Heuristic Algorithms for finding 2-clubs in an undirected graph

Chin-Ping Yang
Dept. of CSIE
CCU, Taiwan
ycp97m@cs.ccu.edu.tw

Hung-Chou Chen
Dept. of CSIE
CCU, Taiwan
allenc427@gmail.com

Sin-Da Hsiea
Dept. of CSIE
CCU, Taiwan
ibmibmibm.tw@gmail.com

Bang Ye Wu
Dept. of CSIE
CCU, Taiwan
bangye@cs.ccu.edu.tw

***Abstract*—In a graph G, a k-club is a node set inducing a subgraph of diameter at most k. The structure is useful to define a community or friend group on a social network. However, similar to the maximum clique problem, finding the largest k-club for any fixed $k \geq 1$ is NP-hard. In this paper, we propose a new heuristic algorithm to determine a largest 2-club. To compare the performance, we also implement the other three methods. The experimental results show that our method indeed improves the previous algorithms.**

***Index Terms*—k-clubs, social network analysis, heuristic algorithms**

I. Introduction

Social and behavioral scientists frequently use network analysis to identify the relationship between groups, individuals or any abstract entities. It's the work related to graph analysis and also one branch of data mining called *social network analysis* (SNA). From a statistical point of view, SNA can be described as directed graphs for there is an actor(node) who knows other by adding a directional edge from himself to the other. Moreover, the applications are widely developed and among these problems scientists usually tend to focus on how to identify dense structures in a graph. For example, Baker [1] analyzed market networks, Snyder and Kick [9] studied transactional interactions between nations, and Mintz and Schward [7] investigated interlocking directorates in major corporations.

The most widely known in this type of research is to find out *clique* defined as a set of nodes all directly linked to each other by an edge, i.e. a complete subgraph. In SNA, finding a friend group can be referred to finding a clique in a graph. The idea of a clique is relatively simple and the subgraph is closely and intensely. However, the size of cliques we found could be small in general since its restricted condition. To overcome this problem, several relaxations have been used [5]. A *k-clique* relaxes the distance constraint such that the distance from each node to any other is at most k but not necessary 1. For example, the set $S = \{1, 2, 3, 4, 5\}$ in Fig.1 is a 2-clique by definition. However, the shortest path between two nodes may pass through a node outside the k-clique. For example, node 4 and 5 passes through node 6 which does not belong to S. It doesn't make sense in some applications like friend group in reality.

Since there are property drawbacks that 1-clique is too restricted and 2-clique is too sparse, there is another usual way to identify more meaningful structures called *club*. A *k-club* is defined as a node set, by which the induced subgraph is of diameter at most k. The diameter of a graph is the largest distance between any two nodes. For example, $\{1, 2, 3, 4\}, \{1, 2, 3, 4, 5, 6\}$ in Fig.1 are 2-clubs but $\{1, 2, 3, 4, 5\}$ is not since the paths from each other can only pass through the nodes within club.

Due to the rapid growth of Internet, Internet service providers(ISP) provide many kinds of social network services(SNS), especially the personal web space such as *facebook*. People can interactive with each other by using various applications in the platform. One important function of such platforms is to introduce latent friends. One way to do this is introducing the friends of friends to a user (such as MSN and facebook currently do). But such a easy way may be not precise at all. Instead people in one 2-club may be latent friends more possibly.

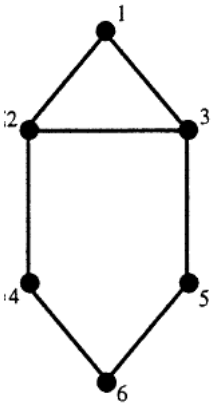

Fig. 1. An example of a 2-clique but not a 2-club

However, finding 2-clubs seems even more difficult than 2-clique although both problems are NP-hard. UCINet is a popular software for SNA [10]. It provides a function to compute k-clique but no function for k-club. Previously Jean-Marie, Gilbert, Gilles [2] showed that the maximum k-club problem (MkCP) in an undirected graph is NP-hard and give an exact branch-and-bound algorithm to obtain optimal within insignificant computing time only in very dense graphs. They also proposed three heuristics [3] in an earlier paper.

In this paper, we propose another heuristic algorithm to find the maximum 2-club (M2CP), which is modified from a previous method called DROP. To show the performance, we test it on randomly generated network. Comparing to the other three previous heuristics, the experimental results show that the new method performs better in many situations.

The paper is organized as follows. The new heuristic, as well as three previous ones are described in the section 2. Then we show the experimental results and discussions in section 3. Finally, conclusions are given in section 4.

II. The algorithms

We describe a new heuristic named VCOVER, as well as three compared methods for the 2-club problem in an undirected graph.

