Int J Data Sci Anal (2016) 2:145–158
DOI 10.1007/s41060-016-0022-1

REGULAR PAPER

A fast and complete algorithm for enumerating pseudo-cliques in large graphs

Hongjie Zhai[1] · **Makoto Haraguchi**[1] · **Yoshiaki Okubo**[1] · **Etsuji Tomita**[2]

Received: 18 July 2016 / Accepted: 22 August 2016 / Published online: 16 September 2016

Abstract This paper discusses a complete and efficient algorithm for enumerating densely connected k-Plexes in networks. A k-Plex is a kind of pseudo-clique which imposes a disconnection upper bound (DUB) involving a parameter k for each constituent vertex. However, because the parameter value is usually set independently of the sizes of the targeted pseudo-cliques, we often obtain k-Plexes that are not densely connected. To overcome this drawback, we introduce another constraint, the connection lower bound (CLB), which involves a parameter j. Using the CLB, we can enjoy monotonic j-core operations and can design an efficient depth-first algorithm, which can exclude both search branches that generate duplicate search nodes and "hopeless" nodes that yield no targets satisfying both DUB and CLB. Our experimental results show that the algorithm can be a useful tool for detecting densely connected pseudo-cliques in large networks, including an example with over 800, 000 vertices.

Keywords Pseudo-clique · k-Plex · j-Core · j-Cored maximal connected k-Plex · Enumeration algorithm

This paper is an extended version of the PAKDD 2016 Long Presentation Paper, "A Fast and Complete Enumeration of Pseudo-Cliques for Large Graphs" [30].

✉ Hongjie Zhai
zhaihj@kb.ist.hokudai.ac.jp

✉ Makoto Haraguchi
mh@ist.hokudai.ac.jp

[1] Graduate School of Information Science and Technology, Hokkaido University, N-14 W-9, 060-0814 Sapporo, Japan

[2] The Advanced Algorithms Research Laboratory, The University of Electro-Communications, Chofugaoka 1-5-1, 182-8585 Chofu, Tokyo, Japan

1 Introduction

Detecting communities in a network is an important aspect of social network analysis [22]. Cliques are typical vertex sets that can be understood as potential communities [20]. Moreover, the class of cliques has an antimonotonicity property that is helpful in designing an efficient clique enumerator. In a real-world network, however, the clique model is too restrictive to capture various communities because it is rare for actual communities to appear as cliques. This has motivated the study of clique relaxation models, with various "pseudo-clique" models having been proposed [20].

An alternative approach to community detection uses graph clustering or partitioning methods, which has been shown to work well (e.g., [14,17]). However, if our aim is to identify small communities, we might prefer to use the (pseudo-) clique detection approach because the alternative approach usually supposes small numbers of relatively large clusters. Our smaller targets would tend to be merged and absorbed into these larger clusters.

In a density-based model of pseudo-cliques (e.g., [1]), indices for measuring the density of vertex sets are presented. Because this class of pseudo-cliques does not satisfy the antimonotonicity property of the pure clique model, efficient but heuristic detectors have been proposed for search purposes [1]. As a result, some possibly valuable vertex sets might be missed. Moreover, even if we had a complete detector, the huge number of pseudo-cliques might prevent the processing of large-scale networks. In spite of these difficulties, it is important to develop an efficient and complete pseudo-clique enumerator for handling large networks because such an enumerator would be useful not only for community discovery, but also for analyzing the nature of large networks in terms of (pseudo-) cliques statistics [25].

Other clique-relaxation models, a distance-based model, k-clique [13], and diameter-based models, k-club and k-clan [2,15], have been proposed. Here, the parameter k controls admissible distances among vertices. As discussed in [27], if we allow long distances, large dense subgraphs appear that are almost cliques even when their subgraphs with respect to the original edge connections are not dense.

On the other hand, a k-Plex model [24] considers the density of the original connections by setting an upper bound for the number of missing edges among the vertices. The class of k-Plexes has an antimonotonicity property that can aid the design of a simple bottom-up enumerator [6,28]. For this reason, we consider k-Plexes in this paper by introducing new constraints to mitigate their weaknesses, which we now discuss.

A vertex set is called a k-Plex if, for each included vertex x, the number of vertices not adjacent to x is at most k. The *DUB* is therefore specified by the parameter k. Note that a k-Plex could have disconnected parts. However, such a set would be unlikely to represent a community and we exclude such sets from further consideration.

For a very small k, a connected k-Plex will be dense if its size is relatively larger than k. However, as the size of the densely connected vertex set increases, the number of disconnected vertices, k per each vertex, shall be nonsmall, depending on the density requirement. In other words, we think that a constraint that considers the sizes of targeted pseudo-cliques is important. For a k-Plex of size n, each vertex has at least $n-k$ adjacent vertices, where $n-k$ must be a particular value, provided the vertex set is to be densely connected. We therefore introduce another constraint, involving a parameter j, that designates a *CLB*. We can then aim to enumerate all maximal connected k-Plexes that meet the CLB constraint.

A naïve strategy, such as computing maximal k-Plexes and then checking the CLB constraint, does not work well for cases involving nonsmall k. This is because every vertex set with size no larger than k is trivially a k-Plex, which forces the examination of an exponentially increasing number of such sets. The key to solving this problem lies in another fact that, if a connected k-Plex X can be extendable to a maximal connected k-Plex under the CLB constraint, X is involved in a "core" of X together with candidates that are potential vertices to be added to X, where the term "core" means the largest subset of vertices with at least j adjacent vertices in the subset [5].

In this paper, j is taken to be fixed beforehand depending on the size of the targeted vertex set. The monotonicity of the core operation suits a standard k-Plex enumerator, which uses the antimonotonicity of the k-Plexness. Based on these monotonicities, we can design an efficient complete depth-first algorithm, which can exclude numerous hopeless k-Plexes that cannot be maximally extended to meet our requirements. This significantly improves the performance of the k-Plex enumerator. In our experiments, we focus on comparing our algorithm with a state-of-the-art maximal k-Plex enumerator that has been proposed very recently [6], in terms of both computational performance and the quality of the solution k-Plexes as pseudo-cliques. For synthetic and large real-world networks, including a Web graph with over 800,000 vertices, the results demonstrate that combining the CLB constraint with our pruning mechanisms is quite effective. Therefore, the proposed algorithm works very well as a practical tool for detecting dense pseudo-cliques.

As an alternative to our formalization of densely connected pseudo-cliques as maximal k-Plexes satisfying the CLB constraint, one might consider maximal vertex sets, which can form k-Plexes satisfying the constraint, where the class of those maximal sets completely subsumes our solutions. This would seem preferable to our formalization because we could miss possibly many densely connected vertex sets. However, our experiments show that the number of the sets actually missed is quite small. Therefore, we can argue that our model of pseudo-cliques is both reasonable and practical.

Our experiments also demonstrate that the proposed method is particularly effective in detecting densely connected pseudo-cliques of relatively small or medium sizes, which are usually outside the target of the standard method for graph clustering or partitioning. In general, those nonlarge pseudo-cliques are often subsumed in large communities and hence would be invisible. However, such a nonlarge community can be considered valuable and informative because it could be a "seed" which potentially grows into a novel community. For example, in a friendship network of SNS users, such a seed absorbed by a major community could be the origin of a new trend. In marketing research, therefore, it would be worth detecting potential seed communities for trend forecasting. The proposed algorithm can play the principal role in accomplishing the important task.

The notion of densely connected pseudo-cliques is fundamental and useful not only for social network analysis, but also for other application domains such as biological systems [9] and financial markets [16]. The method proposed in this paper is a general framework because we do not need to make assumptions specific to some particular domain. Therefore, we expect our algorithm to be applicable to and effective in a variety of application domains in which graph-based formalizations are natural and reasonable.

The remainder of this paper is organized as follows. In Sect. 2, the basic terminology and notations used throughout this paper are introduced. Section 3 presents the notion of maximal connected k-Plexes (k-MPCs). We then define our target sets, namely j-cored maximal connected k-Plexes ((j,k)-MPCs), and show their theoretical properties in Sect. 4. Effective search control rules for the efficient com-

putation of (j, k)-MPCs are discussed in Sect. 5. Section 6 presents our algorithm for enumerating (j, k)-MPCs using the control rules. Our experimental results for synthetic and real-world networks are presented in Sect. 7. Finally, we summarize the paper in Sect. 8 and conclude with our plans for future work.

