Journal of University of Duhok, Vol. 20,No.1(Pure and Eng. Sciences), Pp 175-183, 2017
eISSN: 2521-4861 & pISSN: 1812-7568
https://doi.org/10.26682/sjuod.2017.20.1.16

A FAST METHOD FOR OPTIMIZING THE *K*-CLUSTERING BI-CLIQUE COMPLETION PROBLEM IN TELECOMMUNICATION

SAGVAN ALI SALEH*and MHAND HIFI**
*University of Duhok, Kurdistan Region-Iraq
** Universitѐ de Picardie Jules Verne, 7 rue du Moulin Neuf, 80000 Amiens, France

ABSTRACT

In this work, we present a fast approximate method for solving the *k*-Clustering Minimum Bi-clique Completion Problem (KBCP), a problem belonging to the telecommunication and transportation domains. In KBCP, we are given a set of demands of services from customers and its goal is to determine a subset of *k* multicast sessions that is able to partition the set of the starting demands. Each of the considered service has to belong to a multicast session while each costumer can appear in more sessions. The KBCP is tackled by using a fast approximate method which is based on the principle of neighborhood search techniques. The method can search several solutions belonging to diversified sub-spaces aiming to find the best solution. The performance of the presented method is evaluated on benchmark instances taken from the literature, whereby the provided results are compared to those reached by the Cplex solver and recent methods described in the literature. The results show that, the proposed method is fast and competitive and it is able to reach new bounds.

KEYWORDS: Bi-clique; heuristic; neighborhood; optimization.

1 INTRODUCTION

Combinatorial optimization problems are of great interest both for fundamental research and industrial real-world. The research community has modeled several practical situations as combinatorial optimization problems. Among these problems, we can find a problem belonging to the telecommunication and transportation domains: $k-$Clustering Minimum Bi-clique Completion Problem (abbreviated to KBCP). Such a problem has been already discussed in Faure *et al.* [3], where an application dealing with telecommunications has been tackled. Indeed, the authors studied a multicast session application characterized by a subset of clients that needs the same information, where each of these clients can require some multicast sessions. Then, a telecommunication network cannot manage several multicast sessions at the same time. In this case, some sessions should be grouped into a limited subset of clusters and so, the goal of the problem is to limit the number of unnecessary information to send to the current clients. Hence, such a problem has been modeled in Faure *et al.* [3] as a biclique bipartite graph, where the subset of communications must be minimized.

The KBCP can be graphically represented by a bipartite directed graph G (V, E). V is the set that contains n vertices such that $V = S \cup T$ and $S \cap T = \emptyset$, where S (resp. T) is the first (resp. second) part of nodes of G. The set E contains the interconnected edges between S and T. On the one hand, the couple (S, T) is a bi-clique graph formed with k bi-partite graphs (named as clusters), i.e., $(S_1, T_1)\dots(S_k, T_k)$. On the other hand, all vertices of each couple (S_j, T_j), where $j = 1, \dots, k,$ are interconnected with each others. However, the graph $G(V, E)$ is defined as a general directed graph, searching for a k-clustering. Hence, the aim of the problem is to determine a k bi-partite sub-graph of G with a minimum additional edges that do not belong to the set E. Differently stated, the KBCP is equivalent to determine the best partition of the set S into k clusters realizing a minimum additional edges.

Formally, KBCP can be stated as follows:

$$Min \quad \sum_{P\in K}\sum_{(i,j)\in\overline{E}} Z_{ijp}$$
(1)

$$s.t. \quad xip + yjp \leq 1 + zijp, \forall(i,j) \in \overline{E}, \forall p \in K$$
(2)

$$\sum_{p\in K} xip = 1; \forall i \in S\text{v}$$
(3)

$$xip \leq yjp, \forall(i,j) \in E, \forall p \in K$$
(4)

$$xip, yjp \in \{0,1\}, \forall i \in S, \forall p \in K\, \forall j \in T$$
(5)

$$zijp \in \{0,1\}, \forall(i,j) \in \overline{E}, \forall p \in K$$
(6)

where Equation (1)) represents the objective function in which the goal is to minimize the number of added edges (i.e. the total cost of the clusters). The first constraint (Equation (2)) imposes the link between the variables xip and yjp by making $zijp$ is equal to 1 whenever the both other variables are fixed to 1. The second constraint (Equation (3)) imposes that each vertex $\boldsymbol{i}$ belonging to the set S should be assigned to one and only one cluster. The third constraint (Equation (4)) imposes each vertex $\boldsymbol{j}$ belonging to the set T, which is adjacent to another vertex $\boldsymbol{i}$ belonging to the set S and assigned to the $p^{t□}$ cluster, to be affected to the same cluster (we recall that some vertices of T can be assigned to several clusters). The Forth and last constraints (Equations (5 and 6)) impose the integrality of the variables.

