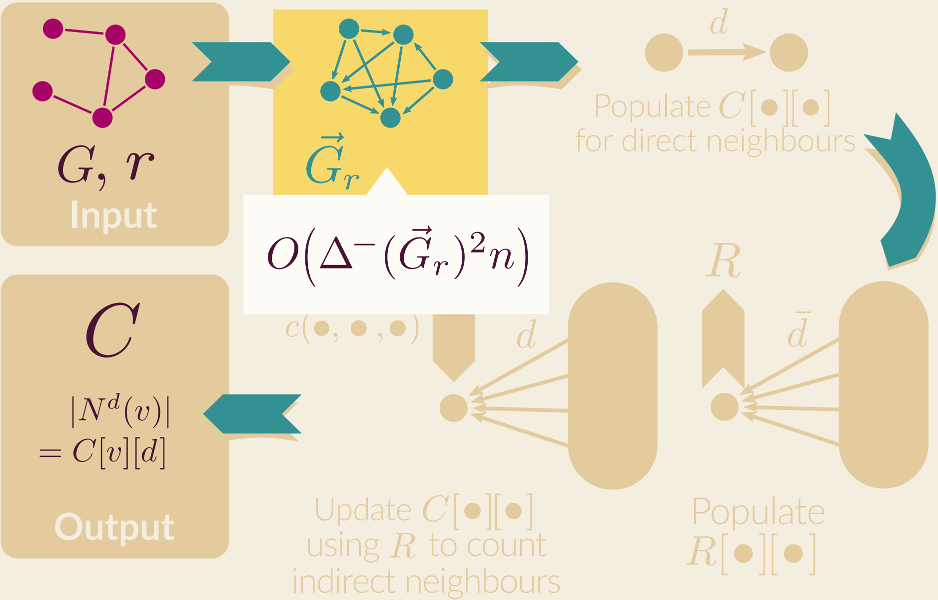
$$C[v][\min(\mathrm{dist}(v, X) + \mathrm{dist}(v, X))] \mathrel{-}= \; 1$$

There are a few more corrections concerning direct neighbours, see paper.

