Vector connectivity in graphs

Martin Milanič

UP IAM and UP FAMNIT, University of Primorska

Raziskovalni matematični seminar, UP FAMNIT, 22. december 2014

The talk is based on joint works with

Endre Boros, Rutgers University

Pinar Heggernes, University of Bergen

Pim van 't Hof, University of Bergen
$\rightarrow$ Rotterdam University of Applied Sciences

Ferdinando Cicalese, University of Salerno
$\rightarrow$ University of Verona

Romeo Rizzi, University of Verona

Vector domination in graphs

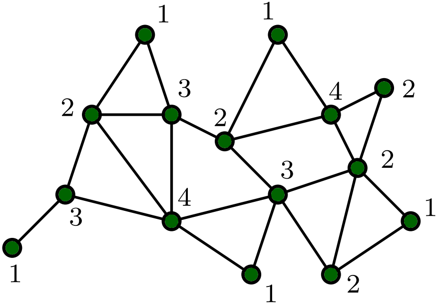
Given: a graph $G = (V, E)$
For every vertex v, an integer $r(v)$

A set $S \subseteq V$ is a vector dominating set for (G, r) if every vertex in $V \setminus S$ has at least $r(v)$ neighbors in S.

Vector domination in graphs

Given: a graph $G = (V, E)$
For every vertex v, an integer $r(v)$

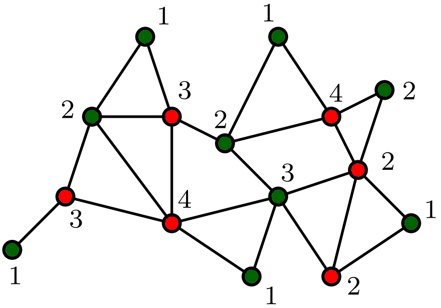
A set $S \subseteq V$ is a vector dominating set for (G, r) if every vertex in $V \setminus S$ has at least $r(v)$ neighbors in S.



Vector domination in graphs

Given: a graph $G = (V, E)$
For every vertex v, an integer $r(v)$

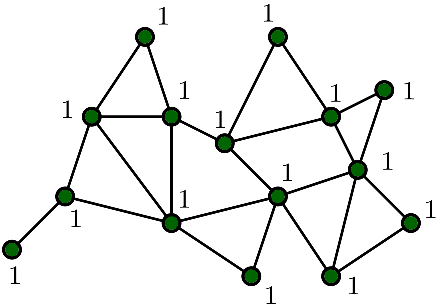
A set $S \subseteq V$ is a vector dominating set for (G, r) if every vertex in $V \setminus S$ has at least $r(v)$ neighbors in S.



Vector domination in graphs

Vector domination generalizes:

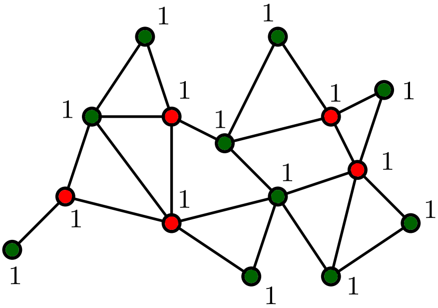
- **domination:** $r(v) = 1$ for all v



Vector domination in graphs

Vector domination generalizes:

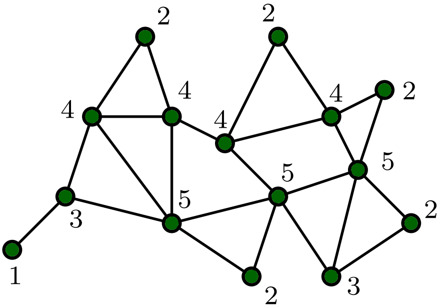
- **domination:** $r(v) = 1$ for all v



Vector domination in graphs

Vector domination generalizes:

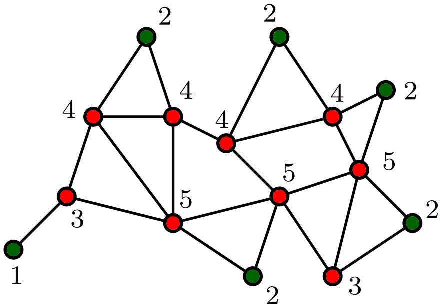
- domination: $r(v) = 1$ for all v
- **vertex cover:** $r(v) = d(v)$ for all v



Vector domination in graphs

Vector domination generalizes:

- domination: $r(v) = 1$ for all v
- **vertex cover:** $r(v) = d(v)$ for all v



Vector connectivity in graphs

Relaxation: edges $\rightsquigarrow$ paths

Given: a graph $G = (V, E)$
For every vertex v, an integer $r(v)$

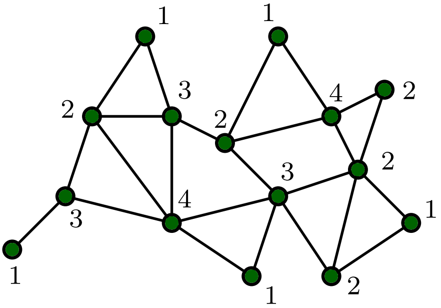
A set $S \subseteq V$ is a vector connectivity set for (G, r) if every vertex in $V \setminus S$ has at least $r(v)$ disjoint paths to S.

