European Journal of Operational Research 138 (2002) 21–28

EUROPEAN JOURNAL OF OPERATIONAL RESEARCH

www.elsevier.com/locate/dsw

Discrete Optimization

An exact algorithm for the maximum k-club problem in an undirected graph

Jean-Marie Bourjolly [a,b], Gilbert Laporte [b,c,*], Gilles Pesant [c,d]

[a] *Department of Decision Sciences and MIS, Concordia University, 1455 de Maisonneuve boulevard West, Montréal, Canada H3G 1M8*

[b] *Centre de recherche sur les transports, Université de Montréal, Case postale 6128, Succursale "Centre-ville", Montréal, Canada H3C 3J7*

[c] *GERAD and École des Hautes Études Commerciales, 3000 chemin de la Côte-Sainte-Catherine, Montréal, Que., Canada H3T 2A7*

[d] *Département de génie électrique et de génie informatique, École Polytechnique de Montréal, Case Postale 6079, Succursale "Centre-ville", Montréal, Canada H3C 3J7*

Received 15 May 2000; accepted 1 March 2001

Abstract

In this paper, we prove that the maximum k-club problem (MkCP) defined on an undirected graph is NP-hard. We also give an integer programming formulation for this problem as well as an exact branch-and-bound algorithm and computational results on instances involving up to 200 vertices. Instances defined on very dense graphs can be solved to optimality within insignificant computing times. When $k = 2$, the most difficult cases appear to be those where the graph density is around 0.15. © 2002 Elsevier Science B.V. All rights reserved.

1. Introduction

The purpose of this article is to develop an exact branch-and-bound algorithm for the maximum k-club problem (MkCP) defined as follows. Let $G = (V, E)$ be an undirected connected graph, where $V = \{1, \dots, n\}$ is the vertex set, and the edge set E is a subset of $\{(i,j) : i,j \in V,\ i < j\}$. A subset S of V induces a subgraph $H(S)$ of G defined as $H(S) = (S, F(S))$, where $F(S) = \{(i,j) \in E : i, j \in S\}$. Denoted by c_{ij}^S the length of a shortest chain (defined as the number of its edges) between two vertices i and j of S using only edges of $F(S)$. The *diameter* of $H(S)$ is defined as $\max_{i,j \in S}\{c_{ij}^S\}$. A *$k$-club* is a vertex set S inducing a subgraph $H(S)$ of diameter at most k. The MkCP consists of determining a maximum cardinality k-club in G.

The concept of k-club is related to, but different from that of k-clique (or simply clique if $k = 1$). A k-clique is a set S of vertices such that every pair is linked by a chain in G, but not necessarily in $H(S)$, having a length at most equal to k. In other words, some of these chains may use vertices outside S, which is not the case of a k-club. While a k-club is always a k-clique, the reverse is not true as

* Corresponding author. Tel.: +1-514-343-6143; fax: +1-514-343-7121.

E-mail address: gilbert@crt.unmontreal.ca (G. Laporte).

illustrated by Fig. 1 taken from Alba [1,2] and Mokken [11]. Here $S = \{1,2,3,4,5\}$ is a 2-clique but not a 2-club since the only chain of length 2 linking 4 and 5 passes through 6 which is outside S.

Social and behavioral scientists frequently use network analysis to identify highly connected structures in societies and organizations. For example, Baker [3] has analyzed market networks, Snyder and Kick [13] have studied transactional interactions between nations, and Mintz and Schwartz [10] have investigated interlocking directorates in major corporations. Studies of this type rely on the identification of dense structures in a graph of which k-clubs are the most strongly interconnected. A common tool used in network analysis is the UCINET [14] package developed at the University of California at Irvine which makes use of the enumerative algorithm of Bron and Kerbosch [5] to identify dense structures in graphs. Because of its nature, this type of approach is limited to relatively small graph sizes. Heuristics are more than often necessary. In a previous study [4], we have developed three constructive heuristics for the MkCP. Here, we propose for the first time an exact enumerative algorithm for the same problem.