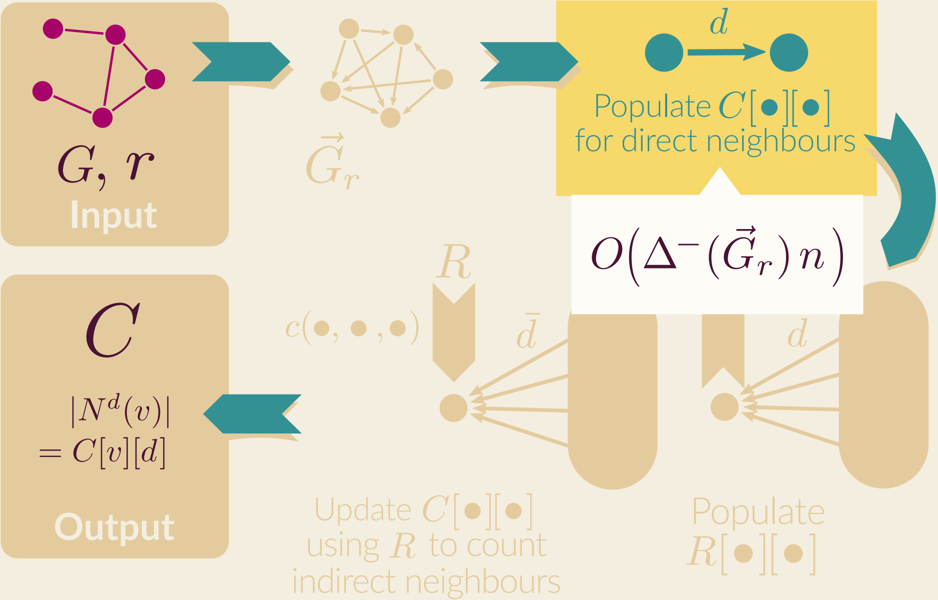
Counting using dtf-augmentations



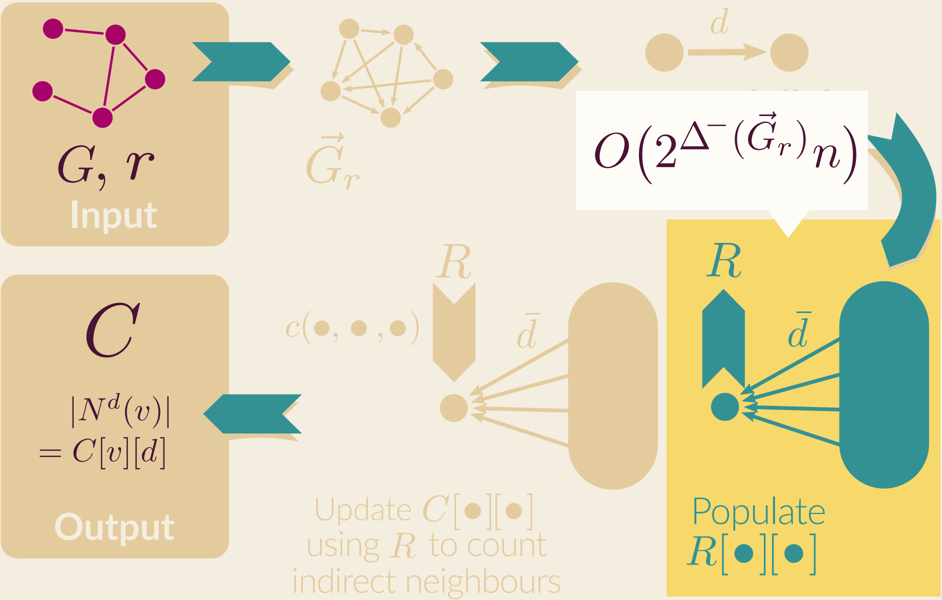
Counting using dtf-augmentations



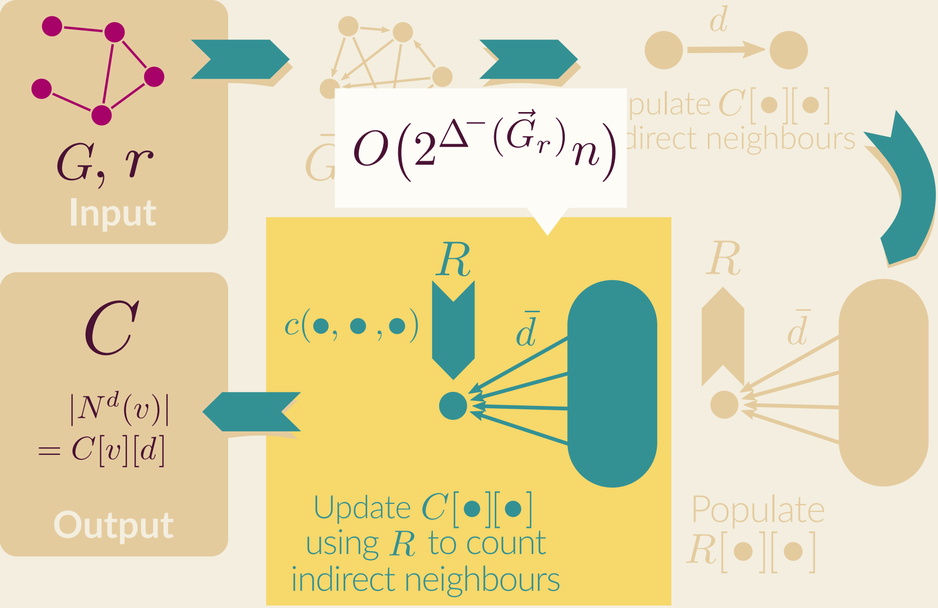
Counting using dtf-augmentations



Counting using dtf-augmentations



Counting using dtf-augmentations



Counting using dtf-augmentations

Thm. Given a graph G and an integer r, we can compute the size of $|N^d(v)|$ for all $v \in G$ and $1 \leqslant d \leqslant r$ in total time $O\big(2^{\Delta^-(\vec{G}_r)}n\big)$.

Counting using dtf-augmentations

Thm. Given a graph G and an integer r, we can compute the size of $|N^d(v)|$ for all $v \in G$ and $1 \leqslant d \leqslant r$ in total time $O(2^{\Delta^-(\vec{G}_r)}n)$.

- Exponential vs quadratic?
- Does not scale to nowhere dense graphs!