Vector connectivity in graphs

Relaxation: edges $\rightsquigarrow$ paths

Given: a graph $G = (V, E)$
For every vertex v, an integer $r(v)$

A set $S \subseteq V$ is a vector connectivity set for (G, r) if every vertex in $V \setminus S$ has at least $r(v)$ disjoint paths to S.

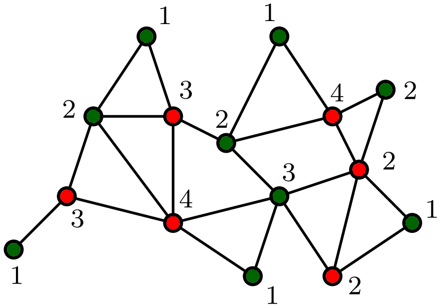


Vector connectivity in graphs

Relaxation: edges $\rightsquigarrow$ paths

Given: a graph $G = (V, E)$
For every vertex v, an integer $r(v)$

A set $S \subseteq V$ is a vector connectivity set for (G, r) if every vertex in $V \setminus S$ has at least $r(v)$ disjoint paths to S.

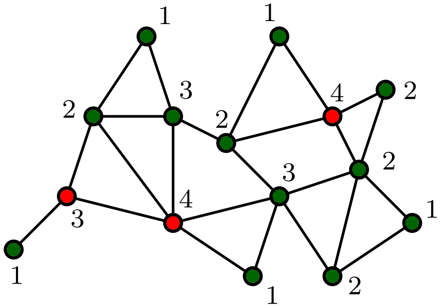


Vector connectivity in graphs

Relaxation: edges $\rightsquigarrow$ paths

Given: a graph $G = (V, E)$
For every vertex v, an integer $r(v)$

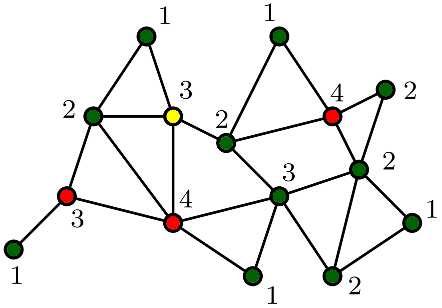
A set $S \subseteq V$ is a vector connectivity set for (G, r) if every vertex in $V \setminus S$ has at least $r(v)$ disjoint paths to S.



Relaxation: edges $\rightsquigarrow$ paths

Given: a graph $G = (V, E)$
For every vertex v, an integer $r(v)$

A set $S \subseteq V$ is a vector connectivity set for (G, r) if every vertex in $V \setminus S$ has at least $r(v)$ disjoint paths to S.

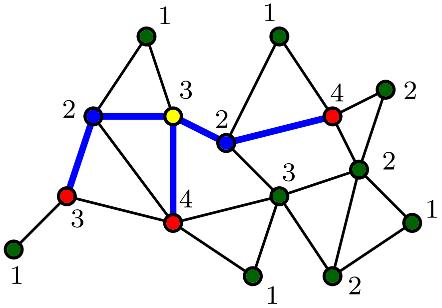


Vector connectivity in graphs

Relaxation: edges $\rightsquigarrow$ paths

Given: a graph $G = (V, E)$
For every vertex v, an integer $r(v)$

A set $S \subseteq V$ is a vector connectivity set for (G, r) if every vertex in $V \setminus S$ has at least $r(v)$ disjoint paths to S.

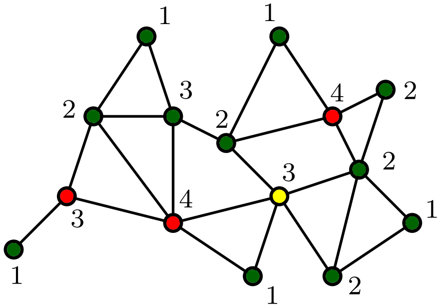


Vector connectivity in graphs

Relaxation: edges $\rightsquigarrow$ paths

Given: a graph $G = (V, E)$
For every vertex v, an integer $r(v)$

A set $S \subseteq V$ is a vector connectivity set for (G, r) if every vertex in $V \setminus S$ has at least $r(v)$ disjoint paths to S.