The rest of this paper is organized as following. Section 2, presents some previous works on KBCP. Section 3 introduces a fast method for approximately solving the KBCP. Section 4 studies the performance of the proposed method, where the given results are compared to those results obtained by the recent solution methods available in the literature. Finally, Section 5 concludes the contribution of the work.

2 RELATED WORK

The KBCP has been firstly addressed by Faure et al. [3], where they proved the hardness of the problem. The authors presented a mathematical model for optimizing to optimality instances with small sizes. They also presented a column generation-based heuristic, to optimize instances of large scale of the problem.

Gualandi [4] proposed a hybrid method which combines constraint programming and semidefinite programming. Gualandi et al. [5] presented a method based upon branch-and-price. The method accelerates the search procedure, at the same time, it improves the upper bounds quality with the use of the Cplex solver. Hifi et al. [6] proposed an adaptive neighborhood search method. The method uses an intensification and diversification procedures in order to diversify the solution process and explore unvisited solutions' spaces. It is based upon a greedy random walk procedure where two phases are used in order to iteratively improve the solutions at hand. Finally, Al-Iedani et al. [1] presented a variable neighborhood search heuristic, where the method starts with an initial solution with moderate quality. Then, an iterative solution procedure is used to improve a current solution in its neighborhood. In the experimental part of Al-Iedani et al. [1], the authors demonstrated experimentally the superiority of the neighborhood search when comparing its results to those achieved by the method proposed in Hifi et al. [6]. In this paper, a fast approximate method is proposed for solving the KBCP. It is based upon neighborhood search techniques that tries to converge to high quality solutions (high upper bounds) within short average runtime. Such a method explores iteratively a series of sub-solution spaces, with the aim of finding the optimal solution through the search process. We recall that, an instance of the KBCP is described as a bipartite directed graph. The objective of the problem is to divide the S vertices into k clusters, such that the number of adding edges to form the k bi-cliques is minimize (i.e., minimize the cost of the problem).

3. A FAST APPROXIMATE METHOD FOR SOLVING THE KBCP

Neighborhood search techniques are of high interest techniques that have shown to be so efficient for optimizing a large variety of combinatorial optimization problems (cf., Shaw [9], Dasgupta *et al.* [2]). Although such techniques produce approximate solution methods, they allow us to present fast algorithms that yield high quality solutions within a short average runtime. In general, neighborhood search-based algorithms consists of two complementary procedures: a building procedure and an exploring procedure. The building procedure used some techniques to yield a reduced solution spaces, while the aim of the exploring procedure is to search the yielded space to find a local optimum solution.

This section presents our proposed method for approximately solving the KBCP. The used method is based on the principle of exploring a series of sub-solution spaces with the aim of exploring a more promising search space and reaching better solution. In fact, the proposed method is composed of three main steps (cf., Hifi *et al.* [7]):

