Dinur's Proof of the PCP Theorem

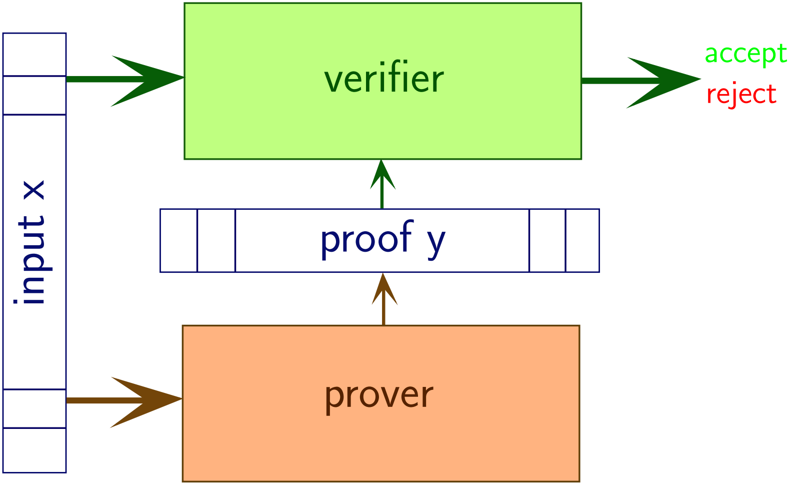
Zampetakis Manolis

School of Electrical and Computer Engineering
National Technical University of Athens

December 22, 2014

Table of contents

Characterization of NP



Characterization of NP

Prover

The prover of a language L has to output a correct proof y with $|y| \leq poly(|x|)$ when $x \in L$. When $x \notin L$ prover could output anything.

Verifier

The verifier of a language $L \in NP$ has to be efficient (polynomialy time bounded).

Characterization of NP

NP prover-verifier

For every language $L \in NP$ there exists a prover P and an efficient verifier V, that reads :

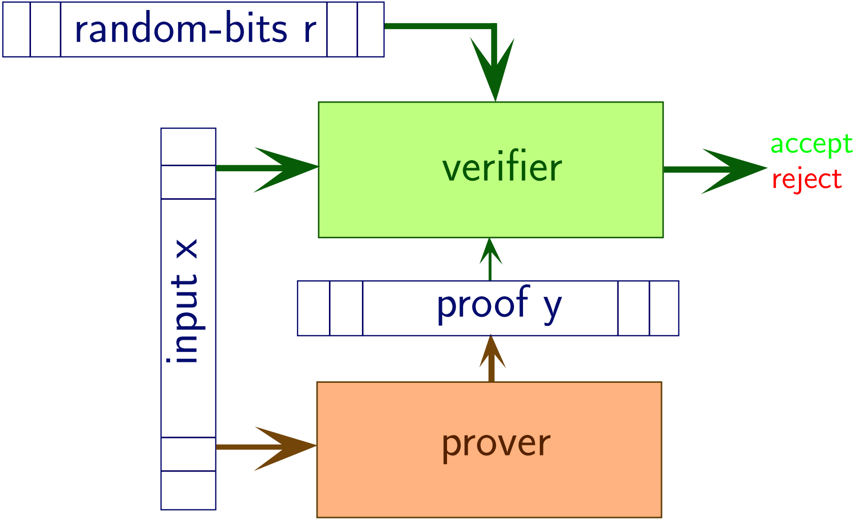
- the input string x
- the proof string y

and accepts or rejects such that :

Completeness : If $x \in L$ then P outputs a proof y such that V accepts.

Soundness : If $x \notin L$ then V rejects every proof y.

New characterization of NP [Arora, Safra]



New characterization of NP [Arora, Safra]

Probabilistically Checkable of Proofs

Definition : PCP $[p(n), q(n)]$

For every $L \in PCP[p(n),\ q(n)]$ there exists a prover P and an efficient verifier V, that reads :

- an input string x with length n
- **a random string r with length p(n)**
- **q(n) bits** randomly from the proof y

and accepts or rejects, such that :

Completeness : If $x \in L$ then P outputs a proof y such that V accepts **with probability 1**.

Soundness : If $x \notin L$ then V accepts **with probability** $\leq \frac{1}{2}$.

New characterization of NP [Arora, Safra]

Probabilistically Checkable of Proofs

Definition : PCP $[O(\log n), O(1)]$

For every $L \in PCP[O(\log n),\ O(1)]$ there exists a prover P and an efficient verifier V, that reads :

- an input string x with length n
- **a random string r with length c $\log$ n**
- **k bits** randomly from the proof y

with $c, k \in \mathbb{R}^+$ and accepts or rejects, such that :

Completeness : If $x \in L$ then P outputs a proof y such that V accepts **with probability 1**.

Soundness : If $x \notin L$ then V accepts **with probability** $\leq \frac{1}{2}$.

