Generalized Partial Vertex Cover

Sayan Bandyapadhyay [1]
Zachary Friggstad [2] and Ramin Mousavi [2]

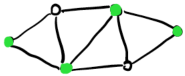
[1]Portland State University

[2]University of Alberta

WADS 2023

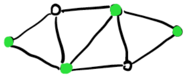
Partial vertex cover

Vertex cover: choose k vertices that cover all the edges.



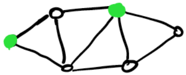
Partial vertex cover

Vertex cover: choose k vertices that cover all the edges.



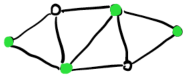
Partial vertex cover: choose k vertices that cover at least c edges.
c is the coverage requirement.

c = 5

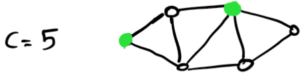


Partial vertex cover

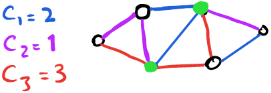
Vertex cover: choose k vertices that cover all the edges.



Partial vertex cover: choose k vertices that cover at least c edges.
c is the coverage requirement.



Generalized partial vertex cover: choose k vertices that cover at least c_1 edges from E_1, at least c_2 edges from E_2,..., at least c_m edges from E_m.
E_i is color class i and $E = E_1 \uplus E_2 \ldots \uplus E_m$.



Motivation

- Vertex cover:
  1. 2-approx, folklore.
  2. FPT in k, i.e., solvable in $f(k) \cdot \mathrm{poly}(n)$ time, Chen et al. 2005.

Motivation

- Vertex cover:
  1. 2-approx, folklore.
  2. FPT in k, i.e., solvable in $f(k) \cdot \mathrm{poly}(n)$ time, Chen et al. 2005.
- Partial vertex cover:
  1. 2-approx on k, Bshouty and Burroughs 1998.
  2. 3/4-approx on c, Ageev and Sviridenko 1999.
  3. FPT in c, Bläser 2003.
  4. no FPT in k, Guo et al. 2005.
  5. APX-hard on c Petrank 1994.
  6. $1 - \epsilon$ approx on c in time $f(k, \epsilon) \cdot \mathrm{poly}(n)$, Marx 2008.

Motivation

- Vertex cover:
  1. 2-approx, folklore.
  2. FPT in k, i.e., solvable in $f(k) \cdot \mathrm{poly}(n)$ time, Chen et al. 2005.
- Partial vertex cover:
  1. 2-approx on k, Bshouty and Burroughs 1998.
  2. $3/4$-approx on c, Ageev and Sviridenko 1999.
  3. FPT in c, Bläser 2003.
  4. no FPT in k, Guo et al. 2005.
  5. APX-hard on c Petrank 1994.
  6. $1 - \epsilon$ approx on c in time $f(k, \epsilon) \cdot \mathrm{poly}(n)$, Marx 2008.

For generalized partial vertex cover, what can be achieved?

What can be achieved?

Known results:

- $O(\log m)$-approx on k where m is the number of color classes. Tight! Bera et al. 2014.
- ≈ 2-approx on k for constant m. Bandyapadhyay et al. 2021.
- find k vertices that cover at least 0.63% of each coverage requirements if m is constant, Chekuri et al. 2009.
- no FPT in k.
- APX-hard even for $m = 1$. So no $(1-\epsilon)$-approx on c_i's in time $f(m, \epsilon) \cdot \mathrm{poly}(n)$.

Theorem (Bandyapadhyay, Friggstad, and M.)

1. *no α-approx for $\alpha \leq 1$ in time $f(k) \cdot n^{o(k)}$ assuming ETH.*
2. *$(1-\epsilon)$-approx on c_i's in time $2^{O(\frac{m \cdot k^2 \cdot \log k}{\epsilon})} \cdot \mathrm{poly}(n)$.*

Main tool

FPT approximation scheme for partial vertex cover.
Partial vertex cover: find k vertices that cover at least c edges.

- if c is small $(< \frac{\binom{k}{2}}{\epsilon})$:

Main tool

FPT approximation scheme for partial vertex cover.
Partial vertex cover: find k vertices that cover at least c edges.

- if c is small ($< \frac{\binom{k}{2}}{\epsilon}$):
  1. label coding: randomly assign one label to each edge among c many labels

Main tool

FPT approximation scheme for partial vertex cover.
Partial vertex cover: find k vertices that cover at least c edges.

