The COMAD 2014 Programming Contest

Akhil Arora Arnab Bhattacharya
akhil.arora@xerox.com arnabb@cse.iitk.ac.in

1 Motivation

Mining interesting graph patterns has attracted much attention of late, with interests in studying protein interaction networks [PJZ05, SIKS06, YHC08], statistically significant pattern mining [HS06, YCHY08, RS09, SB12, ASB14, LPP14], and mining dense subgraphs and cliques [MYTM09, LW08, WZZ06, ZWZK06]. To take this further and to revisit the state-of-the-art, we take this opportunity to organize the COMAD 2014 programming contest and invite interesting solutions on *mining densest subgraphs from a given graph*.

The problem of finding relatively highly dense substructures in various social networks such as communication networks, collaboration networks and web networks has received a lot of attention [Gol84, Cha00, GKT05, KP93, Law01]. Experiments have suggested that such subgraphs correspond to *communities* in those networks. Formally, given an undirected un-weighted graph, the problem is to find the *densest subgraph of the given graph*.

The combinatorial algorithms based on max flow computation for finding maximum density subgraphs have been studied extensively [Gol84, Law01]. However, the time complexity of exactly finding the densest component using these algorithms is cubic or worse, and hence, they are not scalable for large graphs in real-world settings. Although the previous stated results are the best that any deterministic algorithm can achieve, there exist fast linear time algorithms for computing approximate solutions [Cha00, KP93]. However, they are destructive and, thus, return only a single subgraph.

There are other important variations such as finding the *densest k subgraphs*. This problem, unfortunately, is NP-complete [FPK01].

2 Problem Statement

We first define the density of a subgraph.

Definition 1 (Density of a Subgraph) *Suppose $S = (V', E')$ be any subgraph of $G(V, E)$ where $V' \subseteq V$ and an edge $e' = (u', v') \in E'$ if and only if $e' \in E$. The* density *of the subgraph S, denoted by ρ_S, is defined as the ratio of the number of edges $m' = |E'|$ to the number of vertices $n' = |V'|$:*

$$\rho_S = \frac{m'}{n'} \tag{1}$$

We invite participants to submit novel and interesting solutions that aim at solving the following technical problems.

- Given an *undirected* and *unlabeled* graph G, find the *densest* subgraph, i.e., S for which ρ_S is *maximal*.
- Moreover, extend this algorithm to find the *top-k densest* subgraphs that do not share any vertex with the previous top subgraphs, i.e., a set of $\{S_1, \ldots, S_k\}$ such that ρ_{S_i} are *maximal* and $\forall_{i,j}(S_i \cap S_j) = \Phi$.

Participants can choose to solve either just the first problem or both of them. Depending on the problem being solved, the result could either contain information about a single subgraph or k subgraphs.

The results must be returned according to the following format:

- The first line in the output should contain a positive integer, indicating the number of subgraphs returned by the code. If only the first problem is attempted, this line should simply contain 1. Otherwise, this should be the number of subgraphs returned.
- The subsequent lines, whose number will depend on the integer mentioned in the first line, should contain 3 tab-separated values:
  1. Density, ρ_S, of the returned subgraph, S. This should be a single real number, e.g., 0.67.
  2. Set of vertices, V', of the returned subgraph, S. These should be enveloped between curly braces ($\{\}$) and should internally be comma-separated, e.g., $\{v'_1, v'_2, v'_3\}$.

3. Set of edges, E', of the returned subgraph, S. An edge e' should be represented as a pair of two vertices (u', v'). These should be enveloped between curly braces ({}) and should internally be comma-separated, e.g., $\{(v'_1, v'_2), (v'_2, v'_3)\}$.

We will use the publicly available graph data sets from the Stanford Large Network Database collection (SNAP, http://snap.stanford.edu/data), to evaluate the solutions. We will consider input graphs from the following two categories of datasets:

1. Networks with ground-truth communities
2. Collaboration networks

The code written by participants should handle the input format of the corresponding mentioned categories as published on the website. More specifically, the participants should expose a command-line option with two input values:

1. The first is a positive integer. If only the first problem is attempted, it should be simply 1. Otherwise, this can be any k.
2. The second argument is the graph input file in the format directly available from the website.

A note of caution is the fact that many of these graphs are big enough to not fit in the memory directly.