Heuristic 1: DROP

The algorithm DROP starts from a larger node set and removes the nodes one-by-one until it becomes a 2-club.

Algorithm DROP
For each v_i, do the following and choose the best.
1: Calculate all-to-all distance;
2: Initially set $W = V_i$;
Compute for each node v of W the number $q_v = |\{u \in V(W) : d(u, v) > 2\}|$;
If $q_v = 0$ for every node v, stop; W is a 2-club.
3: Delete from W the node with the smallest degree among the largest value of q_v.
4: Update the distances for W. Goto step 2.

For each node v_i, we first find the nodes whose distance to v_i is at most two. Let V_i be the set of such nodes. To speed up the computation, the algorithm starts from V_i instead of the whole node set. The algorithm is a modification of the one in [2]. Their algorithm first finds a maximum 2-clique and then removes the nodes one by one, therefore, it's less efficient when the number of node is large.

Heuristic 2: VCOVER

The heuristic DROP removes nodes one by one. Although it is easy but may be lack of globe view. To improve this drawback, we design a new heuristic VCOVER. The intuition is as follows. Let H be a network and F the set of node pair whose distance larger than two, i.e., $F = \{(u, v)|d_H(u, v) > 2\}$. Clearly for each pair (u, v) in F, at least one of the two nodes must be removed to make the remaining a 2-club. Therefore, at each iteration, we find minimum number of nodes to cover all the pairs in F, which is exactly the so-call "minimum vertex cover" of the graph induced by the edge set F.

The minimum vertex cover problem is NP-hard [4]. We use the algorithm in [8]. Fortunately the instance size for the problem is small and the algorithm is practically efficient.

Algorithm VCOVER
For each v_i, do the following and choose the best.
1: Let $H = G[V_i]$;
2: find $F = \{(u, v)|d_H(u, v) > 2\}$;

3: if $F = \emptyset$ return H as the 2-club;
4: construct graph P induced by the edge F;
5: find a minimum vector cover C of P;
6: remove C from $V(H)$ and goto Step 2;

To exhibit the performance of VCOVER, in the following we introduce two previous methods for comparison. One is INC [6], and the other is STAR [2].

Heuristic 3: INC

Different from the first two methods, INC uses the concept of incrementally adding nodes, one by one or two by two. The algorithm is described as follows:

Algorithm INC
/* Let $V = \{v_i | 1 \leq i \leq n\}$ be the node set */
For i = 1 to n do the following p times and choose the best
    $C \leftarrow \{v_i\}$;
    repeat forever
        call ADD_1;
        if ADD_1 returns FAIL then call ADD_2;
        if both ADD_1 and ADD_2 return FAIL then
            compare C with best result and goto next iteration;
    end repeat;
end for;

Procedure ADD_1
Input: A 2-club C
find all nodes $v \in V - C$ such that $C \cup \{v\}$ is a 2-club;
put all such nodes into S;
if S is empty then
    return FAIL;
else
    randomly choose a node from S and insert it into C;
    return SUCCESS;

The procedure ADD_2 is similar to ADD_1 and is omitted here. The only difference is that it tries to insert two nodes at a time instead of one node at a time. We can see that it is a randomized algorithm, and the solution varies with the different choices. Therefore, the quality of the result depends on the times we repeat the program. The repetitions are used in three places of the program: The first one is that we use every node as an beginning to build a 2-club. The second is that for each staring node we repeat p times. The third is that we do the whole procedure q times.

Heuristic 4: STAR

This algorithm simply determines a node of maximum degree in G and output the node as well as all its neighbors as a 2-club. Although it is a simple method, the solution would be fine in some certain cases and of course the least execution time. The analysis of these cases and experimental results would be shown later on.

III. Experimental results

A. Environment and data instance

These algorithms were coded in C++ and run on a PC. The test graphs are generated randomly by an algorithm used in [2]. The graph density is controlled by two density parameters a and b, $0 \leq a \leq b \leq 1$. For each node v, we first generate a random number $p[v]$ drawn uniformly from the interval $[a, b]$. Then, for each node u and v, the edge (u, v) exists with probability $(p[u] + p[v])/2$. The excepted edge density of the graph is equal to $(a + b)/2$ and the node degree variance increases with $b - a$. Tests were performed on 17 cases controlled by several combination of a and b of 20-node and 50-node graphs. There are 100 instances tested for each case.

B. Running time

Theoretical time Complexities:

- The most time-consuming step of the DROP algorithm is to compute all-pairs shortest path. In our implementation we reduce the time complexity to $O(|V_i|^2 + |V_i||E_i|)$, in which V_i is the set of nodes with distance at most two to node v_i and E_i is the corresponding edge set. For small degree graphs, $|V_i|$ and $|E_i|$ are small. Since at most $|V_i|$ nodes are deleted and we do the procedure for each node

v_i, the overall time complexity of DROP is $O(\sum_i(|V_i|^3 + |V_i|^2|E_i|))$.

