Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 346 (2019) 667–676

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Approximations for Restrictions of The Budgeted and Generalized Maximum Coverage Problems

Breno Piva[1]

Departamento de Computação
Universidade Federal de Sergipe
São Cristóvão SE, Brasil

Abstract

In this paper we present approximation preserving reductions from the Budgeted and Generalized Maximum Coverage Problems to the Knapsack Problem with Conflict Graphs. The reductions are used to yield Polynomial Time Approximation Schemes for special classes of instances of these problems. Using these approximation schemes, the existence of pseudo-polynomial algorithms are proven and, in more particular cases, these algorithms are shown to have polynomial time complexity. Moreover, the characteristics of the instances that admit these algorithms are analyzed.

Keywords: Budgeted Maximum Coverage, Generalized Maximum Coverage, Knapsack with Conflict Graph, Approximation Algorithm, Pseudo-polynomial Algorithm

1 Introduction

The Budgeted Maximum Coverage Problem (BMC) [6] is a generalization of the classical Maximum Coverage Problem where an additional knapsack constraint is given. Formally, an instance of BMC is given by a collection of sets $S = \{s_1, s_2, ..., s_m\}$ defined over a domain of elements $X = \{x_1, x_2, ..., x_n\}$, a cost function $\delta : S \to \mathbb{Z}$, a budget L and a weight function $\omega : X \to \mathbb{Z}$. A solution is a collection $S' \subseteq S$ with total cost not greater than L and maximizing the total weight of the covered elements. For clarity reasons we present below an integer programming (IP) formulation for BMC.

[1] Email:brenopiva@dcomp.ufs.br

[2] The author would like to thank the anonymous referees for their invaluable suggestions.

https://doi.org/10.1016/j.entcs.2019.08.058
1571-0661/

$$
\begin{aligned}
&(\text{BMC}) && z = \max \sum_{x \in X} \omega(x) p_x && && (1)\\
&\text{subject to} && \sum_{s \in S} \delta(s) q_s \le L && && (2)\\
& && p_x \le \sum_{s \in S: x \in s} q_s && \forall\, x \in X && (3)\\
& && p_x, q_s \in \{0,1\}; x \in X, s \in S && && (4)
\end{aligned}
$$

In the IP model for BMC variables p_x are binary and indicate whether element $x \in X$ is covered or not and variables q_s, also binary, indicate if set $s \in S$ is in the solution or not. The inequality (2) is a knapsack inequality and defines that the cost of the solution is no greater than the budget. Meanwhile, restrictions (3) relate the two kinds of variables, defining that an element can only be covered if a set containing it is in the solution.

This problem has important applications in several areas including: recovery from power outage [5], location of network monitors [10], automatic text summarization [11], software test-case prioritization [12], recruitment for participatory sensing data collections [9] and news recommendation systems [7] just to mention a few.

The Generalized Maximum Coverage Problem (GMC) [3] generalizes BMC by giving different weights and costs to items depending on the set used to cover it. Formally the problem can be defined as: given a collection of sets $S = \{s_1, s_2, ..., s_m\}$ defined over a domain of elements $X = \{x_1, x_2, ..., x_n\}$, a cost function $\delta : S \to \mathbb{Z}$, a second cost function $\delta : X \times S \to \mathbb{Z}$, a budget L and a weight function $\omega : X \times S \to \mathbb{Z}$, find a triple (S', X', f) where $S' \subseteq S$, $X' \subseteq X$ and f is an assignment function from X' to S' where the sum of the costs of sets in S' plus the sum of the costs of covering the elements is no greater than L and maximizing the weighted sum of elements coverage. An IP model for GMC is presented below.

$$
\begin{aligned}
&(\text{GMC}) && z = \max \sum_{s \in S} \sum_{x \in s} \omega(x,s) p_{xs} && && (5)\\
&\text{subject to} && \sum_{s \in S} \delta(s) q_s + \sum_{s \in S} \sum_{x \in s} \delta(x,s) p_{xs} \le L && && (6)\\
& && p_{xs} \le q_s && \forall\, s \in S, \forall\, x \in s, && (7)\\
& && \sum_{s \in S: x \in s} p_{xs} \le 1, && \forall x \in X && (8)\\
& && p_{xs}, q_s \in \{0,1\}; x \in X, s \in S && && (9)
\end{aligned}
$$

Binary variables p_{xs} in model (GMC) define whether an element $x \in X$ is covered by a set $s \in S$. The variables q_s are also binary and indicate if a set $s \in S$ is in the solution. Inequality (6) enforces that the sum of costs of covering the elements plus the cost of the selected sets must fit in the budget. Restrictions (7) state that an element $x \in X$ can only be covered by a set $s \in S$ if s is in the solution. Finally, inequalities (8) define that at most one set in S can cover an element in X.