The solutions would be evaluated based on the following two parameters:

1. **Optimal Density:** The optimal density here is the density of the subgraph returned after running the Goldberg's algorithm [Gol84] on our given graph. The solutions should aim at reducing the gap between the output from their approach and the optimal density as computed from the Goldberg's algorithm.
2. **Running Time:** The running time to obtain the optimal density again can be checked by using the Goldberg's algorithm. The participants need to reduce this time as much as possible, while keeping the density as close to the optimal as possible.

3 Submissions

The submission guidelines and the execution environment details are provided below:

- Each submission should be a zip file and should be named as `<team-name>_<submission_number>.zip`. If your team name is *XYZ* and you are submitting your solution for the third time, then the file name should be `XYZ_3.zip`.
- Each submission should contain all the necessary code, scripts, dependencies, libraries etc. to execute your program. Apart from this, it should also contain a detailed `README` (preferably with a `makefile`) that describes a step-by-step procedure to compile and run the submission.
- In addition, the final *executable* should be also included.
- Each submission should follow the input and output specification as mentioned in Section 2.
- Each submission would be evaluated against the datasets (chosen by the judges) from the ones as mentioned in Section 2. The participants can judge the complexity and nature of the datasets by looking on the ones mentioned on the SNAP website.
- The codes would be run sequentially (no multi-threading, no map-reduce) on an Intel(R) Xeon(R) E5645 12-core machine with 2.4 GHz CPU, 92 GB RAM, and 2 TB HDD running Linux Debian 6.0.7. The participants can choose any programming language/technology/tool, with the constraint that it should be installed freely and readily on a Linux Debian desktop.
- Finally, the participants are free to use any open/existing code-bases, libraries to facilitate an easy, neat and faster code for the same. However, the participants will need to properly acknowledge this work in their reports.
- Each team is allowed up to 3 submissions.

4 Evaluation

The evaluation will be done using a scoring function that takes into account both the density of the solution and the running time. Currently, the function can be simply thought of as a ratio of the density and the time:

$$\text{score} = \frac{\text{density}}{\text{running time}}$$

The solution with the *highest* score wins.

However, to prevent trivial solutions, those that return subgraph(s) with density less than $\tau\%$ of the optimal density will be rejected. Currently, τ is set to be 50%.

The committee reserves the right to change both the scoring function and the threshold τ with adequate notice to the participants.

5 Resources

We will also like to provide the participants with a headstart to the resources or papers, that we think, would be helpful in arriving at an interesting and novel solution. However, these resources are in no way limiting.

- Andrew V. Goldberg: Finding a Maximum Density Subgraph. http://www.eecs.berkeley.edu/Pubs/TechRpts/1984/CSD-84-171.pdf
- Victor E. Lee, Ning Ruan, Ruoming Jin, Charu C. Aggarwal: A Survey of Algorithms for Dense Subgraph Discovery. Managing and Mining Graph Data 2010: 303-336. http://charuaggarwal.net/dense-survey
- Akhil Arora, Mayank Sachan, Arnab Bhattacharya: Mining statistically significant connected subgraphs in vertex labeled graphs. SIGMOD, 2014: 1003-1014.
- David Gibson, Ravi Kumar, Andrew Tomkins: Discovering Large Dense Subgraphs in Massive Graphs. VLDB, 2005: 721-732.
- Charalampos E. Tsourakakis, Francesco Bonchi, Aristides Gionis, Francesco Gullo, Maria A. Tsiarli: Denser than the densest subgraph: extracting optimal quasi-cliques with quality guarantees. KDD, 2013: 104-112. http://videolectures.net/kdd2013_tsourakakis_quasi_cliques/
- Samir Khuller, Barna Saha: On Finding Dense Subgraphs. http://www.cs.umd.edu/~samir/talks/ismp-dense.pdf
- Finding Dense Subgraphs. http://www.cs.princeton.edu/~zdvir/apx11slides/charikar-slides.pdf

6 Rules

The competition is open to all. Participants can be in teams. There is a limit of 5 on the team size.

7 Deadline

- Release of problem statement: 15th September.
- Registration deadline of teams: 3rd November.
- Submission: 5th November.
- Announcement of results: 12th November.

8 Prizes

The top-3 solutions and/or novel/interesting solutions will be invited to present a poster and demonstrate their software in the conference. In addition, the committee may invite the authors of those solutions to write a joint paper which may be published as part of the COMAD conference or at some other venue or journal. The top-3 teams will be presented with a certificate as well.