The remainder of this article is organized as follows. In Section 2, we prove that the MkCP is NP-hard. In Section 3, we formulate the problem as an integer linear program. The enumerative algorithm is described in Section 4, followed by computational results in Section 5.

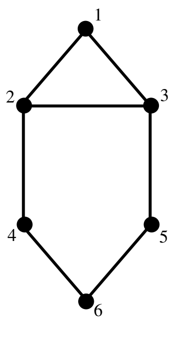

Fig. 1. $S = \{1,2,3,4,5\}$ is a 2-clique, but not a 2-club.

2. Complexity analysis

We prove in this section that the MkCP is NP-complete by reduction from the clique problem. The MkCP in decision form can be stated as follows.

> Given a graph $G = (V,E)$ and an integer $t \leqslant |V|$, does there exist $S \subseteq V$ with $|S| \geqslant t$ and such that in the subgraph induced by S there exists a chain of length not exceeding k between each pair of vertices?

Proposition 1. *The MkCP is in NP.*

Proof. One can verify in polynomial time that the vertex set of an induced subgraph of G is a k-club by determining a shortest chain between each pair of its vertices (using, for example, Floyd's algorithm [6]). □

To prove that the MkCP is NP-complete, we will use two types of reduction: one for the even values of k and one for the odd values of k. The idea is to replace each original edge by a chain of k edges, and the subtlety lies in how these chains are interconnected. Let $G' = (V',E')$ be a connected graph and suppose that k is even. For each edge $e = (i',j')$ of G' create $k-1$ *artificial* vertices $v_1^e, v_2^e, \dots, v_{k-1}^e$, together with the chain $(i', v_1^e, v_2^e, \dots, v_{k-1}^e, j')$. Let $\tilde{V}$ be the set of all artificial vertices and let $V = V' \cup \tilde{V}$. The vertices of V' are said to be *natural*. The vertices $v_{k/2}^e$, i.e., the middle vertices of all chains are pairwise joined together by an edge to form a complete subgraph (see Fig. 2(a)). Finally, let E be the set of all edges that either belong to a chain or to the complete subgraph.

When k is odd, the reduction is slightly different. Create an extra artificial vertex v, and let again $V = V' \cup \tilde{V}$. The vertices $v_{\lfloor k/2 \rfloor}^e$ and $v_{\lceil k/2 \rceil}^e$ of each chain are joined to v by an edge (see Fig. 2(b)). Let E be the set of all edges that either belong to a chain or are incident to v. In Fig. 3, we illustrate the construction for $k = 2$, 3 and 4 on a particular graph.

The following observations follow from the construction of $G = (V,E)$:

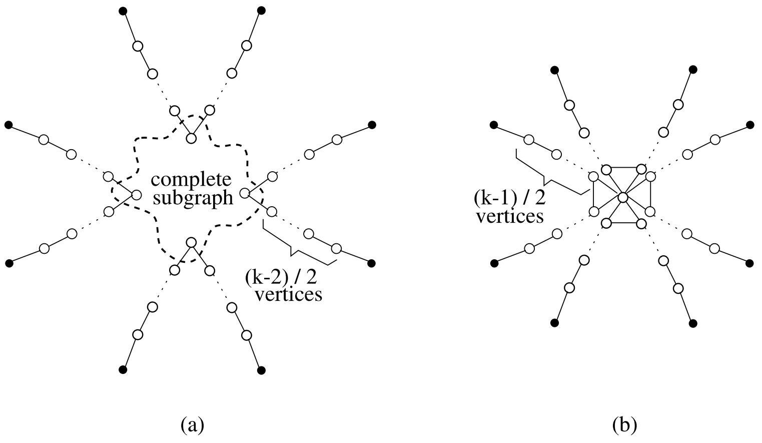

Fig. 2. Illustration of the structure of the artificial vertices for k even (a) and odd (b). The natural vertices are shown in black and the artificial vertices in white.