Besides being a generalization of BMC (since any instance of BMC can be viewed as an instance of GMC where the cost of any element been covered is zero and the weight of an element is always the same regardless of the set covering it) and thus having all the applications BMC has, GMC has applications in finding overlapping communities in graphs and social networks [4], planning sightseeing tours [2], link-based sensor selection [1] etc.

The approximation threshold proved by [6] and the obvious reduction from the Maximum Coverage Problem prove that BMC is strongly $\mathcal{NP}$-hard and does not admit a Polynomial Time Approximation Scheme (PTAS) unless $\mathcal{P}= \mathcal{NP}$. Since GMC is a generalization of BMC these innaproximability results extend to it.

The existence of a PTAS or pseudo-polynomial algorithms for the two mentioned problems is unlikely (unless $\mathcal{P}= \mathcal{NP}$). It is possible, however, that such algorithms exist for special cases of the problems. In the current work we are going to show that such group of instances do exist and give necessary and sufficient conditions to identify subsets of these instances.

In the next sections we are going to present the polynomial time approximation preserving reductions from BMC and GMC to the Knapsack Problem with Conflict Graphs (KCG) and prove that they yield approximation algorithms for restrictions of these problems (Section 2). In Section 3 the characteristics of subsets of the instances of BMC and GMC that satisfy the conditions of the algorithms are displayed. Section 4 describes exact algorithms obtained from the approximations for special cases. Finally, in Section 5 we present some conclusions and future directions of research.

2 Reductions and Approximations

There are several knapsack problems with side constraints, one of these problems is KCG. An instance of this problem is given by a graph $G = (V, E)$ where its vertices correspond to the knapsack items and its edges correspond to conflicts between items, a budget B, a cost function $\delta : V \to \mathbb{Z}$ and a weight function $\omega : V \to \mathbb{Z}$. The goal is to select a set of items $V' \subseteq V$ with maximum weight and respecting the budget and conflict constraints. Another way of looking at this problem is as a maximum weight independent set problem with knapsack constraint.

Since KCG can be viewed as a generalization of the maximum weight independent set problem, it is strongly $\mathcal{NP}$-hard and does not admit a PTAS. However it was proven in [8] that KCG has an FPTAS for instances where the conflict graphs are weakly chordal or have bounded treewidth.

In this section we are going to describe a reduction R that given an instance $I_1 = (S, X, L, \delta_1, \omega_1)$ of BMC produces $R(I_1) = I_2 = (G = (V, E), B, \delta_2, \omega_2)$, an instance of KCG. Our hope is that using this reduction it is possible to obtain an approximation algorithm for BMC.

Let the set of vertices of G be $V = V_1 \cup V_2 \cup V_3$ where $V_1 = \{u_s | s \in S\}$, that is, there is a vertex for each set in collection S representing the presence of that set in the solution. $V_2 = \{w_s | s \in S\}$, meaning there is also a vertex for each set in S representing the absence of that set in the solution. At last, $V_3 = \{v_{xs} | s \in S, x \in s\}$, i.e., for each element x of X there is a vertex for each set of $s \in S$ where $x \in s$, these vertices represent that x is covered by that set.

The set of edges is defined as $E = E_1 \cup E_2 \cup E_3$ where $E_1 = \{(u_s, w_s) | u_s \in V_1, w_s \in V_2$ and $s \in S\}$, that is, the vertices that represent the presence and the absence of the same set of S in the solution cannot be part of the solution simultaneously. $E_2 = \{(w_s, v_{xs}) | w_s \in V_2, v_{xs} \in V_3, x \in X$ and $s \in S\}$ meaning that if a set in collection S is absent in the solution, it cannot be used to cover any element of X. Finally, $E_3 = \bigcup_{x \in X} \{(v_{xr}, v_{xs}) | v_{xr}, v_{xs} \in V_3$ and $r, s \in S\}$ meaning that at most one of the sets in S having an element $x \in X$ will count as covering X. The definition of E_3 also means that for each $x \in X$ there is a clique in G with vertices in V_3 representing the sets $s \in S$ where $x \in s$.