- For VCOVER, we implement a fixed parameter algorithm of a search tree of size $O(1.33^k)$, in which k is the solution size [8]. Although the minimum vertex cover problem is NP-hard, it's not surprising that VCOVER takes not much time since the small number of nodes in the auxiliary graphs. Another reason to make VCOVER efficient is that, in any iteration, the larger k is, the more nodes we delete. The efficiency is shown more obviously on middle density graphs.
- The most cost step of INC algorithm is to check nodes to be added into a 2-club, which takes $O(|V| + |E|)$ time. Since using every node as a beginning and repeating $p*q$ times, the overall time complexity is $O(pq(|V|^2 + |V||E|))$, where p and q are repetition times parameters controlled by users.
- The STAR algorithm is the simplest and fast. It takes $O(|V|+|E|)$ to find the maximum degree node.

Practical running time: Since the computing time on 20-node instances is very fast, the following discussion focuses on 50-node graphs.

- Among the four heuristic, INC and STAR cost almost no time.
- For DROP and VCOVER, the most cost case happened on the density around 0.1, spending about 1.3 seconds and 0.28 seconds per instance, respectively. But in low and high density, DROP and VCOVER take less time, about 0.4 seconds and 0.13 seconds, respectively.
- In summary, VCOVER runs about 4 times faster than DROP although the former involves an exponential time solver for an NP-hard problem.

C. Performance

The density parameters a and b are given in the column ab with the form $aabb$. For example, by $ab = 0515$ we mean that $a = 0.05$ and $b = 0.15$.

In Tables I and III we list the winning percentage of each of the four heuristics for all the cases of ab. For example, the first row indicates that, in 98 of the 100 test cases of $a = 0.05$ and $b = 0.05$, the clubs found by DROP are best. The total percentage of each row may exceed 100% when two or more methods tie in some instances.

Tables II and IV show the average number of nodes of the found 2-clubs on 20-node and 50-node graphs, respectively. In Table II we also list the sizes of optimal solutions, which were found by a brute-force exact algorithm.

By the experimental results, we make some discussions on the performances of the heuristics in the following:

For 20-node graphs:

- When density is 0.05, a maximum 2-clubs is almost just a star. When density is up to 0.25, almost the whole graph is a 2-club. That's the reason why we list the experimental results for densities between the two extremes.
- INC and STAR perform well only for very low density (0.05). As the density increases, we notice that the performances of INC and STAR are getting worse. But DROP and VCOVER still perform very close to the exact algorithm.

For 50-node graphs:

- Just as the 20-node cases, when density is 0.05, a maximum 2-clubs is almost just only a star. For density is 0.2 and 0.25, almost the whole graph is a 2-club. The most important cases are density=0.1 and 0.15.
- Same as 20-node cases, these four heuristics perform almost the same when density=0.05. DROP and VCOVER have the best solution.
- When density=0.1 and 0.15, INC and STAR work significantly worse. DROP performs better than VCOVER in some cases of density=0.1 but they almost tie when density=0.15.

In Summary:

- From these two experiments of 20-node and 50-node graphs, we observe that STAR behaves well only on graphs of very low density. For other cases, DROP and VCOVER are the best choices not only for giving good solutions but also taking not much time.
- In the cases of density=0.1 and 50 nodes, VCOVER performs worse but takes much less time than DROP. It's a trade-off for users between time and quality. But in the case of density=0.15, VCOVER is better than the others in most of the cases, both in quality

TABLE I
THE WINNING PERCENTAGE (20 NODES)

Density	ab	DROP	VCOVER	INC	STAR
0.05	0505	100	100	90	94
	0010	100	100	93	95
0.1	1010	98	99	36	49
	0515	98	97	30	43
	0020	100	100	40	68
0.15	1515	98	98	6	4
	1020	96	97	3	4
	0525	98	97	9	10
0.2	2020	99	100	12	0
	1525	100	98	13	0
	1030	99	98	13	1
0.25	2525	100	100	56	0
	2030	100	100	57	0
	1535	100	100	50	0

TABLE II
AVERAGE SIZE OF 2-CLUBS FOR FOUR DIFFERENT HEURISTICS (20 NODES)

