Finding Dense Subgraphs

Moses Charikar

Center for Computational Intractability

Dept of Computer Science
Princeton University
The Dense Subgraph Problem

Given $G$, find dense subgraph $S$
Dense subgraphs are everywhere!

- A useful subroutine for many applications.
Social Networks

• Trawling the Web for emerging cyber-communities [KRRT ‘99]
  – Web communities are characterized by dense bipartite subgraphs
Communities on gitweb
• Mining coherent dense subgraphs across massive biological networks for functional discovery [HYHHHZ '05]

  – dense protein interaction subgraph corresponds to a protein complex [BD’03] [SM’03]

  – dense co-expression subgraph represent tight co-expression cluster [SS ‘05]
Dense subgraphs are everywhere!

• A useful subroutine for many applications.

• A useful candidate hard problem with many consequences
Public Key Cryptography [ABW ‘10]

- Hardness assumption
Complexity of Financial Derivatives

• Computational Complexity and Information Asymmetry in Financial Products [ABBG ’10]
  – Evaluating the fair value of a derivative is a hard problem
  – Tampered derivatives (CDOs) can be hard to detect.
  – Derivative designer can gain a lot from small asymmetry in information (lemon cost).
Simplest Model

6σ lemons, default w.p. ½

M CDOs

D assets per CDO

N Asset classes

I can cluster lemons to create tampered CDOs.

I hope lemons are spread evenly over CDOs.

Dense Subgraph

L Lemons
Summary so far

- Finding dense subgraphs is useful, both as a subroutine as well as a candidate hard problem

- So, what do we know about the problem?
  - Formal definition
  - New results
  - New results on related problems
**Densest \( k \)-subgraph**

**Problem.** Given \( G \), find a subgraph of size \( k \) with the maximum number of edges (think of \( k \) as \( n^{1/2} \))

---

**Problems of similar flavor**

- Max clique
- Max density subgraph – find \( H \) to maximize the ratio:
  \[
  \frac{\# \text{edges}(H)}{|H|}
  \]
Approximation Algorithm

• Exact problem is hard, prove that efficient heuristic finds good solution.

• Approximation ratio = \frac{Value of heuristic solution}{Value of optimal solution}

• Solution value = number of edges in subgraph
Densest $k$-subgraph

**Problem.** Given $G$, find a subgraph of size $k$ with the maximum number of edges (think of $k$ as $n^{1/2}$)

[Feige, Kortsarz, Peleg 93] $O(n^{1/3 - 1/90})$ approximation

[Feige, Schechtman 97] $\Omega(n^{1/3})$ integrality gap for natural SDP

[Feige 03] Constant hardness under the Random 3-SAT assumption

[Khot 05] There is no PTAS unless $\text{NP} \subseteq \text{BPTIME}(\text{sub-exp})$
Main Result


Theorem. $O(n^{1/4 + \varepsilon})$ approximation for DkS in time $O(n^{1/\varepsilon})$

[Bhaskara, C, Chlamtac, Feige, Vijayaraghavan ‘10]
Outline

• Introduce two average case problems
• ‘Local counting’ based algorithms for these
• Notion of log-density
• Techniques lead to algorithms for the DkS problem
Planted problems related to DkS

Yes

G, n

H, k

- Assume G does not have dense subgraphs
- Good algorithm for DkS ⇒ we can distinguish

No

G, n

Two natural questions:
1. Random in Random: G(k,q) planted in G(n,p)
2. Arbitrary in Random: Some dense subgraph planted in G(n,p)
**Question.** How large should $q$ be so as to distinguish between

- **YES:** $G(n,p)$ with $G(k,q)$ planted in it
- **NO:** $G(n,p)$

When would looking for the presence of a subgraph help distinguish?

Eg. $K_{2,3}$
**Question.** How large should $q$ be so as to distinguish between

**YES:** $G(n,p)$ with $G(k,q)$ planted in it

**NO:** $G(n,p)$

[**Erdos-Renyi**]:

- Appears w.h.p. in $G(n,p)$ if $n^5 p^6 \gg 1$, i.e., degree $\gg n^{1/6}$
- Does *not* appear w.h.p. in $G(n,p)$ if $n^5 p^6 \ll 1$, i.e., degree $\ll n^{1/6}$

Valid distinguishing algorithm if: $k^5 q^6 \gg 1$, and $n^5 p^6 \ll 1$

I.e., degree $\ll n^{1/6}$, and planted-degree $\gg k^{1/6}$

Center for Computational Intractability, Princeton University
**Question.** How large should $q$ be so as to distinguish between

**YES:** $G(n,p)$ with $G(k,q)$ planted in it

**NO:** $G(n,p)$

In general, suppose degree $< n^{\delta}$, and planted-degree $> k^{\delta+\epsilon}$

Find a rational number $1 - r/s$ between $\delta$ and $\delta+\epsilon$, and use a graph with $r$ vertices and $s$ edges to distinguish.
Log density

A graph on $n$ vertices has **log-density** $\delta$ if the average degree is $n^\delta$

$$\delta = \frac{\log d_{\text{avg}}}{\log |V|}$$

**Question.** Given $G$, can we detect the presence of a subgraph on $k$ vertices, with higher log-density?
Dense vs. Random

**Problem.** Distinguish $G \sim G(n,p)$, log-density $\delta$ from a graph which has a $k$-subgraph of log-density $\delta + \epsilon$

( Note. $kp = k(n^\delta/n) = k^\delta(k/n)^{1-\delta} < k^\delta$ )

More difficult than the planted model earlier (graph inside is no longer random)

Eg. $k$-subgraph could have log-density=1 and not have triangles
Main idea

**Example.** Say $\delta = 2/3$, i.e., degree $= n^{2/3}$

random graph $G(n, n^{-1/3})$:

any three vertices have $O(\log n)$ common neighbors w.h.p.

planted graph: size $k$, log-density $2/3 + \varepsilon$:

triple with $k^{3\varepsilon}$ common neighbors
Main idea (contd.)