1. The first step yields a fast feasible solution.
2. The second step is used in order to enhance the solution at hand, by using what is known as *intensification strategy*. The intensification strategy presented used a combination of neighborhood search techniques, such as exchanging, 2-opt, and 3-opt procedures.
3. The third and last step diversifies the search process, by using what is known as *diversification strategy*. The diversification strategy introduces two strategies: degrading and re-optimizing strategies.
Both intensification and diversification strategies are used in order to explore more promising feasible solution spaces with the aim of escaping from a series of local optimum solution.

```
Algorithm 1 A starting feasible solution
Require: $PKBCP$ An instance of the problem.
Ensure: $SKBCP$ A feasible solution.
1: Initialization:
2: Let $Si$ be the set services, where $i = \{1, 2, 3... m\}$, $m$ is the number of services.
3: Let $Kk$ be the set clusters, where $k = \{1, 2, 3... n\}$, $n$ is the number of clusters.
4: Set $S^t \leftarrow S$, $Kk = \emptyset$, and counter = 0;
5: Sort all services of ($S'$) in decreasing order of their degrees;
6: while $S' \neq \emptyset$ do
7: Let $i = argmax\{S'_i,\ S'_i \geq S'_j,\ j \in S'\}$
8:         if (counter = n) then
9:         counter = 0;
10:        end if
11:        counter = counter + 1;
12:        Set $Kcounter = Kcounter \cup \{i\}$.
13:        Remove $i$ from $S^t$.
14: end while
15: Each cluster $k$ is completed with the additional edges to form the $k$-bicliques.
16: Compute the cost.
17: $SKBCP \leftarrow Kk$
18” return SKBCP as a feasible solution of PKBCP
```

3.1 An initial solution for KBCP

The first step involves the use of a greedy procedure in order to provide a quick starting feasible solution. We recall that, greedy solution procedures are a type of

heuristics that construct a fast solution piece by piece focusing on an immediate improvement without looking ahead (i.e., without considering the consequences). Generally, this type of approach does not ensure the optimality of the solution but it is very fast for determining a starting feasible solution. For this reason, we considered a simple greedy procedure, which can be viewed as a variation of the well- known Johnson’s algorithm (cf., Johnson [8]). The main steps of this procedure in given in Algorithm 1. It illustrates the main steps of the proposed greedy method. Where a solution is constructed sequentially. However, the algorithm starts by sorting the services in decreasing order according to their degrees. Lines 6-14 is the main loop, which describes the main steps of the method, where the clusters Kk are iteratively filling with the services belonging to S^l. Such that, the service realizing the greatest degree is selected and inserted into a cluster Kk, then, the next service into another cluster and so on. Such a process is iterated until all services in *S'* are inserted into Kk.After that, each cluster is completed by adding edges to form the *k*-bicliques of the graph *G* (cf., Line 15). Then, the objective value is evaluated (the cost of the feasible solution).

3.2 Minimizing the cost of the solution

Minimizing the cost of a given solution is equivalent to enhance the solution's quality achieved by the greedy procedure (cf., Algorithm 1). For this reason, the second step is introduced which explores a series of neighborhoods of a solution at hand with the aim of improving its quality. In fact, a combination of neighborhood search techniques are used which work together as an intensification procedure. In other words, a number of local search techniques are used to improve the quality of solutions at hand, including: *k-exchanging* and a *k-opt strategies*. The main idea of both strategies consists of replacing some services (vertices) by others. This is allowed if the current solution is improved. However, the *k*-exchanging strategy performs swapping between some services of the cur- rent clusters, while the *k*-opt strategy performs many moves of services between clusters. However, the aforementioned strategies are performed as the following sequence:

1. *K-exchanging* strategy: swap *k*-vertex in a cluster with other *k*-vertex in an- other cluster; this is allowed when the resulting solution improves the best neighborhood solution.
2. The best solution obtained from the previous step is modified by introducing a *k*-opt procedure as following: (a) select a set of *k*-vertices from a cluster then move them to other clusters (if it is possible) and then, (b) recombine the clusters and provide a KBCP's feasible solution.

3.3 Diversifying the search process

Local search procedures (cf., Section 3.2) can improve a solution, when they try to search a series of neighborhoods in a sub-solution space, where the neighborhoods are close to the current solution. Thus, the obtained solutions from such methods are not of high quality, because they are local optimum. Further- more, it is necessary to diversify the solution procedure, i.e., change the search space which induce the exploration of new neighborhoods and so, some other new and better solutions may arise. Thus, a diversification process is introduced for searching a series of new solutions and escape from a series of local optimum solutions. The diversification process diversifies the search process by degrading the quality of the solution. This is done by applying a random destroying strat- egy that removes randomly a percentage of services from the current clusters and produces a reduced problem. The reduced problem is then optimized using the greedy procedure presented in Section 3.1 and the local search strategies presented in Section 3.2.

Now we are going to present an overview of the proposed heuristic for approximating the KBCP. The main steps of the proposed method are illustrated in Algorithm 2.