2 Preliminaries and notations

An undirected graph G is denoted by $G = (V, \Gamma)$, where $V = \{v_1, \ldots, v_{|V|}\}$ is a set of vertices and $\Gamma(v_n) = \{v_m \in V \mid v_n, v_m \text{ are adjacent}\}$. $\Gamma(v_n)$ is assumed to not include v_n itself, i.e., $v_n \notin \Gamma(v_n)$. An ordering $\prec$ over V is defined by $v_i \prec v_j$ and v_i is said to be younger than v_j if and only if $i < j$. (If the identifier i of v_i does not need to be specified, we will abbreviate the notation v_i as v.) For a vertex set $X \subseteq V$, $G[X]$ is a subgraph of G induced by X. For a vertex $x \in X$, $\Gamma_X(x)$ denotes $\Gamma(x) \cap X$. $|\Gamma_X(x)|$ is often referred to as $deg_X(x)$.

A vertex set X is called a k-Plex if $|X - \Gamma_X(x)| \leq k$ for any $x \in X$. It is easy to see that, for a k-Plex Y, any subset X of Y is also a k-Plex. Therefore, the class of k-Plexes has an antimonotonicity property, which is often used to design maximal k-Plex enumerators. A vertex $y \notin X$ is called a k-Plex candidate if Xy remains a k-Plex, where Xy is an abbreviation of $X \cup \{y\}$. $Cand(X)$ denotes the set of all k-Plex candidates for X. In this paper, we are particularly interested in connected k-Plexes (c-k-Plexes).

For a vertex set X and a vertex y, the distance between X and y, denoted $dist(X, y)$, is given by the minimum length of paths in G from X to y, where $dist(X, x) = 0$ whenever $x \in X$. $dist(X, y) = \infty$ only when y is not reachable from X. $D_n(X)$ is defined as $\{y \in V \mid dist(X, y) = n\}$. $K_1(X) = D_1(X) \cap Cand(X)$ is the set of k-Plex candidates directly connected to X and plays a key role in the discussion of c-k-Plexes. It is called a K_1-candidate set at X.

These basic notations are summarized in Table 1.

3 Maximal connected k-Plex

A maximal c-k-Plex (k-MPC) is a c-k-Plex that is maximal among c-k-Plexes. It is clear that a c-k-Plex X is extendable to its super c-k-Plex if and only if $K_1(X) \neq \emptyset$. Therefore, the extension can be made by adding K_1-candidates at X to X.

A formation X^f of c-k-Plex X with respect to an indexing function f is a sequence of vertices $v_{f(i)}$, $(v_{f(1)}, v_{f(2)}, \ldots, v_{f(|X|)})$, where $f(i)$ is the identifier of the i-th vertex added to form $X = \{v_{f(1)}, \ldots, v_{f(|X|)}\}$ as sets, and each of the intermediate $X_i^f = \{v_{f(1)}, \ldots, v_{f(i)}\}$ must be a c-k-Plex. That is,

$$v_{f(i+1)} \in K_1\left(X_i^f\right). \tag{1}$$

The sequence $(X_1^f, \ldots, X_{|X|}^f)$ is also called a formation of X. For any k-MPC Z, its formation ends at $Z = Z_{|Z|}^f$ with $K_1(Z) = \emptyset$.

We have multiple formations Z^f for a k-MPC Z, depending on the order of the vertex addition. For an arbitrary initial vertex $v_{n_1} \in Z$, we can repeat the addition of $v_{n_i} \in Z$ such that v_{n_i} is adjacent to some vertex added prior to v_{n_i}. Any intermediate $X_i (\subseteq Z)$ is then a c-k-Plex because the monotonicity of k-Plexes and Z^f with the index f such that $f(i) = n_i$ gives a formation of Z. In Sect. 5, we introduce some control rules to enable us to disregard useless formations, given the restriction that formations must be those for k-MPCs whose density is guaranteed by another parameter j.

Table 1 Basic notations

Notation	Definition	Meaning/interpretation
G	(V, Γ), where $\Gamma \subseteq (V \times V) \setminus \{(x, x) \mid x \in V\}$	An undirected graph with a set V of vertices and an adjacency relation Γ on V
$\Gamma(x)$	$\{v \in V \mid (x, v) \in \Gamma\}$	The set of vertices in V adjacent to a vertex x
$G[X]$	$G[X] = (X, \Gamma \cap (X \times X))$	The subgraph of G induced by a set of vertices X
$\Gamma_X(x)$	$\Gamma(x) \cap X$	The set of vertices adjacent to a vertex x in $G[X]$
$deg_X(x)$	$\lvert\Gamma_X(x)\rvert$	The degree of a vertex x in $G[X]$
$Cand(X)$	$\{y \in V \mid Xy \text{ is a } k\text{-Plex}\}$	The set of k-Plex candidates of a vertex set X
$dist(X, y)$	the minimum length of paths in G from X to y	The distance between a vertex set X and a vertex y
$D_n(X)$	$\{y \in V \mid dist(X, y) = n\}$	The set of vertices whose distance from a vertex set X is exactly n
$K_1(X)$	$D_1(X) \cap Cand(X)$	The set of k-Plex candidates adjacent to a vertex set X

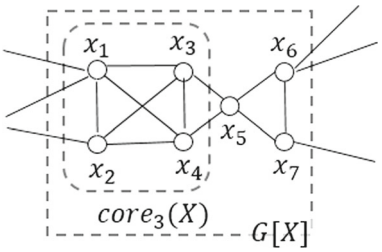


Fig. 1 Example of $core_j(X)$ for the $j = 3$ case

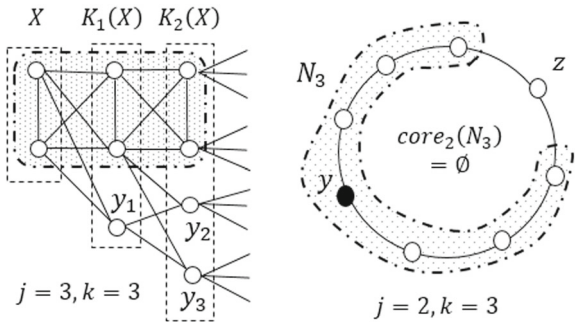


Fig. 2 Examples showing how the j-core operation is useful

4 j-Cored k-MPC

In this section, we define a particular class of k-MPCs to be our targets, which are dense in the sense that each vertex has at least j adjacent vertices. We then discuss how to enumerate those targets completely and efficiently.

Given a graph $G = (V, \Gamma)$, a vertex set X is said to be j-cored if $deg_X(x) \geq j$ for any $x \in X$. The largest j-cored set, denoted by $core_j(V)$, is called the j-core of G [5].[1] For a vertex set X, $core_j(X)$ is the j-core of the induced subgraph $G[X]$. We can observe a monotonicity in j-core operations, as follows.

Fact (Monotonicity of j-cores): For vertex sets X_1, X_2 of V, $core_j(X_1) \subseteq core_j(X_2)$ holds whenever $X_1 \subseteq X_2$.

Verification: It is easy to see that $core_j(X_1) \subseteq X_1 \subseteq X_2$. For any $x_1 \in core_j(X_1)$, because $deg_{X_1}(x_1) \geq j$, we have $deg_{X_2}(x_1) \geq deg_{X_1}(x_1) \geq j$. This means that $x_1 \in core_j(X_2)$ for any $x_1 \in core_j(X_1)$. Therefore, $core_j(X_1) \subseteq core_j(X_2)$. □

The construction of $core_j(X)$ for a vertex set X is simple. Essentially, $core_j(X)$ can be obtained by iteratively removing vertices of degree less than j from X. Because removing some vertices usually decreases the degree of other vertices, we can iterate the removal process until no more vertices can be removed.

This j-core process is illustrated in Fig. 1. Any connection toward outside of X is ignored as we consider the subgraph $G[X]$. For this X, both x_6 and x_7 can be removed because $deg_X(x_6) = deg_X(x_7) = 2 < j = 3$. After this removal, $deg_{X\setminus\{x_6,x_7\}}(x_5)$ becomes 2. Therefore, x_5 can now be removed. No further removal can be made using the lower-bound constraint in this case because all remaining vertices x_1, x_2, x_3 and x_4 retain at least j adjacent vertices after the previous removals.

Our target can now be defined as a k-MPC which is j-cored, called a j-cored k-MPC ((j, k)-MPC). We now consider an efficient algorithm for enumerating every (j, k)-MPC.

[1] The notion of j-core was originally defined in [23]. In this paper, we use the definition and construction method for j-core given in [5].

Unlike k-Plexes, the class of j-cored vertex sets does not have the antimonotonicity property. Despite this, we can give a sufficient condition for c-k-Plexes to become j-cored. We can then use the condition in reverse to reject hopeless c-k-Plexes from appearing in formations.