$2^{n^\varepsilon}n$

Counting using dtf-augmentations

Thm. Given a graph G and an integer r, we can compute the size of $|N^d(v)|$ for all $v \in G$ and $1 \leqslant d \leqslant r$ in total time $O(2^{\Delta^-(\vec{G}_r)} n)$.

- Exponential vs quadratic?
- Does not scale to nowhere dense graphs!

$2^{n^{\varepsilon}} n$

Can we do ***better***?

Some Bad News

Can we do better?

Closed 2-Neighbourhood Sizes

Input: A graph G.

Output: $|N^2[v]|$ for every $v \in G$.

Can we do better?

CLOSED 2-NEIGHBOURHOOD SIZES

Input: A graph G.
Output: $|N^2[v]|$ for every $v \in G$.

Thm. Unless SETH fails, 2-CNBS cannot be solved in time

1. $O(|G|^{2-\varepsilon})$
2. $O(2^{o(\Delta^-(\vec{G}_2))} n^{2-\varepsilon})$

Gutin G, Mertzios GB, Reidl F.
Lower and Upper Bound for Computing the Size of All Second Neighbourhoods.
arXiv preprint arXiv:1805.01684. 2018 May 4

Lower bound tool: SETH

r-CNF SAT

Input: A CNF formula ϕ on n variables and m clauses of size $\leqslant r$.

Problem: Is ϕ satisfiable?

Strong exponential time hypothesis

For every $\varepsilon > 0$ there exists an r_ε such that r_ε-CNF SAT cannot be solved in time $O(2^{\varepsilon n})$.

Lower bound tool: SETH

r-CNF SAT

Input: A CNF formula ϕ on n variables and m clauses of size $\leqslant r$.

Problem: Is ϕ satisfiable?

Strong exponential time hypothesis

For every $\varepsilon > 0$ there exists an r_ε such that r_ε-CNF SAT cannot be solved in time $O(2^{\varepsilon n})$.

Can we do better?

CLOSED 2-NEIGHBOURHOOD SIZES

Input: A graph G.

Output: $|N^2[v]|$ for every $v \in G$.

Thm. Unless SETH fails, 2-CNBS cannot be solved in time

1. $O(|G|^{2-\varepsilon})$
2. $O(2^{o(\Delta^-(\vec{G}_2))} n^{2-\varepsilon})$

Gutin G, Mertzios GB, Reidl F.
Lower and Upper Bound for Computing the Size of All Second Neighbourhoods.
arXiv preprint arXiv:1805.01684. 2018 May 4

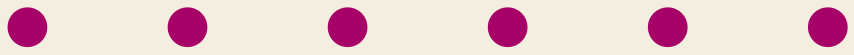
Reduction: $\mathrm{SAT} \leqslant \text{2-CNBS}$

We begin with a SAT formula on n variables with m clauses: $\phi(x_1, \ldots, x_n) = C_1 \wedge \ldots \wedge C_m$

Using the sparsification lemma, we can assume in the following that $m = O(n)$.

Reduction: $\mathrm{SAT} \leqslant 2\text{-CNBS}$

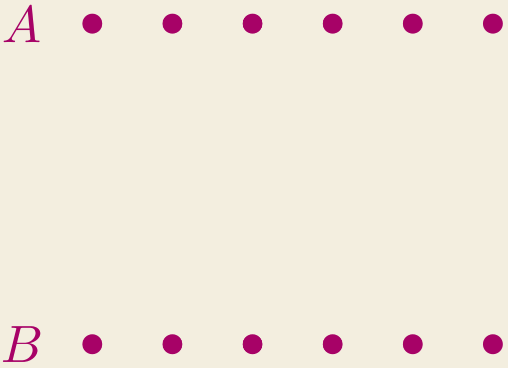
We begin with a SAT formula on n variables with m clauses: $\phi(x_1, \ldots, x_n) = C_1 \wedge \ldots \wedge C_m$

A 

Let A contain all $2^{n/2}$ assignments of the variables $x_1, \ldots, x_{n/2}$

Reduction: $\mathrm{SAT} \leqslant \text{2-CNBS}$

We begin with a SAT formula on n variables with m clauses: $\phi(x_1, \ldots, x_n) = C_1 \wedge \ldots \wedge C_m$

