The Generalized Max-Controlled Set Problem

Ivairton M. Santos [1]
Inst. Universitário do Araguaia, Univ. Federal de Mato Grosso-UFMT
Pontal do Araguaia-MT, Brazil

Carlos A. Martinhon [2] and Luiz S. Ochi [3]
Inst. de Computação, Universidade Federal Fluminense-UFF
Niterói-RJ, Brazil

Abstract

In this work we deal with sandwich graphs $G = (V, E)$ and present the notion of vertices f-controlled by a subset $M \subseteq V$. We introduce the GENERALIZED MAX-CONTROLLED SET PROBLEM (GMCSP), where gaps and positive weights are associated to each vertex of V. In this case, the objective is find a sandwich graph G in order to maximize the sum of the weights associated to all vertices f-controlled by M. We present a $\frac{1}{2}$-approximation algorithm for the GMCSP and a new procedure for finding feasible solutions based on a linear relaxation. The best solution is then used as starting point in a local search procedure (Tabu Search with Path Relinking). Finally, we present some computational results and compare the performance of our heuristics with the optimum solution value of some instances of the problem.

Keywords: Sandwich Graph Problems, Approximation Algorithms, Construction Heuristics, Tabu Search, Path Relinking.

1 Introduction

Given two graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ such that $E_1 \subseteq E_2$, we say that $G = (V, E)$, where $E_1 \subseteq E \subseteq E_2$, is a *sandwich graph* for some property Π if $G = (V, E)$ satisfies Π. A *sandwich problem* consists of deciding whether there exists some sandwich graph satisfying Π. We denote *optional* and *fixed edges*, the edges belonging, respectively, to $E_2 \backslash E_1$ and E_1 (see [2]).

Given an undirected graph $G = (V, E)$ and a set of vertices $M \subseteq V$, a vertex $i \in V$ is said to be *controlled* by M if $|N_G[i] \cap M| \geq |N_G[i] \cap U|$, where

[1] Email: `ivairton@ufmt.br`
[2] Email: `mart@dcc.ic.uff.br`
[3] Email: `satoru@dcc.ic.uff.br`

$N_G[i] = \{i\} \cup \{j \in V | (i,j) \in E\}$ and $U = V \backslash M$. The set M defines a *monopoly* in G if every vertex $i \in V$ is controlled by M. Therefore, if $cont(G, M)$ denotes the set of vertices controlled by M in G, M will be a monopoly in G if and only if $cont(G, M) = V$.

Prior to define the GENERALIZED MAX-CONTROLLED SET PROBLEM (GMCSP), we first consider the MONOPOLY VERIFICATION PROBLEM (MVP) and the MAX-CONTROLLED SET PROBLEM (MCSP) as defined in [3]. In the MVP, given a set $M \subseteq V$ and two graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$, where $E_1 \subseteq E_2$, the question is to decide whether there exists a set E such that $E_1 \subseteq E \subseteq E_2$ and M is a monopoly in $G = (V, E)$. If the answer of the MVP is "NO", we then consider the MCSP, whose goal is to find a set E such that $E_1 \subseteq E \subseteq E_2$ and the number of vertices controlled by M in $G = (V, E)$ is maximized. The MVP can be solved in polynomial time by formulating it as a network flow problem. Unfortunately, the MCSP is NP-hard, even for those instances where G_1 is an empty graph and G_2 is a complete graph (see [3] for details).

Now, consider the notion of f-controlled vertices (as introduced in Makino *et. al.*[3]) where f denotes a function on V. Thus, given a value $f_i \in \mathbb{Z}$, the vertex $i \in V$ is said to be f-*controlled* by M if and only if $|N_G[i] \cap M| - |N_G[i] \cap U| \geq f_i$. The constant f_i represents the gap necessary to f-control the vertex $i \in V$. Note, for instance, that if $f_i = -\infty$ ($f_i = +\infty$), vertex i is always (never) f-controlled by M.

Finally, if positive weights w_i are assigned to each $i \in V$, our goal in the GMCSP is to find a sandwich graph $G = (V, E)$ (where $E_1 \subseteq E \subseteq E_2$) in order to maximize the sum of the weights associated to all vertices f-controlled by M. The GMCSP is obviously NP-hard since it generalizes the MCSP (particular instance of the GMCSP where $f_i = 0$ and $w_i = 1$, $\forall\ i \in V$).

2 Reduction rules

Now we generalize the reduction rules as described in [3,4] for the MCSP. In this case, it suffice to change the definition of controlled by f-controlled vertices. As will be observed later, these rules will be helpful in the definition of a tight linear integer programming formulation for the GMCSP.

For $A, B \subseteq V$, we denote by $D(A, B) = \{(i, j) \in E_2 \backslash E_1 \mid i \in A,\ j \in B\}$, the set of optional edges with both ends belonging to A and B respectively. Two rules are used: a new edge set E_1^* is obtained by the union of E_1 and $D(M, M)$ and a new edge set E_2^* is obtained by removing $D(U, U)$ from E_2. Therefore, the set E in the sandwich graph G must satisfy: $E_1 \cup D(M, M) \subseteq E \subseteq E_1 \cup D(M, M) \cup D(U, M)$. For simplicity, assume from now on $E_1 = E_1^*$ (Reduction Rule 1) and $E_2 = E_2^*$ (Reduction Rule 2).