Fact (Hopeful c-k-Plex): Let X be a c-k-Plex extendable to some (j, k)-MPC by adding vertices to X. Then,

$$X \subseteq U(X), \quad \text{where} \tag{2}$$

$$U(X) = core_j(X \cup Cand(X)). \tag{3}$$

Verification: As $Xz \subseteq Z$ for any $z \in Z - X$ and Z is a k-Plex, Xz is also a k-Plex because of the antimonotonicity of k-Plexes. Therefore, $Z \subseteq X \cup Cand(X)$. Z is j-cored, giving $X \subseteq Z \subseteq core_j(X \cup Cand(X))$. □

An X satisfying (2) is said to be "hopeful." Conversely, "hopeless" sets are those satisfying the negation of (2).

$$\textbf{(Hopelessness)} \quad X - U(X) \neq \emptyset \tag{4}$$

The definition (3) of $U(X)$ that takes k-Plex candidates $Cand(X)$ before the j-core operation is weak, particularly for small X of size less than k. In this case, a large number of vertices appear as k-Plex candidates independently of the length of the paths connecting X and the candidates. As X is extended to include a vertex y further from X, the shortest path connecting X and y in the extension will become longer. Consequently, such an extended Y will not be k-Plex, because there will be many vertices not adjacent to y on the path. If we are targeting connected k-Plexes including X, the distance limit becomes the key to defining $U_1(X) = U(X) \cap D_1(X)$ as a set of potential candidates for (j, k)-MPCs.

Figure 2 shows that the j-core operation is useful under such a distance limit and also suggests that the construction of $U(X)$ depends on the sizes of X. In the right-hand figure, for a singleton Y of the black circle y, $Y \cup Cand(Y)$ is the whole vertex set that is j-cored, where $j = 2$. Therefore, nothing is removed by performing the j-core operation for $Y \cup Cand(Y)$. Nevertheless, note that the vertex z cannot be consistent with Y in making a c-k-Plex, because

$dist(Y, z) = 4 > k = 3$. More precisely, on the paths connecting Y and z, there are four vertices not adjacent to z, including z itself. Therefore, there can be no c-3-Plex involving Yz. For this reason, we should remove z before we test if Y is extendable to j-cored c-k-Plexes. Let N_3 be the set of remaining candidates together with Y. Then $core_j(N_3) = \emptyset$ asserts that there exists no j-cored c-k-Plex containing Y.

Conversely, the left-hand figure shows a case of X with $|X| = 2$. Similarly, when $k = 3$, vertices with $dist(X, z) > 3$ cannot be members of a c-3-Plex that includes X. After excluding vertices that violate the distance limit, we perform a 3-core operation. We first remove y_2 and y_3, then y_1. The remaining part, surrounded by the dot–dash line and shaded, is 3-cored and is a maximal 3-Plex in this case.

4.1 Small c-k-Plex

A c-k-Plex X is said to be small if $|X| < k$. Note here that any small connected X is a c-k-Plex. In this case, therefore, $Cand(X)$ is given as $V - X$ which is usually large. To omit useless candidates, we check whether there exists some c-k-Plex Z such that $Xy \subseteq Z$ for a small c-k-Plex X and $y \notin X$. Because $Z = Xy$ is a trivial positive result for $dist(X, y) = 1$, we analyze the cases where $\ell = dist(X, y) \geq 2$. Consider the shortest path $p = (y_0, y_1, \ldots, y_{\ell'} = y)$ from X to y in $G[Z]$, where $y_0 \in X$ and $y_1, \ldots, y_{\ell'} \in Z$. Then, $\ell \leq \ell'$, and y is not adjacent to any of $X, y_1, \ldots, y_{\ell'-2}$ and $y_{\ell'} = y$ itself. Because Z is a k-Plex, it must be $|X| + (\ell' - 2) + 1 \leq k$. Therefore, we have $\ell \leq \ell' \leq k - |X| + 1$. In other words, y with $dist(X, y) > k - |X| + 1$ can never be a member of any c-k-Plex including X. Therefore, for small X, the following definition for $U(X)$ is more accurate.

$$U(X) = core_j(X \cup K(X)),$$
$$K(X) = \bigcup_{i=1}^{k-|X|+1} D_i(X),$$

where $k - |X| + 1$ is the distance limit.

Because the distance limit decreases as X becomes larger, $K(X)$ is monotonically decreasing.

4.2 Medium c-k-Plex

We say that a c-k-Plex X is medium if $k \leq |X| < j + k$. For a medium c-k-Plex$X$, any vertex whose distance from X is greater than 1 can never be a k-Plex candidate because it has no connection with at least k vertices in X. Therefore, $Cand(X) = K_1(X)$, and $U(X)$ can be given exactly as

$$U(X) = core_j(X \cup K_1(X)).$$

4.3 Large c-k-Plex

We say that c-k-Plex X is large if $|X| \geq j + k$. Any large X is j-cored because $j \leq |\Gamma_X(x)|$ is derived from $j + k \leq |X|$ and $|X - \Gamma_X(x)| = |X| - |\Gamma_X(x)| \leq k$. We need not perform the j-core operation, which gives $U(X) = X \cap Cand(X)$. Therefore, we have the following rule for updating U.

$$U(Xu) = U(X) \cap Cand(Xu), \quad \text{where} \quad u \in U_1(X).$$

The rule for large X is the same as that for k-Plexes.

4.4 Revised formations for (j, k)-MPCs

Using these $U(X)$ definitions that depend on $|X|$, a formation $Z^f = (v_{f(1)}, \ldots, v_{f(|V|)})$ of (j, k)-MPC Z satisfies

$$U\left(Z_{i+1}^f = Z_i^f v_{f(i+1)}\right) \subseteq U\left(Z_i^f\right) \quad \text{and} \tag{5}$$
$$v_{f(i+1)} \in U_1\left(Z_i^f\right), \quad \text{where } U_1 \text{depends on} \quad |Z_i^f|, \tag{6}$$
$$U_1\left(Z = Z_{|Z|}^f\right) = \emptyset. \tag{7}$$

Condition (6) is stronger than (1) because $U_1(Z_i^f) \subseteq K_1(Z_i^f)$. The Condition (7) holds because $U_1(Z) \subseteq K_1(Z)$ and $K_1(Z) = \emptyset$ by the property of (j, k)-MPC as a k-MPC.

5 Right and left search control rules

If a c-k-Plex Z is more densely connected, the number of possible formations of Z will be greater. To achieve efficient computation by avoiding useless and duplicated formations, we can deploy two search control rules, namely "right" and "left" rules.

Using a right candidate control (RCC), we can exclude many useless formations. A formation $Z^f = (v_{f(1)}, \ldots, v_{f(|Z|)})$ can be obtained by extending an intermediate Z_i^f with a vertex in $U_1(Z_i^f)$. In other words, for complete enumeration of formations, we have to examine every vertex in $U_1(Z_i^f)$ to extend Z_i^f. Some of them, however, will result in nonmaximal k-Plexes. Extending an idea discussed in [8,26], we can identify the candidate set $R(Z_i^f)$ using the property that adding member vertices in $R(Z_i^f)$ only to the present Z_i^f will never achieve a maximal k-Plex. Therefore, the set of vertices actually used to extend Z_i^f is given as $NR(Z_i^f) = U_1(Z_i^f) - R(Z_i^f)$ and called the nonright candidates. The RCC used here can be regarded as an extended version of the RCCs in [28] in the sense that it is more widely applicable to nonsmall Z_i^f than that in [28]. Although we skip the details because of space limitations, interested readers can refer to [18].

In addition to using RCC, we can deploy a left candidate control (LCC) enabling just one formation for each (j,k)-MPC to be composed. The LCC is in some sense a standard technique for set enumeration [21] and is similar to what is discussed in [26]. In essence, if we extend an intermediate Z_i^f, we do not need to consider any vertex y such that $y \prec v_{f(i)}$, called a left candidate because any formation obtained by extending Z_i^f with such a y can be composed by extending another intermediate formation with some left candidate $\ell(\neq y)$. Therefore, the set of vertices we actually use to extend Z_i^f is given by $NR(Z_i^f) - L(Z_i^f)$, where $L(Z_i^f)$ is the set of left candidates. Although we do not go into detail, the effects of the control rules in forming the search tree are illustrated in the next section.

6 Algorithm for (j,k)-MPCs

In Fig. 3, we present an algorithm for making formations for (j,k)-MPCs. At each intermediate c-k-Plex $X = Z_i^c$, only the nonleft and nonright candidates x at X are added to extend the partial formations, provided that X together with x is hopeful. Therefore, any hopeless cases are omitted immediately. Although the whole control structure appears simple, it should be noted that the construction of $U_1(X)$ and $R(X)$ depends on the size $|X|$.

The algorithm is written using recursive calls to procedure $Expand$. This will realize a depth-first search for (j,k)-MPCs.