The cost function is given by $\delta_2(u_s) = \delta_1(s)$ if $u_s \in V_1$ and $\delta_2(v) = 0$ if $v \notin V_1$. Budget B is equal to L. Finally, the weight function is given by $\omega_2(v_{xs}) = \omega_1(x)$ if $v_{xs} \in V_3$ and $\omega_2(u_s) = 1 + \sum_{x \in s} \omega_1(x)$ if $u_s \in V_1 \cup V_2$.

Figure 1 presents an instance of KCG obtained from reducing instance $\Upsilon = (S, X, L, \delta_1, \omega_1)$ of BMC where $S = \{s_1 = \{x_1\}, s_2 = \{x_1, x_2, x_3\}, s_3 = \{x_1, x_2, x_3\}, s_4 = \{x_1, x_2, x_4\}\}$, $X = \{x_1, x_2, x_3, x_4\}$, $L = 11$, $\delta_1 = \{(s_1, 2), (s_2, 9), (s_3, 6), (s_4, 4)\}$ and $\omega_1 = \{(x_1, 2), (x_2, 5), (x_3, 6), (x_4, 3)\}$.

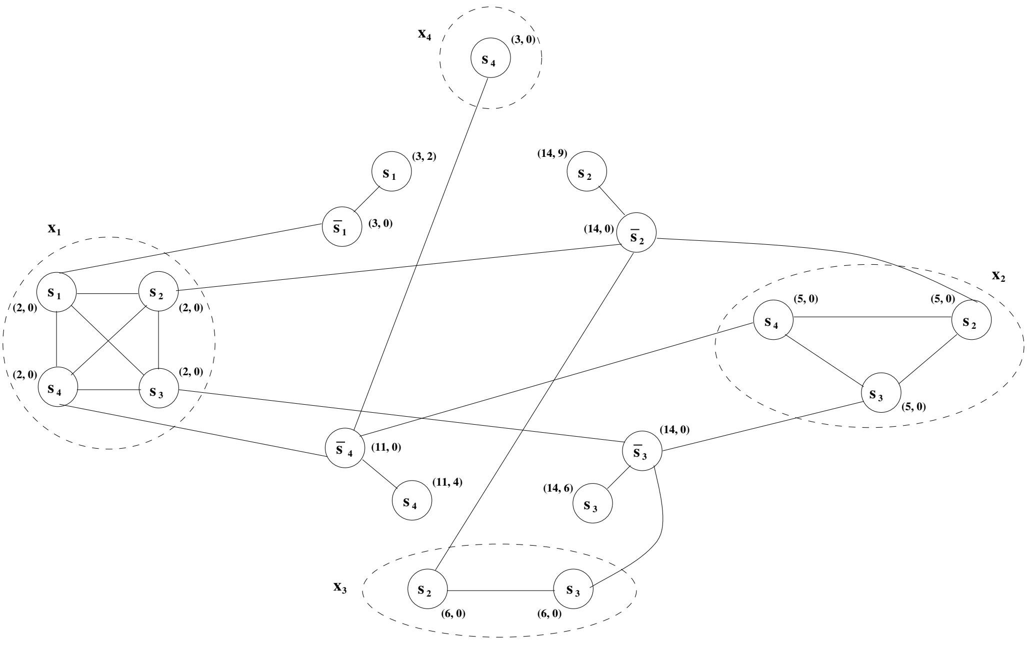

Fig. 1. Instance of KCG obtained from reduction R applied to Υ. The pair of numbers in parentheses represent, respectively the weight and cost of the vertex. The vertices with over-lined labels are the vertices in set V_2 (representing the absence of that set in the solution) and their neighbor with degree one are the vertices in set V_1 (representing the presence of that set in the solution). The remaining vertices compose set V_3. The dashed ellipses put in evidence the cliques representing the vertices covering each element of X.

Proposition 2.1 *An instance I_1 of* BMC *has optimal solution with value $OPT(I_1)$ if and only if $I_2 = R(I_1)$, instance of* KCG*, has optimal solution with value $OPT(I_2) = \Gamma + OPT(I_1)$ where $\Gamma = |S| + \sum_{s \in S} \sum_{x \in s} \omega_1(x)$.*