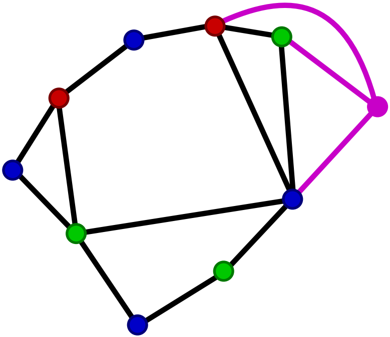
New characterization of NP [Arora, Safra]

Probabilistically Checkable of Proofs

PCP Theorem [Arora-Lund-Motwani-Sudan-Szegedy 92]

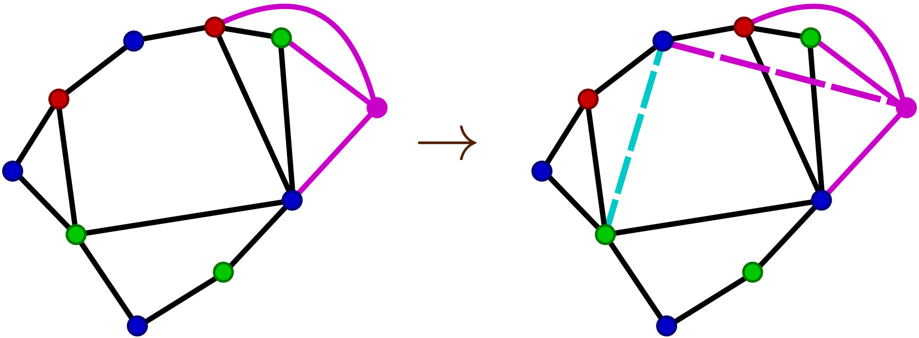
$$\mathbf{NP} = \mathbf{PCP}[\mathbf{O}(\log \mathbf{n}), \mathbf{O}(1)]$$

Key Idea
Graph Coloring Problem - Constraint Graph Problem



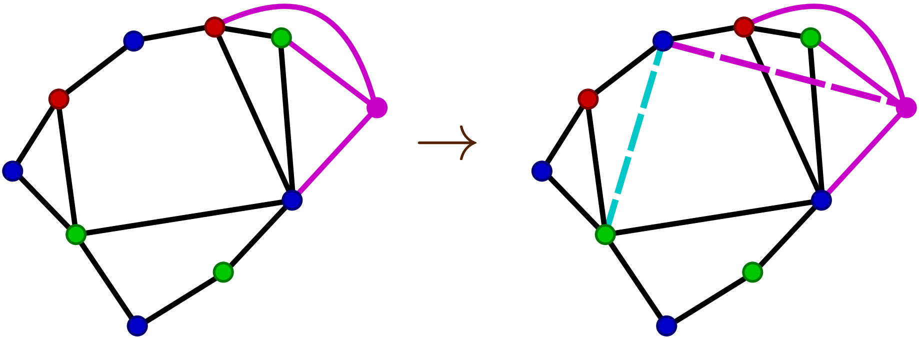
Key Idea

Graph Coloring Problem - Constraint Graph Problem



Key Idea

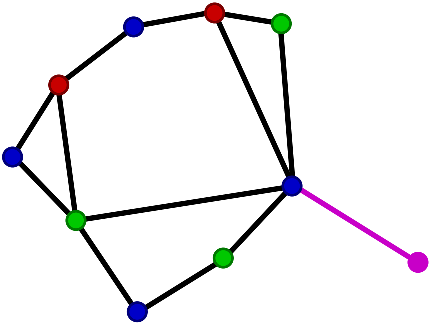
Graph Coloring Problem - Constraint Graph Problem



Key Idea

Add contraints for every path with length $\leq t$ in the graph G.

Good Idea but...



A Lemma on expanders...