In Fig. 4, we show the process of invoking $Expand$ in the form of search tree with c-k-Plexes as its nodes. In the search

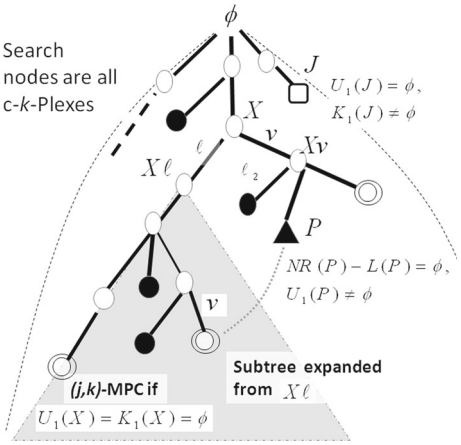


Fig. 4 Search tree

```
procedure MAIN(G^input):
  [Input] G^input = (V^input, Γ^input): an input graph.
  [Output] All maximal j-cored c-k-Plexes including (j,k)-MPCs
  begin
    for each connected component C of core_j(V^input)
      Let G = (C, Γ = Γ_C^input);
      ũ = arg max_{u∈C} deg(u);
      NR = C − Γ(ũ); L = ∅;
      X = ∅; U = NR;
      Expand(X, NR, L, U);
    endfor
  end

procedure EXPAND(X, NR, L, U):
      // X: c-k-Plex, NR = NR(X), L = L(X); U = U(X) (depending on |X|)
  begin
    U_1 = U ∩ D_1(X); //for non-small case, U_1 = U
    if (U_1 = ∅) then
      print X; return; // maximal j-cored c-k-Plex including (j,k)-MPC
    endif
    for each v ∈ NR − L // the order accords to ≺
      Xnew = Xv;
      Unew = U(Xnew);
      if Xnew − Unew ≠ ∅ then continue; // Xnew is hopeless
      NRnew = NR(Xnew); Lnew = (L(X) ∪ {v}) ∩ Unew;
      Expand(Xnew, NRnew, Lnew, Unew);
    endfor
  end
```

Fig. 3 Enumeration algorithm for (j,k)-MPCs

tree, a path from the root, the empty set $\emptyset$, to a leaf just corresponds to the formation of a c-k-Plex. Dark circles indicate hopeless c-k-Plexes from which no branch is expanded. The double circle is a (j, k)-MPC. Each of the remaining single circles has a chance of being extended by choosing some of $NR(X) - L(X) \neq \emptyset$. The dark triangle is a c-k-Plex P such that every nonright candidate is left. This means that the (j, k)-MPC Z obtained by extending X is generated by another branch with some left candidates ℓ along its path. In other words, using some initial segment X of Z and its nonleft $\ell \in NR(X)$, Z appears in the subtree rooted by $X\ell$. Finally, the white square J is a j-cored c-k-Plex, which is not a k-MPC. It is maximal in the sense that there exists no j-cored c-k-Plexes that include J. Although it is straightforward to exclude such a J with the condition $K_1(X) \neq \emptyset$, we allow the output of J in addition to all possible (j, k)-MPCs.

We should also note that every (j, k)-MPC is a subset of the global j-core $core_j(V)$. In addition, because $core_j(V)$ comprises several connected components and our targets must be connected, we run the algorithm for each connected component C of $core_j(V)$.

In setting parameters k and j, we assume a preferred size range $[n_1, n_2]$ and a density parameter τ. Then the CLB will be $j = \tau \cdot n_1$, and the DUB will be $k = (1 - \tau) \cdot n_2$.

7 Experimental results

In this section, we present our experimental results. The proposed system, referred to as JKMPC, was coded in Java and executed on a PC with an Intel® Core™-i7 (1.7 GHz) processor and 8 GB memory. For several datasets, we observed the computation times and the quality of the solutions as representing pseudo-cliques.

To investigate the practical efficiency of JKMPC, we compared it with MaxKplexEnum, a state-of-the-art maximal connected k-Plex enumerator [6].[2] In Ref. [6], it was shown that MaxKplexEnum could enumerate maximal (connected) k-Plexes much faster than Pemp, a pioneer system for enumerating maximal k-Plexes [28].[3]

7.1 Datasets

For our experiments, we prepared a collection of both synthetic and real networks.

As a synthetic dataset, we created a scale-free network, referred to as BA, based on the *Barabási-Albert model* [3]. This dataset involved the parameters $N = 10,000$ and $m = 10$, where N was the number of nodes and m the number of edges to be attached from each new node to existing nodes. We also created a synthetic small-world network, referred to as WS, based on the *Watts–Strogatz model* [27]. This dataset involved the parameters $N = 50,000$, $K = 20$ and $p = 0.1$, where N was the number of nodes, K the initial degree of each node and p the probability of rewriting edges.

For the real datasets, we prepared four benchmark networks, called GrQc [10,11], DAYS [4,7], DBLP [11,29], and Google [11,12]. GrQc was a network representing scientific collaborations among authors of submitted arXiv papers under the category of General Relativity and Quantum Cosmology. The researchers were the nodes, with co-authors of a paper being connected by edges to each other. DAYS was a word co-occurrence network created from a collection of Reuters news articles. If a pair of words co-occurred in a certain number of sentences, they would be connected by an edge. DBLP was also a collaboration network constructed from the DBLP computer science bibliography.[4] Authors as nodes were connected if they had published a paper together in registered journals or conferences. Google was a Web graph comprising Web pages and their hyperlinks, and was released as a part of a Google programming contest in 2002.[5]

7.1.1 Characteristics

Characteristics and degree distributions of those networks are presented in Fig. 5. Note here that since both BA and WS were artificially generated in systematic ways, they had characteristic properties which are rather rare in the real world. Concretely speaking, although every vertex in BA had at least 10 edges, where the value was determined by the parameter m, the minimum degree in most real networks was just a small integer, say 1. This implies that the average degree of BA would be larger than those of most real networks. Moreover, the vertex degrees in WS were distributed around 20 determined by the parameter K as the peak. In other words, the degree distribution did not follow a power law. That is, WS was not scale-free which is a typical property of most real networks, whereas it was a small-world network. From these observations, we tried BA and WS to examine whether our algorithm can work well even for unrealistic networks.

Because the others were all real networks, their degree distributions seemed to almost follow power laws as shown in Fig. 5. However, we found several characteristic differences among them. The density of GrQc was clearly higher than those of the others, whereas the cluster coefficient was not characteristic. For DAYS, because the number of edges was about a half of the number of vertices, the average degree

[2] Its Python-based source code was kindly provided by the authors of [6].

[3] As discussed in [6,20], Pemp was the only system available for the task until recently.

[4] http://dblp.org.

[5] http://www.google.com/programming-contest.

Name	# of Nodes	# of Edges	Density	Degree			Cluster Coeff.
				max.	min.	ave.	
BA	10,000	99,900	0.001998	531	10	19.98	0.01
WS	50,000	500,000	0.000400	26	14	20.00	0.52
GrQc	5,242	14,496	0.001054	81	0	5.53	0.53
DAYS	13,332	5,616	0.000063	252	0	0.84	0.04
DBLP	317,080	1,049,866	0.000021	341	1	6.62	0.63
Google	875,713	4,322,051	0.000011	6,332	1	9.87	0.51