- if c is small ($< \frac{\binom{k}{2}}{\epsilon}$):
  1. label coding: randomly assign one label to each edge among c many labels
  2. with "good" probability the OPT edges are all labeled differently

Main tool

FPT approximation scheme for partial vertex cover.
Partial vertex cover: find k vertices that cover at least c edges.

- if c is small $\left(< \frac{\binom{k}{2}}{\epsilon}\right)$:
  1. label coding: randomly assign one label to each edge among c many labels
  2. with "good" probability the OPT edges are all labeled differently
  3. guess all the configuration of vertices in OPT. k^c many guesses

Main tool

FPT approximation scheme for partial vertex cover.
Partial vertex cover: find k vertices that cover at least c edges.

- if c is small ($< \frac{\binom{k}{2}}{\epsilon}$):
  1. label coding: randomly assign one label to each edge among c many labels
  2. with "good" probability the OPT edges are all labeled differently
  3. guess all the configuration of vertices in OPT. k^c many guesses
  4. brute-force

Main tool

FPT approximation scheme for partial vertex cover.
Partial vertex cover: find k vertices that cover at least c edges.

- if c is small ($< \frac{\binom{k}{2}}{\epsilon}$):
  1. label coding: randomly assign one label to each edge among c many labels
  2. with "good" probability the OPT edges are all labeled differently
  3. guess all the configuration of vertices in OPT. k^c many guesses
  4. brute-force
- if c is large ($\geq \frac{\binom{k}{2}}{\epsilon}$):

Main tool

FPT approximation scheme for partial vertex cover.
Partial vertex cover: find k vertices that cover at least c edges.

- if c is small ($< \frac{\binom{k}{2}}{\epsilon}$):
  1. label coding: randomly assign one label to each edge among c many labels
  2. with "good" probability the OPT edges are all labeled differently
  3. guess all the configuration of vertices in OPT. k^c many guesses
  4. brute-force
- if c is large ($\geq \frac{\binom{k}{2}}{\epsilon}$):
  1. $d_1 \geq d_2 \ldots \geq d_n$

Main tool

FPT approximation scheme for partial vertex cover.
Partial vertex cover: find k vertices that cover at least c edges.

- if c is small ($< \frac{\binom{k}{2}}{\epsilon}$):
  1. label coding: randomly assign one label to each edge among c many labels
  2. with "good" probability the OPT edges are all labeled differently
  3. guess all the configuration of vertices in OPT. k^c many guesses
  4. brute-force
- if c is large ($\geq \frac{\binom{k}{2}}{\epsilon}$):
  1. $d_1 \geq d_2 \ldots \geq d_n$
  2. choose the first k vertices

Main tool

FPT approximation scheme for partial vertex cover.
Partial vertex cover: find k vertices that cover at least c edges.

- if c is small $(< \frac{\binom{k}{2}}{\epsilon})$:
  1. label coding: randomly assign one label to each edge among c many labels
  2. with "good" probability the OPT edges are all labeled differently
  3. guess all the configuration of vertices in OPT. k^c many guesses
  4. brute-force
- if c is large $(\geq \frac{\binom{k}{2}}{\epsilon})$:
  1. $d_1 \geq d_2 \ldots \geq d_n$
  2. choose the first k vertices
  3. $\sum_{i=1}^{k} d_i \geq c$ (edges are double counted)

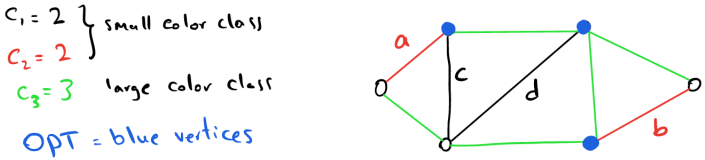
Main tool

FPT approximation scheme for partial vertex cover.
Partial vertex cover: find k vertices that cover at least c edges.

