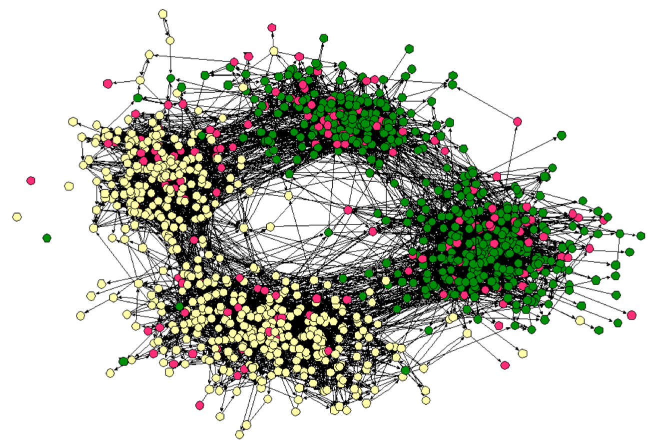
Dense Subgraph Discovery in Large Graphs

Charalampos (Babis) E. Tsourakakis

babis@seas.harvard.edu

Motivation



- Anomaly detection in "who-calls-whom" network, large sets of vertices which look like cliques are suspicious.
  - Vertices correspond to humans
  - Edges denote at least one phone call exchange
- Many more applications rely on dense subgraph discovery, correlation mining, graph visualization, mining Twitter data, bioinformatics.

Related work

- Find $S \subseteq V$ that maximizes the degree density $\rho(S) = \frac{e(S)}{|S|}$.
  - The densest subgraph problem (DSP) is solvable in polynomial time.
  - 2-approximation peeling algorithm which uses linear space $O(n+m)$ and runs in linear time $O(n+m)$ due to Charikar.
- Unfortunately, optimizing the DSP does not always result in finding "clique"-like sets.
  - Football network ($n = 115, m = 613$). The densest subgraph is the whole network with resulting edge density $f_e(S) = \frac{e(S)}{\binom{|S|}{2}} = 0.094$.
  - (Semi)-Streaming algorithm. $O(\log(n)/\epsilon)$ passes over the edge stream, achieves $(2+\epsilon)$ approximation and requires $\tilde{O}(n)$ space due to Bahmani et al. 2012.
  - Dynamic graphs. There exists $(2+\epsilon)$-approximation algorithm, $O(polylog(n)) = \tilde{O}(1)$ amortized time per update, $O(n+m)$ space under the assumption that deletions are *random* due to Epasto et al., 2015.

Main contributions

Theorem 1 (STOC'15) *Let $\epsilon \in (0,1)$, $\lambda > 1$ constant and $T = \lceil n^{\lambda} \rceil$.*

- *There is an algorithm that processes the first T updates in the dynamic stream such that:*
  - *It uses $\tilde{O}(n)$ space* (**Space efficiency**)
  - *It maintains a value $\textsc{Output}^{(t)}$ at each $t \in [T]$ such that for all $t \in [T]$ whp*

$$\textsc{Opt}^{(t)}/(4+\Theta(\epsilon)) \leq \textsc{Output}^{(t)} \leq \textsc{Opt}^{(t)}.$$

Also, the total amount of computation performed while processing the first T updates in the dynamic stream is $O(T \text{ poly}\log n)$. (**Time efficiency**)

Theorem 2 (STOC'15) *We can process a dynamic stream of updates in the graph G in $\tilde{O}(n)$ space, with a single pass and with high probability return a $(2+O(\epsilon))$-approximation of $d^* = \max_{S \subseteq V} \rho(S)$ at the end of the stream.*

Theorem 3 (KDD'15) *Sample each edge $e \in E_{\mathcal{H}}$ independently with probability $p = \frac{6}{\epsilon^2}\frac{\log n}{D}$. Then, the following statements hold simultaneously with high probability:*

- *For all $U \subseteq V$ such that $\rho(U) \geq D$, $\tilde{\rho}(U) \geq (1-\epsilon)C\log n$ for any $\epsilon > 0$.*
- *For all $U \subseteq V$ such that $\rho(U) < (1-2\epsilon)D$, $\tilde{\rho}(U) < (1-\epsilon)C\log n$ for any $\epsilon > 0$.*