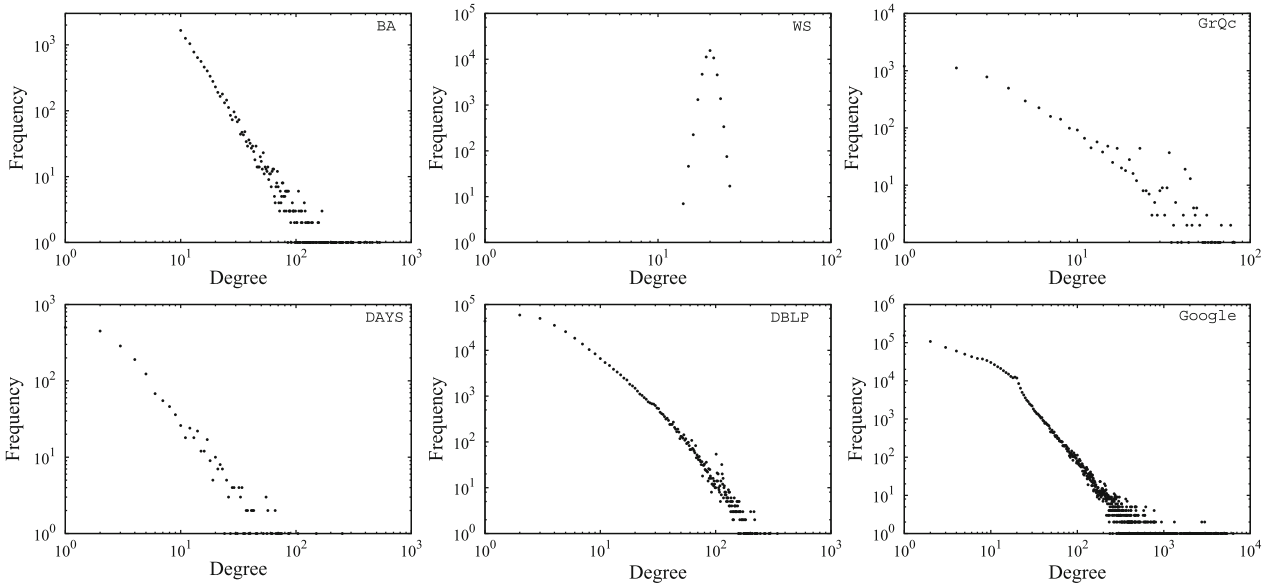


Fig. 5 Scales and degree distributions for networks

and the cluster coefficient were much smaller than those of the others. The cluster coefficient of DBLP was clearly larger than those of the others, whereas the maximum degree was small for the number of vertices. Because Google was the largest real network in the experiments, the degree of each vertex was in a fairly wide range from 1 to over 6000.

From the above, the networks used in the experiments had their own characteristic features. Using the networks, therefore, we could examine the proposed algorithm in various cases.

7.1.2 Preprocessing for real networks

In our preparation process for the real networks, DAYS and Google, we conducted the following preprocessing.

The word co-occurrence network, DAYS, was originally provided as a network with 13, 332 vertices (words) and 243, 447 edges, where each edge connected a pair of words co-occurred in the same text sentence. Moreover, the weight of each edge was defined as the co-occurrence frequency. For the original network, we removed every edge that assigned the weight less than 4, to exclude insignificant co-occurrences of too small frequencies. The resultant network with 5616 edges was actually used for the experiments.

The Web graph, Google, was originally available as a directed network with 875, 713 Web pages as vertices and 5, 105, 039 hyperlinks as directed edges. To obtain an undirected network, we neglected the directions of the hyperlinks and identified any edges connecting the same pair of Web pages with a single undirected edge. For the experiments, we actually used the resultant network with 4, 322, 051 edges as Google.

7.2 Computational performance

We know of no existing algorithm devoted to enumerating all (j, k)-MPCs. This makes it difficult to compare JKMPC directly with other systems. However, our target k-MPCs can be obtained by any maximal k-Plex enumerator with a j-coreness check. Therefore, we measured the computational performance of a state-of-the-art enumerator, MaxKplexEnum [6], and of our JKMPC to investigate the practical efficiency of our system. Given a network, MaxKplexEnum runs in FPT (fixed-parameter-tractable) incremental polynomial time with respect to k and can enumerate a designated number N of maximal k-Plexes. That is, to completely obtain our target (j, k)-MPCs using MaxKplexEnum, there has to be a sufficiently large value for N for it to generate a set of candidates that includes all of our solutions. However, it is impossible to identify such a suitable N in advance. Therefore, in our comparison experiments, we first ran JKMPC and identified

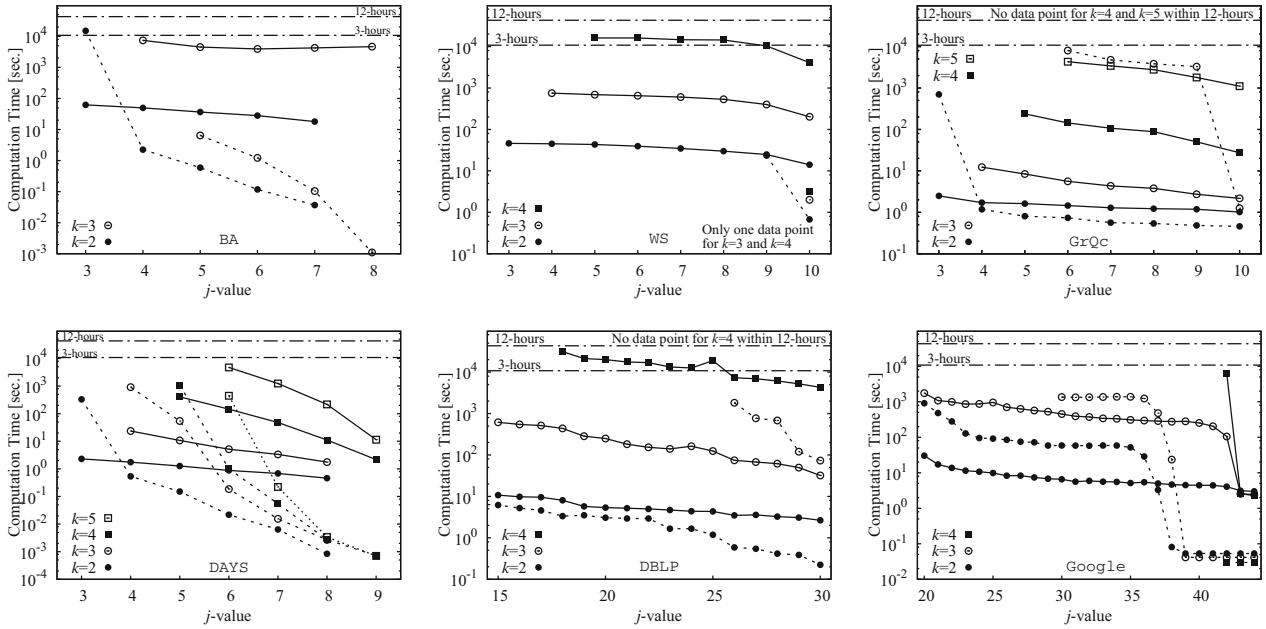


Fig. 6 Computation times

the number of solutions $\tilde{N}$. We then tried to detect $\tilde{N}$ maximal k-Plexes using `MaxKplexEnum`. Note that the k-Plexes extracted by `MaxKplexEnum` do not always contain all of our solution (j, k)-MPCs. That is, using the value of $\tilde{N}$ can provide `MaxKplexEnum` with the best-case scenario that the k-Plexes found by `MaxKplexEnum` would exactly match our solutions. Using this approach would never unfairly advantage our `JKMPC`.

Figure 6 shows the computation times for both systems, where the dotted lines indicate `MaxKplexEnum` and the solid lines indicate `JKMPC`. Data points for each k-value are distinguished by point types (e.g., ●, □). For each computation, we set a time limit of 12 h. If a computation for some j and k did not complete within the limit, the corresponding data point is omitted from the graphs. In our experiments, we assumed that the values of j were larger than k to exclude undesirable solutions of low density.