Proof. Let $\varsigma_1 \subseteq S$ be an optimal solution for I_1 and let $\Xi_1 \subset X$ be the set of elements covered by this optimal solution, thus $OPT(I_1) = \sum_{x \in \Xi_1} \omega_1(x)$. Consider the following solution of KCG: take into the solution all the vertices of subset V_1 corresponding to elements of ς_1 and vertices of subset V_2 corresponding to elements of $S \backslash \varsigma_1$. The sum of the weights of these vertices is equal to $|S| + \sum_{s \in S} \sum_{x \in s} \omega_1(x) = \Gamma$. Now, for each element $x \in \Xi_1$ include in the solution a vertex from V_3 corresponding to an element of ς_1 that covers x. Since all the vertices in V_3 corresponding to sets covering x have weight $\omega_1(x)$, the total weight of the solution is $\Gamma + \sum_{x \in \Xi_1} \omega_1(x)$. Moreover, we must notice that there is no conflict between the chosen vertices since only one vertex corresponding to a set covering each x was chosen (no violation of set E_3), only the vertices in V_2 corresponding to sets not in ς_1 were included in the solution (no violation of set E_2) and, finally, the set of vertices chosen from sets V_1 and V_2 have no intersection (no violation of set E_1).

Now to show the converse, let $\varsigma_2 \in V$ be an optimal solution for $I_2 = R(I_1)$ with value $OPT(I_2)$. First we must notice that any optimal solution of I_2 must contain exactly one element of each pair of vertices where one is in V_1 and the other in V_2, indicating the presence or absence of a set from S in the solution. The reason why exactly one of them must be in the solution is that they cannot be both in the solution (set E_1) and each one of them weight more than all its neighbors combined (except its pair which weights the same). The presence of these vertices in any optimal solution contribute with Γ to the value of the solution. The remaining of the solution's value comes from the weights of vertices in V_3, let us denote this set of vertices by V_3'. Therefore, if $OPT(I_2) = \Gamma + OPT(I_1)$, that means that $\sum_{v_{xs} \in V_3'} \omega_2(v) = OPT(I_1)$. Notice that from E_3, two vertices representing sets in S covering the same element of X cannot be in the solution at the same time. Hence, each vertex in V_3' covers a different element of X. Thus, there is also a solution of I_1 covering the same set and thus having value $OPT(I_1)$.

□

A reduction from GMC to KCG can be obtained simply by altering the weight and cost functions in the reduction. In GMC there is a cost function $\delta_1 : S \to \mathbb{Z}$ for each element of S but there is also a cost function $\delta_1 : X \times S \to \mathbb{Z}$ for an element $x \in X$ been covered by an element of S. Likewise, the weight function $\omega_1 : X \times S \to \mathbb{Z}$ indicates the weight of an element $x \in X$ been covered by an element of S. Define $\delta_2(u_s) = \delta_1(s)$ if $u_s \in V_1$ and $\delta_2(v_{xs}) = \delta_1(x, s)$ if $v_{xs} \in V_3$. $\delta_2(v) = 0$ otherwise. The weight function is defined as $\omega_2(u_s) = 1 + \sum_{x \in s} \omega_1(x, s)$ if $u_s \in V_1 \cup V_2$ and $\omega_2(v_{xs}) = \omega_1(x, s)$ if $v_{xs} \in V_3$.

Next we show that the reduction R described above preserves approximation, thus yielding a polynomial time approximation algorithm for a subset of BMC's instances.

Proposition 2.2 *If there is a $(1-\varepsilon)$-approximation polynomial time algorithm for* KCG*, then there is a polynomial time algorithm for* BMC *that for any given instance I_1 produces a solution with value $z_1 \geq (1-\varepsilon)OPT(I_1) - \varepsilon\Gamma$.*

Proof. First of all, it is easy to notice that reduction R is a polynomial time reduction. Then, from Proposition 2.1 we know that an instance I_1 of BMC with optimal solution value $OPT(I_1)$ has a corresponding instance $I_2 = R(I_1)$ of KCG with optimal solution value $OPT(I_2) = \Gamma + OPT(I_1)$. Besides, any solution ς_2 of I_2 can be split in the vertices in ς_2 that are in V_3 and the vertices in ς_2 that are in $\{V_1 \cup V_2\}$. The vertices in $\{V_1 \cup V_2\}$ contribute with value at most Γ while the vertices in V_3 correspond to a solution for I_1 with value z_1. Hence, any solution ς_2 of I_2 has a value $z_2 \leq z_1 + \Gamma$. Therefore, if there is a $(1-\varepsilon)$-approximation for I_2, we have $(1-\varepsilon)OPT(I_2) \leq z_2 \leq OPT(I_2)$, hence $(1-\varepsilon)(\Gamma + OPT(I_1)) \leq z_2 \leq z_1 + \Gamma \leq \Gamma + OPT(I_1)$ and then, $(1-\varepsilon)OPT(I_1) - \varepsilon\Gamma \leq z_1 \leq OPT(I_1)$. □