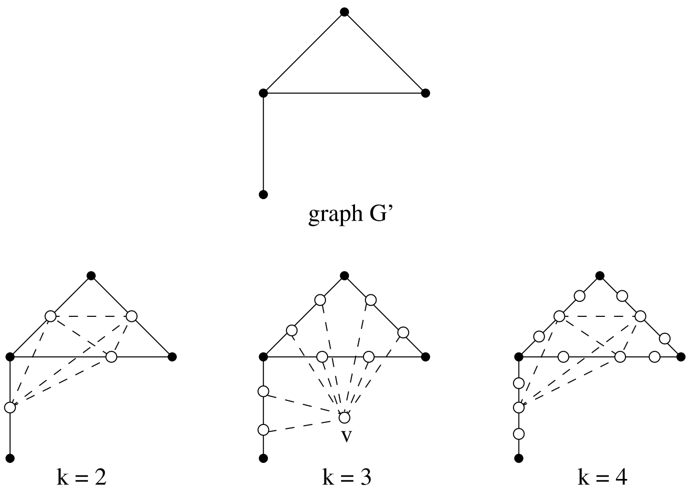

Fig. 3. Illustration of the construction for $k = 2,\ 3$ and 4 on a graph G'.

1. For any two artificial vertices there exists a chain of length at most $k-1$ between them involving only artificial vertices.
2. For any natural vertex and any artificial vertex there exists a chain of length at most k between them involving only artificial vertices.
3. The shortest distance between two natural vertices is k if they are joined by an edge in G', and $k+1$ otherwise. All intermediate vertices of such a shortest chain are artificial.

It follows from observations 1 and 2 that

Lemma 1. *If S is a k-club of G, then so too is $S \cup \tilde{V}$.*

In other words, if all the artificial vertices are added to a k-club of G, it remains a k-club.

Lemma 2. *There exists a clique C' in G' of size t' if and only if there exists a k-club C in G whose size is $t = t' + |\tilde{V}|$.*

Proof. $\Rightarrow$: follows directly from observation 3 and Lemma 1 by taking $C = C' \cup \tilde{V}$.

$\Leftarrow$: C must contain at least t' natural vertices. By observation 3, every pair of these vertices must be joined by an edge in G' and hence form a clique. □

Proposition 2. *The k-club problem is NP-complete.*

Proof. The thesis follows from Proposition 1, Lemma 2, and the fact that the construction is polynomial in the input size. □

3. Formulation

The MkCP can be formulated as an integer linear program as follows. For any two vertices $i,j \in V$, let C^k_{ij} be the sets of all chains of length at most k linking i and j, and denoted by V_t the vertex set of a chain t. For every $i \in V$, let x_i be a binary variable equal to 1 if and only if i belongs to the solution. Let y_t be an auxiliary binary variable associated with every chain $t \in \cup_{i,j\in V} C^k_{ij} = C$. The problem is then

$$\text{maximize} \quad \sum_{i\in V} x_i \tag{1}$$

subject to

$$\sum_{t\in C^k_{ij}} y_t \geqslant x_i + x_j - 1$$

$$\forall (i,j) \notin E, \quad C^k_{ij} \neq \emptyset, \tag{2}$$

$$y_t \leqslant x_r \quad \forall t \in C, \ \forall r \in V_t, \tag{3}$$

$$x_i + x_j \leqslant 1 \quad \forall (i,j) \notin E, \quad C^k_{ij} = \emptyset, \tag{4}$$

$$x_i = 0 \text{ or } 1 \quad \forall i \in V, \tag{5}$$

$$y_t = 0 \text{ or } 1 \quad \forall t \in C. \tag{6}$$

In this formulation, constraints (2) express the fact that if two vertices i and j are not linked by an edge but are linked by a chain of length at most k, then both vertices can belong to the solution only if y_t is equal to 1 for at least one such chain. Constraints (3) guarantee that all vertices of a chain t for which $y_t = 1$ belong to the solution. Constraints (4) mean that two vertices not linked by an edge or a chain of length at most k cannot simultaneously belong to the solution.