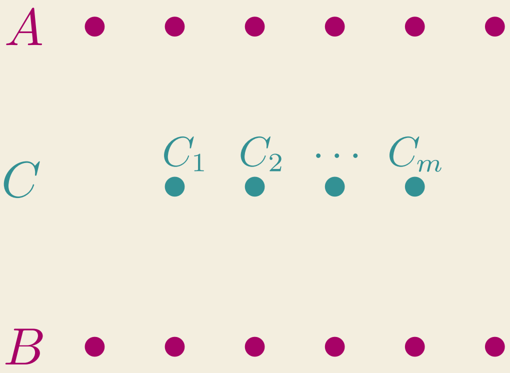


Let A contain all $2^{n/2}$ assignments of the variables $x_1, \ldots, x_{n/2}$

Let B contain all $2^{n/2}$ assignments of the variables $x_{n/2+1}, \ldots, x_n$

Reduction: $\mathrm{SAT} \leqslant \text{2-CNBS}$

We begin with a SAT formula on n variables with m clauses: $\phi(x_1, \ldots, x_n) = C_1 \land \ldots \land C_m$



Let A contain all $2^{n/2}$ assignments of the variables $x_1, \ldots, x_{n/2}$

Let C contain all clauses of ϕ

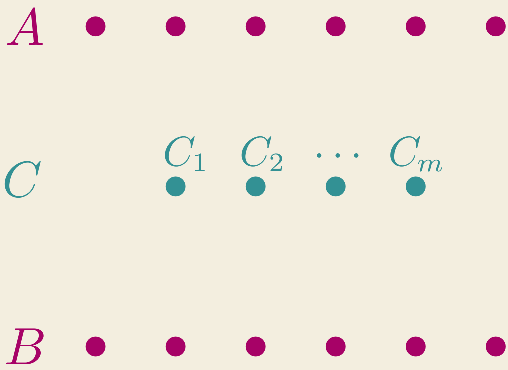
Let B contain all $2^{n/2}$ assignments of the variables $x_{n/2+1}, \ldots, x_n$

Reduction: $\mathrm{SAT} \leqslant \text{2-CNBS}$

Let A contain all $2^{n/2}$ assignments of the variables $x_1, \ldots, x_{n/2}$

Let C contain all clauses of ϕ

Let B contain all $2^{n/2}$ assignments of the variables $x_{n/2+1}, \ldots, x_n$

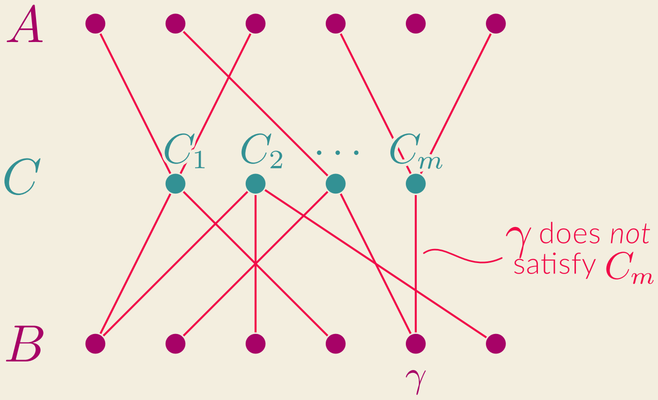


Reduction: $\mathrm{SAT} \leqslant 2\text{-CNBS}$

Let A contain all $2^{n/2}$ assignments of the variables $x_1, \ldots, x_{n/2}$

Let C contain all clauses of ϕ

Let B contain all $2^{n/2}$ assignments of the variables $x_{n/2+1}, \ldots, x_n$

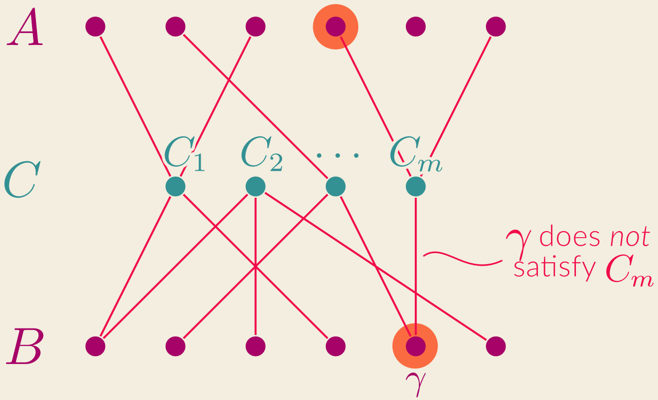


Reduction: SAT $\leqslant$ 2-CNBS

Let A contain all $2^{n/2}$ assignments of the variables $x_1, \ldots, x_{n/2}$

Let C contain all clauses of ϕ

Let B contain all $2^{n/2}$ assignments of the variables $x_{n/2+1}, \ldots, x_n$