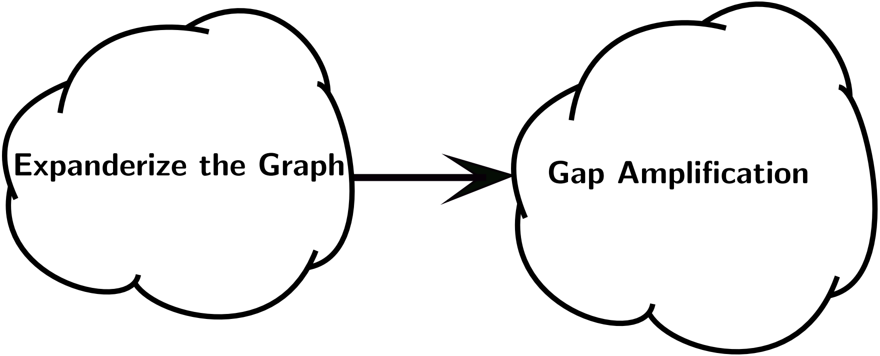
from Trevisan's lectures

Lemma

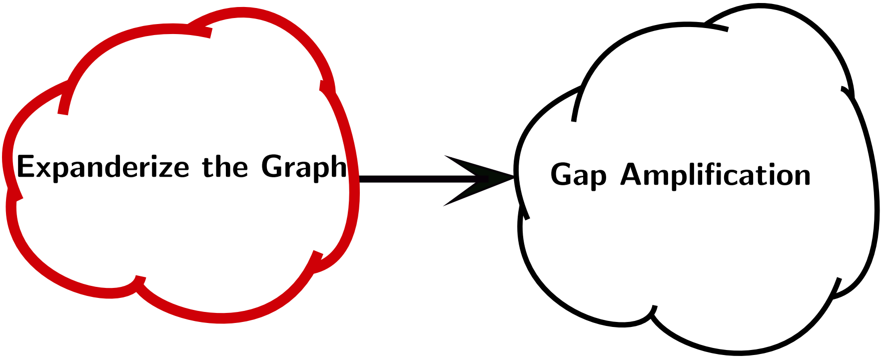
Let $G = (V, E)$ *be an expander and* $F \subseteq E$ *then for every path* p *in* G *with length* t

$$\Pr[p \textit{ completely misses } F] \leq \left(1 - t\frac{|F|}{|E|}\right)$$

Proof Overview



Proof Overview



Introduction to Expanders

Definition

The edge expansion of a graph $G = (V, E)$, denoted by $\phi(G)$, is defined as

$$\phi(G) = \min_{S \subseteq V, |S| \leq |V|/2} \frac{|E(S, \bar{S})|}{|S|}$$

Introduction to Expanders

Lemma

There exists a constant ϕ_0 such that for every $n \in \mathbb{N}$ and $d < n$ there is an efficient algorithm to construct a $d-$regular graph G with $\phi(G) \geq \phi_0$.

Introduction to Expanders

Lemma

There exists a constant ϕ_0 *such that for every* $n \in \mathbb{N}$ *and* $d < n$ *there is an efficient algorithm to construct a* $d-$*regular graph* G *with* $\phi(G) \geq \phi_0$.

Lemma

Let $G = (V, E)$ *be an expander and* $F \subseteq E$ *then for every path* p *in* G *with length* t

$$\Pr[p \text{ completely misses } F] \leq \left(1 - t\frac{|F|}{|E|}\right)$$

Introduction to Expanders

Lemma

There exists a constant ϕ_0 *such that for every* $n \in \mathbb{N}$ *and* $d < n$ *there is an efficient algorithm to construct a* $d-$*regular graph* G *with* $\phi(G) \geq \phi_0$.

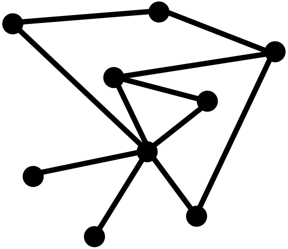
Lemma

Let $G = (V, E)$ *be an expander and* $F \subseteq E$ *then for every path* p *in* G *with length* t

$$\Pr[p \text{ completely misses } F] \leq \left(1 - t\frac{|F|}{|E|}\right)$$

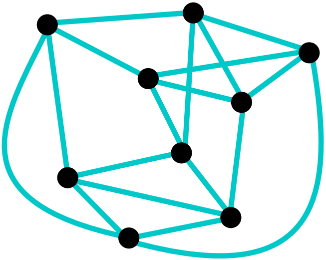
useful property: If a we add edges to a graph which is expander then the graph remains expander.

