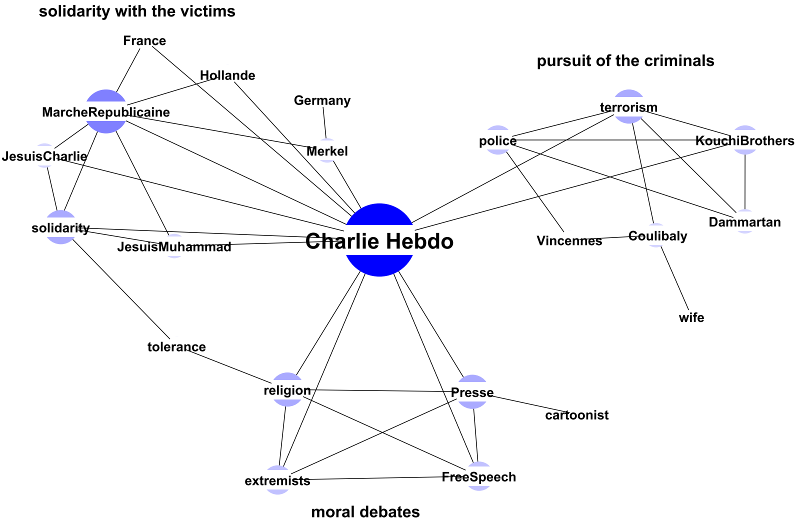
Finding Subgraphs with Maximum Total Density and Limited Overlap

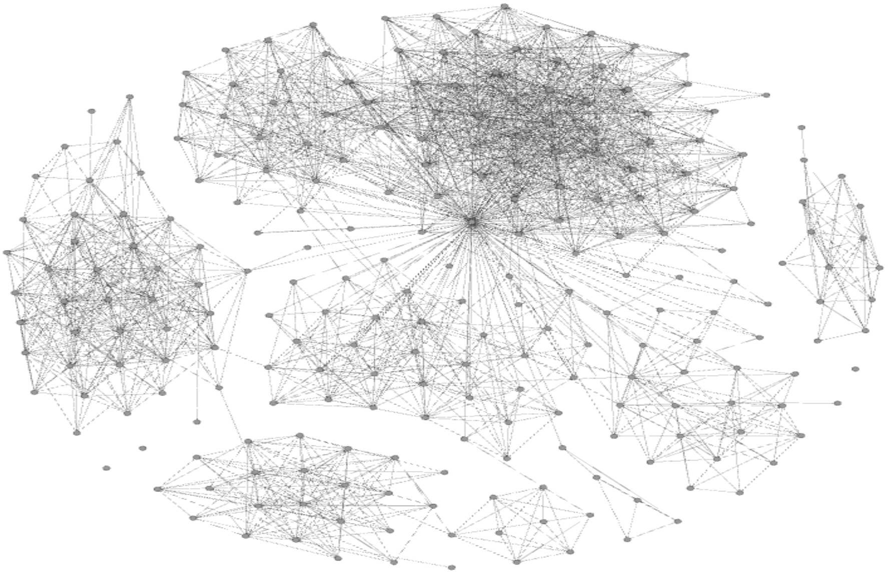
Oana Balalau[1], Francesco Bonchi[2], T-H . Hubert Chan[3], Francesco Gullo[2] and Mauro Sozio[1]

[1]Telecom ParisTech University [2]Yahoo Labs [3]The University of Hong Kong

Motivation


solidarity with the victims
France
Hollande
Germany
MarcheRepublicaine
Merkel
JesuisCharlie
solidarity
JesuisMuhammad
Charlie Hebdo
pursuit of the criminals
terrorism
police
KouchiBrothers
Dammartan
Vincennes
Coulibaly
wife
tolerance
religion
Presse
cartoonist
extremists
FreeSpeech
moral debates

Motivation



Related work

Finding multiple dense subgraphs
Find one densest subgraph in the current graph, remove all its vertices and edges, and iterate at most k times.