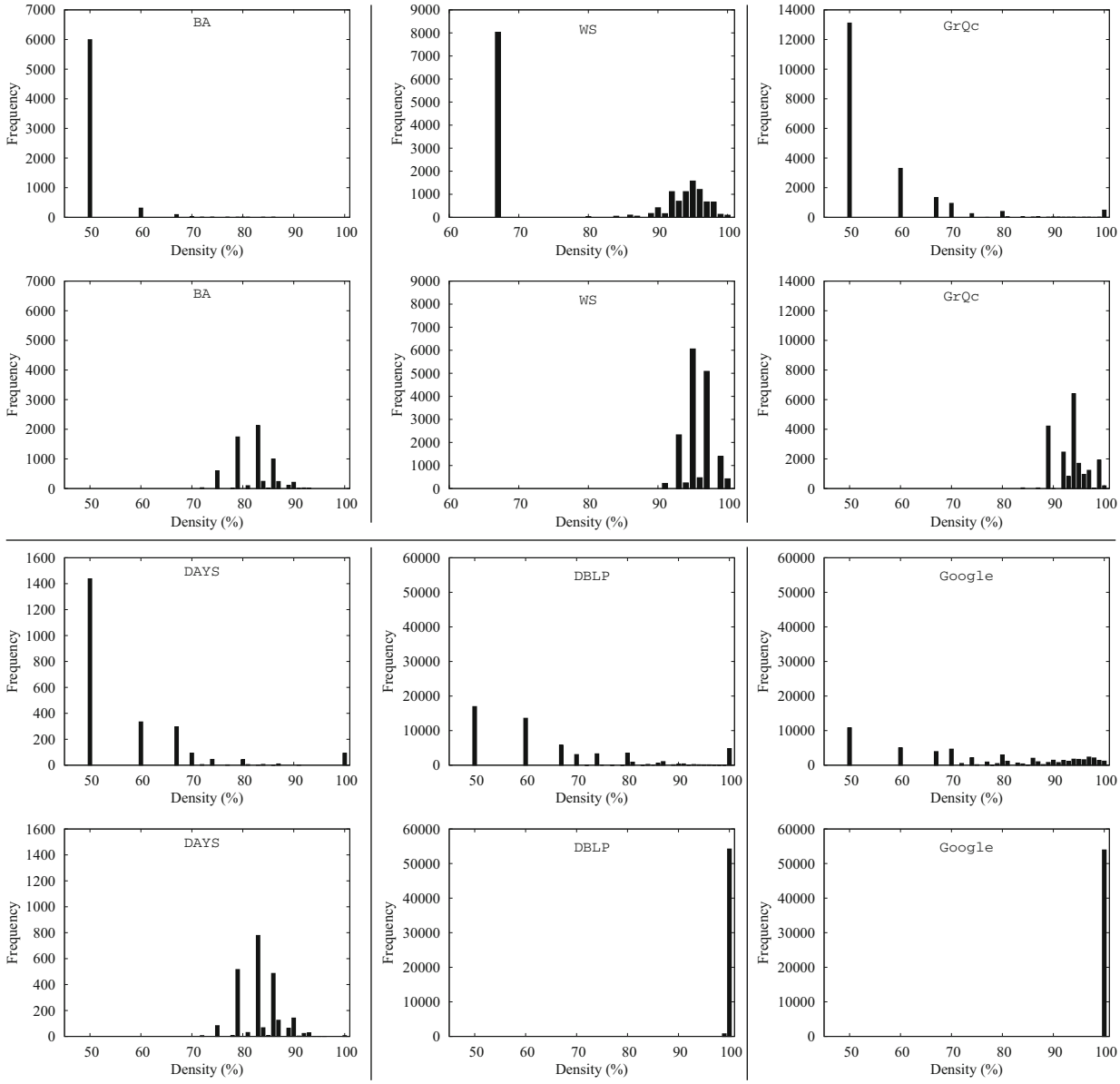
The larger a j-value is, the smaller the number of (j, k)-MPCs becomes. `MaxKplexEnum` finds the computation easier for higher ranges of j because it is required to enumerate a smaller number of maximal k-Plexes. In fact, `MaxKplexEnum` executes faster than our system for higher ranges of j. Its performance, however, suddenly deteriorates as j decreases. In many cases, it failed to complete the enumeration within the time limit. One such example was the case of $j = 3$ and $k = 2$ for `BA`, where `MaxKplexEnum` was required to extract 19, 138 maximal k-Plexes.

The number of solutions we have to enumerate is also affected by the value of k. In most cases, it increases as k increases. It turns out that `MaxKplexEnum` can work well for a small k, say 2. For $k = 3$ or $k = 4$, however, we often observed low performance by `MaxKplexEnum` even for the larger j-values. For example, for `WS`, the computation was completed within the time limit only for $j = 10$. Moreover, no data points for $k = 4$ and $k = 5$ were observed with `GrQc`, or for $k = 4$ with `DBLP`.

In contrast, the performance of `JKMPC` was almost stable. Even for cases that were difficult for `MaxKplexEnum`, our system could enumerate all solutions. In other words, `JKMPC` has the ability to extract (j, k)-MPCs even for relatively small sizes. This is a remarkable advantage of our system because it is usually difficult for standard methods of graph clustering and partitioning to detect such small dense subgraphs. Therefore, these results demonstrate that `JKMPC` is an efficient and practical system for enumerating (j, k)-MPCs.

7.3 Quality of solutions as pseudo-cliques

Because the notion of a k-Plex was originally proposed as a relaxation model for a clique, any maximal k-Plexes with lower densities would be undesirable. Therefore, we would like our solution k-Plexes to have sufficient density. To investigate the quality of solution k-Plexes from the viewpoint of pseudo-cliques, we observed the density distributions for solutions obtained by `MaxKplexEnum` and `JKMPC`. As with the computational performance experiments, given a pair of j and k for each network, we first enumerated (j, k)-MPCs using `JKMPC`. If the number of obtained solutions was $\tilde{N}$, we then tried to enumerate $\tilde{N}$ connected maximal k-Plexes using `MaxKplexEnum`. Finally, for the solution k-



	BA		WS		GrQc		DAYS		DBLP		Google	
	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.	Ave.	S.D.
MaxKplexEnum	0.509	0.035	0.803	0.137	0.562	0.107	0.576	0.120	0.657	0.154	0.750	0.170
JKMPC	0.816	0.038	0.953	0.018	0.933	0.030	0.829	0.039	0.992	0.002	0.992	0.001

Ave.: Average Density S.D.: Standard Deviation of Density

Fig. 7 Density distributions of solution k-Plexes

Plexes enumerated by each system, we recorded their density distributions.

The density distributions of solutions for each network are presented in Fig. 7. For each network, the upper and lower graphs show the MaxKplexEnum and JKMPC distributions, respectively. Clearly, the distributions are quite different for each network. For more precise analysis, we also summarize the averages and standard deviations of the densities, as shown in Fig. 7.

For BA, although the standard deviations by both systems are low and almost the same, the average of 0.816 for JKMPC is much better than that of 0.509 for MaxKplexEnum.

For WS, the average of 0.803 for MaxKplexEnum seems fine. However, the actual densities are spread out away from the average, as indicated by the high standard deviation of 0.137. Additionally, the densities given for JKMPC are dis-

tributed around a high average of 0.953 with a very low standard deviation of 0.018.

For GrQc, the average of 0.562 for MaxKplexEnum is low and the actual densities are distributed in a wide range from 0.5 to 1.0, whereas the densities for JKMPC give a high average and a low deviation. Similar behavior can be observed for DAYS.

For DBLP and Google, the densities for MaxKplexEnumare somewhat low, giving averages of 0.657 and 0.750, respectively. Moreover, the standard deviations are also high, at 0.154 and 0.170, respectively. The densities for JKMPC are distributed quite differently. For both networks, we observe very high average densities, above 0.99, and very low standard deviations near zero.

These results demonstrate that the solutions using JKMPC have sufficient density. Without doubt, all solutions would be regarded as pseudo-cliques. However, the solutions using MaxKplexEnum include many maximal k-Plexes of relatively low density. In other words, the quality of solutions using MaxKplexEnum can be unstable, but JKMPC can work well in practice as an effective pseudo-clique detector.

7.4 Effectiveness of j-coreness constraint

Our (j, k)-MPC is defined as a maximal k-Plex that satisfies the j-coreness constraint. The main purpose of imposing the constraint is to exclude pseudo-cliques that are not densely connected. To investigate the effectiveness of the constraint, we compared the numbers of maximal k-Plexes and (j, k)-MPCs. The difference between them indicates the number of undesirable maximal k-Plexes, which we cannot consider densely connected.

Because there may be a huge number of maximal k-Plexes, even in a network of moderate size, enumerating all maximal k-Plexes in a large network such as DBLP or Google is quite impractical. Therefore, we present results only for WS and GrQc. More precisely, for the set of (j, k)-MPCs, we aim to enumerate (and count) the maximal k-Plexes of size larger than j because the minimum size for a (j, k)-MPC is given by $j + 1$ (for any k). For this enumeration, we implemented a simple extended version of Pemp [28] with branch-and-bound pruning based on the minimum size of the solutions.

Figure 8 shows the numbers of maximal k-Plexes and (j, k)-MPCs for each network with various j and k values. In the graphs, k-Plexes are represented by dotted lines and (j, k)-MPCs by solid lines. For each j and k, the difference between the former and the latter gives the number of maximal k-Plexes that cannot satisfy the j-coreness constraint. To clarify the differences, we show them separately in the graphs. The graphs show that the differences almost coincide with the numbers of maximal k-Plexes, particularly for larger values of k. This indicates that most of the maximal k-Plexes cannot be regarded as densely connected. Our j-coreness constraint can therefore exclude a high proportion of these undesirable maximal k-Plexes. Because our algorithm deploys several search control mechanisms based on this constraint, a large number of useless search branches