Union with expander



G

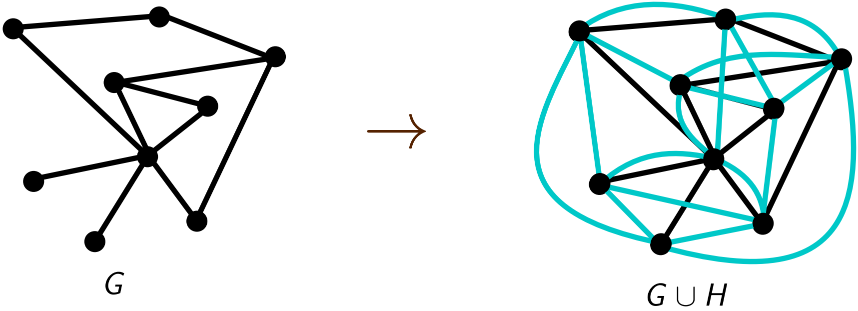
$\cup$



H

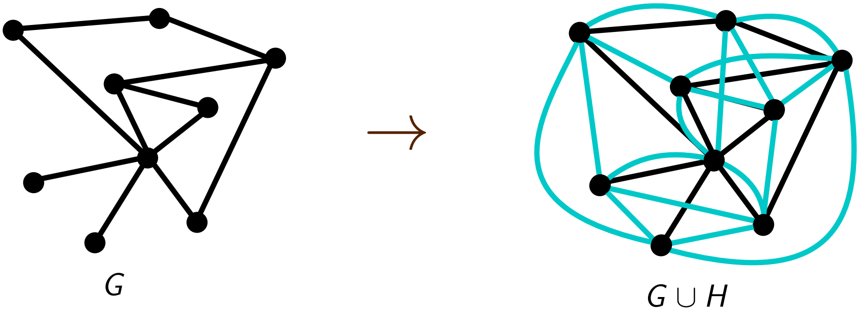
$d-$regular expander

Union with expander



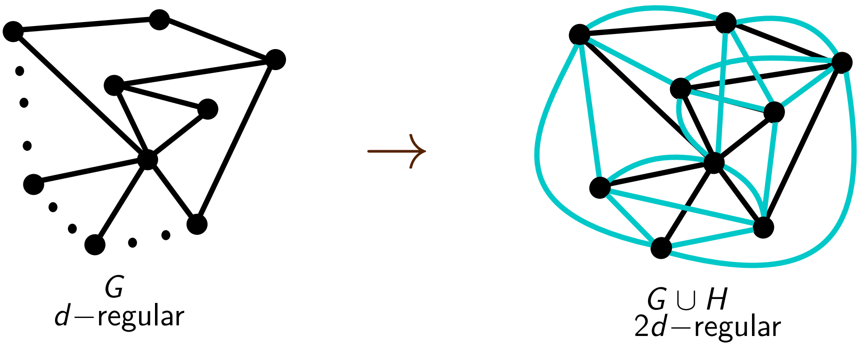
G is satifiable iff $G \cup H$ is satisfiable

Union with expander



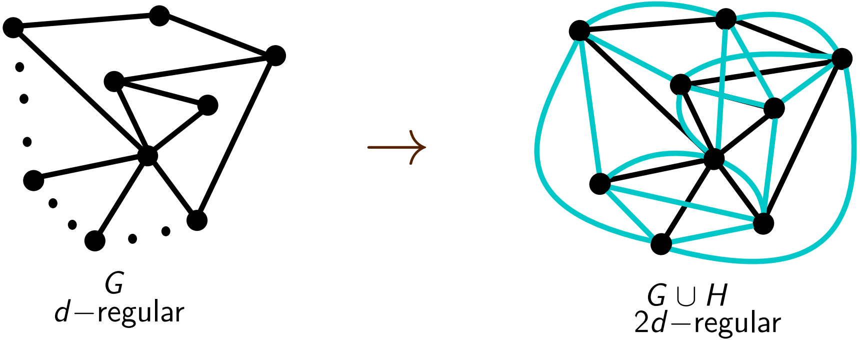
what about gap?

Union with expander



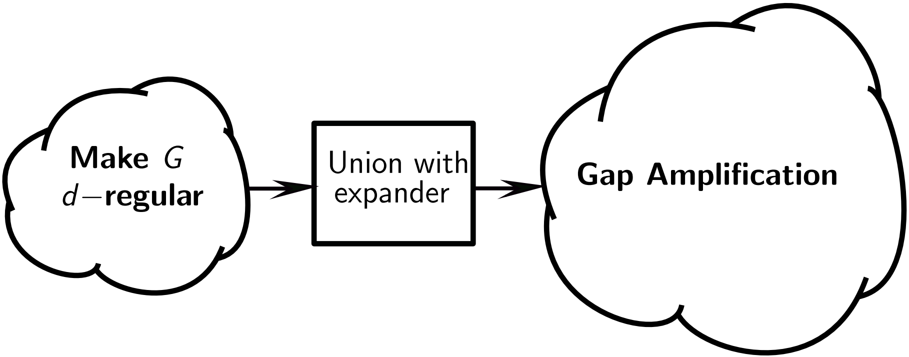
G is satifiable iff $G \cup H$ is satisfiable

