Finding Small Subgraphs of Given Minimum Degree

Ignasi Sau-Valls
Joint work with O. Amini, S. Perénnes, D. Peleg, and S. Saurabh

Mascotte Project - INRIA/CNRS-I3S/UNSA - FRANCE
Applied Mathematics IV Department of UPC - SPAIN

Outline of the talk

- Statement of the problem
- (Motivations)
- Preliminaries
- Hardness result
- (Approximation algorithm for minor free graphs)
- Conclusions and open problems

Definition of the problem

- **MINIMUM SUBGRAPH OF MINIMUM DEGREE $\geq d$ (MSMD$_d$)**:

  Input: an undirected graph $G = (V, E)$ and an integer $d \geq 3$.

  Output: a subset $S \subseteq V$ with $\delta(G[S]) \geq d$, s.t. $|S|$ is minimum.

- For $d = 2$ it is the GIRTH problem (find the length of a shortest cycle), which is in P.

- We will see that for $d \geq 3$, MSMD$_d$ does not accept any constant-factor approximation (in particular, it is NP-complete).

Definition of the problem

- **Minimum Subgraph of Minimum Degree $\geq d$ (MSMD$_d$)**:

  Input: an undirected graph $G = (V, E)$ and an integer $d \geq 3$.

  Output: a subset $S \subseteq V$ with $\delta(G[S]) \geq d$, s.t. $|S|$ is minimum.

- For $d = 2$ it is the GIRTH problem (find the length of a shortest cycle), which is in P.

- We will see that for $d \geq 3$, MSMD$_d$ does not accept any constant-factor approximation (in particular, it is NP-complete).

Definition of the problem

- **MINIMUM SUBGRAPH OF MINIMUM DEGREE** $\geq d$ **(MSMD$_d$)**:

  Input: an undirected graph $G = (V, E)$ and an integer $d \geq 3$.

  Output: a subset $S \subseteq V$ with $\delta(G[S]) \geq d$, s.t. $|S|$ is minimum.

- For $d = 2$ it is the GIRTH problem (find the length of a shortest cycle), which is in P.

- We will see that for $d \geq 3$, MSMD$_d$ does not accept any constant-factor approximation (in particular, it is NP-complete).

Definition of the problem

- **MINIMUM SUBGRAPH OF MINIMUM DEGREE $\geq d$ (MSMD$_d$)**:

  Input: an undirected graph $G = (V, E)$ and an integer $d \geq 3$.

  Output: a subset $S \subseteq V$ with $\delta(G[S]) \geq d$, s.t. $|S|$ is minimum.

- For $d = 2$ it is the GIRTH problem (find the length of a shortest cycle), which is in P.

- We will see that for $d \geq 3$, MSMD$_d$ does not accept any constant-factor approximation (in particular, it is NP-complete).

Motivation 1: relation with the DENSE-k-SUBGRAPH

- *Density* $\rho(G)$ of a graph $G = (V, E)$:

$$\rho(G) := \frac{|E(G)|}{|V(G)|}$$

More generally, for $S \subset V(G)$:

$$\rho(S) := \rho(G[S])$$

- DENSE k-SUBGRAPH problem:

DENSE k-SUBGRAPH (DkS):
Input: a graph $G = (V, E)$ and a positive integer k.
Output: a subset $S \subseteq V$ with $|S| = k$, maximizing $\rho(S)$.

(U. Feige, D. Peleg and G. Kortsarz, Algorithmica'01)
(S. Khot, FOCS'04)

Motivation 1: relation with the DENSE-k-SUBGRAPH

- *Density* $\rho(G)$ of a graph $G = (V, E)$:

$$\rho(G) := \frac{|E(G)|}{|V(G)|}$$

More generally, for $S \subset V(G)$:

$$\rho(S) := \rho(G[S])$$

- DENSE k-SUBGRAPH problem:

  DENSE k-SUBGRAPH (DkS):
  Input: a graph $G = (V, E)$ and a positive integer k.
  Output: a subset $S \subseteq V$ with $|S| = k$, maximizing $\rho(S)$.

(U. Feige, D. Peleg and G. Kortsarz, Algorithmica'01)
(S. Khot, FOCS'04)

Motivation 1: relation with the DENSE-k-SUBGRAPH (II)

- Suppose that we want to find an induced subgraph $G[S]$ of size at most k and density at least ρ.
  We can suppose that S is minimal, i.e. there is no subset of S with density greater than $\rho(S)$.

  1) All the vertices of $G[S]$ have degree at least $\rho/2$.
     If there exists a vertex v with degree strictly smaller than $\rho/2$, then the removal of v results in a smaller subgraph with higher density.

  2) If we have a subgraph $G[S]$ with minimum degree at least ρ, then S is a subset with density at least $\rho/2$.

- So, **modulo a constant factor**, looking for a densest subgraph of G with size at most k is as hard as looking for the greatest ρ such that there exists a subgraph of G with size at most k and minimum degree at least ρ.

Motivation 1: relation with the DENSE-k-SUBGRAPH (II)

- Suppose that we want to find an induced subgraph $G[S]$ of size at most k and density at least ρ.
  We can suppose that S is minimal, i.e. there is no subset of S with density greater than $\rho(S)$.

  1) All the vertices of $G[S]$ have degree at least $\rho/2$.
     If there exists a vertex v with degree strictly smaller than $\rho/2$, then the removal of v results in a smaller subgraph with higher density.

  2) If we have a subgraph $G[S]$ with minimum degree at least ρ, then S is a subset with density at least $\rho/2$.

- So, **modulo a constant factor**, looking for a densest subgraph of G with size at most k is as hard as looking for the greatest ρ such that there exists a subgraph of G with size at most k and minimum degree at least ρ.

Motivation 1: relation with the DENSE-k-SUBGRAPH (II)

- Suppose that we want to find an induced subgraph $G[S]$ of size at most k and density at least ρ.
  We can suppose that S is minimal, i.e. there is no subset of S with density greater than $\rho(S)$.