Proposition 2.3 *Let I_1 be an instance of* BMC *and let $I_2 = R(I_1)$ be an instance of* KCG *obtained from I_1 using the reduction above. If the graph in I_2 is weakly chordal or has bounded treewidth then there is a family of approximation algorithms for I_1 with approximation guarantee $(1-\varepsilon)OPT(I_1) - \varepsilon\Gamma$ and running time that is polynomial in $|I_1|$ and $\frac{1}{\varepsilon}$. Moreover, this family of approximation algorithms for I_1 defines a* PTAS*.*

Proof. The existence of a family of approximation algorithms for I_1 comes straight from the existence of an Fully Polynomial Time Approximation Scheme (FPTAS) for KCG when the conflict graph is weakly chordal or has bounded degree [8] and from Proposition 2.2. Since $\Gamma \leq (M-1)OPT(I_1)$ where $M = 2|S|\max(|s|)\max(\omega_1(x))$, we can define $\varepsilon' = \frac{\varepsilon}{M}$. Hence, there is a solution z_1 of I_1 satisfying $(1-\varepsilon')OPT(I_1) - \varepsilon'\Gamma \leq z_1 \leq OPT(I_1)$. By replacing ε' for $\frac{\varepsilon}{M}$ we get $(1-\varepsilon)OPT(I_1) \leq z_1 \leq OPT(I_1)$. □

Similar results as the ones in Proposition 2.2 and Proposition 2.3 can be achieved for GMC using the same ideas and the reduction described above.

3 Characterization of BMC and GMC Instances

From Proposition 2.3 we know there is an approximation algorithm for BMC whenever the corresponding KCG instance is weakly chordal or have bounded treewidth. However it would be far more useful to know the characteristics of BMC instances that have corresponding KCG instances of these classes. In the current section we define what a BMC instance should look like for having a corresponding instance of KCG with weakly chordal conflict graph.

Proposition 3.1 *The graphs generated by reduction R cannot contain an anti-hole.*

Proof. Let $I_2 = (G, B, \delta, \omega)$ be an instance of KCG obtained from R. First of all, notice that all the vertices in set V_1 have degree one and, therefore, can not be in an anti-hole. Now, let us try to construct an anti-hole using the vertices in V_2 and V_3.

The first case we consider is that of an anti-hole composed only by vertices in V_3. Notice that the vertices in V_3 are grouped in cliques corresponding to the sets in S covering the same element of X and vertices from two different cliques are non-adjacent. Without loss of generality, let v_1, v_2 and v_3, be an anti-path composed by vertices in V_3 and part of an anti-hole. We have that v_1 and v_3 must be in the same clique and v_2 in a different one. Let v_4 be a fourth vertex in the anti-hole, following v_3 in the anti-cycle. Since it follows v_3 it must be from a different clique but since it is not a successor nor a predecessor of v_1, v_4 and v_1 must be adjacent and therefore, must be in the same clique, but v_1 and v_3 are in the same clique. Contradiction.

Now suppose the anti-hole have vertices from V_2. If we take any three vertices of any anti-hole, there is at least one edge connecting them, moreover, V_2 induces an independent set in G. Therefore, any anti-hole can have at most two vertices from V_2 and if there are two vertices, they must be consecutive. Since an odd-hole must have at least five vertices, if there is an anti-hole in G it must have vertices from both V_2 and V_3. Also notice that any vertex in V_3 has exactly one edge connecting it to vertices in V_2. Suppose without loss of generality that v_1, v_2 and v_3 compose an anti-path that is part of an anti-hole where $v_1 \in V_2$ and $v_2, v_3 \in V_3$, therefore, v_2 and v_3 must be from different cliques and we must have $v_1 = w_{s_k}$, $v_3 = v_{x_i s_k}$ and $v_2 = v_{x_j s_l}$ for some $x_i, x_j \in X$, $s_k, s_l \in S$, $x_i \neq x_j$ and $s_k \neq s_l$. Now, let v_4 be a fourth vertex in the anti-hole, following v_3 in the anti-cycle. We have $v_4 \notin V_2$ since if there are two vertices from V_2 they must be consecutive, hence $v_4 \in V_3$. Also, we must have $v_4 = v_{x_j s_k}$ and thus, $(v_2, v_4) \in E_3$ and $(v_1, v_4) \in E_2$. Finally, let v_5 be a fifth vertex in the anti-hole, following v_4 in the anti-cycle. Suppose $v_5 \in V_2$ then if $v_5 = w_{s_p}$, it must be true that $v_2 = v_{x_j s_p}$ and $v_3 = v_{x_i s_p}$, but since $v_3 = v_{x_i s_k}$ and $v_2 = v_{x_j s_l}$ and $s_k \neq s_l$ this is not possible. Suppose then $v_5 \in V_3$, hence v_2, v_3 and v_5 must be in the same clique which is also impossible. Contradiction. □