Union with expander

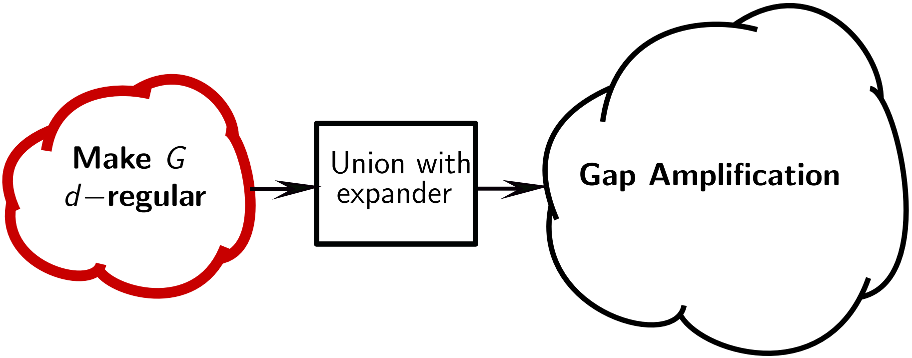


$$gap(G \cup H) \geq 1/2 gap(G)$$

Proof Overview

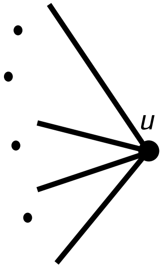


Proof Overview

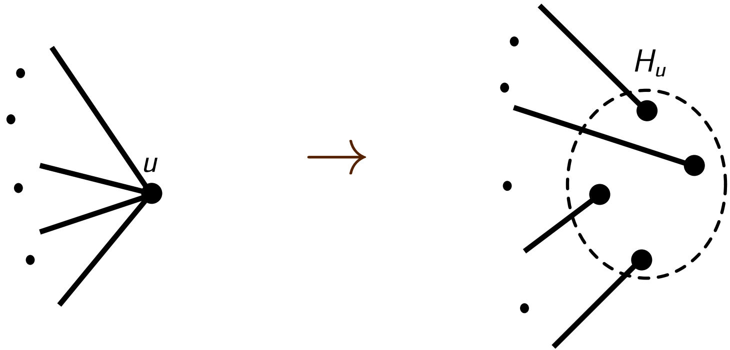


Make G $d-$regular

Papadimitriou and Yannakakis

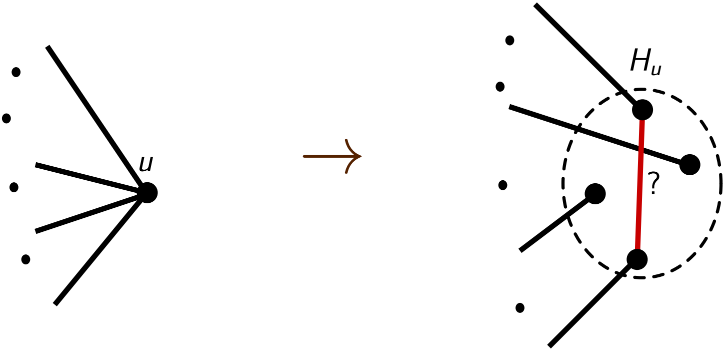


Make G $d-$regular
Papadimitriou and Yannakakis



Make G $d-$regular

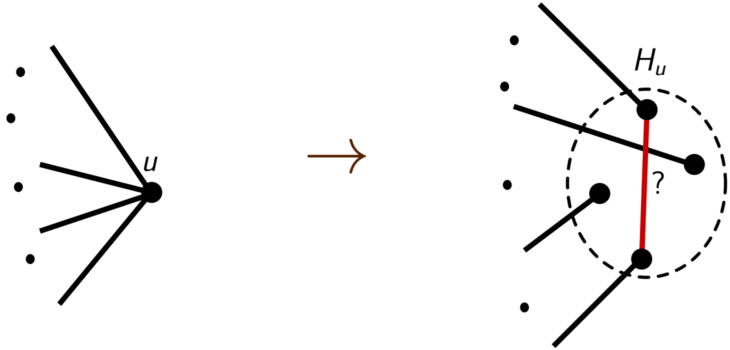
Papadimitriou and Yannakakis



What kind of constraints in H_u ?

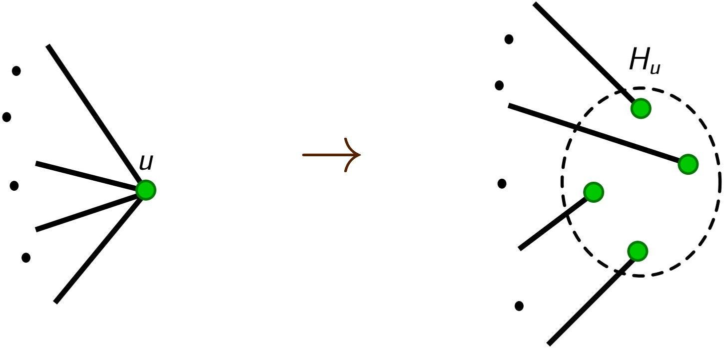
Make G $d-$regular

Papadimitriou and Yannakakis



What kind of constraints in H_u ?
EQUALITY

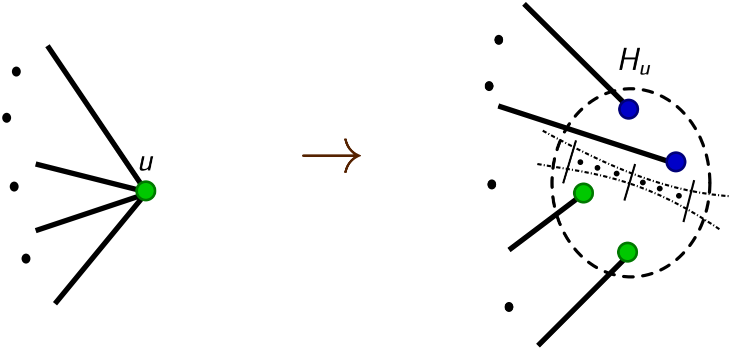
Make G $d-$regular
Papadimitriou and Yannakakis