Drawbacks:

- it is costly to compute a densest subgraph
- the subgraphs found are disjoint
- no formal definition for the problem
- we can compute a "bad" solution

Related work

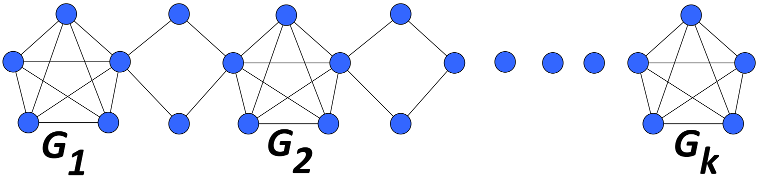


Figure: Each clique has density 2 as well as the entire graph.

Densest subgraph definition

Given an undirected graph G, its **density** is defined as the number of edges divided by the number of nodes.

Densest subgraph problem: finding a subgraph with maximum density. Solutions in polynomial time:

- max-flow algorithm (Goldberg)
- **linear-programming formulation (Charikar)**.

Heuristic : 1/2 approximation algorithm (linear in the size of the input).

Problem definition
Multiple dense subgraphs with limited overlap

Given

- an undirected graph $G = (V, E)$
- an integer $k > 0$
- a rational number $\alpha \in [0, 1]$

we want to find at most k subgraphs of G such that their total density is maximum and the pairwise Jaccard coefficient on the sets of nodes $\leq \alpha$.

Problem definition
Multiple dense subgraphs with limited overlap

Given

- an undirected graph $G = (V, E)$
- an integer $k > 0$
- a rational number $\alpha \in [0, 1]$

we want to find at most k subgraphs of G such that their total density is maximum and the pairwise Jaccard coefficient on the sets of nodes $\leq \alpha$.

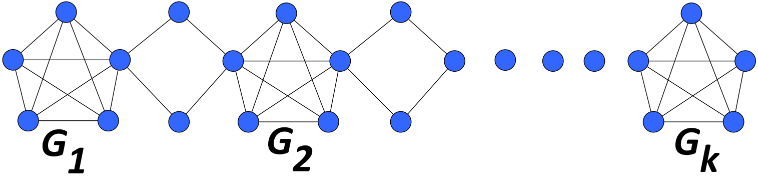
Theorem

The problem is NP-hard.

Proof.

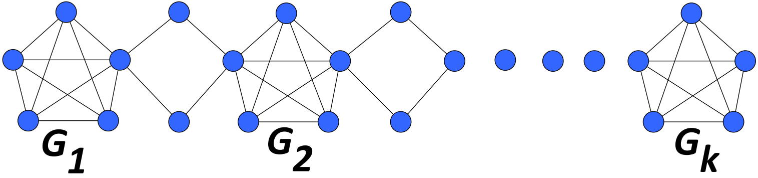
Reduction from the maximum independent set problem. □

Minimal densest subgraphs



An undirected graph G is a **minimal densest graph** if its density is maximum and it doesn't contain a proper subgraph with the same density.

Minimal densest subgraphs



An undirected graph G is a **minimal densest graph** if its density is maximum and it doesn't contain a proper subgraph with the same density.

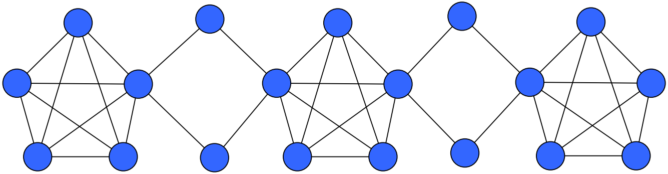
Can we compute minimality efficiently? Yes.

Computing minimal densest subgraphs

- faster algorithm for the densest subgraph (via pruning the search space)
- faster rounding scheme for the rounding of the fractional linear programming solution (order of n versus order of $nlog(n) + m$)
- minimality by solving at most $4log_{4/3}(n)$ number of linear programs

MinAndRemove

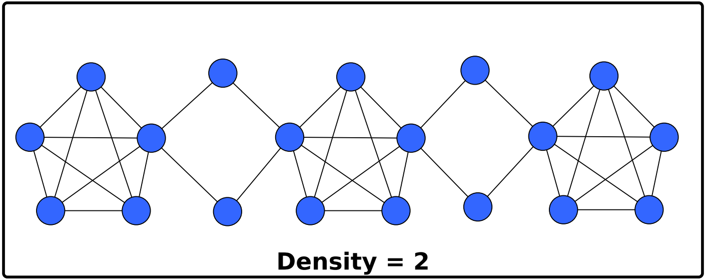
Find $k = 3$ subgraphs that have an overlap of at most $\alpha = 0.25$.