This formulation can be simplified if $k = 2$. For $i \in V$, let $N_i = \{j \in V : (i,j) \text{ or } (j,i) \in E\}$. Then variables y_t are not needed and the formulation reduces to

$$\text{maximize} \quad \sum_{i\in V} x_i, \tag{7}$$

subject to

$$\sum_{r\in N_i \cap N_j} x_r \geqslant x_i + x_j - 1$$

$$\forall (i,j) \notin E, \quad N_i \cap N_j \neq \emptyset, \tag{8}$$

$$x_i + x_j \leqslant 1 \quad \forall (i,j) \notin E, \quad N_i \cap N_j = \emptyset, \tag{9}$$

$$x_i = 0 \text{ or } 1 \quad \forall i \in V. \tag{10}$$

Constraint (8) play the same role as (2) and (3), while constraints (9) are a rewriting of (4).

4. Algorithm

Our exact algorithm is based on DROP [4], an earlier heuristic which gave good results for this problem. The DROP heuristic proceeds by removing vertices one at a time until those that are left form a k-club. A vertex lying too far from the largest number of vertices is favored for removal at each step. In order to efficiently identify such a vertex, DROP relies on an algorithm that incrementally updates shortest chain lengths between pairs of vertices [12].

The branch-and-bound algorithm we developed uses the DROP heuristic to guide its branching. It also uses solutions to the maximum stable set problem on an auxiliary graph for its bounding. At the root node of the search tree, two branches are generated corresponding to removing or keeping the vertex selected by a single iteration of the DROP heuristic. The branch that removes the vertex is explored first, thus trusting the heuristic. The process is then recursively applied until a terminal node is reached, yielding a depth-first search. Note that the first terminal node reached corresponds to the solution given by the DROP heuristic.

Deciding to remove a vertex may increase some of the shortest chain lengths since this vertex can no longer be used. Deciding to keep a given vertex

v may also increase shortest chain lengths since every vertex whose shortest chain to v is longer than k should immediately be deleted: shortest chains certainly do not get shorter as vertices are removed. This reasoning applies not only at the node where the decision to keep v is made, but also throughout the subtree rooted at that node, as new vertices come to lie too far from v. In fact, a pair of vertices kept in a tentative solution may become too far from one another, in which case a terminal failure node has been reached.

We now describe the upper bound computed at the nodes of the search tree. Let $G = (V, E)$ be the current graph at a given node. Associate to it an auxiliary graph $H = (V, F)$ on the same vertex set, where $F = \{(u, v)$: the shortest chain linking u and v in G has length greater than $k\}$. Clearly, if $(u, v) \in F$, vertices u and v cannot both belong to the same k-club in G. In fact, the largest stable set in H, that is a subset of its vertices without any edge between a pair of them, provides an upper bound on the size of the largest k-club in G. The maximum stable set problem [7] is well known to be NP-complete and several heuristics, but also exact algorithms, have been implemented to solve it [9]. Note that we require an exact solution to the stable set problem since otherwise the upper bound derived may not be valid. Note also that although running an exact algorithm for this problem can in principle be costly, H often contains several isolated vertices that can be set apart, so that the size of the auxiliary problem is considerably reduced (see Table 3), which ensures the effectiveness of our approach. This upper bound is not necessarily tight as shown in Fig. 4. Here, H yields a maximum stable set of cardinality 6, while the maximum 2-club in G is $\{2, 3, 4, 5, 6\}$.

Table 1
2-Clubs, each entry corresponds to an average over 30 random instances (three different density ranges with 10 instances each)

n	Average density	Largest 2-club		Computation time		Number of terminal nodes	
		DROP heuristic	DROP exact	No upper bound	Stable set	No upper bound	Stable set
50	0.05	6.8	7.5	2.7	2.9	47.9	44.4
	0.1	8.6	11.1	1.9	1.6	156.2	49.4
	0.15	13.2	15.8	2.2	1.2	397.8	60.9
	0.2	18.8	23.4	3.5	1.6	688.4	83.3
	0.25	30.9	34.5	1.9	0.5	390.9	47.1
	0.3	44.4	44.9	0.2	0.1	33.6	7.7
100	0.05	8.3	12.5	77.5	77.7	299.1	102.0
	0.1	12.3	20.1	120.9	51.6	3306.4	268.5
	0.15	23.6	32.1	1752.2	931.3	57 833.1	2912.5
	0.2	60.2	68.2	1620.8	181.2	39 613.0	747.0
	0.25	93.5	94.1	2.2	0.6	58.1	8.6
	0.3	99.5	99.5	0.4	0.3	1.9	1.5
150	0.05	7.9	17.8	582.6	351.8	1560.3	215.9
	0.1	17.1	29.3	7354.3	3150.3	82 846.0	1626.2
	0.15	41.3	–	–	–	–	–
	0.2	127.2	134.2	603.7	11.3	6174.0	77.0
	0.25	149.5	149.5	1.2	1.2	1.6	1.7
	0.3	150.0	150.0	1.2	1.2	1.1	1.1
200	0.05	8.4	22.0	3082.9	2198.4	4464.4	390.4
	0.1	18.4	–	–	–	–	–
	0.15	63.0	–	–	–	–	–
	0.2	194.0	194.3	109.8	4.3	638.3	8.6
	0.25	200.0	200.0	2.9	2.9	1.0	1.0
	0.3	200.0	200.0	3.0	3.0	1.0	1.0