G is satifiable iff G' is satisfiable

Make G $d-$regular

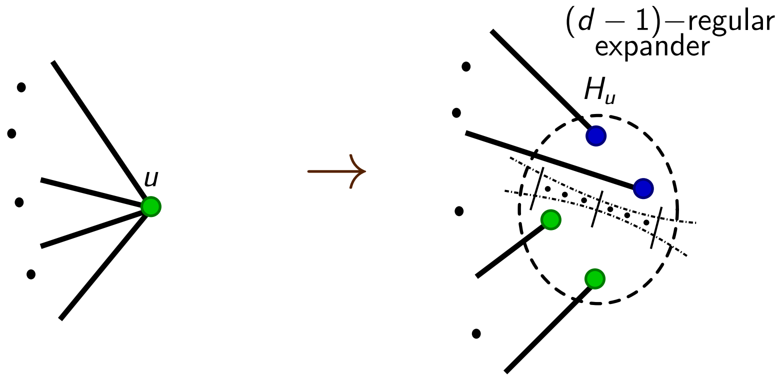
Papadimitriou and Yannakakis



What about gap?

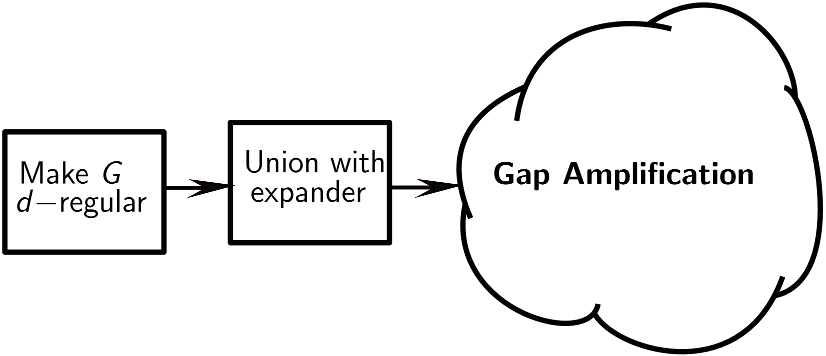
Make G $d-$regular

Papadimitriou and Yannakakis

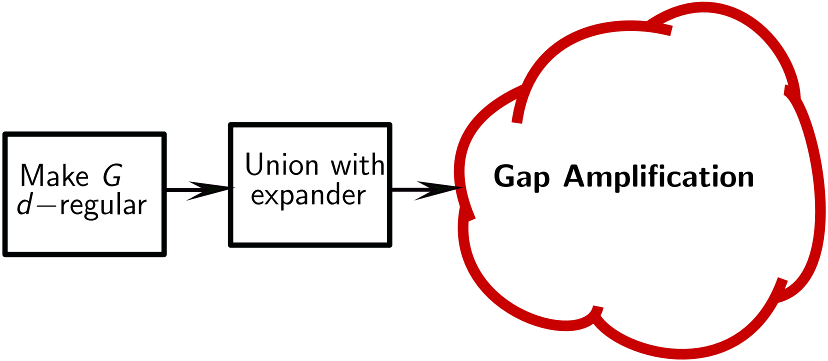


$$gap(G') \geq 1/O(1)gap(G)$$

Proof Overview



Proof Overview



Gap Amplification

Key Idea

Add contraints for every path with length $\leq t$ in the graph G.

Gap Amplification

Key Idea

Add contraints for every path with length $\leq t$ in the graph G.

for t = 2

Our initial alphabet is $\Sigma = \{red, green, blue\}$.
Now every vertex u has an *opinion* about the color of the vertices in $N(u)$ and therefore the alphabet becomes $\Sigma' = \Sigma \times \Sigma^d$.
For every path $\{u, w, v\}$ with length 2 we add an edge $\{u, v\}$ with constraint : $c_{\{u,v\}}$ is true if the opinion u has about w is the same as the opinion v has about w.

Gap Amplification

Key Idea

Add contraints for every path with length $\leq t$ in the graph G.

for every t

Our initial alphabet is $\Sigma = \{red, green, blue\}$.
Now every vertex u has an *opinion* about the color of the vertices in $N(u)$ and therefore the alphabet becomes $\Sigma' = \Sigma \times \Sigma^d \times \cdots \times \Sigma^{d^t}$.
For every path $\{u, w, \ldots, v\}$ with length t we add an edge $\{u, v\}$ with constraint : $c_{\{u,v\}}$ is true if the opinion u has about every internal vertex w is the same as the opinion v has about w, and every internal edge of G is valid using this opinion.