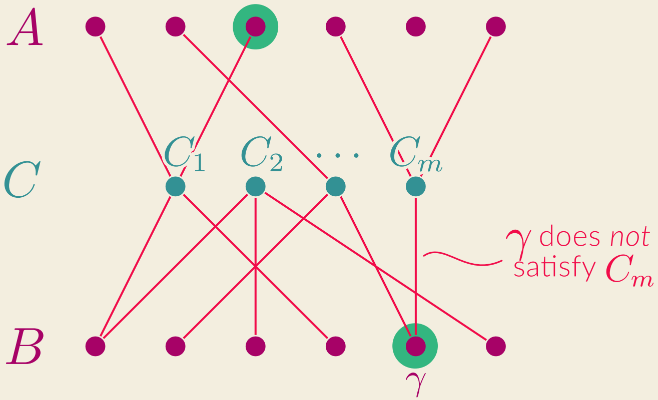


Reduction: $\mathrm{SAT} \leqslant \text{2-CNBS}$

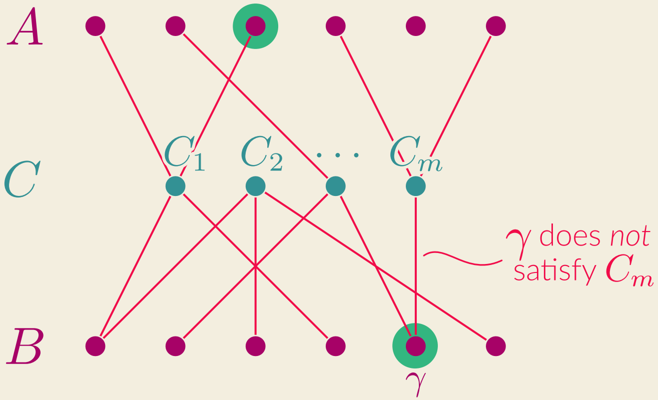
Let A contain all $2^{n/2}$ assignments of the variables $x_1, \ldots, x_{n/2}$

Let C contain all clauses of ϕ

Let B contain all $2^{n/2}$ assignments of the variables $x_{n/2+1}, \ldots, x_n$

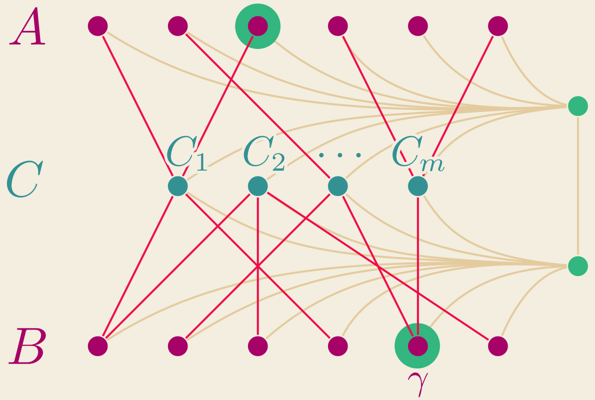


Reduction: $\mathrm{SAT} \leqslant \text{2-CNBS}$



ϕ is satisfiable iff there exist two vertices $\alpha \in A$, $\beta \in B$ with $N(\alpha) \cap N(\beta) = \varnothing$.

Reduction: $\mathrm{SAT} \leqslant \text{2-CNBS}$



ϕ is satisfiable iff there exists a vertex $\gamma \in A \cup B$ with $|N^2[\gamma]| < 2^{\frac{n}{2}+1} + m + 2$.

Reduction: $\mathrm{SAT} \leqslant \text{2-CNBS}$

ϕ is satisfiable iff there exists a vertex $\gamma \in A \cup B$ with $|N^2[\gamma]| < 2^{\frac{n}{2}+1} + m + 2$.

Assume we can solve 2-CNBS in time $O(|G|^{2-\varepsilon})$.