Algorithm 2 : A fast heuristic algorithm for the KBCP
Require: $SKBCP$, a starting solution of $PKBCP$.
Ensure: S^*_{KBCP}, alocal optimum solution of PKBCP.

```
1: Set $S^*_{KBCP} \leftarrow SKBCP$
```

```
2: while (the time limit is not performed) do
3: Apply a random destroying strategy (remove $\alpha$% of services) in order to find a reduced problem $S^r_{KBCP}$ according to $S^*_{KBCP}$
4: Call Algorithm 1 in order to complete and optimize $S^r_{KBCP}$ solution and reaching a new solution SKBCP.
5: Apply 1-exchanging, and 2-exchanging strategies in order to Improve $SKBCP$.
6: Apply 2-opt, and 3-opt procedures in order to Improve $SKBCP$.
7: Update $S^*_{KBCP}$ with the best solution.
8: end while
9: return $S^*_{KBCP}$
```

Note that, Algorithm 2 can be viewed as an iterative yield that iteratively pro- duce a series of degraded solutions of the KBCP and their optimized ones. That is, the diversification process. First, the algorithm starts with an initial solution obtained from Algorithm 1. The main loop (line 2-line 8) presents the degrading and re-optimizing procedures which are used in order to end the search process when the sopping criteria is satisfied; herein, a limited runtime is considered (as shown in the experimental part). In line 3, a random degrading strategy is used where α% of services are randomly removed from random cluster; in this case a reduced problem S^r_{KBCP} is obtained. In line 4, the reduced problem is first treated by using Algorithm 1. Then, the obtained solution is optimized using k-exchanging and k-opt (cf., line 5 and line 6). Finally, the best solution obtained is updated.

4 COMPUTATIONAL RESULTS

This section evaluates the effectiveness of the proposed heuristic: Fast Method for solving the KBCP (noted FM-KBCP). We first describe some benchmark instances extracted from the literature (taken from Gualandi et al. [5]). Next, we have already mentioned that Al-Iedani et al.'s [1] algorithm outperformed Hifi et al.'s [6] algorithm, especially when comparing the quality of the achieved results by both algorithm. We then decided to evaluate the performance of the proposed method by comparing its achieved results to those reached by Al- Iedani et al.'s [1] algorithm (noted VNS). Finally, the results obtained by the proposed FM-KBCP are compared to the best results of the literature (cf., Al- Iedani et al. [1]), and to those reached by the Cplex solver version 12.5, where the time limit is fixed to 3600 seconds and with the default setting. The proposed FM-KBCP was coded using C++ and tested on Pentium Core i5, 2.8GHz.

Table 1 shows the set of instances obtained from the literature (cf., Gualandi *et al.* [5] and Al-Iedani *et al.* [1]) for testing. Column 1 indicates the in- stance label. Columns 2 and 3 indicate the cardinalities of the set *S* (services) and the set *T* (costumers), respectively. Column 4 (*k*) displays the number of clusters required. Finally, column 5 (*d*) indicates the graph density related to each instance. One can see that, the instances are varied in their number of services S, costumers T, clusters k, and density of the graph d.

Table (1): Instances' characteristics.

Instanc	*S*	*T*	*k*	*d*
IA.1	15	15	2	0.3
IA.2	15	15	2	0.5
IA.	15	15	2	0.7
IB.1	15	15	5	0.3
IB.2	15	15	5	0.5
IB.	15	15	5	0.7
IC.1	18	18	2	0.3
IC.2	18	18	2	0.5
IC.	18	18	2	0.7
ID.1	18	18	5	0.3
ID.2	18	18	5	0.5
ID.	18	18	5	0.7
IE.1	20	20	5	0.3
IE.2	20	20	5	0.5
IE.	20	20	5	0.7
IF.1	50	50	5	0.3
IF.2	50	50	5	0.5
IF.	50	50	5	0.7
IG.1	50	50	10	0.3
IG.2	50	50	10	0.5
IG.3	50	50	10	0.7

4.1 Effect of the parameter α

In this section, we evaluate the behavior of the proposed heuristic when varying the parameters used by such a heuristic. Indeed, *FM-KBCP* involves two parameters: the stopping criteria and the percentage α%. The first parameter represents the runtime limit, where it is fixed to 30 seconds for FM-KBCP. This fixation is for computational purposes (in the literature, the cpu-time is fixed to 100 second). The second parameter α%, which is the percentage of the removed vertices belonging to the solution at hand, from different clusters. For this parameter, the behavior of FM-KBCP is evaluated by varying the value of α in the discrete interval: α ∈ {5; 10; 20; 30; 40}, as shown in Table 2.