  1) All the vertices of $G[S]$ have degree at least $\rho/2$.
     If there exists a vertex v with degree strictly smaller than $\rho/2$, then the removal of v results in a smaller subgraph with higher density.

  2) If we have a subgraph $G[S]$ with minimum degree at least ρ, then S is a subset with density at least $\rho/2$.

- So, **modulo a constant factor**, looking for a densest subgraph of G with size at most k is as hard as looking for the greatest ρ such that there exists a subgraph of G with size at most k and minimum degree at least ρ.

Motivation 1: relation with the DENSE-k-SUBGRAPH (II)

- Suppose that we want to find an induced subgraph $G[S]$ of size at most k and density at least ρ.
  We can suppose that S is minimal, i.e. there is no subset of S with density greater than $\rho(S)$.

  1) All the vertices of $G[S]$ have degree at least $\rho/2$.
     If there exists a vertex v with degree strictly smaller than $\rho/2$, then the removal of v results in a smaller subgraph with higher density.

  2) If we have a subgraph $G[S]$ with minimum degree at least ρ, then S is a subset with density at least $\rho/2$.

- So, **modulo a constant factor**, looking for a densest subgraph of G with size at most k is as hard as looking for the greatest ρ such that there exists a subgraph of G with size at most k and minimum degree at least ρ.

Motivation 2: relation with TRAFFIC GROOMING

- Roughly, (one model of) TRAFFIC GROOMING consists in finding subgraphs with **high density** and *bounded* number of edges.
- I.e., we want to find subgraphs with **high average degree** (and *bounded* number of edges).
- Density and average degree of a graph differ by a factor 2.
- So, if we can find small subgraphs with prescribed minimum degree, we can also find small subgraphs with *good* density.

Motivation 2: relation with TRAFFIC GROOMING

- Roughly, (one model of) TRAFFIC GROOMING consists in finding subgraphs with **high density** and *bounded* number of edges.
- I.e., we want to find subgraphs with **high average degree** (and *bounded* number of edges).
- Density and average degree of a graph differ by a factor 2.
- So, if we can find small subgraphs with prescribed minimum degree, we can also find small subgraphs with *good* density.

Motivation 2: relation with TRAFFIC GROOMING

- Roughly, (one model of) TRAFFIC GROOMING consists in finding subgraphs with **high density** and *bounded* number of edges.
- I.e., we want to find subgraphs with **high average degree** (and *bounded* number of edges).
- Density and average degree of a graph differ by a factor 2.
- So, if we can find small subgraphs with prescribed minimum degree, we can also find small subgraphs with *good* density.

Hardness result

Preliminaries: hardness of approximation

- **Class APX (Approximable):**

  an NP-complete optimization problem is in APX if it can be approximated within a constant factor.

  Example: VERTEX COVER

- **Class PTAS (Polynomial-Time Approximation Scheme):**

  an NP-complete optimization problem is in PTAS if it can be approximated within a constant factor $1 + \varepsilon$, for all $\varepsilon > 0$ (the best one can hope for an NP-complete problem).

  Example: MAXIMUM KNAPSACK

Preliminaries: hardness of approximation

- **Class APX (Approximable):**

  an NP-complete optimization problem is in APX if it can be approximated within a constant factor.

  Example: VERTEX COVER

- **Class PTAS (Polynomial-Time Approximation Scheme):**

  an NP-complete optimization problem is in PTAS if it can be approximated within a constant factor $1 + \varepsilon$, for all $\varepsilon > 0$ (the best one can hope for an NP-complete problem).

  Example: MAXIMUM KNAPSACK

Idea of the proof for $d = 3$

(1) First we will see that $\mathrm{MSMD}_3 \notin$ PTAS.

(2) Then we will see that $\mathrm{MSMD}_3 \notin$ APX.

(1) $MSMD_3$ is not in $PTAS$

- Reduction from VERTEX COVER:

  Instance H of VERTEX COVER $\rightarrow$ Instance G of MSMD$_3$

- We will see that

  $$\text{PTAS for } G \Rightarrow \text{PTAS for } H$$

- And so,

  $$\nexists \text{ PTAS for MSMD}_3$$

- We can suppose $|E(H)| = 3 \cdot 2^m$

(1) *$MSMD_3$* is not in *PTAS*

- Reduction from VERTEX COVER:

  Instance H of VERTEX COVER $\rightarrow$ Instance G of $\mathbf{MSMD_3}$

- We will see that

$$\text{PTAS for } G \Rightarrow \text{PTAS for } H$$

- And so,

$$\nexists \text{ PTAS for } \text{MSMD}_3$$

- We can suppose $|E(H)| = 3 \cdot 2^m$

(1) *$MSMD_3$* is not in *PTAS*

- Reduction from VERTEX COVER:

  Instance H of VERTEX COVER $\rightarrow$ Instance G of $\mathbf{MSMD_3}$

- We will see that

$$\text{PTAS for } G \Rightarrow \text{PTAS for } H$$

- And so,

$$\nexists \text{ PTAS for } \text{MSMD}_3$$

- We can suppose $|E(H)| = 3 \cdot 2^m$

(1) *MSMD*$_3$ is not in *PTAS*

- Reduction from VERTEX COVER:

  Instance H of VERTEX COVER $\rightarrow$ Instance G of MSMD$_3$

- We will see that

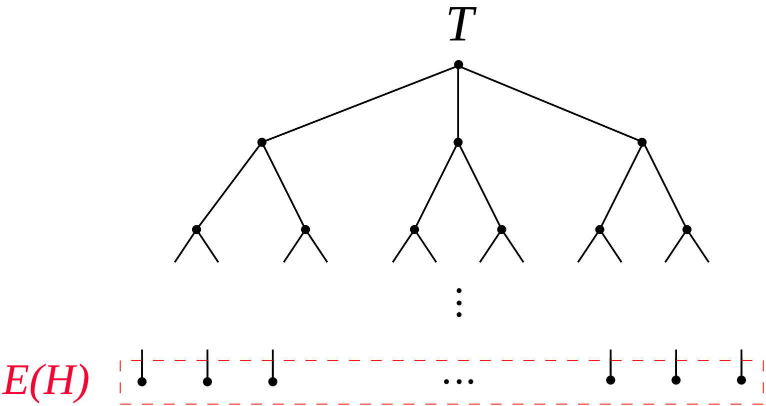
$$\text{PTAS for } G \Rightarrow \text{PTAS for } H$$