Reduction: $\mathrm{SAT} \leqslant 2\text{-}\mathrm{CNBS}$

ϕ is satisfiable iff there exists a vertex $\gamma \in A \cup B$ with $|N^2[\gamma]| < 2^{\frac{n}{2}+1} + m + 2$.

Assume we can solve 2-CNBS in time $O(|G|^{2-\varepsilon})$.
The output consists of $|G|$ numbers, thus in time

$$O(2^{\frac{n}{2}} m + |G|^{2-\varepsilon} + |G| \log |G|)$$

we can check whether ϕ is satisfiable,

Reduction: $\mathrm{SAT} \leqslant \text{2-CNBS}$

ϕ is satisfiable iff there exists a vertex $\gamma \in A \cup B$ with $|N^2[\gamma]| < 2^{\frac{n}{2}+1} + m + 2$.

Assume we can solve 2-CNBS in time $O(|G|^{2-\varepsilon})$. The output consists of $|G|$ numbers, thus in time

$$O(2^{\frac{n}{2}}m + |G|^{2-\varepsilon} + |G|\log|G|)$$
$$= O(2^{\frac{n}{2}}m + (2^{\frac{n}{2}+1} + m + 2)^{2-\varepsilon})$$
$$= 2^{n(1-\varepsilon/2)}m^{O(1)} = 2^{\varepsilon' n}m^{O(1)}$$

we can check whether ϕ is satisfiable, contradicting SETH.

Can we do better?

CLOSED 2-NEIGHBOURHOOD SIZES

Input: A graph G.

Output: $|N^2[v]|$ for every $v \in G$.

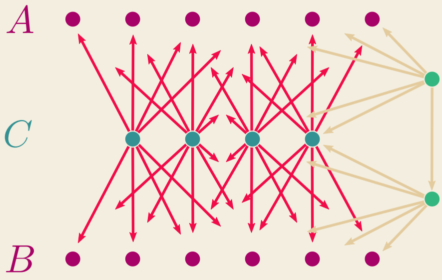
Thm. Unless SETH fails, 2-CNBS cannot be solved in time

1. $O(|G|^{2-\varepsilon})$
2. $O(2^{o(\Delta^-(\vec{G}_2))} n^{2-\varepsilon})$

Gutin G, Mertzios GB, Reidl F.
Lower and Upper Bound for Computing the Size of All Second Neighbourhoods.
arXiv preprint arXiv:1805.01684. 2018 May 4

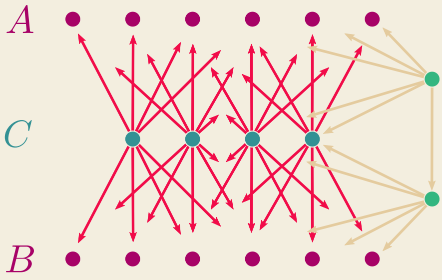
Reduction: $\mathrm{SAT} \leqslant \mathrm{2\text{-}CNBS}$

How big is $\Delta^{-}(\vec{G}_r)$?



Reduction: $\mathrm{SAT} \leqslant \text{2-CNBS}$

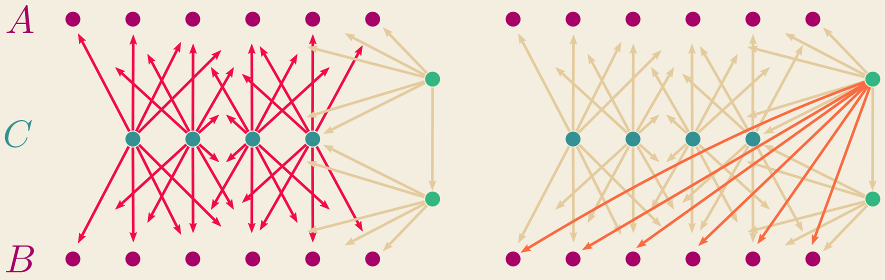
How big is $\Delta^-(\vec{G}_r)$?



$$\Delta^-(\vec{G}_1) \leqslant m + 1$$

Reduction: $\mathrm{SAT} \leqslant \text{2-CNBS}$

How big is $\Delta^-(\vec{G}_r)$?