Table 2 illustrates the average bounds (solution values) reached by FM- KBCP over all tested instances by limiting the runtime to 30 second (on average). The first column represents the instance label. Columns from 2 to 6 tally the average results when varying α from 5% to 40%, according to the discrete interval used. From Table 2, one can observe that the best average solution value is reached when α equals to 30. Indeed, in this case the average upper bound is equal to 351.0 which represents the smallest average value when compared to the rest of the bounds. Because FM-KBCP seems to give better results for α = 30, we then tune proposed heuristic with such value for the rest of the paper.

Table (2): The effect of the parameter α on the solutions' quality.

#Inst	Variation of α				
	5%	10%	20%	30%	40%
IA.1	125	126	115	115	117
IA.2	99	95	95	95	95
IA.3	64	64	63	61	63
IB.1	60	56	52	52	56
IB.2	54	54	51	51	54
IB.3	47	42	40	40	40
IC.1	186	184	184	180	184
IC.2	146	126	126	126	126
IC.3	91	91	89	89	91
ID.1	135	109	101	98	101
ID.2	108	106	90	90	96
ID.3	77	68	63	63	68
IE.1	131	131	130	130	130
IE.2	137	137	123	123	132
IE.3	104	98	98	90	96
IF.1	1508	1501	1490	1490	1501
IF.2	1120	1104	1104	1104	1104
IF.3	703	680	680	678	680
IG.1	1185	1185	1175	1175	1185
IG.2	958	930	930	930	930
IG.3	605	605	605	591	591
Av Sol	364.0	356.8	352.6	351.0	354.3

4.2 Effect of the degrading and re-optimizing procedures

In this section, we evaluate the behavior of the proposed FM-KBCP on the in- stances considered in the last section. Its obtained bounds are displayed and compared to those reached by the best results available in the literature (extracted from Al-Iedani et al. [1]) and to those achieved by the Cplex solver 12.5.

Journal of University of Duhok, Vol. 20,No.1(Pure and Eng. Sciences), Pp 175-183, 2017
eISSN: 2521-4861 & pISSN: 1812-7568
https://doi.org/10.26682/sjuod.2017.20.1.16

Table (3): Performance of FSA-KBCP vs VNS and Cplex.

#Inst	Cplex 3600s	VNS 100s	FM-KBCP's solutions						
			1	2	3	4	5	Av Sol	Best
IA.1	156	123	113	117	113	113	117	114.6	113
IA.2	93	95	93	92	92	93	93	92.6	92
IA.3	63	63	63	63	61	63	63	62.6	61
IA.1	52	73	54	52	52	52	52	52.4	52
IA.2	51	62	52	52	51	52	52	51.8	51
IA.3	41	47	42	42	40	40	40	40.8	40
IA.1	180	190	182	182	180	182	180	181.2	180
IA.2	127	136	128	126	128	126	126	126.8	126
IA.3	89	89	91	89	89	91	89	89.8	89
IA.1	94	126	92	92	94	94	94	93.2	92
IA.2	87	98	87	86	86	86	86	86.2	86
IA.3	63	67	63	65	63	65	63	63.8	63
IA.1	120	174	126	129	126	126	126	126.6	126
IA.2	124	132	132	128	128	132	123	128.6	123
IA.3	92	90	82	82	82	82	96	84.8	82
IA.1	1485	1507	1490	1490	1490	1490	1501	1492.2	1490
IA.2	1100	1090	1104	1104	1104	1104	1104	1104	1104
IA.3	685	681	678	678	678	678	680	678.4	678
IA.1	1151	1213	1175	1175	1175	1175	1175	1175	1175
IA.2	936	928	930	930	930	930	930	930	930
IA.3	588	607	591	591	591	591	591	591	591
Average	351.3	361.5	350.9	350.7	350.1	350.7	351.5	350.8	349.7