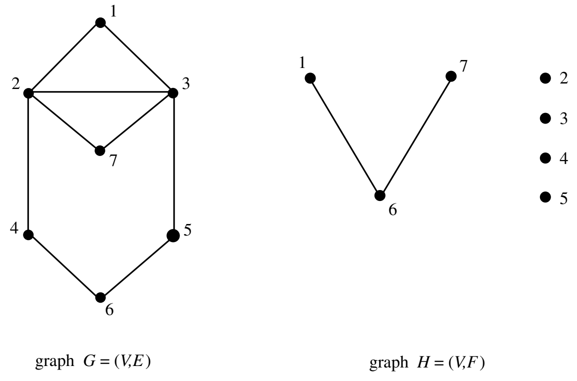

Fig. 4. Example for which the upper bound is not tight.

5. Computational results

The algorithm was coded in C++ and run on a Sun Sparc Ultra 10 workstation (440 MHz). It was tested on randomly generated graphs controlled by two density parameters a and b, using an algorithm described in [8]. The expected edge density of the graphs is equal to $(a+b)/2$ and the vertex degree variance increases with $b-a$. Tests were performed on 50-vertex to 200-vertex graphs for combinations of a and b that yielded meaningful instances: when the graph density becomes too large, all vertices belong to a maximum k-club and the corresponding problem is then trivial. We also computed 2-, 3-, and 4-clubs. A maximum of 10 000 seconds was allowed to solve any individual instance.

Table 1 summarizes our computational results for finding maximum 2-clubs on the generated graphs. The first column gives the number of vertices while the second one represents the average density. Note that within an average density class, different density ranges are generated by varying the parameters a and b. For example, for an average density of 0.2 we use pairs of values $a=0.2$; $b=0.2$, $a=0.15$; $b=0.25$, and $a=0.1$; $b=0.3$. Since for every parameter setting we generate 10 instances, each line in the table represents an average over 30 instances. In the third and fourth columns we compare the results of our algorithm with those of our earlier heuristic. The fifth and sixth columns provide the computation time in seconds with and without the use of our stable set upper bound, whereas the last two columns give the number of terminal nodes reached in the search tree, again with and without the bound.

A first observation stemming from these results is that the heuristic yields near-optimal solutions for high density values but leaves a substantial gap otherwise. Another observation is that for $k=2$, the hardest instances seem to occur around a density of 0.15, already known to be difficult for the stable set problem. For graphs involving 150 and 200 vertices, we were not able to consistently solve instances of that density within the prescribed time limit. The number of terminal nodes indicates that the upper bound is very effective in pruning the search tree and, despite its inherent cost, it also cuts down on the overall computation time.

Table 2 compiles results for the 2-, 3-, and 4-club problem on 100-vertex graphs. Different graph densities had to be used in order to maintain an interesting range of instances. Again an entry corresponds to an average over 30 instances. We observe that the computation time does not appear to increase with k.

