Found Graph Data and Planted Vertex Covers

Austin R. Benson and Jon Kleinberg

arb@cs.cornell.edu, kleinber@cs.cornell.edu

Code & data ⟶ https://github.com/arbenson/FGDnPVC

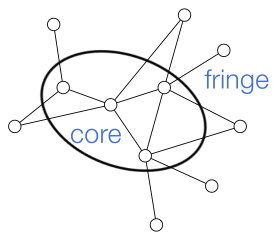
Cornell University

Partially measured graphs & core-fringe structure.

We often measure graph data by recording interactions involving a *core* set of nodes:

- Email of company employees
- Phone calls of all customers of a service provider
- Friendships of a set of users

We end up with a dataset that includes the core along with a potentially much larger set of *fringe* nodes.

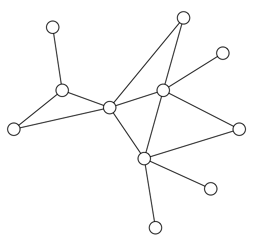


Core nodes (set C) are a vertex cover of the graph.

Planted vertex covers (unknown cores).

Sometimes we *find graph data* without core-fringe labels.

- Intelligence and counter-surveillance: find email graph from compromised accounts but don't know who is compromised
- Data provenance: metadata is lost over time



Main question.
Can we recover the core without labels?
Or, can we recover a planted vertex cover in found graph data?

Why? We need the labels for better network analysis (e.g., to to deal with compromised accounts in counter-surveillance).

In general, the answer is "No" but…

Theorem. If C is a minimal planted vertex cover with $|C| \le k$, then we can find a set D of size $O(k^2)$ that must contain C. Specifically, D is the *union of all minimal covers* of size $\le k$.

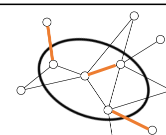
Computation exponential in k. Need something more practical.

Maximal matchings & partial recovery.

Algorithm 1: Greedy maximal matching.

Input: Graph $G = (V, E)$
Output: Vertex cover M with $|M| \le 2k^*$
$M \leftarrow \emptyset$
for $e = (u, v) \in E$ **do**
  if $u, v \notin M$ **then** $M \leftarrow M \cup \{u, v\}$

A simple building block.
Greedy maximal matching finds 2-approximation to minimum vertex cover.



Theorem. If C is any planted vertex cover with $|C| \le ck^*$, then *any* greedy matching M contains at least $|C| / (2c)$ nodes in C.

The union of minimal vertex covers (UMVC) algorithm for planted vertex cover recovery.

1. Run greedy maximal matching several times with random initializations.
2. Prune maximal matchings to get minimal covers.
3. Take union of these minimal vertex covers as guess at the planted vertex cover.

```julia
using SparseArrays, Random
function UMVC(A::SparseMatrixCSC{Int64,Int64},
              ncovers::Int64=300)
  edges = filter(e->e[1] < e[2],
                 collect(zip(findnz(A)[1:2]...)))
  umvc = zeros(Int64,size(A,1))
  for _ in 1:ncovers
    # Run 2-approximation with random edge ordering
    vc = zeros(Int64, size(A,1))
    for (i, j) in shuffle(edges)
      if vc[[i,j]] == [0,0]; vc[[i,j]] .= 1; end
    end
    # Reduce to a minimal cover
    while true
      vc_size = sum(vc)
      for c in shuffle(findall(vc .== 1))
        nbrs = findnz(A[:,c])[1]
        if sum(vc[nbrs]) == length(nbrs); vc[c] = 0; end
      end
      if sum(vc) == vc_size; break; end
    end
    umvc[findall(vc .== 1)] .= 1
  end
  degs = vec(sum(A, dims=1)) # node degrees
  return sortperm(collect(zip(umvc, degs)), rev=true)
end
```

Entire Julia implementation: bit.ly/UMVC-julia

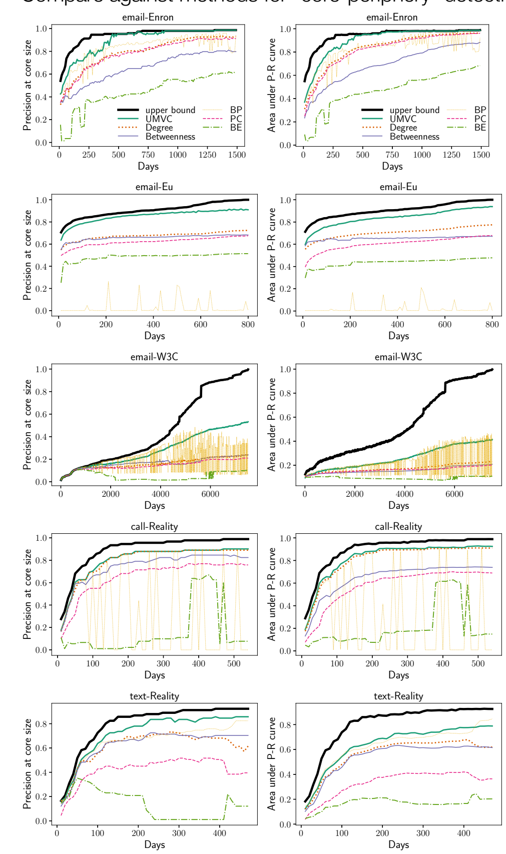
Other fun theory.

U = union of *all* minimal vertex covers of size $\le k = |C|$.

1. High-degree nodes are in U.
   If the degree of node v is at least k, then v is in U.
2. Nodes connected to the fringe are in U.
   If node v connects to a fringe node w, then v is in U.
3. Nodes connected to deeply embedded core nodes are in U.
   If all of v's neighbors are in the core and (u, v) is an edge, then u is in U.
4. Stochastic block modeling makes U small.
   SBM block probabilities p (core-core), q (core-fringe), 0 (fringe-fringe) for p, q constants. Suppose ck core nodes for constant c. Then w.h.p. as function of k, $|U| \le O(k \log k)$.
5. Unions can't grow too fast.
   If M, N come from greedy max. matching, then $|M \cap N| \ge ¼\ |M|$.

Empirical evaluation.

- Evaluate recovery on datasets containing known measured core C in email and telecommunications data.
- Edges all have timestamps (measure recovery over time).
- Compare against methods for "core-periphery" detection.



The algorithm is also pretty fast.

Dataset	UMVC	degree	betweenness	PC scores	BE scores	Belief Prop.
email-W3C	6.5 secs	< 0.01 secs	2.8 mins	1.1 hours	0.1 secs	1.0 mins
email-Enron	8.4 secs	< 0.01 secs	2.5 mins	1.8 hours	0.1 secs	20.2 mins
email-Eu	1.2 mins	< 0.01 secs	11.8 hours	> 3 days	0.9 secs	15.0 mins
call-Reality	2.2 secs	< 0.01 secs	27.9 secs	6.1 mins	1.8 secs	4.3 secs
text-Reality	0.5 secs	< 0.01 secs	0.8 secs	11.4 secs	< 0.1 secs	6.8 secs

Acknowledgements. This research was supported by a Simons Investigator Award, NSF TRIPODS Award 1740822, and NSF Award DMS-1830274.