Corollary 1 (KDD'15) *We improve the approximation guarantee of the single pass dynamic streaming algorithm to $(1+\Theta(\epsilon))$.*

Theorem 4 (WWW'15) *Consider the following generalization of the DSP, the k-clique DSP. The goal is to maximize the k-clique density $h_k(S)$, $k \geq 2$ as $h_k(S) = \frac{c_k(S)}{s}$, where $c_k(S)$ is the number of k-cliques induced by S and $s = |S|$.*

- *For any constant K, the K-clique densest subgraph problem can be solved exactly in polynomial time.*
- *Furthermore, we can $\frac{1}{k}$-approximate it using any K-clique counting algorithm as subroutine.*

Key concept – (α, d, L)-decomp.

Definition 1 *Fix any $\alpha \geq 1$, $d \geq 0$, and any positive integer L. Consider a family of subsets $Z_1 \supseteq \cdots \supseteq Z_L$. The tuple $(Z_1, \ldots, Z_L)$ is an (α, d, L)-decomposition of the input graph $G = (V,E)$ iff $Z_1 = V$ and, for every $i \in [L-1]$, we have $Z_{i+1} \supseteq \{v \in Z_i : D_v(Z_i) > \alpha d\}$ and $Z_{i+1} \cap \{v \in Z_i : D_v(Z_i) < d\} = \emptyset$.*

Two key properties of the (α, d, L)-decomposition follow.

Theorem 5 *Fix any $\alpha \geq 1$, $d \geq 0$, $\epsilon \in (0,1)$, $L \leftarrow 2 + \lceil \log_{(1+\epsilon)} n \rceil$. Let $d^* \leftarrow \max_{S \subseteq V} \rho(S)$ be the maximum density of any subgraph in $G = (V,E)$, and let $(Z_1, \ldots, Z_L)$ be an (α, d, L)-decomposition of $G = (V,E)$. We have: (1) If $d > 2(1+\epsilon)d^*$, then $Z_L = \emptyset$, and (2) if $d < d^*/\alpha$, then $Z_L \neq \emptyset$.*

(Rough) Idea of how to turn the previous theorem into an algorithm.

- Discretize the range of d^* as $d_k \leftarrow (1+\epsilon)^{k-1} \cdot \frac{m}{n}$, $k \in [K]$ where $K = O(\log_{1+\epsilon}(n))$.
- For every $k \in [K]$, construct an (α, d_k, L)-decomposition $(Z_1(k), \ldots, Z_L(k))$, where $L = O(\log_{1+\epsilon}(n))$.
- Let $k' \leftarrow \max\{k \in [K] : Z_L(k) \neq \emptyset\}$.

Then we have the following guarantees:

1. $d^*/(\alpha(1+\epsilon)) \leq d_{k'} \leq 2(1+\epsilon) \cdot d^*$.
2. There exists an index $j' \in [L]$ such that $\rho(Z_{j'}) \geq d_{k'}/(2(1+\epsilon))$.

Sketching the idea of the streaming algorithm. The key lemma on which we rely on is the following. Using a collection of $cm(L-1)\log n/d$ mutually independent simple random edges, we can construct from S an (α, d, L)-decomposition whp. The total space used is $O((n + m/d) \text{ poly}\log n) = \tilde{O}(n)$

- "Guess" the number of edges m.
- For each guess of m, build $O(\log n/\epsilon)$ $(\alpha, d_k = (1+\epsilon)^{k-1}\frac{m}{n}, L)$-decompositions, one for each density guess d_k. Set $\alpha = \frac{1+\epsilon}{1-\epsilon}$.
- For each guess of d_k maintain a sample S of $cm(L-1)\log n/d_k = \tilde{O}(n)$ random edges.
- Perform peeling based on expected values and find k'.