- And so,

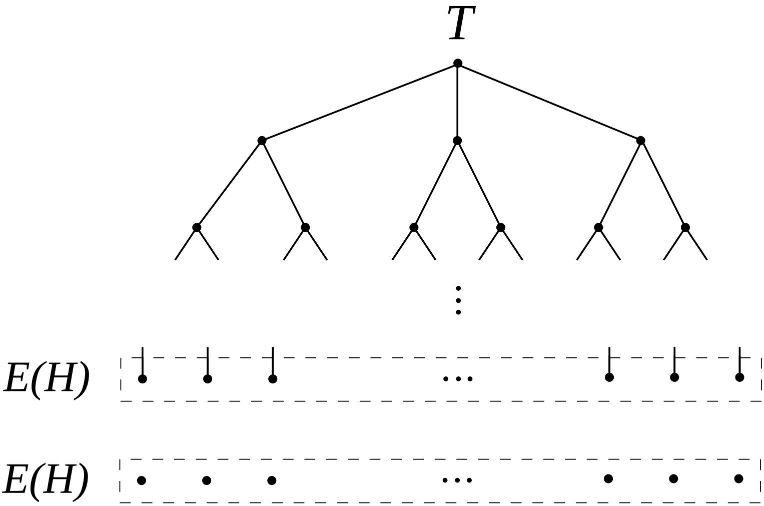
$$\nexists \text{ PTAS for MSMD}_3$$

- We can suppose $|E(H)| = 3 \cdot 2^m$

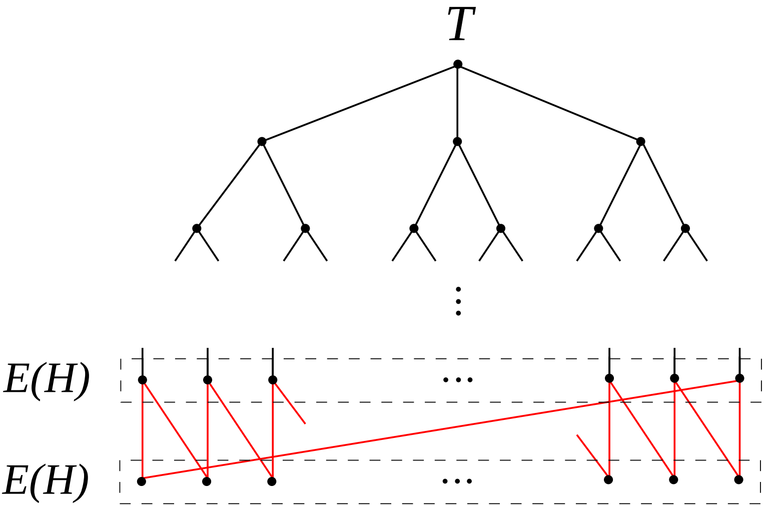
We build a complete ternary tree with $|E(H)| = 3 \cdot 2^m$ leaves:



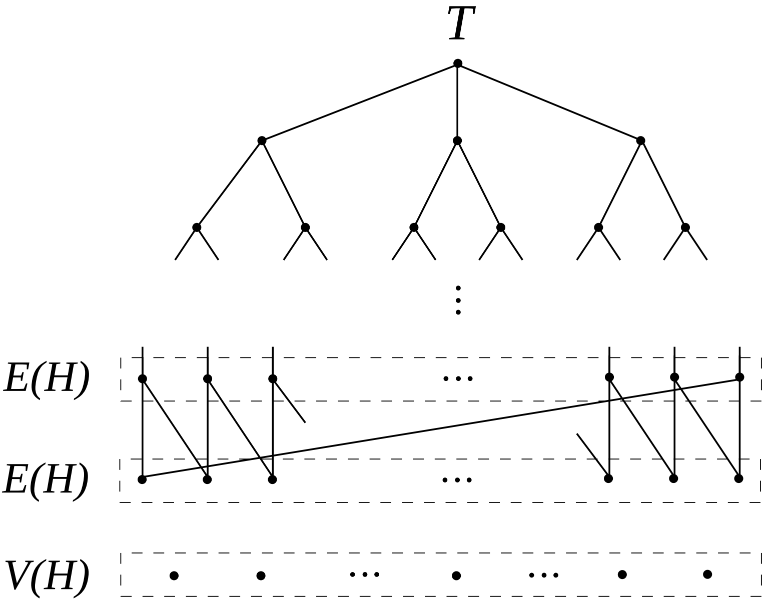
We add a copy of the set of leaves $E(H)$:



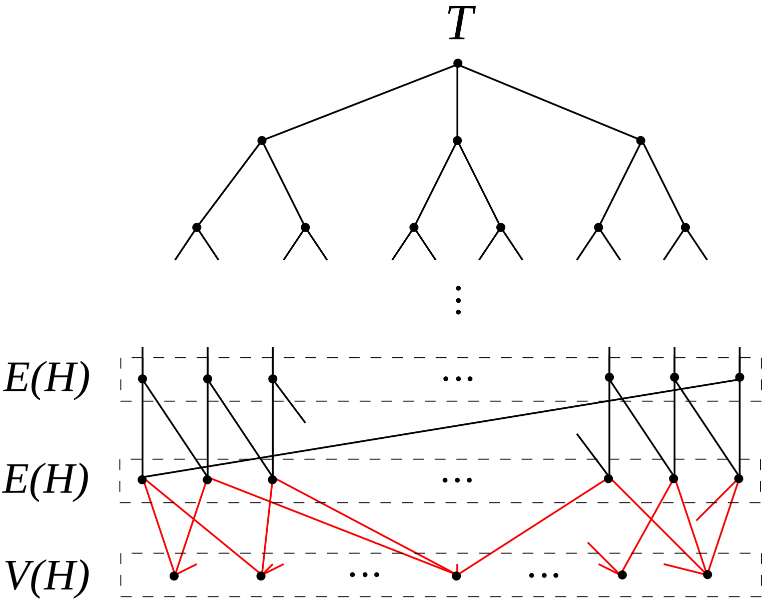
We join both sets with a Hamiltonian cycle (for technical reasons):