Density	ab	DROP	VCOVER	INC	STAR	Exact
0.05	0505	5.58	5.58	5.48	5.52	5.58
	0010	6.07	6.07	6.00	6.02	6.07
0.1	1010	8.87	8.88	7.92	8.13	8.89
	0515	9.17	9.16	8.11	8.35	9.19
	0020	8.89	8.89	8.08	8.41	8.89
0.15	1515	13.38	13.36	10.27	10.42	13.41
	1020	13.29	13.31	10.13	10.51	13.34
	0525	12.88	12.85	10.02	10.48	12.9
0.2	2020	17.52	17.52	13.82	12.53	17.53
	1525	17.64	17.64	13.8	12.67	17.66
	1030	16.73	16.72	13.2	12.54	16.76
0.25	2525	19.64	19.64	17.69	14.52	19.64
	2030	19.57	19.57	17.95	14.56	19.57
	1535	19.31	19.31	17.35	14.47	19.31

TABLE III
THE WINNING PERCENTAGE (50 NODES)

Density	ab	DROP	VCOVER	INC	STAR
0.05	0505	90	91	66	88
	0010	100	100	30	92
0.1	1010	83	44	0	2
	0515	87	53	0	4
	0020	88	84	0	13
0.15	1515	95	90	0	0
	1020	88	91	0	0
	0525	86	91	0	0
0.2	2020	100	100	5	0
	1525	100	100	4	0
	1030	98	99	1	0
0.25	2525	100	100	41	0
	2030	100	100	44	0
	1535	99	99	30	0

TABLE IV
AVERAGE SIZE OF 2-CLUBS FOR FOUR DIFFERENT HEURISTICS (50 NODES)

Density	ab	DROP	VCOVER	INC	STAR
0.05	0505	10.92	10.92	9.24	10.85
	0010	12.22	12.22	10.11	12.13
0.1	1010	22.47	21.7	12.95	17.25
	0515	23.57	22.94	13.74	18.09
	0020	23.32	23.29	14.38	18.91
0.15	1515	45.21	45.15	18.25	23.15
	1020	43.6	43.62	18.07	23.36
	0525	40.47	40.54	18.12	24.41
0.2	2020	49.83	49.83	26.3	28.37
	1525	49.79	49.79	26.33	28.84
	1030	49.17	49.19	25.85	29.52
0.25	2525	50	50	39.7	33.41
	2030	50	50	41.46	33.6
	1535	49.99	49.99	36.78	33.5

and in efficiency. Therefore, if quality is the most important consideration, we suggest to use both VCOVER and DROP algorithms and then choose the better.

- For a specified node v, the 2-club can only include nodes with distance at most two to v. The locality property makes us determine the number of nodes in our experiments. It may be less of meaning to do experiment on graphs of too many nodes.

IV. CONCLUSION

We have described four heuristics for determining a maximum 2-club in an undirected graph. The experiment result on 20-node graph suggests that DROP and VCOVER are effective and promising since the performance of 17 cases is almost as good as exact solution.

The result that DROP wins VCOVER in some cases points out that maybe we should give more considerations on the centrality (importance) of nodes in the deletion step of VCOVER, such as degree, betweenness and distance.

The 2-club problem has several applications nowadays due to the rapid growth of Internet. We will keep working on finding k-club for $k > 2$. Another interesting problem is the club problem on directed graph [6].

ACKNOWLEDGMENTS

The work was supported in part by an NSC Grant NSC97-2221-E-366-001-MY3 from the National Science Council, Taiwan.

REFERENCES

[1] W.E. Baker, Market networks and corporate behavior. *American Journal of Mathematical Sociology*, 12:191–223, 1986.

[2] J.M. Bourjolly, G. Laporte and G. Pesant, Heuristics for finding k-clubs in an undirected graph, *Computers Operations Research*, 27:559–569, 2000.

[3] J.M. Bourjolly, G. Laporte and G. Pesant, An exact algorithm for the maximum k-club problem in an undirected graph, *European Journal of Operational Research*, 138:21–28, 2002.

[4] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to The Theory of NP-Completeness*, Freeman, NewYork, 1979.

[5] R.A. Hanneman and M. Riddle, *Introduction to social network methods*, online at http://www.faculty.ucr.edu/hanneman/nettext/, 2005.

[6] S.T. Kuan, B.Y. Wu, and W.J. Lee, Finding friend groups in Blogsphere, in the Proceedings of *22nd International Conference on Advanced Information Networking and Applications*, 1046–1050, 2008.

[7] B. Mintz and M. Schwartz, Interlocking directorates and interest group formation. *American Sociological Review*, 46:851–69, 1981.

[8] R. Niedermeier, *Invitation to Fixed Parameter Algorithms*, in *Oxford Lecture Series in Mathematics and Its Applications*, 2006.

[9] D. Snyder, and EL. Kick, Structural position in the world system and economic growth, 1955–1970: a multiple-network analysis of transactional interactions. *American Journal of Sociology*, 84:1096–126, 1979.

[10] UCINet, Release 6.0, Mathematical Social Science Group, Social School of Science, University of California at Irvine.

[11] S. Wasserman and K. Faust, *Social Network Analysis*, Cambridge University Press, Cambridge, 1994.