Table 2
k-Clubs, with $n=100$ and stable set upper bound

k	Average density	Largest k-club	Computation time
2	0.5	12.5	77.7
	0.1	20.1	51.6
	0.15	32.1	931.3
	0.2	68.2	181.2
	0.25	94.1	0.6
	0.3	99.5	0.3
3	0.25	14.1	103.5
	0.5	28.3	56.8
	0.75	63.8	60.8
	0.1	93.3	1.0
	0.125	99.4	0.3
	0.15	99.9	0.3
4	0.013	12.9	13.4
	0.025	25.7	88.3
	0.038	47.6	44.3
	0.05	76.2	11.1
	0.063	94.7	1.1
	0.075	99.2	0.4

Table 3
Size of stable set subproblems solved

n	k											
	2				3				4			
	Avg. size	Avg. density	Max. size	Max. density	Avg. size	Avg. density	Max. size	Max. density	Avg. size	Avg. density	Max. size	Max. density
50	9.9	0.35	38	0.13								
100	16.1	0.31	80	0.11	11.9	0.43	74	0.06	10.7	0.36	68	0.09
150	18.2	0.41	98	0.18								
200	14.8	0.36	62	0.43								

Table 3 gives an idea of the size of the stable set subproblems solved at each node of the search tree to yield an upper bound. For each combination of n and k considered, it provides an average size and density of the subproblems over all relevant instances. It also gives the size of the largest subproblem solved, with its density. The average subproblem is seen to be rather small and so quite tractable.

6. Conclusion

We have analyzed, formulated and solved the MkCP on undirected graphs. This problem arises naturally in the determination of highly interconnected structures in networks. Applications of this problem are encountered in several areas of organization theory and political science. We have proved that the MkCP is NP-hard. We have developed a branch-and-bound procedure that maintains shortest path information and makes use of upper bounds based on the computation of maximal stable sets in a graph. Using this technique, instances of up to 200 vertices were solved to optimality. Problem difficulty is directly dependent of the graph density. Thus, instances defined on very dense graphs can be solved within insignificant amounts of time. For $k = 2$, the most difficult cases appear to be those for which the density is around 0.15.

Acknowledgements

This work was partially supported by the Canadian Natural Sciences and Engineering Research Council under grants OGP0039359, OGP0039682, and OGP0005052. This support is gratefully acknowledged. Thanks are also due to one referee for his valuable comments.

References

[1] R.D. Alba, A graph-theoretic definition of a sociometric clique, Journal of Mathematical Sociology 3 (1973) 113–126.
[2] R.D. Alba, Taking stock of network analysis: A decade's result, Research in the Sociology of Organizations 1 (1982) 39–74.
[3] W.E. Baker, Market networks and corporate behavior, American Journal of Mathematical Sociology 12 (1986) 191–223.
[4] J.-M. Bourjolly, G. Laporte, G. Pesant, Heuristics for finding k-clubs in an undirected graph, Computers & Operations Research 27 (2000) 559–569.
[5] C. Bron, J. Kerbosch, Finding all cliques of an undirected graph, Communications of the ACM 16 (1973) 575–577.
[6] R.W. Floyd, Algorithm 97: Shortest path, Communications of the ACM 5 (1965) 345.
[7] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, San Francisco, 1979.
[8] M. Gendreau, P. Soriano, L. Salvail, Solving the maximum clique problem using a tabu search approach, Annals of Operations Research 41 (1993) 385–403.
[9] D.S. Johnson, M.A. Trick (Eds.), Cliques, Coloring and Satisfiability, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 26, American Mathematical Society, Providence, RI, 1996.
[10] B. Mintz, M. Schwartz, Interlocking directorates and interest group formation, American Sociological Review 46 (1981) 851–869.
[11] R.J. Mokken, Cliques, clubs and clans, Quality and Quantity 13 (1979) 161–173.
[12] A. Ronert, A dynamization of the all pairs least cost path problem, in: Proceedings of STACS'85, Lecture Notes in Computer Science, vol. 182, Springer, Berlin, 1985, pp. 279–286.

[13] D. Snyder, E.L. Kick, Structural position in the world system and economic growth, 1955–1970: A multiple-network analysis of transactional interactions, American Journal of Sociology 84 (1979) 1096–1126.

[14] UCINET, Release 3.0, Mathematical Social Science Group, Social School of Social Sciences, University of California at Irvine.