$\Delta^-(\vec{G}_1) \leqslant m+1$

$\Delta^-(\vec{G}_r) \leqslant m+2$

Reduction: $\mathrm{SAT} \leqslant 2\text{-CNBS}$

Assume we can solve 2-CNBS in time $O(2^{o(\Delta^-(\vec{G}_2))} n^{2-\varepsilon})$. Thus in time

$$O(2^{o(\Delta^-(\vec{G}_2))}|G|^{2-\varepsilon} + |G|\log|G|)$$

we can check whether ϕ is satisfiable,

Reduction: $\mathrm{SAT} \leqslant 2\text{-CNBS}$

Assume we can solve 2-CNBS in time $O(2^{o(\Delta^-(\vec{G}_2))} n^{2-\varepsilon})$. Thus in time

$$O(2^{o(\Delta^-(\vec{G}_2))} |G|^{2-\varepsilon} + |G| \log |G|)$$
$$= O(2^{o(m)} 2^{\frac{n}{2}(2-\varepsilon)} + 2^{\frac{n}{2}} n)$$

we can check whether ϕ is satisfiable,

Reduction: $\mathrm{SAT} \leqslant \text{2-CNBS}$

Assume we can solve 2-CNBS in time $O(2^{o(\Delta^-(\vec{G}_2))} n^{2-\varepsilon})$. Thus in time

$$O(2^{o(\Delta^-(\vec{G}_2))}|G|^{2-\varepsilon} + |G| \log |G|)$$
$$= O(2^{o(m)} 2^{\frac{n}{2}(2-\varepsilon)} + 2^{\frac{n}{2}} n)$$
$$= O(2^{(1-\frac{\varepsilon}{2})n + o(n)} + 2^{\frac{n}{2}} n)$$

we can check whether ϕ is satisfiable,

Reduction: $\mathrm{SAT} \leqslant \text{2-CNBS}$

Assume we can solve 2-CNBS in time $O(2^{o(\Delta^-(\vec{G}_2))} n^{2-\varepsilon})$. Thus in time

$$O(2^{o(\Delta^-(\vec{G}_2))} |G|^{2-\varepsilon} + |G| \log |G|)$$

$$= O(2^{o(m)} 2^{\frac{n}{2}(2-\varepsilon)} + 2^{\frac{n}{2}} n)$$

$$= O(2^{(1-\frac{\varepsilon}{2})n + o(n)} + 2^{\frac{n}{2}} n) = O(2^{\varepsilon' n})$$

we can check whether ϕ is satisfiable, contradicting SETH.

Reduction: $\mathrm{SAT} \leqslant \mathrm{2\text{-}CNBS}$

Assume we can solve 2-CNBS in time $O(2^{o(\Delta^-(\vec{G}_2))} n^{2-\varepsilon})$. Thus in time

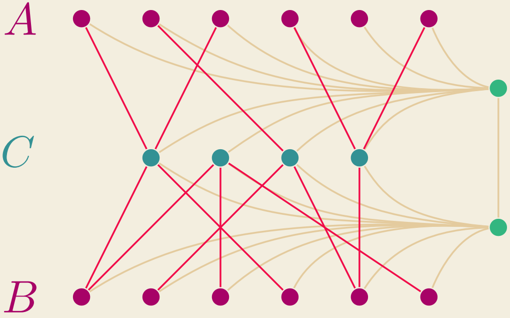
$$
\begin{aligned}
& O(2^{o(\Delta^-(\vec{G}_2))} |G|^{2-\varepsilon} + |G| \log |G|) \\
= & O(2^{o(m)} 2^{\frac{n}{2}(2-\varepsilon)} + 2^{\frac{n}{2}} n) \\
= & O(2^{(1-\frac{\varepsilon}{2})n + o(n)} + 2^{\frac{n}{2}} n) = O(2^{\varepsilon' n})
\end{aligned}
$$

we can check whether ϕ is satisfiable, contradicting SETH.

Unless the SETH fails, 2-CNBS cannot be solved in time $O(2^{o(\Delta^-(\vec{G}_2))} n^{2-\varepsilon})$.

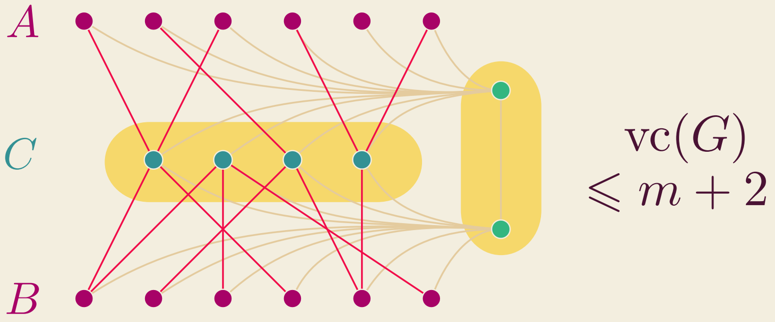
Reduction: $\mathrm{SAT} \leqslant \text{2-CNBS}$

What about other parameters?