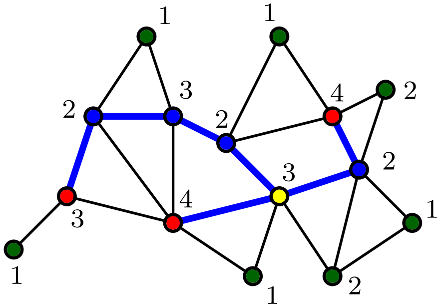


Vector connectivity in graphs

Relaxation: edges $\rightsquigarrow$ paths

Given: a graph $G = (V, E)$
For every vertex v, an integer $r(v)$

A set $S \subseteq V$ is a vector connectivity set for (G, r) if every vertex in $V \setminus S$ has at least $r(v)$ disjoint paths to S.



The problem

Vector connectivity:

Input: A graph $G = (V, E)$ and $r(v) \in \mathbb{Z}_+$ for each $v \in V$

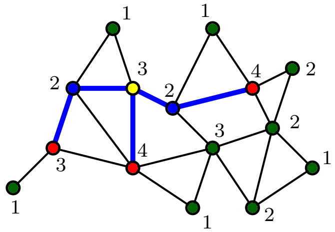
Task: Find a minimum vector connectivity set for $(G, \boldsymbol{r})$.

Introduced in: Boros, Heggernes, van 't Hof, M.
TAMC 2013, Networks 2014

The talk is also based on:
Cicalese, M., Rizzi, 2014, On the complexity of the vector connectivity problem, http://arxiv.org/abs/1412.2559

Application 1 – viral marketing

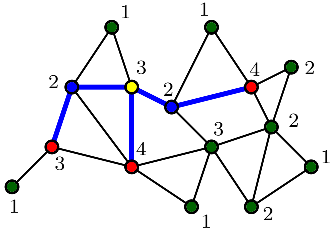
Suppose that the graph models a social network.



- In a viral marketing campaign one of the problems is to identify a set of targets in the network that can be influenced (e.g., on the goodness of a product) and such that from them most/all the network can be influenced (e.g., convinced to buy the product).
- Usual assumption: each vertex has a threshold $r(v)$ such that when $r(v)$ neighbors are influenced, v will get influenced too.

Application 1 – viral marketing

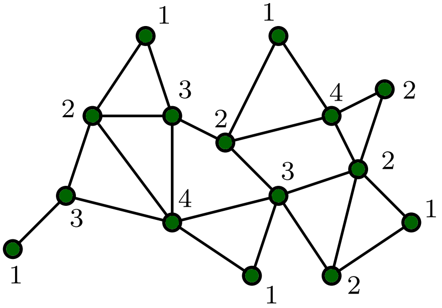
- Assume now that for v it is not enough that $r(v)$ neighbors are convinced about the product. Vertex v also requires that **their motivations are independent.**



- A customer will buy the product if he/she has at least $r(v)$ "independent" ways of learning about it.
- A minimum set of customers that the company should 'target' directly so that all individuals in the network will buy the product is exactly a minimum vector connectivity set.

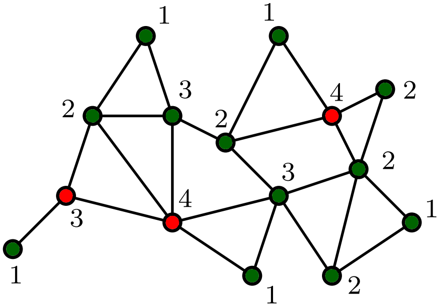
Application 2 – warehouse placement

- Each vertex in a network produces a certain amount of a given good.



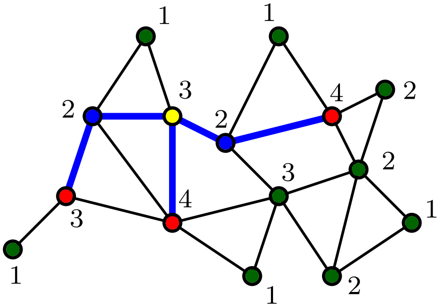
Application 2 – warehouse placement

- We want to place in the network warehouses where the good can be stored.



Application 2 – warehouse placement

- For security/resilience reasons it is better if from each source to each destination (warehouse) only a small amount of the good (e.g., one unit) travels at once.
- In particular, it is preferred if the units of good **from one location to the different warehouses travel on different routes.**



Application 2 – warehouse placement

- This reduces the risk that if delivery gets intercepted or attacked or interrupted by a fault on the network a large amount of the good gets lost.
- Finding the minimum number of warehouses given the amount of units produced at each vertex coincides with the vector connectivity problem.