We add all the vertices of H:



We add the incidence relations between $E(H)$ and $V(H) \rightarrow G$:



(1) MSMD_3 is not in PTAS

- If we touch a vertex of $G \setminus V(H)$, we have to touch all the vertices of $G \setminus V(H)$

- Thus, MSMD_3 in G is equivalent to minimize the number of selected vertices in $V(H)$

  $\rightarrow$ this is **exactly** VERTEX COVER in H !!

- Thus,

$$OPT_{\text{MSMD}_3}(G) = OPT_{\text{VC}}(H) + |V(G \setminus V(H))| =$$
$$= OPT_{\text{VC}}(H) + 9 \cdot 2^m$$

- This clearly proves that

$$\text{PTAS for MSMD}_3 \Rightarrow \text{PTAS for VERTEX COVER}$$

(1) MSMD_3 is not in PTAS

- If we touch a vertex of $G \setminus V(H)$, we have to touch all the vertices of $G \setminus V(H)$
- Thus, MSMD_3 in G is equivalent to minimize the number of selected vertices in $V(H)$

  $\rightarrow$ this is **exactly** VERTEX COVER in H !!

- Thus,

$$OPT_{\mathrm{MSMD}_3}(G) = OPT_{\mathrm{VC}}(H) + |V(G \setminus V(H))| =$$
$$= OPT_{\mathrm{VC}}(H) + 9 \cdot 2^m$$

- This clearly proves that

$$\text{PTAS for } \mathrm{MSMD}_3 \Rightarrow \text{PTAS for VERTEX COVER}$$

(1) MSMD_3 is not in PTAS

- If we touch a vertex of $G \setminus V(H)$, we have to touch all the vertices of $G \setminus V(H)$
- Thus, MSMD_3 in G is equivalent to minimize the number of selected vertices in $V(H)$

  $\rightarrow$ this is **exactly** VERTEX COVER in H !!

- Thus,

$$OPT_{\text{MSMD}_3}(G) = OPT_{\text{VC}}(H) + |V(G \setminus V(H))| =$$
$$= OPT_{\text{VC}}(H) + 9 \cdot 2^m$$

- This clearly proves that

$$\text{PTAS for MSMD}_3 \Rightarrow \text{PTAS for VERTEX COVER}$$

(1) $MSMD_3$ is not in PTAS

- If we touch a vertex of $G \setminus V(H)$, we have to touch all the vertices of $G \setminus V(H)$

- Thus, $MSMD_3$ in G is equivalent to minimize the number of selected vertices in $V(H)$

  $\rightarrow$ this is **exactly** VERTEX COVER in H !!

- Thus,

$$OPT_{\text{MSMD}_3}(G) = OPT_{\text{VC}}(H) + |V(G \setminus V(H))| =$$
$$= OPT_{\text{VC}}(H) + 9 \cdot 2^m$$

- This clearly proves that

$$\text{PTAS for MSMD}_3 \;\; \Rightarrow \;\; \text{PTAS for VERTEX COVER}$$

(2) $MSMD_3$ is not in APX

- Let $\alpha > 1$ be the factor of inapproximability of $MSMD_3$
- We use a technique called **error amplification**:
  - We build a sequence of families of graphs $\mathcal{G}_k$, such that $MSMD_3$ is hard to approximate in $\mathcal{G}_k$ within a factor α^k
  - This proves that the problem is not in APX

    (for any constant C, $\exists\ k > 0$ such that $\alpha^k > C$)
- Let $G_1 = G$.

  We explain the construction of G_2: first take our graph G and...

(2) $MSMD_3$ is not in APX

- Let $\alpha > 1$ be the factor of inapproximability of $MSMD_3$
- We use a technique called **error amplification**:
  - We build a sequence of families of graphs $\mathcal{G}_k$, such that $MSMD_3$ is hard to approximate in $\mathcal{G}_k$ within a factor α^k
  - This proves that the problem is not in APX
    (for any constant C, $\exists\ k > 0$ such that $\alpha^k > C$)
- Let $G_1 = G$.
  We explain the construction of G_2: first take our graph G and...

(2) $MSMD_3$ is not in APX

- Let $\alpha > 1$ be the factor of inapproximability of $MSMD_3$
- We use a technique called **error amplification**:
  - We build a sequence of families of graphs $\mathcal{G}_k$, such that $MSMD_3$ is hard to approximate in $\mathcal{G}_k$ within a factor α^k
  - This proves that the problem is not in APX

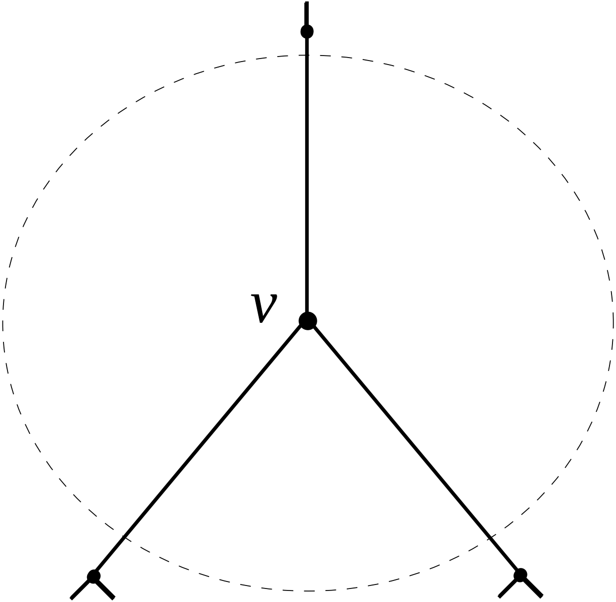
    (for any constant C, $\exists\ k > 0$ such that $\alpha^k > C$)
- Let $G_1 = G$.
  We explain the construction of G_2: first take our graph G and...