MinAndRemove

Find $k = 3$ subgraphs that have an overlap of at most $\alpha = 0.25$.

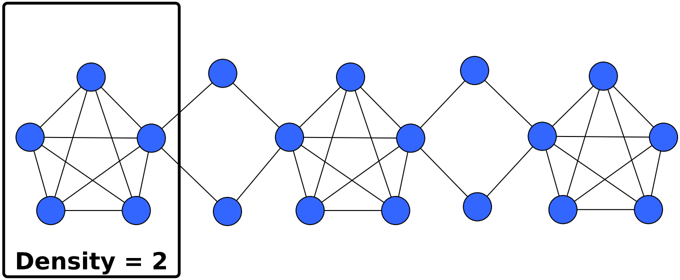
- Find a densest subgraph



MinAndRemove

Find $k = 3$ subgraphs that have an overlap of at most $\alpha = 0.25$.

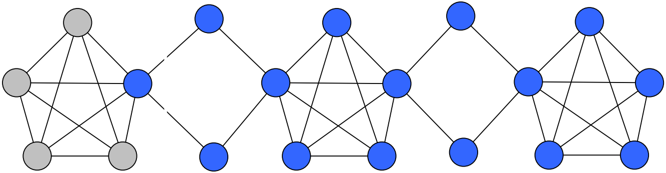
- Find a densest subgraph
- Make it minimal



MinAndRemove

Find $k = 3$ subgraphs that have an overlap of at most $\alpha = 0.25$.

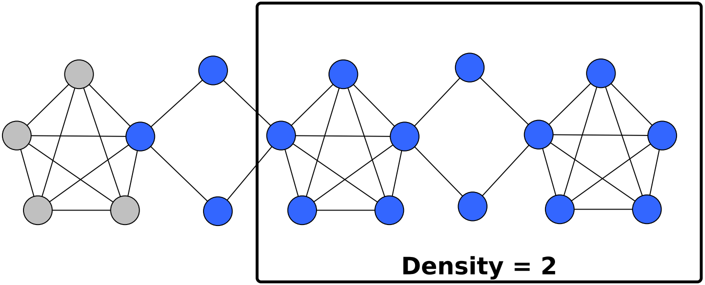
- Find a densest subgraph
- Make it minimal
- Remove 75% of the subgraph's nodes



MinAndRemove

Find $k = 3$ subgraphs that have an overlap of at most $\alpha = 0.25$.

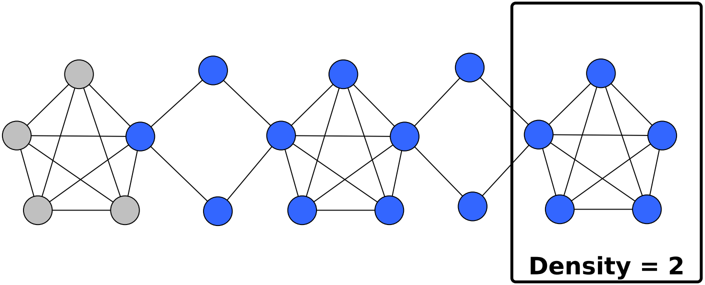
- Find a densest subgraph
- Make it minimal
- Remove 75% of the subgraph's nodes
- Iterate



MinAndRemove

Find $k = 3$ subgraphs that have an overlap of at most $\alpha = 0.25$.

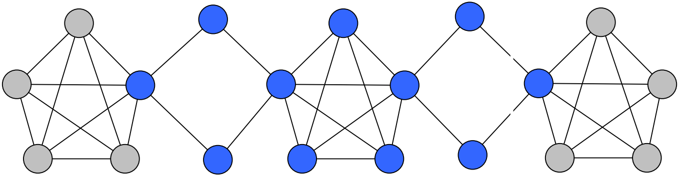
- Find a densest subgraph
- Make it minimal
- Remove 75% of the subgraph's nodes
- Iterate



MinAndRemove

Find $k = 3$ subgraphs that have an overlap of at most $\alpha = 0.25$.