Consider the following two conditions regarding an instance $I_1 = (S, X, L, \delta, \omega)$ of BMC and the existence of a hole in the conflict graph of an instance $I_2 = R(I_1)$ of KCG.

$$\exists\ s_i, s_j \in S \text{ where } |s_i \cap s_j| \geq 2 \tag{10}$$

$$\begin{aligned}&\forall\ s_i, s_j \in S, |s_i \cap s_j| \leq 1 \text{ and } \exists\ S' \subseteq S,\ |S'| > 2 \text{ where there is an order}\\ &s'_0, s'_1, ..., s'_{(|S'|-1)} \text{ of its elements and } s'_l \cap s'_{(l+1)\text{mod}|S'|} \neq \emptyset \\ &\qquad \forall\ l \in \{0, ..., (|S'|-1)\}\end{aligned} \tag{11}$$

Proposition 3.2 *The conflict graph of an instance I_2 of KCG obtained from $R(I_1)$ contain a hole if and only if condition (10) or (12) is satisfied.*

Proof. ($\Rightarrow$) Suppose I_1 satisfies condition (10). Then, there are two sets s_i and s_j in collection S such that $|s_i \cap s_j| \geq 2$, let us say that $x_k, x_l \in s_i \cap s_j$. Therefore, in graph G of I_2, the sequence of vertices $v_{x_k s_i}$, $v_{x_k s_j}$, w_{s_j}, $v_{x_l s_j}$, $v_{x_l s_i}$, w_{s_i}, $v_{x_k s_i}$, where $v_{x_k s_i}$, $v_{x_k s_j}$, $v_{x_l s_j}$, $v_{x_l s_i} \in V_3$ and w_{s_j}, $w_{s_i} \in V_2$, induces a hole.

Assume, on the other hand, that I_1 satisfies condition (12). Denote by x_{ij} the unique element in $s_i \cap s_j$ for some pair of consecutive sets s_i, s_j in the ordering of S'. Then, the following sequence of vertices where $w_{s'_0}$, $w_{s'_1}$, ..., $w_{s'_{(|S'|-1)}} \in V_2$ and $v_{x_{01} s'_0}$, $v_{x_{01} s'_1}$, $v_{x_{12} s'_1}$, $v_{x_{12} s'_2}$, ..., $v_{x_{(|S'|-1)0} s'_{(|S'|-1)}}$, $v_{x_{(|S'|-1)0} s'_0} \in V_3$ induces a hole in G: $v_{x_{(|S'|-1)0} s'_0}$, $w_{s'_0}$, $v_{x_{01} s'_0}$, $v_{x_{01} s'_1}$, $w_{s'_1}$, ..., $w_{s'_{(|S'|-1)}}$, $v_{x_{(|S'|-1)0} s'_{(|S'|-1)}}$, $v_{x_{(|S'|-1)0} s'_0}$.

($\Leftarrow$) Let us first consider some facts about graph G in I_2 : (1) all vertices in V_1 have degree one, therefore, no vertex in V_1 can be in hole. (2) V_3 can be partitioned in cliques; there is one clique partition for each $x \in X$ and the vertices in a same clique correspond to the sets $s \in S$ such that $x \in s$. (3) there is no edge $(v_{x_i s_k}, v_{x_j s_l}) \in E$ where $v_{x_i s_k}, v_{x_j s_l} \in V_3$ and $x_i \neq x_j$, i.e., there is no edge connecting vertices from different clique partitions. (4) V_2 induces an independent set in G and if a vertex in V_2 have two vertices in V_3 as neighbors, then they must be in different clique partitions of V_3, i.e, if $(w_s, v_{x_i s}) \in E$ and $(w_s, v_{x_j s}) \in E$, where $w_s \in V_2$ and $v_{x_i s}, v_{x_j s} \in V_3$, then $x_i \neq x_j$. (5) any vertex in V_3 has exactly one neighbor in V_2. (6) since any vertex u in a hole H has exactly two neighbors v, w and v and w are not adjacent, H can have at most two vertices from the same clique partition of V_3. (7) for the same reason as in fact (6) and from facts (5) and (6), any hole H containing vertices from a clique partition of V_3 must contain exactly two vertices from that partition and they are consecutive. (8) from (4), if a hole H contain a vertex $u \in V_2$ then it must be true that, its two neighbors, $v, w \in V_3$. (9) from facts (3) and (7), if a hole H contain a pair of vertices $u, v \in V_3$, the pair must be preceded and succeeded by vertices in V_2. (10) from facts (7), (8) and (9) we know that any hole in G must have vertices from both V_2 and V_3. Moreover, the vertices in V_3 can only be taken in pairs and each pair must be preceded and succeeded by vertices in V_2 which in turn must have two vertices in V_3 as its neighbors. Therefore, any hole in G is composed by a sequence of two vertices in V_3 (from the same clique partition) and one in V_2, thus having a number of vertices that is a multiple of 3.