9 Sample Code

A sample program is given in `sample.py`. It works on the input file `CA-GrQc.csv` and produces the output file `dense-CA-GrQc.csv`.

References

[ASB14] Akhil Arora, Mayank Sachan, and Arnab Bhattacharya. Mining statistically significant connected subgraphs in vertex labeled graphs. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, SIGMOD '14, pages 1003–1014, 2014.

[Cha00] Moses Charikar. Greedy approximation algorithms for finding dense components in a graph. In *Proceedings of the Third International Workshop on Approximation Algorithms for Combinatorial Optimization*, APPROX '00, pages 84–95, 2000.

[FPK01] U. Feige, D. Peleg, and G. Korsatz. The dense k-subgraph problem. *Algorithmica*, 29(3):410–421, 2001.

[GKT05] David Gibson, Ravi Kumar, and Andrew Tomkins. Discovering large dense subgraphs in massive graphs. In *Proceedings of the 31st International Conference on Very Large Data Bases*, VLDB '05, pages 721–732, 2005.

[Gol84] A. V. Goldberg. Finding a maximum density subgraph. Technical report, University of California, Berkeley, 1984.

[HS06] Huahai He and Ambuj K. Singh. Graphrank: Statistical modeling and mining of significant subgraphs in the feature space. In *Proceedings of the Sixth International Conference on Data Mining*, ICDM '06, pages 885–890, 2006.

[KP93] G. Kortsarz and D. Peleg. On choosing a dense subgraph. In *Foundations of Computer Science, 1993. Proceedings., 34th Annual Symposium on*, pages 692–701, Nov 1993.

[Law01] Eugene Lawler. *Combinatorial Optimization: Networks and Matroids*. Dover Publications, 2001.

[LPP14] Jefrey Lijffijt, Panagiotis Papapetrou, and Kai Puolamäki. A statistical significance testing approach to mining the most informative set of patterns. *Data Min. Knowl. Discov.*, 28(1):238–263, 2014.

[LW08] Guimei Liu and Limsoon Wong. Effective pruning techniques for mining quasi-cliques. In *Machine Learning and Knowledge Discovery in Databases*, volume 5212 of *Lecture Notes in Computer Science*, pages 33–49. Springer, 2008.

[MYTM09] Tsutomu Matsunaga, Chikara Yonemori, Etsuji Tomita, and Masaaki Muramatsu. Clique-based data mining for related genes in a biomedical database. *BMC Bioinformatics*, 10, 2009.

[PJZ05] Jian Pei, Daxin Jiang, and Aidong Zhang. Mining cross-graph quasi-cliques in gene expression and protein interaction data. In *Proceedings of the 21st International Conference on Data Engineering*, ICDE '05, pages 353–354, 2005.

[RS09] Sayan Ranu and Ambuj K. Singh. Graphsig: A scalable approach to mining significant subgraphs in large graph databases. In *Proceedings of the 2009 IEEE International Conference on Data Engineering*, ICDE '09, pages 844–855, 2009.

[SB12] Mayank Sachan and Arnab Bhattacharya. Mining statistically significant substrings using the chi-square statistic. *Proc. VLDB Endow.*, 5(10):1052–1063, 2012.

[SIKS06] Jacob Scott, Trey Ideker, Richard M. Karp, and Roded Sharan. Efficient algorithms for detecting signaling pathways in protein interaction networks. In *Journal of Computational Biology*, pages 133–144, 2006.

[WZZ06] Jianyong Wang, Zhiping Zeng, and Lizhu Zhou. Clan: An algorithm for mining closed cliques from large dense graph databases. In *ICDE*, pages 73–84, 2006.

[YCHY08] Xifeng Yan, Hong Cheng, Jiawei Han, and Philip S. Yu. Mining significant graph patterns by leap search. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 433–444, 2008.

[YHC08] Chang Hun You, Lawrence B. Holder, and Diane J. Cook. Temporal and structural analysis of biological networks in combination with microarray data. In *CIBCB*, pages 62–69, 2008.

[ZWZK06] Zhiping Zeng, Jianyong Wang, Lizhu Zhou, and George Karypis. Coherent closed quasi-clique discovery from large dense graph databases. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '06, pages 797–802, 2006.