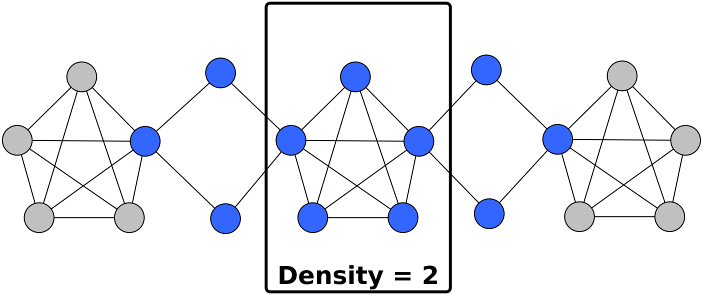
- Find a densest subgraph
- Make it minimal
- Remove 75% of the subgraph's nodes
- Iterate



MinAndRemove

Find $k = 3$ subgraphs that have an overlap of at most $\alpha = 0.25$.

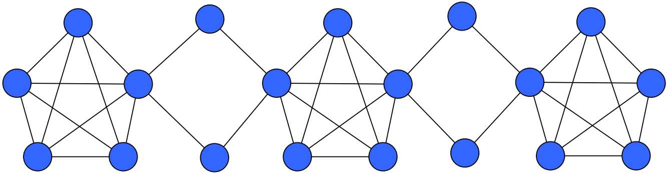
- Find a densest subgraph
- Make it minimal
- Remove 75% of the subgraph's nodes
- Iterate



MinAndRemove

Find $k = 3$ subgraphs that have an overlap of at most $\alpha = 0.25$.

- Find a densest subgraph
- Make it minimal
- Remove 75% of the subgraph's nodes
- Iterate
- Solution $= \{C_1, C_2, C_3\}$



Guarantees

Theorem

The algorithm MinAndRemove *will find the optimum when the input graph contains k disjoint densest subgraphs.*

In the general case, no guarantees.

Experiments

We considered 8 datasets, 2 groups according to size:

- 5 datasets with the number of edges between 2M and 11M
- 3 datasets with the number of edges between 43M and 117M

For solving linear programs we used the Gurobi Optimizer.

Evaluation and upper bound

Let ρ_{max} be the density of the densest subgraph.

$k \cdot \rho_{max}$ gives an upper bound on the optimum solution.

MinAndRemove

The density found by the algorithm as a percentage of the upper bound.

$k = 10$	$\alpha = 0.1$	$\alpha = 0.2$	$\alpha = 0.3$	$\alpha = 0.4$	$\alpha = 0.5$
web-Stanford	71%	73%	76%	79%	81%
com-Youtube	48%	52%	51%	61%	62%
web-Google	80%	80%	80%	80%	80%
Youtube-growth	44%	46%	53%	59%	57%
As-Skitter	58%	59%	59%	62%	64%

FastDSLO

The density found by the algorithm as a percentage of the upper bound.

$k = 10$	$\alpha = 0.1$	$\alpha = 0.2$	$\alpha = 0.3$	$\alpha = 0.4$	$\alpha = 0.5$
LiveJournal	24%	24%	25%	28%	27%
Hollywood-2009	18%	19%	19%	21%	23%
Orkut	18%	20%	21%	25%	27%

Running time

Minimal densest subgraph routine: 15' (the smallest dataset) to 3h (the biggest dataset, 11M edges) to find 10 subgraphs.

Approximation subgraph routine: from 30' to at most 2h20' (117M edges) to find 10 subgraphs.

Conclusions

Contributions

- formulation and analysis of the problem of finding **multiple dense subgraphs with limited overlap**
- fastest algorithm for the minimal densest subgraph (improvement of the LP-based approach of Charikar)
- heuristics for the problem

Future work

- more scalable algorithms
- adapting in a dynamic environment
- finding patterns in real-world graphs