Suppose graph G in I_2 has a hole H. We are going to consider two cases: $|H| \leq 6$ and $|H| > 6$. If $|H| \leq 6$, then $|H| = 6$ since the number of vertices must be a multiple of 3, there must be a sequence of vertices $v_{x_k s_i}$, $v_{x_k s_j}$, w_{s_j}, $v_{x_l s_j}$, $v_{x_l s_i}$, w_{s_i}, $v_{x_k s_i}$, where $v_{x_k s_i}$, $v_{x_k s_j}$, $v_{x_l s_j}$, $v_{x_l s_i} \in V_3$ and w_{s_j}, $w_{s_i} \in V_2$, hence condition (10) is satisfied.

If $|H| > 6$, since $|H|$ must be a multiple of 3, $|H| \geq 3k$ with $k > 2$ and since any hole in G is composed by a sequence of two vertices in V_3 and one in V_2, H must be composed by a sequence of vertices $v_{x_{(|S'|-1)0} s'_{(|S'|-1)}}$, $v_{x_{(|S'|-1)0} s'_0}$, $w_{s'_0}$, $v_{x_{01} s'_0}$, $v_{x_{01} s'_1}$, $w_{s'_1}$, ..., $w_{s'_{(|S'|-1)}}$, $v_{x_{(|S'|-1)0} s'_{(|S'|-1)}}$ where $k = |S'| > 2$, $w_{s'_0}$, $w_{s'_1}$, ..., $w_{s'_{(|S'|-1)}} \in V_2$ and $v_{x_{01} s'_0}$, $v_{x_{01} s'_1}$, $v_{x_{12} s'_1}$, $v_{x_{12} s'_2}$, ..., $v_{x_{(|S'|-1)0} s'_{(|S'|-1)}}$, $v_{x_{(|S'|-1)0} s'_0} \in V_3$. Therefore, condition (12) is satisfied.

□

From Propositions 3.1 and 3.2 we can deduce that an instance I_1 of BMC has a corresponding instance $R(I_1)$ of KCG that has weakly chordal conflict graph if and only if I_1 does not satisfy condition (10) nor condition (12).

Notice that the generalization provided by GMC has no influence in the structure of the graph obtained from the reduction to KCG. Therefore, all the results for BMC regarding the characteristics of the instances also hold for GMC.

4 Exact Algorithms

The PTAS obtained from the reduction in Section 2 can be used to produce a pseudo-polynomial time algorithm for a restriction of BMC, i.e. instances of BMC with corresponding KCG instances with graphs that are weakly chordal or have bounded treewidth. This fact is described by Proposition 4.1.

Proposition 4.1 *The algorithm from Proposition 2.3 can be used to produce a pseudo-polynomial time algorithm for that restriction of* BMC.

Proof. Let $I_2 = R(I_1)$ be an instance of KCG obtained from the application of reduction R to an instance I_1 of BMC. Let $OPT(I_1)$ be the value of an optimal solution for I_1 and $OPT(I_2)$ be the value of an optimal solution for I_2. From the definition of R we know that $\Gamma = |S| + \sum_{v \in V_1} \omega_2(v) \geq OPT(I_1)$. Thus we can define a bound $B > \Gamma \geq OPT(I_1)$ and we can define $\varepsilon = \frac{1}{2B}$ in the algorithm from Proposition 2.3. Since the solution of the algorithm has value $z_1 \geq (1-\varepsilon)OPT(I_1) - \varepsilon\Gamma$, we have $z_1 \geq OPT(I_1) - \varepsilon OPT(I_1) - \varepsilon\Gamma > OPT(I_1) - \varepsilon B - \varepsilon B = OPT(I_1) - 1$. Therefore z_1 must be equal to $OPT(I_1)$.

