Vertex Covers Revisited:

New and Simple FPT Algorithms

CAI Leizhen

Chinese Univ of Hong Kong

Outline

- Introduction
- Iterative Compression
- Color Coding
- Random Separation (Indirect Certificate)
- Conclusion

Introduction

Vertex Cover

Input: Graph G = (V,E), parameter k.

Question: Does G contain k vertices that cover all edges?

Vertex v covers edge e if v is incident with e.

FPT algorithm: f(k)$n^{O(1)}$ time.


BAHAMAS
JAMAICA


9
8
7
4
BAILEY
OMEGA
© PA


© Sergei Gladyshev/Hotspot Media

Introduction

Known FPT algorithms

Bounded search tree

- For any edge uv,
  either u or v must be in a solution → $O(2^k kn)$
- Path P_3 → $O(1.618^k kn)$
- Vertex of degree at least 3 → $O(1.5^k kn)$
- Chan, Kanj, and Xia (2010) → $O(1.2738^k + kn)$

Introduction

Kernelization

Vertex of degree > k must be in soln → O(k^2)

Crown decomposition → 3k vertices

Linear programming → 2k vertices

Introduction

Matching

Papadimitriou and Yannakakis (1993) → O(3^kkn)

Graph minor

Fellow and Langston (1986) → O(f(k)n^3)

f(k) astronomical

Johnson (1987) → O(f(k)n^2)

f(k) ≈ $2^{2^{500k}}$

Introduction

化腐朽为神奇

Do bad things in clever ways

- Iterative compression
- Color coding
- Random separation (Indirect Certificate)
- Representative sets
- Randomized divide and conquer

Introduction

Motivations

- Better understanding
- Training students
- Intellectually challenging

Iterative Compression

Reed, Smith, and Vetta (2004)

Idea: Given a (k+1)-solution, obtain a k-solution in FPT time.

For vertex cover:

Given a vertex cover X of G,
is there a smaller vertex cover for G?

Iterative Compression

Characterization of minimum vertex cover

Theorem 1 (Cai 2005). A vertex cover X of G = (V, E) is a minimum vertex cover iff for every independent set S in G[X], $|N^*(S)| \geq |S|$.

Outside neighborhood $N^*(S)$: $N(S) \cap (V - X)$.

Iterative Compression

Compression routine: determine whether there a vertex cover smaller than a given (k+1)-vertex cover X.

- Consider every independent set S in X.
- If for some S, |N*(S)| < |S|, then we obtain a smaller vertex cover. Otherwise no k-vertex cover.

Time: $O(2^k kn)$.

Iterative Compression

Another characterization:

Theorem 2. A vertex cover X of G is a minimum vertex cover iff for every maximal independent set S in G[X], S is a minimum vertex cover of G[S U N*(S)].

Theorem 3 (Moon and Moser 1965)

An n-vertex graph contains at most $3^{n/3}$ maximal independent sets.

Iterative Compression

Compression routine: X = (k+1)-vertex cover of G.

- Consider every maximal independent set S in X, and find a minimum vertex cover X(S) of G[S U N*(S)].
- If for some S, X(S) is smaller than X, then we obtain a smaller vertex cover. Otherwise no k-vertex cover.

Time: $O(3^{k/3}n^{2.5})$.

Iterative Compression

Three ways to obtain a (k+1)-vertex cover:

- Recursively: Arbitrarily choose a vertex v, and recursively solve the problem for G – v to obtain a k-vertex cover X of G – v. Then X + v is a (k+1)-vertex cover of G.

- Iteratively: Order vertices as $v_1, \ldots, v_n$, and set G_i = G[$v_1, \ldots, v_i$].
  A k-vertex cover X of G_i yields a (k+1)-vertex cover X + v_{i+1} of G_{i+1},
  which is then compressed into a k-vertex cover of G_{i+1}.

- Approximation: Use 2-approximation algorithm for Vertex Cover to find a k'-vertex cover.
  Then k' ≤ 2k.

Color Coding

Alon, Yuster, and Zwick (1995)

Idea: Randomly color elements and find a colorful k-solution in FPT time.

For vertex cover:

Given a vertex k-colored graph, determine whether the graph contains a colorful vertex cover.

Color Coding

G = (V, E; c): vertex colored graph with c: V → {1, …, t}.

Color class V_i: vertices with color i.

Colorful vertex cover X: all vertices in vertex cover X have distinct colors, i.e., X contains at most one vertex from each color class.

Color Coding

Colorful Vertex Cover

Instance: Vertex colored graph G.

Question: Does G contain a colorful vertex cover X?

NP-complete for following two variations:

- X contains at most 2 vertices from each color class.
- Size of X is upper bounded.

Color Coding

No easier than 2SAT.

Polynomial-time algorithms

- by reduction to 2SAT
- using ideas for 2SAT.

Color Coding

For vertex v, let x_v be its corresponding Boolean variable.

- For edges E, define F(E) = $\Lambda_{uv \in E}\ (x_u \vee x_v)$.
- For each color class V_i, define F(V_i) =$\Lambda_{\text{distinct } u,v \in V_i} \overline{x_u \wedge x_v}$.

Set F(G) = F(E) ∧ ($\Lambda_{i=1}^{t} F(V_i)$).

Color Coding

Theorem 4. Vertex colored graph G admits a colorful vertex cover iff its corresponding formula F(G) is satisfiable. → O(n^2) algorithm

Proof. For vertices v, determine $x_v \in \{0,1\}$ such that

- for every edge uv, $x_u + x_v \geq 1$, and
- for each color class V_i, $\sum_{v \in V_i} x_v \leq 1$, equivalent to, for every distinct u, v in V_i, $x_u + x_v \leq 1$.