(2) MSMD_3 is not in APX

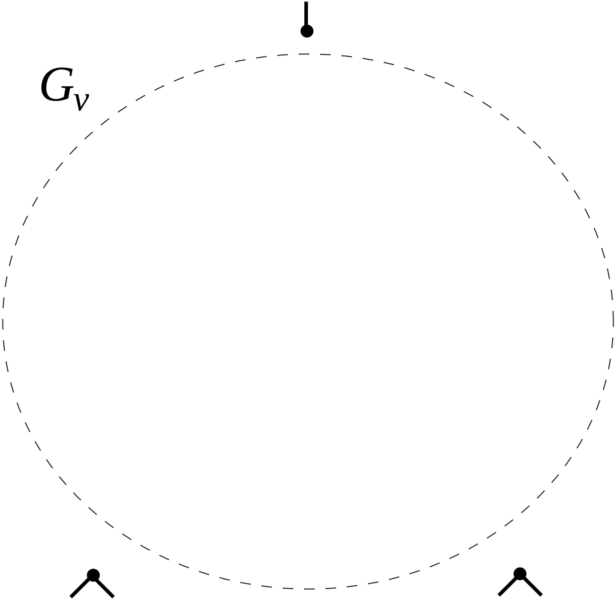
- Let $\alpha > 1$ be the factor of inapproximability of MSMD_3
- We use a technique called **error amplification**:
  - We build a sequence of families of graphs $\mathcal{G}_k$, such that MSMD_3 is hard to approximate in $\mathcal{G}_k$ within a factor α^k
  - This proves that the problem is not in APX

    (for any constant C, $\exists\ k > 0$ such that $\alpha^k > C$)
- Let $G_1 = G$.
  We explain the construction of G_2: first take our graph G and...

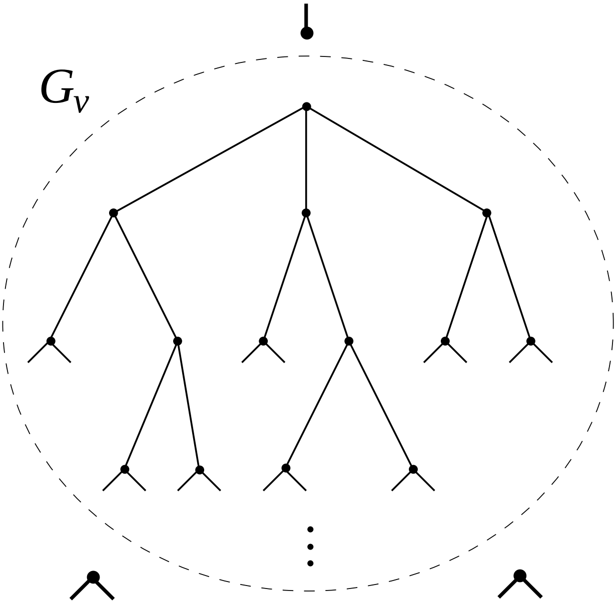
For any vertex v (note its degree by d_v):



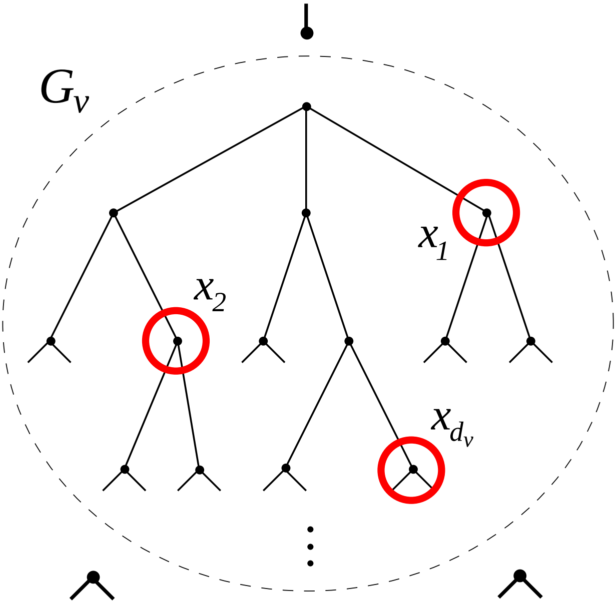
We will replace the vertex v with a graph G_v, built as follows:



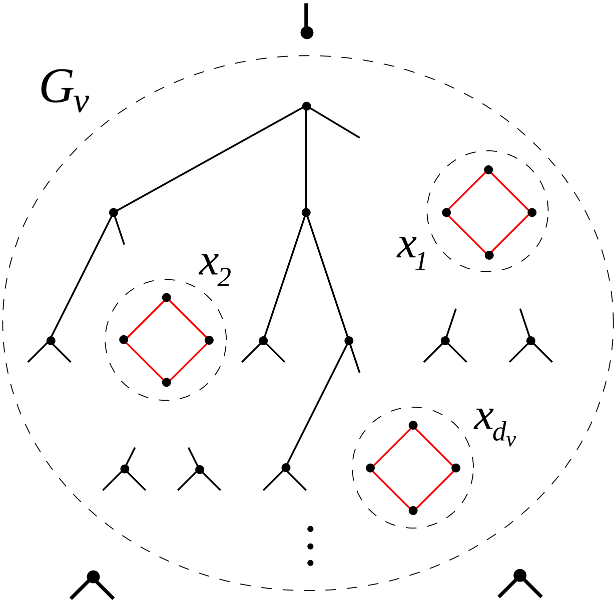
We begin by placing a copy of G (described before):