Experimental results

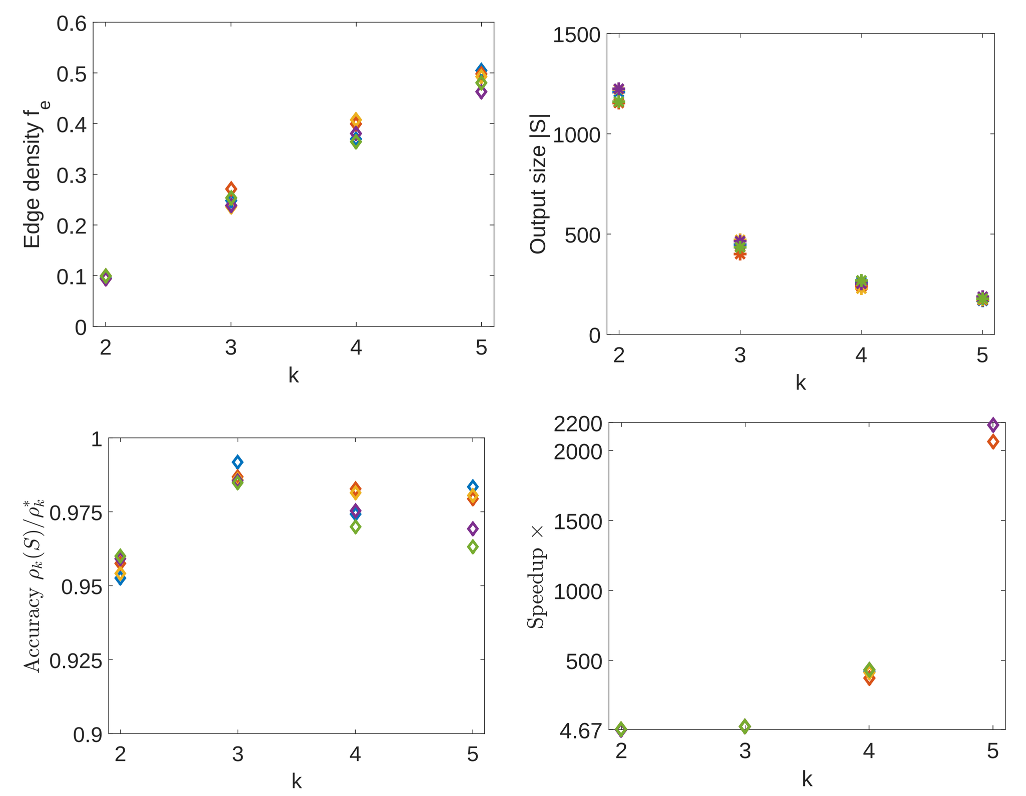
k-cliques

G	$k=2$		$k=3$		$k=4$	
	f_e	$\lvert S\rvert$	f_e	$\lvert S\rvert$	f_e	$\lvert S\rvert$
⋆	0.12	1 012	0.26	432	0.40	235
⊙	0.11	18 686	0.80	76	0.96	62
■	0.19	16 714	0.54	102	0.59	92
⊙	0.13	553	0.38	167	0.48	122

(p,q)-bicliques

G	$(p,q)=(1,1)$		$(p,q)=(2,2)$		$(p,q)=(3,3)$	
	f_e	$\lvert S\rvert$	f_e	$\lvert S\rvert$	f_e	$\lvert S\rvert$
⋆	0.001	9 177	0.06	181	0.30	40
⋆	0.001	6 437	0.41	18	0.43	17

- Effect of sampling on Epinions network.



Open problems

- Can we improve the $(4+\epsilon)$ approximation guarantee? What about weighted graphs?
- Space- and time-efficient fully dynamic algorithm for other graph problems, e.g., single-source shortest paths?

References

[1] Sayan Bhattacharya, Monika Henzinger, Danupon Nanongkai, Charalampos E. Tsourakakis. Space- and Time-Efficient Algorithms for Maintaining Dense Subgraphs on One-Pass Dynamic Streams STOC 2015

[2] Charalampos E. Tsourakakis. The k-clique densest subgraph problem. WWW 2015

[3] Michael Mitzenmacher, Jakub Pachocki, Richard Peng, Charalampos E. Tsourakakis, Shen Chen Xu. Scalable Large Near-Clique Detection in Large-Scale Networks via Sampling KDD 2015