Gap Amplification

Key Idea

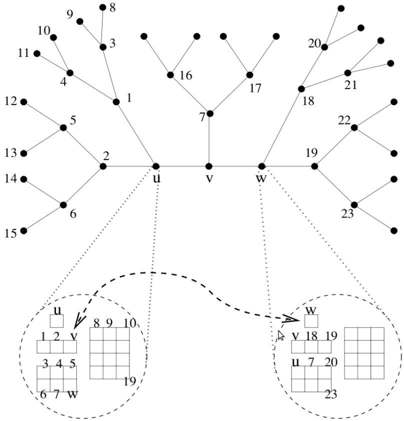
Add contraints for every path with length $\leq t$ in the graph G.

for every t

Our initial alphabet is $\Sigma = \{red, green, blue\}$.
Now every vertex u has an *opinion* about the color of the vertices in $N(u)$ and therefore the alphabet becomes $\Sigma' = \Sigma \times \Sigma^d \times \cdots \times \Sigma^{d^t}$.
For every path $\{u, w, \ldots, v\}$ with length t we add an edge $\{u, v\}$ with constraint : $c_{\{u,v\}}$ is true if the opinion u has about every internal vertex w is the same as the opinion v has about w, and every internal edge of G is valid using this opinion.

G is satifiable iff G' is satisfiable

Gap Amplification



Computational Complexity *Oded Goldreich*

Gap Amplification

We have now to use the following :

Lemma

Let $G = (V, E)$ be an expander and $F \subseteq E$ then for every path p in G with length t

$$\Pr[p \text{ completely misses } F] \leq \left(1 - t\frac{|F|}{|E|}\right)$$

Where we set F the set of unsatisfied constraints in G.

Gap Amplification

Lemma

$$gap(G') \geq \frac{t}{O(1)} gap(G)$$

Proof Sketch

$$\begin{aligned} gap(G') &\geq 1/3 \Pr_{e'}[e' \text{ passes through } F] \\ &\geq 1/3(1 - \Pr_{e'}[e' \text{ completely misses } F]) \\ &\geq 1/3(1 - (t \cdot gap(G))) \\ &= \frac{t}{O(1)} gap(G) \end{aligned}$$

Gap Amplification

If we set $t = O(n)$ we have finished!

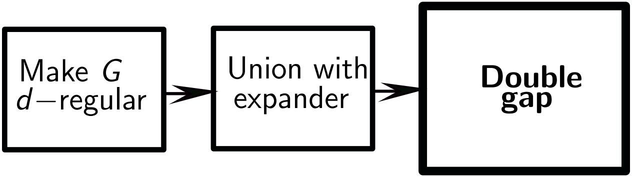
Gap Amplification

If we set $t = O(n)$ we have finished!
But we cannot do this because then $size(G') = O(d^t) = O(d^{O(n)})$ which is inefficient !!!

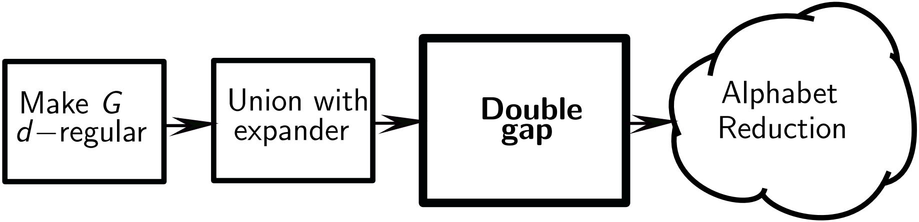
Gap Amplification

If we set $t = O(n)$ we have finished!
But we cannot do this because then $size(G') = O(d^t) = O(d^{O(n)})$ which is inefficient !!!
Therefore t must be a constant!