Table 3 displays the results achieved by FM-KBCP and those reached by the algorithm proposed in Al-Iedani *et al.* [1] (noted VNS), and those realized by the Cplex solver (noted Cplex). The Cplex solver solves the problems to optimality, therefore, in order to use the method as a heuristic, the runtime limit of the Cplex was extended to 3600 second. Column 1 illustrates the instance label. Column 2 shows the bounds realized by the Cplex solver. Column 3 (VNS) displays the best upper bounds reached by VNS when the runtime limit was fixed to 100 seconds (using an equivalent computer). Columns from 4 to 8 display five trials representing the bounds achieved by FM-KBCP when fixing the runtime limit to 30 seconds. Columns 9 shows the average bounds over the five trials. Finally, Column 10 reports the best bounds reached by the five trials for FM-KBCP. From Table 3, one can observe what follows:

1. The average solution values (bounds) achieved by FM-KBCP over the five trials are generally better than those reached by both VNS and Cplex (see the last line of Table 3). Indeed, on the one hand, FM-KBCP is able to realize an average value of 350.8 over all the five trials, whereas VNS realizes an average bound of 361.5. On the other hand, FM-KBCP performs better than the Cplex since the last method achieved also a greatest bounds (361.3).

2. According to the runtime limits, FM-KBCP requires an average runtime smallest than those needed by both VNS and Cplex. Indeed, FM-KBCP requires only 30 seconds to achieve its bounds whereas VNS and Cplex needed 100 and 3600 seconds, respectively to achieve their bounds.

3. According to the best solutions achieved, FAS-KBCP is able to provide eleven new bounds out of the 21 tested instances, which represents a percentage of more than 50% of the best solution values (see column 10).

5 CONCLUSION

In this paper, we proposed a fast heuristic for optimizing a problem belonging to the telecommunication domain, known as the k-clustering minimum biclique completion problem. The proposed method is based upon the principles of neighborhood search techniques. The main idea is that, search several solutions belonging to diversified sub-spaces aiming to find the global optimum. First, the method starts with a greedy procedure for constructing an initial feasible solution. Second, a combination between local search techniques is used as an improvement procedure in order to improve each solution at hand. Third and last, a random destroying strategy is used as a diversification procedure for jumping toward un-searched spaces. Finally, computational results showed that the pro- posed method remains competitive and it is able to reach new solution values within small average runtimes.

REFERENCES

1. Al-Iedani, N., Hifi, M., Saadi, T.: Neighborhood search-based heuristic for the k- clustering minimum biclique completion problem. In Control, Decision and Informa- tion Technologies (CoDIT), International Conference on IEEE, pp. 639–643, (2016).

2. Dasgupta, S., Christos H. P., Umesh V.: Algorithms. McGraw-Hill, Inc., (2016).

3. Faure, N., Chr´etienne, P., Gourdin, E., Sourd, F.: Biclique Completion Problems for Multicast Network Design. Discrete Optimization, 4, pp. 360–377 (2007).

4. Gualandi, S.: k-Clustering Minimum Biclique Completion via a Hybrid CP and SDP Approach. In: van Hoeve, W-J., Hooker, J.N. (eds.) Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. LNCS, vol. 5547, pp. 87–101. Springer Berlin Heidelberg (2009)

5. Gualandi, S., Maffioli, F., Magni, C.: A Branch-and-Price Approach to k-Clustering Minimum Biclique Completion Problem. International Transactions in Operational Research, 20, pp. 101–117 (2013)

6. Hifi, M., Moussa, I., Saadi, T., Saleh, S.: An Adaptive Neighborhood Search for k-Clustering Minimum Bi-clique Completion Problems. In Modelling, Computation and Optimization in Information Systems and Management Sciences, Springer Inter- national Publishing, pp. 15–25, (2015).

7. Hifi, M., Michrafy, M.: A Reactive Local Search-Based Algorithm for the Disjunc- tively Knapsack Problem. Journal of the Operational Research Society, 57, pp. 718– 726 (2006)

8. Johnson, D.S.: Approximation Algorithms for Combinatorial Problems. Journal of Computer and System Sciences, 9, pp. 256-278 (1974)

9. Shaw, P.: Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. In: M. Maher, M., Puget, J.F. (eds.) Principles and Practice of Constraint Programming – CP98. LNCS, vol. 1520, pp. 417–431. Springer Berlin Heidelberg (1998)