Reduction: $\mathrm{SAT} \leqslant \text{2-CNBS}$

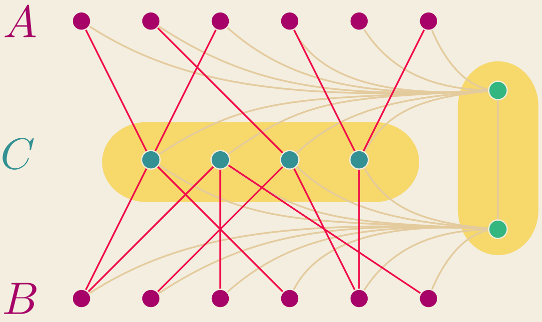
What about other parameters?



Unless the SETH fails, 2-CNBS cannot be solved in time $O(2^{o(\mathrm{vc}(G))} n^{2-\varepsilon})$.

Reduction: $\mathrm{SAT} \leqslant 2\text{-CNBS}$

What about other parameters?



$\mathrm{vc}(G)$
$\leqslant m + 2$

Unless the SETH fails, 2-CNBS cannot be solved in time $O(2^{o(\mathrm{vc}(G))} n^{2-\varepsilon})$.

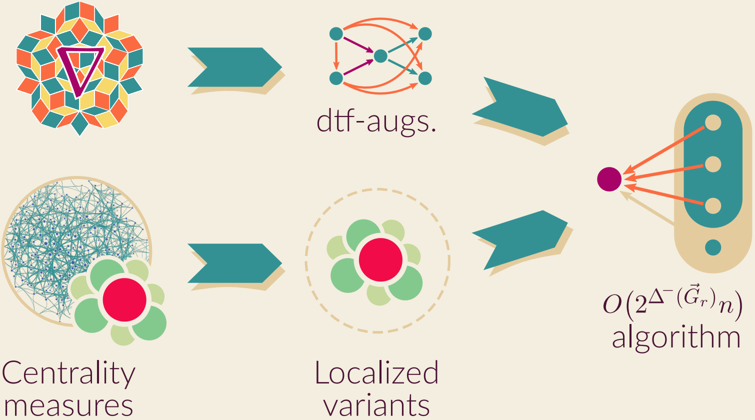
Reduction: $\mathrm{SAT} \leqslant \text{2-CNBS}$

What about other parameters?

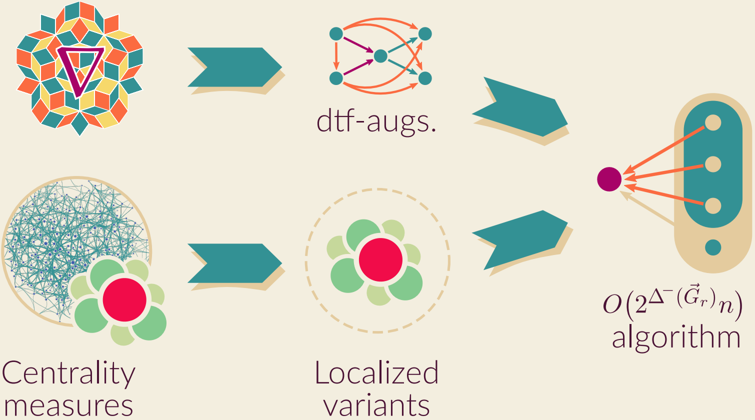
Unless the SETH fails, 2-CNBS cannot be solved in time $O(2^{o(\mathrm{vc}(G))} n^{2-\varepsilon})$.

Unless the SETH fails, 2-CNBS cannot be solved in time $O(2^{o(f(G))} n^{2-\varepsilon})$ for any $f \in \{\mathrm{wcol}_2, \mathrm{vc}, \mathrm{td}, \mathrm{pw}, \mathrm{tw}, \nabla_1, \tilde{\nabla}_1\}$.

The process so far



The process so far



Should we implement
this algorithm?