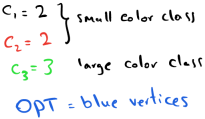
- if c is small ($< \frac{\binom{k}{2}}{\epsilon}$):
  1. label coding: randomly assign one label to each edge among c many labels
  2. with "good" probability the OPT edges are all labeled differently
  3. guess all the configuration of vertices in OPT. k^c many guesses
  4. brute-force
- if c is large ($\geq \frac{\binom{k}{2}}{\epsilon}$):
  1. $d_1 \geq d_2 \ldots \geq d_n$
  2. choose the first k vertices
  3. $\sum_{i=1}^{k} d_i \geq c$ (edges are double counted)
  4. the total number of double counted edges is at most $\binom{k}{2} \leq \epsilon \cdot c$

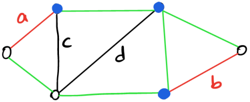
Our algorithm, small color classes

for small color classes $< \frac{\binom{k}{2}}{\epsilon}$ do label-coding as before and guess the configuration of the vertices in OPT on these color classes.

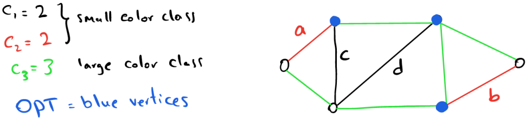


Our algorithm, large color classes





Our algorithm, large color classes



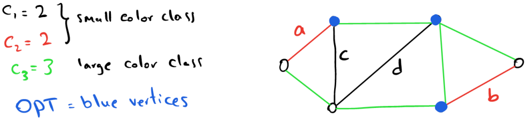
Given the configuration of small color classes in OPT, we use DP to find a solution.

c': the coverage requirements for a subset of large color classes

T: the configuration of some small color classes in OPT.

Order vertices $v_1, v_2, v_3, \ldots, v_n$

Our algorithm, large color classes



Given the configuration of small color classes in OPT, we use DP to find a solution.

c': the coverage requirements for a subset of large color classes

T: the configuration of some small color classes in OPT.

Order vertices $v_1, v_2, v_3, \ldots, v_n$

$g(k', i, T, c') =$ *yes iif there are* k' *vertices among the first* i *vertices compatible with* T *and cover* c' *on large color classes*

Running time

Let's look at our DP table:

$$g(k', i, T, c')$$

T: the configuration of some small color classes in OPT.

c': the coverage requirements for a subset of large color classes

Running time

Let's look at our DP table:

$$g(k', i, T, c')$$

T: the configuration of some small color classes in OPT.
c': the coverage requirements for a subset of large color classes

- The number of possibilities for T is $f(k, m, \epsilon)$.

Running time

Let's look at our DP table:

$$g(k', i, T, c')$$

T: the configuration of some small color classes in OPT.
c': the coverage requirements for a subset of large color classes

- The number of possibilities for T is $f(k, m, \epsilon)$.
- The number of possibilities for c' is $(\text{the number of edges})^m$.
  Not FPT in m :(

Running time

Let's look at our DP table:

$$g(k', i, T, c')$$

T: the configuration of some small color classes in OPT.
c': the coverage requirements for a subset of large color classes

- The number of possibilities for T is $f(k, m, \epsilon)$.
- The number of possibilities for c' is $(\text{the number of edges})^m$. Not FPT in m :(
- Standard scaling and rounding the degree of vertices in large color classes. Then, the number of possibilities reduces to $(\frac{k}{\epsilon})^m$.

Running time

Let's look at our DP table:

$$g(k', i, T, c')$$

T: the configuration of some small color classes in OPT.
c': the coverage requirements for a subset of large color classes

- The number of possibilities for T is $f(k, m, \epsilon)$.
- The number of possibilities for c' is (the number of edges)^m. Not FPT in m :(
- Standard scaling and rounding the degree of vertices in large color classes. Then, the number of possibilities reduces to $(\frac{k}{\epsilon})^m$.

Conclusion: we have a tight FPT approximation scheme in parameters k, m, and ϵ.

Running time

Let's look at our DP table:

$$g(k', i, T, c')$$

T: the configuration of some small color classes in OPT.
c': the coverage requirements for a subset of large color classes

- The number of possibilities for T is $f(k, m, \epsilon)$.
- The number of possibilities for c' is (the number of edges)^m. Not FPT in m :(
- Standard scaling and rounding the degree of vertices in large color classes. Then, the number of possibilities reduces to $(\frac{k}{\epsilon})^m$.

Conclusion: we have a tight FPT approximation scheme in parameters k, m, and ϵ.

THANK YOU!