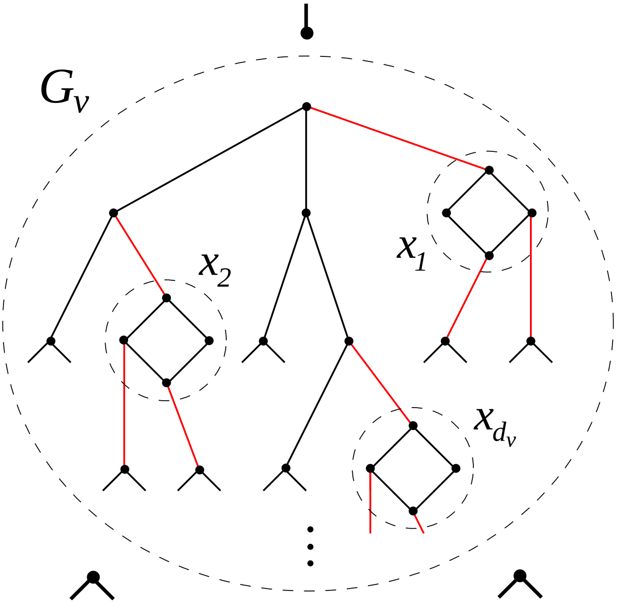
We select d_v vertices of degree 3 in $T \subset G$:



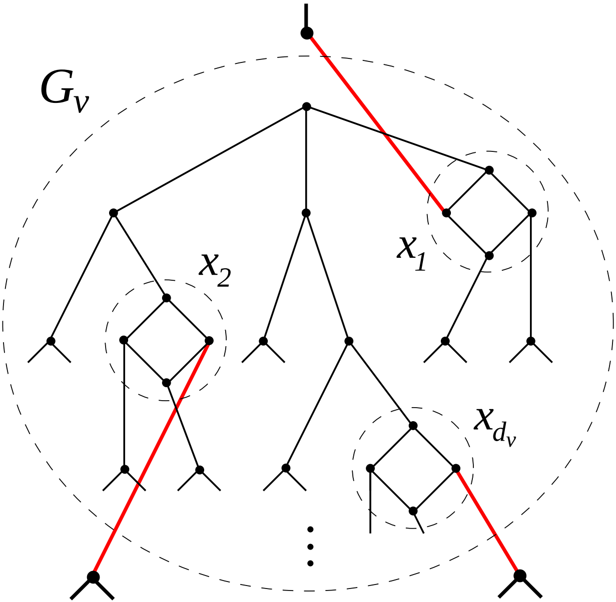
We replace each of these vertices x_i with a C_4:



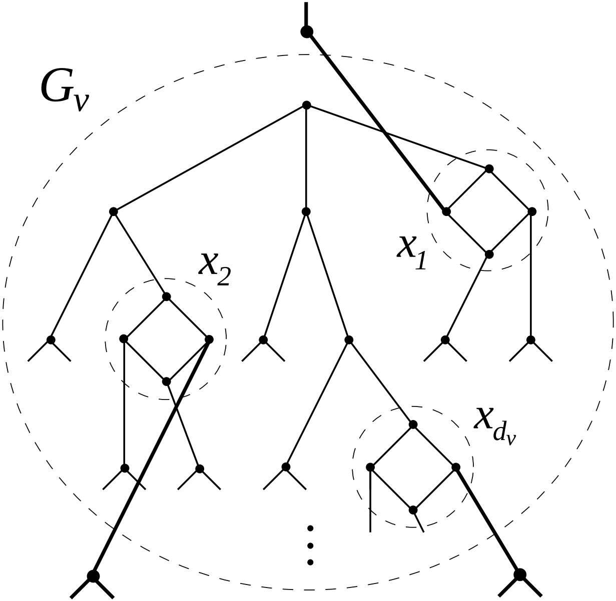
In each C_4, we join 3 of the vertices to the neighbors of x_i:



We join the d_v vertices of degree 2 to the d_v neighbors of v:



This construction for all $v \in G$ defines G_2:



(2) $MSMD_3$ is not in APX

- Once a vertex in one G_v is chosen $\rightarrow$ $MSMD_3$ in G_v (which is hard up to a constant α)
- But minimize the number of v's for which we touch G_v $\rightarrow$ $MSMD_3$ in G (which is also hard up to a constant α)
- Thus, in G_2 the problem is hard to approximate up to a factor $\alpha \cdot \alpha = \alpha^2$
- Inductively we prove that in G_k the problem is hard to approximate up to a factor α^k

(2) $MSMD_3$ is not in APX

- Once a vertex in one G_v is chosen $\rightarrow$ $MSMD_3$ in G_v

  (which is hard up to a constant α)

- But minimize the number of v's for which we touch G_v $\rightarrow$ $MSMD_3$ in G (which is also hard up to a constant α)

- Thus, in G_2 the problem is hard to approximate up to a factor $\alpha \cdot \alpha = \alpha^2$

- Inductively we prove that in G_k the problem is hard to approximate up to a factor α^k

(2) MSMD_3 is not in APX

- Once a vertex in one G_v is chosen $\longrightarrow$ MSMD_3 in G_v

  (which is hard up to a constant α)

- But minimize the number of v's for which we touch G_v $\longrightarrow$ MSMD_3 in G (which is also hard up to a constant α)

- Thus, in G_2 the problem is hard to approximate up to a factor $\alpha \cdot \alpha = \alpha^2$

- Inductively we prove that in G_k the problem is hard to approximate up to a factor α^k

Approximation algorithm for minor free graphs

The problem is in P for graphs of *small* treewidth

Lemma

Let G be a graph on n vertices with treewidth at most t, and let d be a positive integer. Then in time $\mathcal{O}((d+1)^t (t+1)^{d^2} n)$ we can either

- *find a smallest subgraph of minimum degree at least d in G, or*
- *conclude that no such subgraph exists.*

Corollary

Let G be an n-vertex graph with treewidth $\mathcal{O}(\log n)$, and let d be a positive integer. Then in polynomial time one can either

- *find a smallest subgraph of minimum degree at least d in G, or*
- *conclude that no such subgraph exists.*