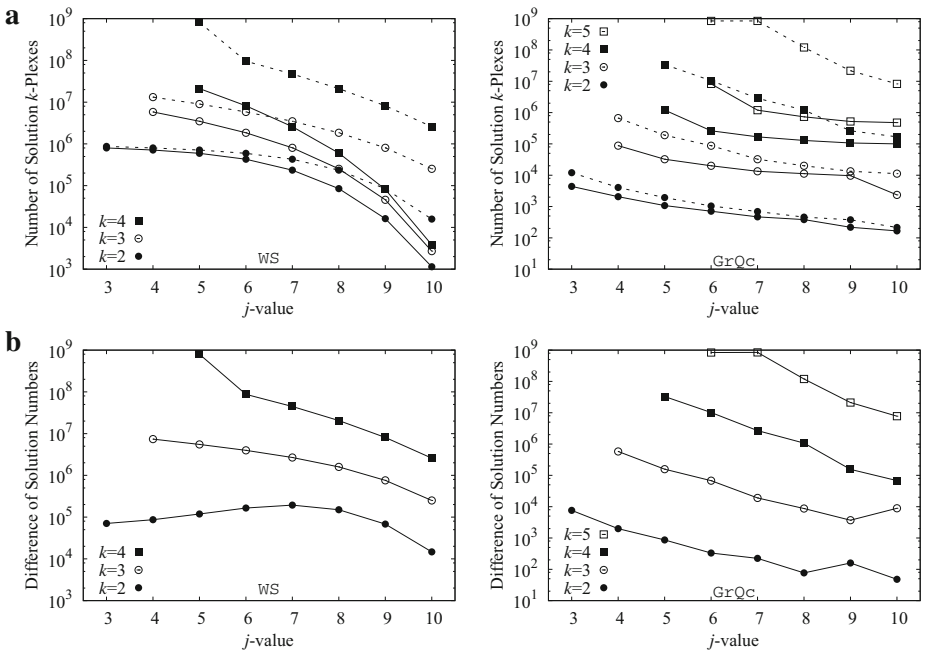


Fig. 8 Numbers of solution k-Plexes. **a** Number of maximal k-Plexes and (j, k)-MPCs. **b** Difference between numbers of maximal k-Plexes and (j, k)-MPCs

(formations) incapable of arriving at solutions can be pruned during our search process.

Note that, from the definition, every maximal k-Plex of size no less than $j+k$ is always a (j,k)-MPC. For any maximal k-Plex not densely connected, its size will be in the interval of $(j, j+k)$. This implies that most of the maximal k-Plexes can be considered as relatively small or medium sized. If our task is to detect densely connected maximal k-Plexes of larger sizes, the standard methods for graph clustering or partitioning would work well. However, they would not be suitable when we try to obtain such k-Plexes of moderate size because those clusters would usually be outside their target. As noted, our algorithm can efficiently enumerate all of (j,k)-MPCs, including small- or medium-sized ones. Moreover, finding nonlarge clusters could be valuable in the sense that they are often invisible or difficult to detect. Without the j-coreness constraint, we might not have the ability to detect such potential candidates.

7.5 Practical utility of (j,k)-MPCs

In our formulation, a densely connected pseudo-clique is defined as a maximal k-Plex that satisfies the j-coreness constraint. That is, the solution k-Plexes we aim to enumerate are intended to belong to the class of maximal k-Plexes. Theoretically speaking, however, there must exist k-Plexes that are not maximal but can satisfy the j-coreness constraint. An example of such a (nonmaximal) k-Plex is illustrated in Fig. 9.

For this graph, we try to enumerate (4, 3)-MPCs, i.e., maximal 3-Plexes that satisfy the 4-coreness constraint. It is easy to find two maximal 3-Plexes, $A = \{\mathtt{a}, 1, 2, 3, 4, 5\}$ and $B = \{\mathtt{b}, 1, 2, 3, 4, 5\}$. However, we see that $deg_A(\mathtt{a}) = 3$ and $deg_B(\mathtt{b}) = 3$, i.e., neither satisfies the constraint. Therefore, no (4, 3)-MPC can be detected for this example.

Alternatively, consider a subset of A (that is also a subset of B), $X = \{1, 2, 3, 4, 5\}$. From the antimonotonicity of a k-Plex, X is a 3-Plex. Moreover, we have $deg_X(x) = 4$ for any $x \in X$. That is, X is a 3-Plex that satisfies the 4-coreness constraint even though it cannot be a solution according to our current formulation.

It could be claimed that such an X should be detected as a solution because it actually satisfies both of the DUB and CLB constraints that we impose. This claim could lead to an alternative formulation of densely connected pseudo-cliques, namely *maximal j-cored k-Plexes* (max-(j,k)-PCs). More precisely, we can define a max-(j,k)-PC as a maximal set of vertices X that forms a k-Plex and satisfies the j-coreness constraint. Again, note that X is not necessarily a maximal k-Plex.

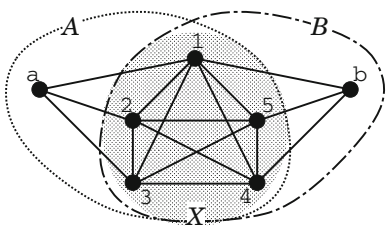


Fig. 9 Example of maximal j-cored k-Plex

Since a (j,k)-MPC is always a max-(j,k)-PC, the class of (j,k)-MPCs is subsumed by that of max-(j,k)-PCs. In our current framework, we will miss every max-(j,k)-PC that is not a maximal k-Plex. Therefore, the class of max-(j,k)-PCs would be preferable to that of (j,k)-MPCs because we have a better chance of finding pseudo-cliques that might be valuable. From a computational point of view, however, the class of (j,k)-MPCs has the advantage, because of the lack of the antimonotonicity in j-coreness. For a set of vertices satisfying the j-coreness constraint, its subset does not necessarily satisfy the constraint. This simple fact requires us to pay a substantial computational cost for detecting the maximal set of vertices with j-coreness. Therefore, (j,k)-MPCs and max-(j,k)-PCs each have their strong and weak points. On balance, we argue that the notion of (j,k)-MPCs is a reasonable formulation for pseudo-cliques from a practical viewpoint. To make this argument more convincing, we recorded the number of max-(j,k)-PCs that were actually missed in our enumeration process for (j,k)-MPCs.

From the definitions, any maximal k-Plex of size no less than $j+k$ satisfies the j-coreness constraint. That is, it can always be regarded as a max-(j,k)-PC. Moreover, no set of vertices of size no larger than j will ever satisfy the constraint. Therefore, for every max-(j,k)-PC we cannot detect, its size must be in the interval $(j, j+k)$. In particular, such a max-(j,k)-PC must appear as a proper subset of a maximal k-Plex not satisfying the j-coreness constraint. Based on these theoretical properties, we can count the number of missing max-(j,k)-PCs for given j and k. Note that we will never miss any max-(j,k)-PC for $k=2$. This is because any max-(j,k)-PC not satisfying the constraint must comprise at most $j+1 (= j+k-1)$ vertices and the size of its proper subset is therefore outside the interval $(j, j+2)$ (i.e., $[j+1, j+1]$).

Let $\mathcal{M}_{j,k}$ be the set of (j,k)-MPCs and $\mathcal{M}_{miss}$ the set of missing max-(j,k)-PCs. From the subsumption relation, the set of all max-(j,k)-PCs, denoted by $\mathcal{M}$, can be given as $\mathcal{M} = \mathcal{M}_{j,k} \cup \mathcal{M}_{miss}$. Therefore, if the ratio of $|\mathcal{M}_{miss}|$ to $|\mathcal{M}_{j,k}|$ is sufficiently low (close to 0.0), we can almost identify the class of (j,k)-MPCs as being that of max-(j,k)-PCs, with few solutions being missed. In such a case, (j,k)-MPCs, which can be computed efficiently, would be considered as a reasonable and practical formalization of pseudo-cliques.

Table 2 lists the ratio of $|\mathcal{M}_{miss}|$ to $|\mathcal{M}_{j,k}|$ for various j and k. Because the size interval of max-(j,k)-PCs in $\mathcal{M}_{miss}$ is defined as $(j, j+k)$, it would seem that we would miss more max-(j,k)-PCs as k increases. For the synthetic net-

Table 2 Ratio of $|\mathcal{M}_{miss}|$ to $|\mathcal{M}_{j,k}|$

	$\lvert\mathcal{M}_{miss}\rvert$	$\lvert\mathcal{M}_{j,k}\rvert$	$\frac{\lvert\mathcal{M}_{miss}\rvert}{\lvert\mathcal{M}_{j,k}\rvert}$
WS			
$k=3$			
$j=4$	0	5, 821, 233	0.000000
$j=5$	0	3, 483, 608	0.000000
$j=6$	0	1, 848, 536	0.000000
$j=7$	0	806, 784	0.000000
$j=8$	16	253, 356	0.000063
$j=9$	50	45, 888	0.001090
$j=10$	33	2706	0.012195
$k=4$			
$j=5$	0	21, 131, 619	0.000000
$j=6$	0	8, 178, 200	0.000000
$j=7$	0	2, 591, 030	0.000000
$j=8$	7	602, 300	0.000012
$j=9$	96	81, 546	0.001177
$j=10$	304	3815	0.079685
GrQc			
$k=3$			
$j=4$	16	87, 786	0.000182
$j=5$	6	32, 407	0.000185
$j=6$	4	20, 014	0.000200
$j=7$	0	13, 364	0.000000
$j=8$	0	11, 247	0.000000
$j=9$	0	9658	0.000000
$j=10$	0	2355	0.000000
$k=4$			
$j=5$	18	1, 204, 135	0.000015
$j=6$	4	262, 717	0.000015
$j=7$	1	168, 025	0.000006
$j=8$	0	128, 942	0.000000
$j=9$	0	107, 120	0.000000
$j=10$	0	100, 080	0.000000
$k=5$			
$j=6$	0	8, 235, 436	0.000000
$j=7$	2	1, 192, 371	0.000002
$j=8$	0	731, 706	0.000000
$j=9$	0	521, 364	0.000000
$j=10$	0	478, 779	0.000000

work WS, such a trend can be observed for larger values of j. For smaller j, however, we miss very few max-(j,k)-PCs. In contrast, for the real network GrQc, the numbers of missing max-(j,k)-PCs are quite low, almost 0, for the whole range of j. These different observations would be caused by the size distributions for the larger solutions ((j,k)-MPCs).