Note x + y ≥ 1 equivalent to $x \vee y$, and
x + y ≤ 1 equivalent to $\overline{x} \vee \overline{y}$. ∎

Color Coding

Linear-time algorithm:

Let u be a vertex with a value → force values on other vertices.

Case x_u = 0: for every edge uv

if x_v = 0 then conflict and stop

else set x_v = 1.

Case x_u = 1: for every other vertex v in the color class containing u

if x_v = 1 then conflict and stop

else set x_v = 0.

Color Coding

For a vertex v, define F_x(v), where x ∈ {0, 1}, to be the set of vertices that are forced to receive a value when x_v = x.

Set F_x(v) = ⦸ if x_v = x causes a conflict of values for the forced vertices.

Lemma 5. If F_x(v) ≠ ⦸ then G admits a colorful vertex cover iff G − F_x(v) admits one.

Color Coding

Repeat the following until G is empty:

Arbitrarily choose a vertex v from G;

In parallel,

compute F_0(v), if F_0(v) not empty then set G $\leftarrow$ G $-$ F_0(v) and assign values to vertices in F_0(v) accordingly;

compute F_1(v), if F_1(v) not empty then set G $\leftarrow$ G $-$ F_1(v) and assign values to vertices in F_1(v) accordingly;

If both F_0(v) and F_1(v) empty then “No Solution”

Note: Parallel part can be simulated sequentially by dovetailing.

Random Separation → Indirect Certificate

Cai, Chan and Chan (2006)

Idea: Randomly partition vertex set of G into two disjoint sets to separate a solution S from the rest of G into connected components, and then select appropriate components to form a solution.

- Typically, we want S to be blue and N(S) to be red.
- The probability of obtaining such a partition is $2^{-(k+p)}$ where p = |N(S)|.
- For probability to be a function of k, G need to have bounded degree → the method usually works for degree-bounded graphs only.

Indirect Certificate

More general than random separation

Idea: Randomly partition elements into two disjoint parts, and use a small structure, called indirect certificate, in one part to obtain a solution in another part in FPT time.

Small structure: size bounded above by a function of k.

Indirect Certificate

First used by Cai, Chan, and Chan, 2006

- Maximum weight k-Independent set in planar graphs.
- Induced k-path, k-cycle for graphs with bounded degeneracy.

Cygan et. al., 2014

- Eulerian deletion.

Cai and Ye, 2016.

- Edge-disjoint paths with length constraints.

Indirect Certificate

Theorem 6. For any vertex cover X of a graph G = (V,E), V – X contains at most |X| vertices C such that N(C) resides in X and forms a vertex cover.

C is an **indirect certificate** and can be used to find a k-vertex cover in polynomial time.

Indirect Certificate

Proof.

- X*: minimum vertex cover inside X.
- Every vertex in X* covers some edge in cut [X*, V –X*].
- For each vertex v in X*, arbitrarily choose an adjacent vertex c(v) in V – X*.
- C = {c(v) : v ∈ X*}, and N(C) has required properties. ∎

Indirect Certificate

Step 1. Randomly and independently color each vertex either red or blue with probability ½ to form red vertices R.

Step 2. Return N(R) as a solution if N(R) contains blue vertices only and forms a k-vertex cover; otherwise return ``No solution".

Note: Algorithm can be derandomized by (n,2k)-universal sets.

Theorem 7. The algorithm finds, with probability at least 4^{-k}, a k-vertex cover of G, if it exists, in O(m + n) time.

Indirect Certificate

Proof. Suppose G has a k-vertex cover X.

- By Theorem 6, V – X contains at most k vertices C dominating a vertex cover inside X.
  Step 1 colors, with probability at least 4^{-k}, X blue and C red.

- Step 2 correctly finds a k-vertex cover consisting of blue vertices only.
  By Theorem 6, X contains a k-vertex cover X* consisting of blue vertices only such that

  $$X^* \subseteq N(C) \subseteq N(R)$$

  N(R) consists of blue vertices only and all vertices in N(R) are needed to cover all crossing edges between blue and red vertices. ∎

Indirect Certificate: Edge-Disjoint Paths

Finding two edge-disjoint (*s*,*t*)-paths in a graph with one path of length ≤ *k*.
Joint work with YE Junjie

Is there an (s,t)-path P of length ≤ k whose deletion does not disconnect s and t?

Indirect certificate for P: O(k^2) special edges outside P.

Indirect Certificate: Edge-Disjoint Paths

Vertex v is a *nearby-vertex* if $d(s, v) + d(v, t) \leq k$, and an edge is a *nearby-edge* if its two endpoints are nearby-vertices.

Lemma 13. Let P be an (s,t)-path of length at most k, and Q a minimum length (s,t)-path edge-disjoint from P. Then

(1) all edges of P are nearby-edges, and

(2) Q contains at most $(k+1)^2$ nearby-edges.

Indirect Certificate: Edge-Disjoint Paths

Algorithm

1. Find all nearby-edges by BFS.

2. With probability ½, randomly color each nearby-edge blue or red.
   Color remaining edges red.

3. If blue graph contains an (s,t)-path P of length $\leq k$, and red graph contains an (s,t)-path Q, then P and Q form a solution. Otherwise no solution.

Conclusion

- Ideas useful for special vertex covers
- Polynomial algorithms for colorful solutions of other problems
- High potential for indirect certificate method