The problem is in P for graphs of *small* treewidth

Lemma

Let G be a graph on n vertices with treewidth at most t, and let d be a positive integer. Then in time $\mathcal{O}((d+1)^t (t+1)^{d^2} n)$ we can either

- *find a smallest subgraph of minimum degree at least d in G, or*
- *conclude that no such subgraph exists.*

Corollary

Let G be an n-vertex graph with treewidth $\mathcal{O}(\log n)$, and let d be a positive integer. Then in polynomial time one can either

- *find a smallest subgraph of minimum degree at least d in G, or*
- *conclude that no such subgraph exists.*

Nice partition of M-minor-free graphs

Theorem

For a fixed graph M, there is a constant c_M such that for any integer $k \geq 1$ and for every M-minor-free graph G, the vertices of G can be partitioned into $k+1$ sets such that any k of the sets induce a graph of treewidth at most $c_M k$.
Furthermore, such a partition can be found in polynomial time.

(E. Demaine, M.T. Hajiaghayi and K.C. Kawarabayashi, FOCS'05)

Approximation algorithm for M-minor-free graphs

(1) Relying on the previous Theorem, partition $V(G)$ in polynomial time into $\log n + 1$ sets $V_0, \ldots, V_{\log n}$ such that any $\log n$ of the sets induce a graph of treewidth at most $c_M \log n$, where c_M is a constant depending only on the excluded graph M.

(2) Run the dynamic programming algorithm of the Lemma on all the subgraphs $G_i = G[V \setminus V_i]$ of $\log n$ sets, $i = 0, \ldots, \log n$.

(3) This procedure finds all the solutions of size at most $\log n$.

(4) If no solution is found, output the whole graph G.

This algorithm provides an $O(n/\log n)$-approximation for MSMD_d in minor-free graphs, for all $d \geq 3$.

The running time of the algorithm is polynomial in n, since in step **(2)**, for each G_i, the dynamic programming algorithm runs in $O((d+1)^{t_i}(t_i+1)^{d^2} n)$ time, where t_i is the treewidth of G_i, which is at most $c_M \log n$.

Approximation algorithm for M-minor-free graphs

(1) Relying on the previous Theorem, partition $V(G)$ in polynomial time into $\log n + 1$ sets $V_0, \ldots, V_{\log n}$ such that any $\log n$ of the sets induce a graph of treewidth at most $c_M \log n$, where c_M is a constant depending only on the excluded graph M.

(2) Run the dynamic programming algorithm of the Lemma on all the subgraphs $G_i = G[V \setminus V_i]$ of $\log n$ sets, $i = 0, \ldots, \log n$.

(3) This procedure finds all the solutions of size at most $\log n$.

(4) If no solution is found, output the whole graph G.

This algorithm provides an $\mathcal{O}(n/\log n)$-approximation for MSMD_d in minor-free graphs, for all $d \geq 3$.

The running time of the algorithm is polynomial in n, since in step **(2)**, for each G_i, the dynamic programming algorithm runs in $\mathcal{O}((d+1)^{t_i}(t_i+1)^{d^2} n)$ time, where t_i is the treewidth of G_i, which is at most $c_M \log n$.

Approximation algorithm for M-minor-free graphs

(1) Relying on the previous Theorem, partition $V(G)$ in polynomial time into $\log n + 1$ sets $V_0, \ldots, V_{\log n}$ such that any $\log n$ of the sets induce a graph of treewidth at most $c_M \log n$, where c_M is a constant depending only on the excluded graph M.

(2) Run the dynamic programming algorithm of the Lemma on all the subgraphs $G_i = G[V \setminus V_i]$ of $\log n$ sets, $i = 0, \ldots, \log n$.

(3) This procedure finds all the solutions of size at most $\log n$.

(4) If no solution is found, output the whole graph G.

This algorithm provides an $\mathcal{O}(n/\log n)$-approximation for MSMD$_d$ in minor-free graphs, for all $d \geq 3$.

The running time of the algorithm is polynomial in n, since in step **(2)**, for each G_i, the dynamic programming algorithm runs in $\mathcal{O}((d+1)^{t_i}(t_i+1)^{d^2} n)$ time, where t_i is the treewidth of G_i, which is at most $c_M \log n$.

Approximation algorithm for M-minor-free graphs

(1) Relying on the previous Theorem, partition $V(G)$ in polynomial time into $\log n + 1$ sets $V_0, \ldots, V_{\log n}$ such that any $\log n$ of the sets induce a graph of treewidth at most $c_M \log n$, where c_M is a constant depending only on the excluded graph M.

(2) Run the dynamic programming algorithm of the Lemma on all the subgraphs $G_i = G[V \setminus V_i]$ of $\log n$ sets, $i = 0, \ldots, \log n$.

(3) This procedure finds all the solutions of size at most $\log n$.

(4) If no solution is found, output the whole graph G.

This algorithm provides an $\mathcal{O}(n/\log n)$-approximation for MSMD_d in minor-free graphs, for all $d \geq 3$.

The running time of the algorithm is polynomial in n, since in step **(2)**, for each G_i, the dynamic programming algorithm runs in $\mathcal{O}((d+1)^{t_i}(t_i+1)^{d^2} n)$ time, where t_i is the treewidth of G_i, which is at most $c_M \log n$.

Approximation algorithm for M-minor-free graphs

(1) Relying on the previous Theorem, partition $V(G)$ in polynomial time into $\log n + 1$ sets $V_0, \ldots, V_{\log n}$ such that any $\log n$ of the sets induce a graph of treewidth at most $c_M \log n$, where c_M is a constant depending only on the excluded graph M.

(2) Run the dynamic programming algorithm of the Lemma on all the subgraphs $G_i = G[V \setminus V_i]$ of $\log n$ sets, $i = 0, \ldots, \log n$.

(3) This procedure finds all the solutions of size at most $\log n$.

(4) If no solution is found, output the whole graph G.