Overview of the talk

1. A characterization of vector connectivity sets
   Cicalese et al., 2014
   - vector connectivity sets as hitting sets of a derived hypergraph

2. An approximation algorithm
   Boros et al., 2014
   - polynomial time algorithm approximating vector connectivity within a factor of $\log n + 2$

3. Inapproximability result Cicalese et al., 2014
   - the problem is APX-hard (no PTAS unless $\mathrm{P} = \mathrm{NP}$)

4. Exact polynomial algorithms for special graph classes
   Boros et al., 2014, Cicalese et al., 2014
   - trees, cographs, split graphs, block graphs

A CHARACTERIZATION OF VECTOR CONNECTIVITY SETS

A characterization

Recall:

- A **hypergraph** is a pair $H = (V, \mathcal{F})$ where V is a finite set and $\mathcal{F}$ is a set of subsets of V (called *hyperedges*).
- A **hitting set** (or: **transversal**) of a hypergraph H is a subset $S \subseteq V$ such that $S \cap X \neq \emptyset$ for all $X \in \mathcal{F}$.

vertex covers of a graph G = hitting sets of G

dominating sets = hitting sets of the closed neighborhood hypergraph of G

vector dominating sets = ...

etc.

A characterization

Menger's Theorem can be used to derive a similar characterization of vector connectivity sets.

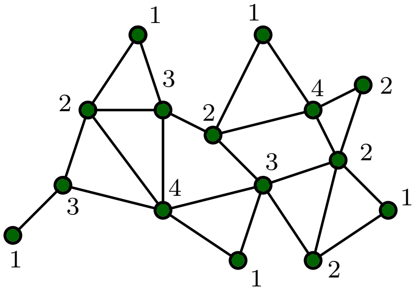
Proposition

For every graph $G = (V, E)$, vertex requirements $r : V \to \mathbb{Z}_+$, and a set $S \subseteq V$, the following conditions are equivalent:

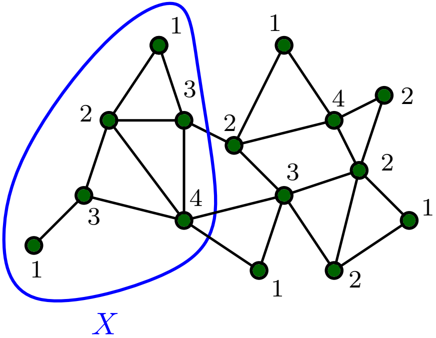
1. *S is a vector connectivity set for (G, r).*
2. *S is the hitting set of the hypergraph consisting of all non-empty sets $X \subseteq V$ such that $G[X]$ is connected and $|N(X)| < R(X)$.*

Here, $R(X) := \max_{x \in X} r(x)$.

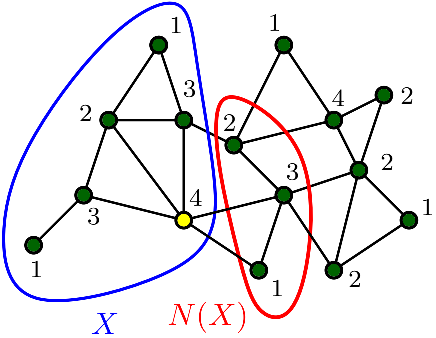
A characterization – example



A characterization – example

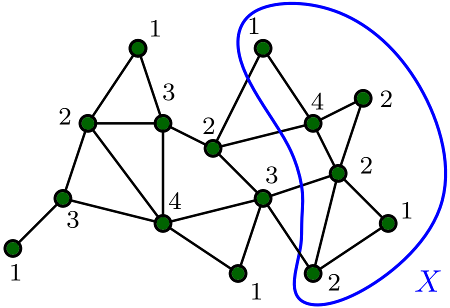


A characterization – example

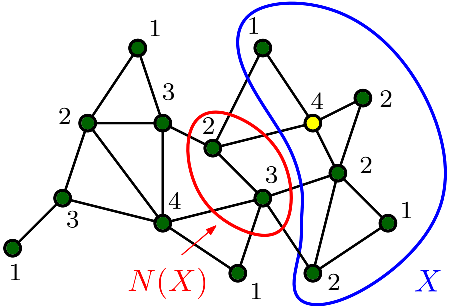


$$\max_{v \in V(X)} r(v) = 4 > |N(X)|$$

A characterization – example



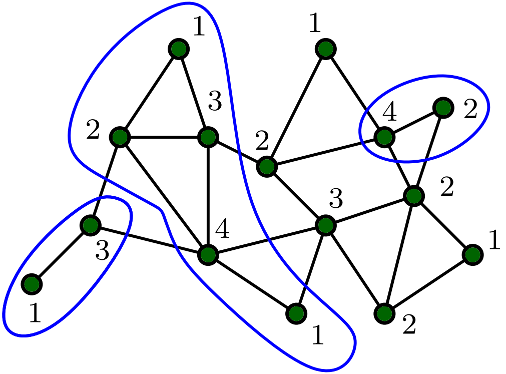
A characterization – example



$$\max_{v \in V(X)} r(v) = 4 > |N(X)|$$

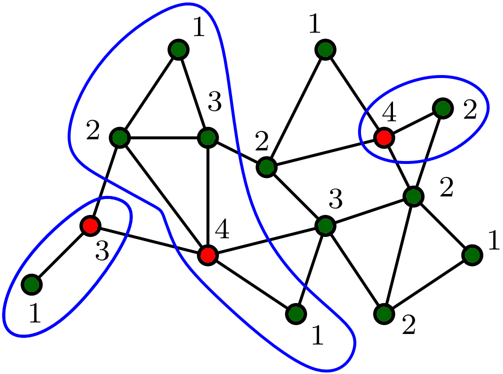
A characterization – example

Every vector connectivity set contains at least one vertex from each of the following three sets:



A characterization – example

It follows that the solution consisting of the three red vertices below is optimal.