More concretely, the WS solution sizes are distributed over the narrow range of 11 to 14 (for $j=10$ and $k=4$). In contrast, for GrQc, we observe a wide range of solution sizes, from 11 to 46 (for $j=10$ and $k=4$). We can note simply that a larger k-Plex can subsume more k-Plexes. For $j=10$ and $k=4$, moreover, the size of the missed max-(j,k)-PCs must be in the interval of [11, 13]. Therefore, the larger solutions in GrQc would tend to make most of the k-Plexes in the interval nonmaximal. As a result, there are several max-(j,k)-PCs we will miss. Conversely, for WS, because there are a small number of (j,k)-MPCs, and all of them are relatively small, k-Plexes in the interval would be less likely to be subsumed. Therefore, we might sometimes miss the detection of a certain number of max-(j,k)-PCs.

Although cases like those for WS are undesirable, similar cases would not often arise in real-world networks. In actual networks, such a narrow range of solution sizes is rare. We emphasize again the ability of our algorithm to detect densely connected pseudo-cliques of relatively small size (for smaller j-values). Therefore, we consider our (j,k)-MPCs to be a useful formalization of densely connected pseudo-cliques in practice.

8 Conclusion

In this paper, we have considered the design of an efficient complete algorithm for enumerating (j,k)-MPCs. For efficient computation, we have discussed several search mechanisms that can prune many useless search nodes effectively. Our experimental results have demonstrated that the algorithm can work well as a practical tool for extracting densely connected pseudo-cliques in large networks.

Although our j-coreness constraint can drastically reduce the number of solutions to be enumerated, it might prove inadequate when dealing with much larger-scale networks. Therefore, it is worth seeking additional constraints that would target the desired solutions more tightly. The authors have investigated a reasonable constraint based on the notion of *variable-j-coreness* [19] that considers the degree of isolation of cliques, and are currently extending it. Based on this work, we expect that the efficiency of our algorithm can be improved further, becoming practical even for huge networks of over a million vertices.

Acknowledgements The authors of [6] have made the program code for MaxKplexEnum available to us. We sincerely appreciate their kindness.

References

1. Abello, J., Resende, M.G.C., Sudarsky, S.: Massive quasi-clique detection. In: Proceedings of the LATIN 2002, LNCS-2286, pp. 598–612 (2002)
2. Alba, R.D.: A graph-theoretic definition of a sociometric clique. J. Math. Sociol. **3**(1), 113–126 (1973)
3. Barabási, A.-L., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999). (AAAS)
4. Batagelj, V., Mrvar, A.: Pajek datasets. http://vlado.fmf.uni-lj.si/pub/networks/data/ (2006)
5. Batagelj, V., Zaversnik, M.: An $O(m)$ Algorithm for Cores Decomposition of Networks, CoRR (2003). cs.DS/0310049 OpenURL
6. Berlowitz, D., Cohen, S., Kimelfeld, B.: Efficient enumeration of maximal k-plexes. In: Proceedings of the 2015 ACM SIGMOD Conference, pp. 431–444 (2015)
7. Corman, S.R., Kuhn, T., Mcphee, R.D., Dooley, K.J.: Studying complex discursive systems: centering resonance analysis of communication. Hum. Commun. Res. **28**(2), 157–206 (2002)
8. Eppstein, D., Strash, D.: Listing all maximal cliques in large sparse real-world graphs. In: Proceedings of the 10th Int'l Symposium on Experimental Algorithms—SEA'11, LNCS-6630, pp. 364–375 (2011)
9. Junker, B.H., Schreiber, F.: Analysis of Biological Networks. Wiley, New York (2008)
10. Leskovec, J., Kleinberg, J. and Faloutsos, C.: Graph evolution: densification and shrinking diameters. ACM Trans. Knowl. Discov. Data 1(1), Article No. 2 (2007)
11. Leskovec, J. and Krevl, A.: SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data (2014)
12. Leskovec, J., Lang, K., Dasgupta, A., Mahoney, M.: Community structure in large networks: natural cluster sizes and the absence of large well-defined clusters. Internet Math. **6**(1), 29–123 (2009)
13. Luce, D.R.: Connectivity and generalized cliques in sociometric group structure. Psychometrika **15**(2), 169–190 (1950)
14. Luxburg, U.: A tutorial on spectral clustering. Stat. Comput. **17**(4), 395–416 (2007)
15. Mokken, R.: Cliques, clubs and clans. Qual. Quant. Int. J. Methodol. **13**(2), 161–173 (1979)
16. Nagurney, A. (ed.): Innovations in Financial and Economic Networks. New Dimensions in Networks Series. Edward Elgar Publishing, Cheltenham (2004)
17. Newman, M.E.J.: Finding community structure in networks using the eigenvectors of matrices. Phys. Rev. E **74**(3), 036104 (2006)
18. Okubo, Y., Haraguchi, M., Tomita, E.: Structural change pattern mining based on constrained maximal k-plex search. In: Proceedings of the 15th Int'l Conference on Discovery Science—DS'12, LNAI-7569, pp. 284–298 (2012)
19. Okubo, Y., Haraguchi, M., Tomita, E.: Enumerating maximal isolated cliques based on vertex-dependent connection lower bound. In: Proceedings of the 12th Int'l Conference on Machine Learning and Data Mining—MLDM 2016, LNAI-9729, pp. 569–583 (2016)
20. Pattillo, J., Youssef, N., Butenko, S.: Clique relaxation models in social network analysis. In: Thai, My T., Pardalos, P.M. (eds.) Handbook of Optimization in Complex Networks: Communication and Social Networks. Springer Optimization and its Applications, vol. 58, pp. 143–162. Springer, New York (2012)
21. Rymon, R.: Search through systematic set enumeration. In: Proceedings of Int'l Conference on Principles of Knowledge Representation Reasoning—KR'92, pp. 539–550 (1992)
22. Scott, J.P., Carrington, P.J. (eds.) The SAGE Handbook of Social Network Analysis. SAGE Publications Ltd, London (2011)
23. Seidman, S.B.: Network structure and minimum degree. Soc. Netw. **5**(3), 269–287 (1983)
24. Seidman, S.B., Foster, B.L.: A graph-theoretic generalization of the clique concept. J. Math. Sociol. **6**(1), 139–154 (1978)
25. Slater, N., Itzchack, R., Louzoun, Y.: Mid size cliques are more common in real world networks than triangles. Netw. Sci. **2**(3), 387–402 (2014)
26. Tomita, E., Tanaka, A., Takahashi, H.: The worst-case time complexity for generating all maximal cliques and computational experiments. Theor. Comput. Sci. **363**(1), 28–42 (2006)
27. Watts, D.J., Strogatz, S.H.: Collective dynamics of small-world networks. Nature **393**(6684), 440–442 (1998)
28. Wu, B., Pei, X.: A parallel algorithm for enumerating all the maximal k-plexes. In: Proceedings of the PAKDD 2007 Workshops, LNAI-4819, pp. 476–483 (2007)
29. Yang, J., Leskovec, J.: Defining and evaluating network communities based on ground-truth. In: Proceedings of the 12th IEEE Int'l Conference on Data Mining—ICDM'12, pp. 745–754 (2012)
30. Zhai, H., Haraguchi, M., Okubo, Y., Tomita, E.: A fast and complete enumeration of pseudo-cliques for large graphs. In: Proceedings of the 20th Pacific Asia Conference on Knowledge Discovery and Data Mining—PAKDD 2016 (Part I), LNAI-9651, pp. 423–435 (2016)