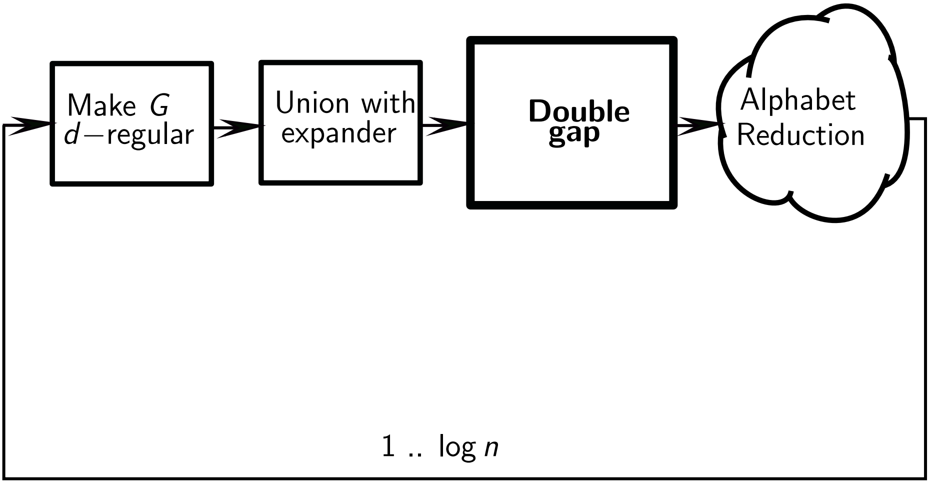
Proof Overview



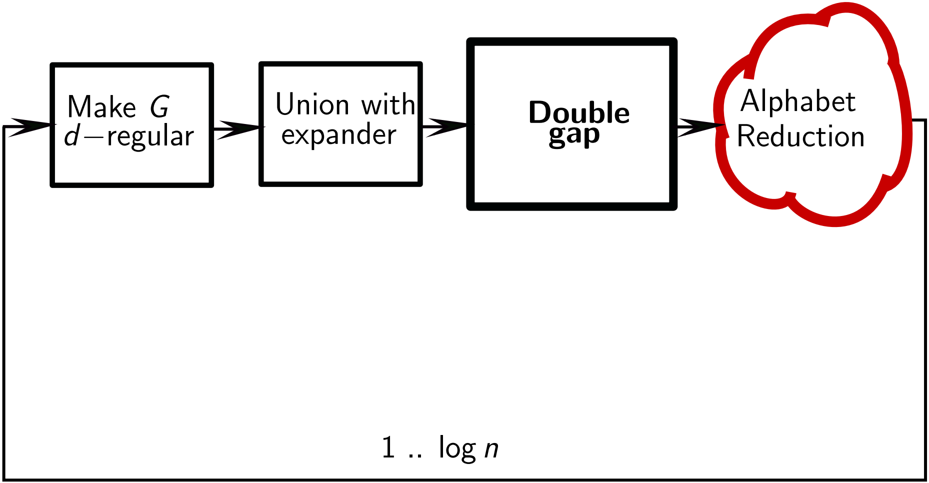
Proof Overview



Proof Overview



Proof Overview



Alphabet Reduction

Effects of the Reduction

- Size increases a constant factor
- Gap decreases a constant factor
- Alphabet size redused to 16

Alphabet Reduction

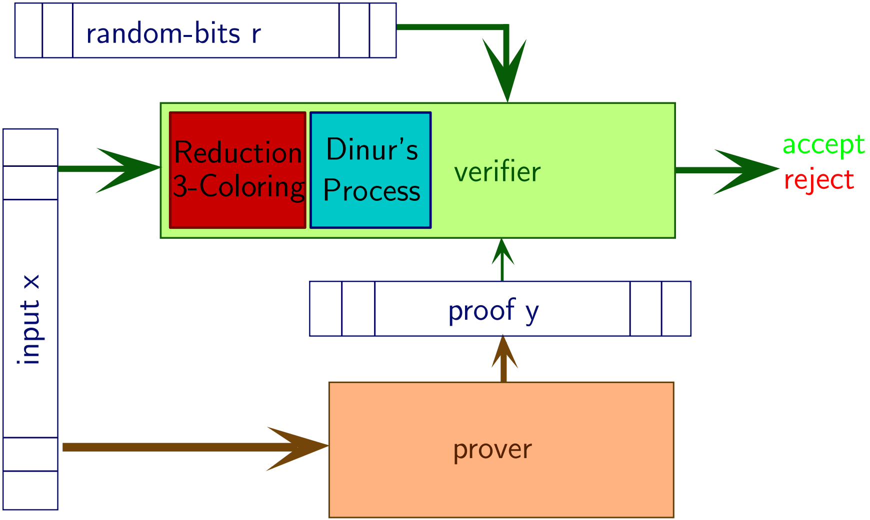
Effects of the Reduction

- Size increases a constant factor
- Gap decreases a constant factor
- Alphabet size redused to 16

Proof Techniques

- Hadamard Codes
- Linearity Testing
- Fourier Analysis

Finishing the proof



Let's flip the coin

Coin

For every language L in NP there is a way to write proofs such that for every instance x :

- If $x \in L$ then there is a correct proof
- If $x \notin L$ then every proof has a lot of errors

Let's flip the coin

Coin

For every language L in NP there is a way to write proofs such that for every instance x :

- If $x \in L$ then there is a correct proof
- If $x \notin L$ then every proof has a lot of errors

One side

PCP Theorem

Let's flip the coin

Coin

For every language L in NP there is a way to write proofs such that for every instance x :

- If $x \in L$ then there is a correct proof
- If $x \notin L$ then every proof has a lot of errors

One side

PCP Theorem

The other side

Hardness of Approximation

Thanks!

Thank you! :)