AN APPROXIMATION ALGORITHM

Approximating vector connectivity

Greedy Strategy

start with $S = \emptyset$

if S is not a vector connectivity set, keep on adding to S a vertex $v \in V \setminus S$ maximizing $f(S \cup \{v\}) - f(S)$

$$\mathrm{argmax}_{v \in V} \left(f(S \cup \{v\}) - f(S)\right)$$

$$f(X) = \sum_{v \in V} f_v(X)\,,\ \text{for all } X \subseteq V, \text{ and}$$

$$f_v(X) = \begin{cases} \min\{\sigma(v, X), r(v)\} & \text{if } v \notin X; \\ r(v) & \text{if } v \in X. \end{cases}$$

$\sigma(v, X) =$ maximum number of disjoint v-X paths

Approximating vector connectivity

$$f(X) = \sum_{v \in V} f_v(X)\,, \text{ for all } X \subseteq V, \text{ and}$$

$$f_v(X) = \begin{cases} \min\{\sigma(v, X), r(v)\} & \text{if } v \notin X; \\ r(v) & \text{if } v \in X. \end{cases}$$

Note that:

- $f(V) = \sum_{v \in V} r(v)$
- $f(X) = f(V)$ if and only if $X \subseteq V$ is a vector connectivity set for (G, r).
- Hence, the vector connectivity problem asks for a smallest set $X \subseteq V$ with $f(X) = f(V)$.

Approximating vector connectivity

It can be shown that f is a (non-decreasing, integer-valued) **submodular set function.**

Hence, the vector connectivity problem is a special case of the **Minimum Submodular Cover** problem:

Input: A finite set V and an integer-valued nondecreasing submodular set function g on subsets of V (given by an oracle).

Task: Find a smallest set $X \subseteq V$ such that $g(X) = g(V)$.

Greedy Strategy

start with $S = \emptyset$

if S is not a vector connectivity set, keep on adding to S a vertex $v \in V \setminus S$ maximizing $f(S \cup \{v\}) - f(S)$

By a result of [Wolsey, 1982] on minimum submodular cover, the greedy strategy approximates OPT by a factor of at most

$$H_{\max f(\{y\})} \leq H_{n+\Delta(G)} \leq \log n + 2\,.$$

$H_k = \sum_{i=1}^{k} \frac{1}{i}$

Approximating vector connectivity

It can be shown that f is a (non-decreasing, integer-valued) **submodular set function.**

Submodularity is a discrete analog of concavity:

$$X \subseteq Y \quad \Rightarrow \quad f(X \cup \{v\}) - f(X) \geq f(Y \cup \{v\}) - f(Y)\,.$$

We have two proofs for submodularity:

1. Derive a monotonicity property of disjoint path systems using a classical result on **stable matchings** [Gale, Shapley, 1962].
2. Observe that f is closely related to the *rank function* of a certain *gammoid*.

A little detour to matroids

Gammoids are special matroids.

Recall:

- A **matroid** is a hypergraph $(V, \mathcal{F})$ such that $\mathcal{F}$ is nonempty and closed under taking subsets, and its elements, called the **independent sets**,

  satisfy the following

- **exchange property**:

  for every two independent sets A and B such that $|A| < |B|$, there exists an element of B whose addition to A results in a larger independent set.

A little detour to matroids

Some basic facts about matroids:

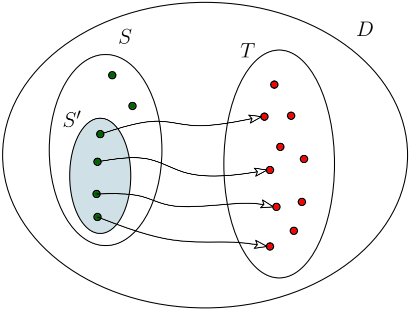
- They can be defined in several equivalent ways.
- For every matrix (over any field), the collection of linearly independent sets of columns of the matrix forms a matroid.
- Matroids are the most general combinatorial structures for which the greedy method always finds a cheapest **basis** (maximal independent set).
- Given a matroid $M = (V, \mathcal{F})$, the **rank function** of M is the function $r_M : \mathcal{P}(V) \to \mathbb{Z}_+$, defined by:

$$r_M(X) = \text{maximum size of an independent set contained in } X\,.$$

The rank function of every matroid is submodular.