Since the complexity time of the algorithm is polynomial in $|I_1|$ and $\frac{1}{\varepsilon}$, with $\varepsilon = \frac{1}{2B}$ it is polynomial in the size of a unary encoding of I_1 thus, pseudo-polynomial. $\square$

A restriction of special interest is the one where the weights of the elements are bounded by some constant k that is independent of the instance. If this restriction applies, then the algorithm is actually polynomial in $|I_1|$ as shown in Proposition 4.2. This case has application, for instance, in the time-aware test-case prioritization problem [12] where the weights are unitary.

Proposition 4.2 *If the weights of the elements in a instance of* BMC *are bounded by some constant k that is independent on the instance, then the algorithm from Proposition 2.3 can be used to produce a polynomial time algorithm for that restriction of* BMC.

Proof. If the weights of the elements are bounded by a constant k then $\Gamma \leq k|S||X| + |S|$ and in the pseudo-polynomial algorithm from Proposition 4.1 we can define $B = k|S||X| + |S| + 1$ and then $\varepsilon = \frac{1}{2B}$ is polynomial in the size of a binary encoding of I_1. $\square$

The same results can be achieved for GMC by making the proper modifications in Proposition 4.1 and Proposition 4.2.

5 Conclusions and Future Work

In the present paper we presented polynomial time reductions from BMC and GMC to KCG. These reductions allowed us to use an existing FPTAS for a restriction of KCG namely instances with weakly chordal graphs or having graphs with bounded treewidth, to obtain polynomial time approximation scheme, pseudo-polynomial exact and polynomial exact algorithms for restrictions of BMC and GMC. We determine a characterization for the instances of BMC and GMC yielding an instance of KCG with weakly chordal graphs.

As a future work we leave the characterization of instances of BMC and GMC that have a corresponding instance of KCG that have conflict graphs with bounded treewidth.

References

[1] Charu C. Aggarwal, Amotz Bar-Noy, and Simon Shamoun. On sensor selection in linked information networks. *Computer Networks*, 126:100 – 113, 2017.

[2] Igo Ramalho Brilhante, Jose Antonio Macedo, Franco Maria Nardini, Raffaele Perego, and Chiara Renso. On planning sightseeing tours with tripbuilder. *Information Processing & Management*, 51(2):1 – 15, 2015.

[3] Reuven Cohen and Liran Katzir. The generalized maximum coverage problem. *Information Processing Letters*, 108(1):15 – 22, 2008.

[4] Esther Galbrun, Aristides Gionis, and Nikolaj Tatti. Overlapping community detection in labeled graphs. *Data Mining and Knowledge Discovery*, 28(5):1586–1610, Sep 2014.

[5] Sudipto Guha, Anna Moss, Joseph (Seffi) Naor, and Baruch Schieber. Efficient recovery from power outage (extended abstract). In *Proceedings of the Thirty-first Annual ACM Symposium on Theory of Computing*, STOC '99, pages 574–582, New York, NY, USA, 1999. ACM.

[6] Samir Khuller, Anna Moss, and Joseph (Seffi) Naor. The budgeted maximum coverage problem. *Information Processing Letters*, 70(1):39 – 45, 1999.

[7] Lei Li, Dingding Wang, Tao Li, Daniel Knox, and Balaji Padmanabhan. Scene: A scalable two-stage personalized news recommendation system. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 125–134, New York, NY, USA, 2011. ACM.

[8] Ulrich Pferschy and Joachim Schauer. Approximation of knapsack problems with conflict and forcing graphs. *Journal of Combinatorial Optimization*, 33(4):1300–1323, May 2017.

[9] Sasank Reddy, Deborah Estrin, and Mani Srivastava. Recruitment framework for participatory sensing data collections. In Patrik Floréen, Antonio Krüger, and Mirjana Spasojevic, editors, *Pervasive Computing*, pages 138–155, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[10] Kyoungwon Suh, Yang Guo, Jim Kurose, and Don Towsley. Locating network monitors: Complexity, heuristics, and coverage. *Computer Communications*, 29(10):1564 – 1577, 2006. Monitoring and Measurements of IP Networks.

[11] Hiroya Takamura and Manabu Okumura. Text summarization model based on maximum coverage problem and its variant. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 781–789, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[12] Lu Zhang, Shan-Shan Hou, Chao Guo, Tao Xie, and Hong Mei. Time-aware test-case prioritization using integer linear programming. In *Proceedings of the Eighteenth International Symposium on Software Testing and Analysis*, ISSTA '09, pages 213–224, New York, NY, USA, 2009. ACM.