Example 2. $\delta = 1/3$, i.e., degree $= n^{1/3}$

random graph $G(n, n^{-1/3})$:
any pair of vertices have $O(\log^2 n)$ paths of length 3, w.h.p.

planted graph: size $k$, log-density $1/3 + \varepsilon$:
exists a pair of vertices with $k^\varepsilon$ paths
Main idea (contd.)

**General strategy:** For each rational $\delta$, consider appropriate `caterpillar’ structures, count how many `supported’ on fixed set of leaves

- Random graph $G(n,p)$, log-density $\delta$: 
  **every** leaf tuple supports polylog($n$) caterpillars
- Planted graph, size $k$, log-density $\delta+\epsilon$ : 
  **some** leaf tuple supports at least $k^\epsilon$ caterpillars

Center for Computational Intractability, Princeton University
Theorem. For every $\varepsilon > 0$, and $0 < \delta < 1$, we can distinguish between $G(n,p)$ of log-density $\delta$, and an arbitrary graph with a $k$-subgraph of log-density $\delta + \varepsilon$, in time $n^{O(1/\varepsilon)}$.

(Pick a rational number between $\delta$ and $\delta + \varepsilon$, and use the caterpillar corresponding to it)
DkS in general graphs
Preliminaries

**Aim.** Obtain a $k$-subgraph of avg degree $\rho$

**Observation 1.** It suffices to return a $\rho$-dense subgraph with $\leq k$ vertices

(removed and repeat)
**Observation 2.** It suffices to return a bipartite subgraph with density \( \rho \), and \( \leq k \) vertices on one side

- Pick the \( |V| \) vertices in \( U \) of largest degree
- Density of the resulting subgraph is

\[
\frac{1}{2 |V|} \cdot \frac{|V|}{|U|} \cdot \rho(|V| + |U|) \geq \frac{\rho}{2}
\]

Density is \( \rho \), so

\[
E(U,V) = \rho(|V| + |U|)
\]
Algorithm using Cat_δ

**Idea.** Look at the ‘set of candidates’ for a non-leaf after fixing a prefix of the leaves

Eg., define $S_{abc}(v) =$ set of ‘candidates’ in $G$ for internal vertex $v$ after fixing $a, b, c$

(for instance, $S_{ab}(u) =$ the set of common nbrs of $a, b$)

Denote $T_{abc}(v) = S_{abc}(v) \cap H$

Given $a, b, ..$ and the structure, we can compute the $S$’s
Algorithm using $\text{Cat}_\delta$ (plot outline)

- For every $a \in V$, perform $\text{LocalSearch}(S_a(u))$
- If it always fails, then $\exists a, b$, s.t. $|S_{ab}(u)| \leq U_1$ and $|T_{ab}(u)| \geq L_1$
- For every $a,b$, perform $\text{LocalSearch}(S_{ab}(u))$
- If it fails each time, then $\exists a, b$, s.t. $|S_{ab}(v)| \leq U_2$ and $|T_{ab}(v)| \geq L_2$
- Keep doing this … At the last step, the parameters give a contradiction!
Main Component – LocalSearch(S)

For each \( i = 1 \ldots k \), do:

- Pick the \( i \) vertices on the right with the most edges to \( S \) (call this \( S_r \)). If \( S \cup S_r \) has density \( \geq \rho \), return it.

If no dense subgraph is found, return Fail.
Linear Programming view

• Can bound the quality of the solution w.r.t value of a Lift-and-project style LP relaxation.

• Algorithm can be viewed as rounding procedure for relaxation via successive conditioning
Subexponential algorithm

- $n^{(1-\varepsilon)/4}$ approximation in time $2^{n^{6\varepsilon}}$

- Guess subsets of size $n^{\varepsilon}$ for every leaf in caterpillar structure.
New developments

• Hardness based on non-standard assumptions
• Integrality gaps for lift-and-project relaxations
Hardness

• [AAMMW ’11]

• No constant factor possible if random k-AND hard to refute.

• No constant possible if planted cliques cannot be found in polynomial time.

• Super constant hardness based on stronger assumption.
Stronger relaxations

Lasserre

Sherali-Adams

Lovasz-Schrijver
Gaps for lift-and-project

- [BCCFV ’10]
  $t$ rounds of Lovasz-Schrijver: gap $n^{\frac{1}{4}+O(1/t)}$

- [BCV ‘11]
  $\Omega\left(\frac{\log n}{\log \log n}\right)$ rounds of Sherali-Adams:
  gap $\tilde{\Omega}(n^{\frac{1}{4}})$

- [GZ ‘11]
  $n^{\Omega(1)}$ rounds of Lasserre: gap $n^{\Omega(1)}$
Open Problem

• Given random graph: $n$ vertices, degree $n^{1/2}$
• Planted subgraph: $n^{1/2}$ vertices, degree $n^{1/4 - \varepsilon}$

• Detect in polynomial time?
Open Problem

Given $G$, find dense subgraph $S$ with size $\sqrt{n}$ and degree $n^{1/4}$. 

Graph $G$ with degree $\sqrt{n}$.