Approximating vector connectivity

A **gammoid** is a hypergraph $\Gamma = (U, \mathcal{F})$ derived from a triple (D, S, T) where $D = (V, A)$ is a digraph and $S, T \subseteq V$ such that $U = S$ and a subset $S' \subseteq S$ forms a hyperedge if and only if D contains $|S'|$ disjoint directed paths connecting S' to T.



Theorem (Perfect 1968, Pym 1969)

Every gammoid is a matroid.

Approximating vector connectivity

$$f(X) = \sum_{v\in V} f_v(X)\,, \text{ for all } X \subseteq V, \text{ and}$$

$$f_v(X) = \begin{cases} \min\{\sigma(v,X), r(v)\} & \text{if } v \notin X; \\ r(v) & \text{if } v \in X. \end{cases}$$

Approximating vector connectivity

$$f(X) = \sum_{v\in V} f_v(X)\,,\ \text{for all } X \subseteq V, \text{ and}$$

$$f_v(X) = \begin{cases} \min\{\sigma(v,X), r(v)\} & \text{if } v \notin X; \\ r(v) & \text{if } v \in X. \end{cases}$$

$$\begin{aligned} \sigma(v,X) &= \text{maximum number of disjoint } v\text{-}X \text{ paths} \\ &= \text{maximum number of disjoint directed paths} \\ &\ \text{from } N(v) \text{ to } X \text{ in the digraph } D \text{ representing } G \\ &= \text{rank of } X \text{ in the gammoid derived from} \\ &\ (D, V\setminus\{v\}, N(v)) \end{aligned}$$

$\Rightarrow \sigma(v,X)$ is submodular

$\Rightarrow f$ is submodular

HARDNESS RESULTS

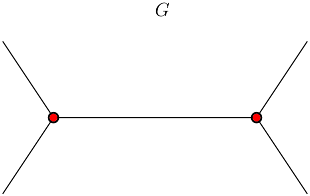
Inapproximability of vector connectivity

Theorem

Vector connectivity is APX*-hard.*

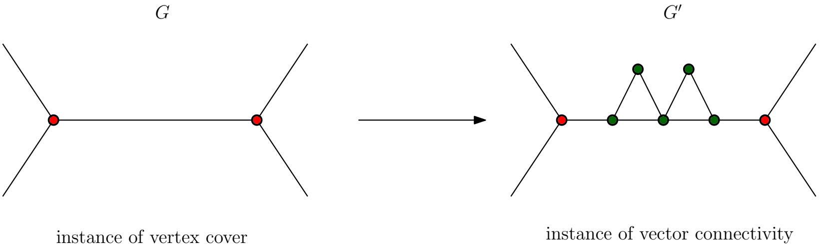
- There exists a constant $\epsilon > 0$ such that vector connectivity cannot be approximated in polynomial time within a factor of $1 + \epsilon$ unless $\mathrm{P} = \mathrm{NP}$.
- Reduction is from vertex cover in cubic graphs.

Inapproximability of vector connectivity



instance of vertex cover

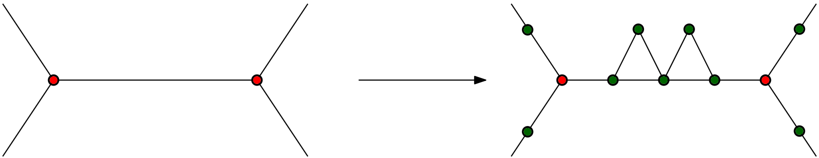
Inapproximability of vector connectivity



Inapproximability of vector connectivity

G

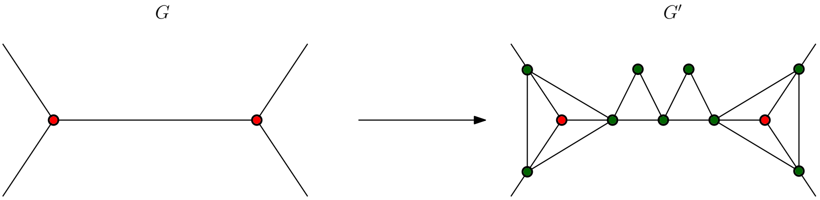
G'



instance of vertex cover

instance of vector connectivity

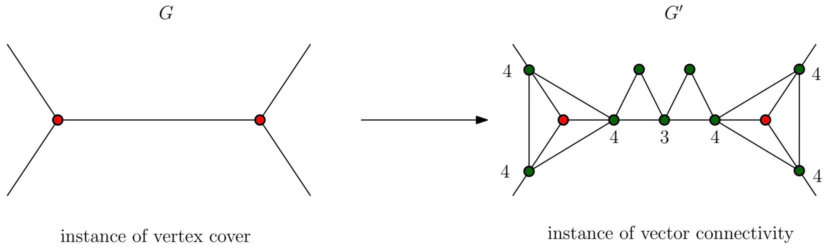
Inapproximability of vector connectivity