This algorithm provides an $\mathcal{O}(n/\log n)$-approximation for MSMD_d in minor-free graphs, for all $d \geq 3$.

The running time of the algorithm is polynomial in n, since in step **(2)**, for each G_i, the dynamic programming algorithm runs in $\mathcal{O}((d+1)^{t_i}(t_i+1)^{d^2} n)$ time, where t_i is the treewidth of G_i, which is at most $c_M \log n$.

Approximation algorithm for M-minor-free graphs

(1) Relying on the previous Theorem, partition $V(G)$ in polynomial time into $\log n + 1$ sets $V_0, \ldots, V_{\log n}$ such that any $\log n$ of the sets induce a graph of treewidth at most $c_M \log n$, where c_M is a constant depending only on the excluded graph M.

(2) Run the dynamic programming algorithm of the Lemma on all the subgraphs $G_i = G[V \setminus V_i]$ of $\log n$ sets, $i = 0, \ldots, \log n$.

(3) This procedure finds all the solutions of size at most $\log n$.

(4) If no solution is found, output the whole graph G.

This algorithm provides an $\mathcal{O}(n/\log n)$-approximation for MSMD_d in minor-free graphs, for all $d \geq 3$.

The running time of the algorithm is polynomial in n, since in step **(2)**, for each G_i, the dynamic programming algorithm runs in $\mathcal{O}((d+1)^{t_i}(t_i+1)^{d^2} n)$ time, where t_i is the treewidth of G_i, which is at most $c_M \log n$.

Conclusions

- We have proved that MSMD_d, $d \geq 3$, is not in APX.
- We have an $\mathcal{O}(n/\log n)$-approximation for minor free graphs.

- **Parameterized version of MSMD_d:**

  Input: an undirected graph $G = (V, E)$, an integer $d \geq 3$, and a parameter k.

  Question: does there exist $S \subseteq V$, with $|S| \leq k$, such that $\delta(G[S]) \geq d$?

- MSMD_d, $d \geq 3$, is W[1]-hard.

  (and thus the problem is not likely to be FPT in general graphs)

- FPT algorithms for minor free graphs.

  (for instance: planar graphs, graphs of bounded local treewidth, graphs of bounded genus,...)

- **Open problem**: find approximation algorithms for general graphs.

Conclusions

- We have proved that MSMD_d, $d \geq 3$, is not in APX.
- We have an $\mathcal{O}(n/\log n)$-approximation for minor free graphs.

- **Parameterized version of MSMD_d:**

  Input: an undirected graph $G = (V, E)$, an integer $d \geq 3$, and a parameter k.

  Question: does there exist $S \subseteq V$, with $|S| \leq k$, such that $\delta(G[S]) \geq d$?

- MSMD_d, $d \geq 3$, is W[1]-hard.

  (and thus the problem is not likely to be FPT in general graphs)

- FPT algorithms for minor free graphs.

  (for instance: planar graphs, graphs of bounded local treewidth, graphs of bounded genus,...)

- **Open problem**: find approximation algorithms for general graphs.

Conclusions

- We have proved that MSMD_d, $d \geq 3$, is not in APX.
- We have an $\mathcal{O}(n/\log n)$-approximation for minor free graphs.

- **Parameterized version of MSMD_d:**

  Input: an undirected graph $G = (V, E)$, an integer $d \geq 3$, and a parameter k.

  Question: does there exist $S \subseteq V$, with $|S| \leq k$, such that $\delta(G[S]) \geq d$?

- MSMD_d, $d \geq 3$, is W[1]-hard.

  (and thus the problem is not likely to be FPT in general graphs)

- FPT algorithms for minor free graphs.

  (for instance: planar graphs, graphs of bounded local treewidth, graphs of bounded genus,...)

- **Open problem**: find approximation algorithms for general graphs.

Conclusions

- We have proved that MSMD_d, $d \geq 3$, is not in APX.
- We have an $\mathcal{O}(n/\log n)$-approximation for minor free graphs.
- **Parameterized version of MSMD_d:**

  Input: an undirected graph $G = (V, E)$, an integer $d \geq 3$, and a parameter k.

  Question: does there exist $S \subseteq V$, with $|S| \leq k$, such that $\delta(G[S]) \geq d$?
- MSMD_d, $d \geq 3$, is W[1]-hard.

  (and thus the problem is not likely to be FPT in general graphs)
- FPT algorithms for minor free graphs.

  (for instance: planar graphs, graphs of bounded local treewidth, graphs of bounded genus,...)
- **Open problem**: find approximation algorithms for general graphs.

Conclusions

- We have proved that MSMD_d, $d \geq 3$, is not in APX.
- We have an $\mathcal{O}(n/\log n)$-approximation for minor free graphs.

- **Parameterized version of MSMD_d:**

  Input: an undirected graph $G = (V, E)$, an integer $d \geq 3$, and a parameter k.

  Question: does there exist $S \subseteq V$, with $|S| \leq k$, such that $\delta(G[S]) \geq d$?

- MSMD_d, $d \geq 3$, is W[1]-hard.

  (and thus the problem is not likely to be FPT in general graphs)

- FPT algorithms for minor free graphs.

  (for instance: planar graphs, graphs of bounded local treewidth, graphs of bounded genus,...)

- **Open problem**: find approximation algorithms for general graphs.

Conclusions

- We have proved that MSMD_d, $d \geq 3$, is not in APX.
- We have an $\mathcal{O}(n/\log n)$-approximation for minor free graphs.
- **Parameterized version of MSMD_d:**

  Input: an undirected graph $G = (V, E)$, an integer $d \geq 3$, and a parameter k.

  Question: does there exist $S \subseteq V$, with $|S| \leq k$, such that $\delta(G[S]) \geq d$?

- MSMD_d, $d \geq 3$, is W[1]-hard.

  (and thus the problem is not likely to be FPT in general graphs)

- FPT algorithms for minor free graphs.

  (for instance: planar graphs, graphs of bounded local treewidth, graphs of bounded genus,...)

- **Open problem**: find approximation algorithms for general graphs.

Thanks!