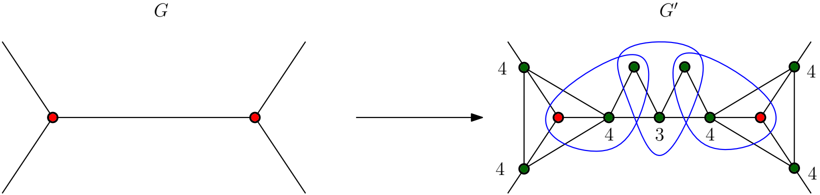
instance of vertex cover

instance of vector connectivity

Inapproximability of vector connectivity

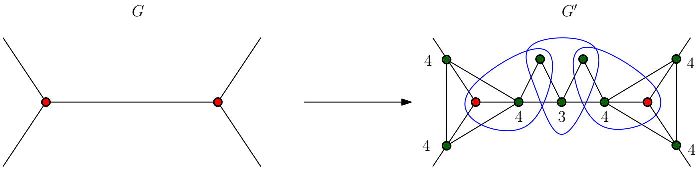


Inapproximability of vector connectivity



instance of vertex cover

instance of vector connectivity



instance of vertex cover   instance of vector connectivity

Idea behind the reduction:

- for each edge of G, every vector connectivity set for (G', r) must contain at least one vertex from each of the circled sets
- the three sets can be hit with two vertices, but not with just one
- no matter whether one or two of the red vertices are used, one additional green vertex must be used

  (and if no red vertex is used, two green vertices must be used)
- the best we can do is to find a vertex cover for G and then add one green vertex for each edge of G.

Inapproximability of vector connectivity

Similar reductions show NP-hardness of vector connectivity in:

- planar line graphs of maximum degree 5,
- planar bipartite graphs of maximum degree 5.

Corollary:

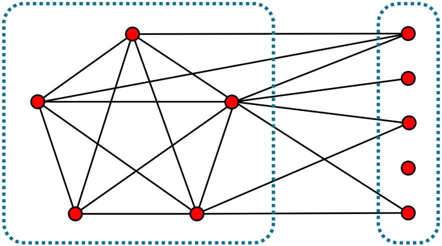
- Vector connectivity is NP-hard for perfect graphs.

Note that vertex cover is polynomially solvable for line graphs and for perfect graphs.

POLYNOMIAL SPECIAL CASES

Split graphs

A graph is split if there exists a partition of its vertex set into a clique and an independent set.



Source: `http://en.wikipedia.org/wiki/Split_graph`

Split graphs

Theorem

Vector connectivity can be solved in polynomial time in **split graphs**.

Recall that **domination** (and hence **vector domination**) is hard to approximate to within a factor of $(1-\epsilon)\log n$, for every constant $\epsilon > 0$, even within the class of split graphs [Chlebík–Chlebíkova, 2008].

Split graphs

Theorem

Vector connectivity can be solved in polynomial time in **split graphs**.

The following very simple **greedy algorithm** is optimal:

- sort the vertices of G by their r-values in non-increasing order
- greedily pick vertices from the start of the sorted list to be in S until we have a vector connectivity set

Caveat: we assume that $r(v) \leq d(v)$ for all $v \in V$

Split graphs

Observation

Let G be a split graph, and let $S \subseteq V$ such that

$$\min_{u \in S} r(u) \geq \max_{v \in V \setminus S} r(v)\,.$$

Then, every $v \in V \setminus S$ has at least $\min\{r(v), |S|\}$ disjoint paths to S.

Cographs

A **cograph** is a P_4-free graph.

Theorem

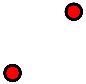
Vector connectivity can be solved in polynomial time in **cographs**.

The algorithm: dynamic programming based on the fact that a cograph on at least two vertices is either disconnected or its complement is disconnected

Example:

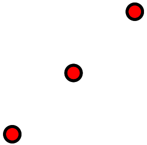
Cographs

Example:



Cographs

Example:

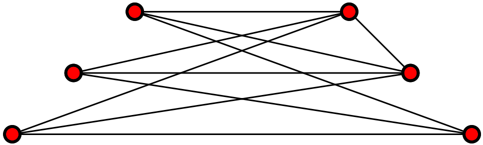


Cographs

Example:

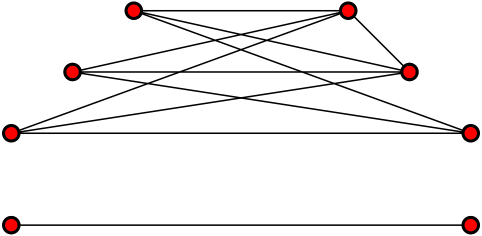
Cographs

Example:



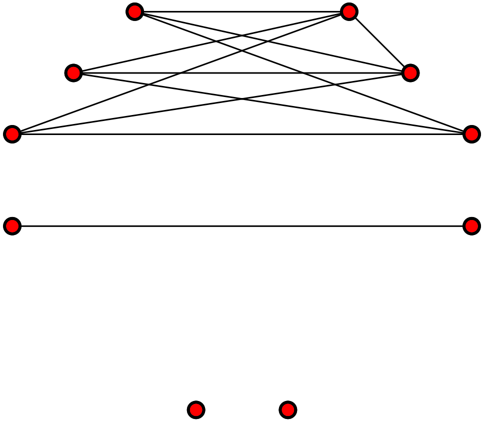
Cographs

Example:



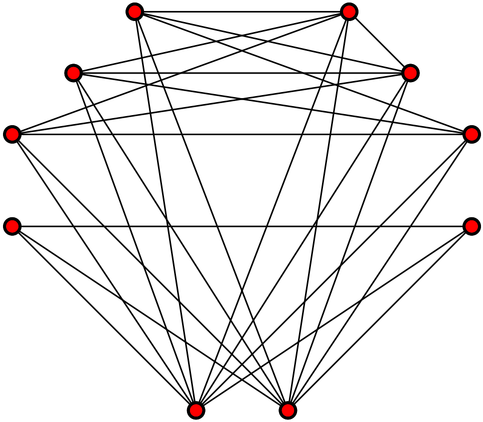
Cographs

Example:



Cographs

Example:



Cographs

Theorem

Vector connectivity can be solved in polynomial time in **cographs**.

- A recursive decomposition of a cograph using disjoint union and join operations is represented by a **cotree**, which can be computed in linear time [Corneil-Perl-Stewart 1985]
- we need to solve a more general problem, in which, given an integer ℓ,
  all vertices v with $r(v) \geq \ell$ must be included in the vector connectivity set

Trees

Theorem

Vector connectivity can be solved in polynomial time **in trees***.*

Idea of the algorithm:
construct a vector connectivity set S for T of minimum size, starting from the leaves of T and traversing the tree bottom up, processing a vertex only after all its children have been processed

Trees

T_v: subtree of T rooted at v

We recursively compute for each vertex v the values of:

- $n(v)$: whether or not there exists a vertex in T_v that "needs" an additional path to a vertex outside of T_v,
- $b(v)$: whether or not there is a vertex of S in the subtree T_v "below" v,
- $c(v)$: the number of children w of v with $b(w) = 1$.

The current node v is added to S only if this is necessary in order to maintain the feasibility in the subtree.

- This is easy to determine using $n(v)$, $b(v)$, $c(v)$.

Trees

The algorithm maintains the following invariants:

Feasibility:

(i) If $n(v) = 0$, then $S \cap V(T_v)$ is a vector connectivity set for T_v.

(ii) If $n(v) = 1$, then $(S \cap V(T_v)) \cup \{p(v)\}$ is a vector connectivity set for the subtree of T induced by $V(T_v) \cup \{p(v)\}$;

Optimality:

(iii) There is no vector connectivity set S' for T such that $|S' \cap V(T_v)| < |S \cap V(T_v)|$.

Block graphs

A **block graph** is a graph every block of which is complete.

Theorem

Vector connectivity can be solved in polynomial time in **block graphs**.

- This generalizes the result for trees.
- The result is obtained by reducing the vector connectivity problem from arbitrary graphs to biconnected graphs (connected graphs without cut vertices),
  and solving the problem on complete graphs.

Reduction to biconnected graphs

The following generalization of the problem is needed for the reduction:

Free-set vector connectivity:

Input: A graph $G = (V, E)$, a function $r : V \to \mathbb{Z}_+$, and a set $F \subseteq V$.

Task: Find a minimum set S such that for every vertex $v \in V \setminus S$ is connected via $r(v)$ disjoint paths to $S \cup F$.
(For $v \in F \setminus S$, one of the paths can be trivial.)

Reduction to biconnected graphs

Theorem

Suppose that free-set vector connectivity can be solved in polynomial time on graphs from a class $\mathcal{G}$.

Then, the problem can be solved in polynomial time on graphs every block of which is in $\mathcal{G}$.

The result for block graphs follows by taking $\mathcal{G}$ to be the set of complete graphs.

CONCLUSION

Conclusion

We have see several aspects of vector connectivity:

1. A characterization of vector connectivity sets
2. A $(\log n + 2)$-approximation algorithm
3. An APX-hardness result
   - NP-hardness for bipartite graphs
   - NP-hardness for line graphs
4. Exact polynomial algorithms for trees, cographs, split graphs, and block graphs
5. A reduction to biconnected graphs

Conclusion

Some open questions:

1. What is the exact (in)approximability status of vector connectivity?
   In particular, is it constant-factor approximable?

2. On what other graph classes is the problem polynomial?
   In particular, is it polynomial for:
   - chordal graphs?
   - P_k-free graphs (for every fixed k)?
   - graphs of bounded clique-width?

3. Could IP formulations lead to new tractable cases?

The end

Thank you for your attention!