Prior to describe the remaining reduction rules, consider the following partition of V: we denote by M_{AC} and U_{AC}, respectively, the subset of vertices belonging to M and U which are always f-controlled by M in any sandwich graph G. Analogously, we denote by M_{NC} and U_{NC} the subset of vertices which are never f-controlled by M in any sandwich graph. Finally, we define the subsets $M_R = M \backslash (M_{AC} \cup M_{NC})$ and $U_R = U \backslash (U_{AC} \cup U_{NC})$.

In order to construct this partition of V it is sufficient to look at "worst case" assignments. Thus, after setting $E = E_2$ we identify all vertices belonging to $M_{AC} \cup U_{NC}$. Similarly, if we set $E = E_1$, we determine $U_{AC} \cup M_{NC}$. The reduction rules are listed in the sequel:

- Add to E_1 all edges belonging to $D(M_{AC} \cup M_{NC}, U_R)$ *(Rule 3)*;
- Remove from E_2 all edges belonging to $D(M_R, U_{AC} \cup U_{NC})$ *(Rule 4)*;
- Add/Remove at random all edges in $D(M_{AC} \cup M_{NC}, U_{AC} \cup U_{NC})$ *(Rule 5)*.

3 Generalized Max-Controlled Set Problem - GMCSP

3.1 A $\frac{1}{2}$-approximation algorithm

The $\frac{1}{2}$-approximation algorithm for the MCSP proposed by Makino *et.al.* [3] may be easily extended to the GMCSP (see Algorithm 1). First of all, consider W_1 and W_2 the sum of the weights associated to all vertices f-controlled by M in $G = (V, E)$ for $E = E_1$ and $E = E_2$ respectively. Hence, we have the following $\frac{1}{2}$-approximation algorithm for the GMCSP (see [5] for the proof):

Algorithm 1 : Based MYK - $\frac{1}{2}$-aproximation algorithm for the GMCSP

1: $W_1 \leftarrow$ Sum of the weights associated to all vertices f-controlled by M in the graph $G = (V, E)$, for $E = E_1$;
2: $W_2 \leftarrow$ Sum of the weights associated to all vertices f-controlled by M in the graph $G = (V, E)$, for $E = E_2$;
3: $z_{H1} \leftarrow max\{W_1, W_2\}$;

3.2 An Heuristic Based in the Linear Programming - LP

In order to describe our *Based LP* procedure for the GMCSP (Algorithm 2), we first introduce an integer programming formulation. We define binary variables $z_i \in \{0, 1\}$ for every $i \in V$, which determine whether vertex i is f-controlled or not by M. Binary variables x_{ij} are used to decide whether optional edges belonging to $E_2 \backslash E_1$ will be considered or not in the optimal sandwich graph. The constants $w_i \in \mathbb{Z}^+$ and $f_i \in \mathbb{Z}$ denote, respectively, the weight and gap of vertex $i \in V$. Binary constants $a_{ij} \in \{0, 1\}$ are associated to $(i,j) \in E_2$ with $a_{ij} = 1$ if and only if $i = j$ or $(i,j) \in E_2$. Further, we assume that $a_{ij} = a_{ji}, \forall i,j \in V$.

Consider the sets $M_R, M_{AC}, M_{NC}, U_R, U_{AC}, U_{NC}$ as described in the reduction rules. In order to define a tight formulation for the GMCSP, we first define some constants b_i, denoting the worst possible gap associated to each vertex $i \in M_R \cup U_R$. Initially, consider the following auxiliary expression:

$$b_i = \left| \sum_{j \in M} a_{ij}x_{ij} - \sum_{j \in U} a_{ij}x_{ij} - f_i \right|, \forall\, i \in M_R \cup U_R \tag{1}$$

The constants b_i are computed in the following way: if $i \in M_R$, we set $x_{ij} = 1,\ \forall\ (i,j) \in E_2 \backslash E_1$. Analogously, if $i \in U_R$, we set $x_{ij} = 0,\ \ \forall\ (i,j) \in E_2 \backslash E_1$. Obviously, in both cases we have $x_{ij} = 1,\ \forall\ (i,j) \in E_1$ (fixed edges). Then, we define the following tight linear integer programming formulation for the GMCSP:

$$z_{max} = maximize \left(\sum_{i \in V} w_i z_i \right) \tag{2}$$

s.t.:

$$\sum_{j \in M} \left(\frac{a_{ij}}{b_i} \right) x_{ij} - \sum_{j \in U} \left(\frac{a_{ij}}{b_i} \right) x_{ij} - \frac{f_i}{b_i} + 1 \geq z_i, \forall\, i \in M_R \cup U_R \tag{3}$$

$$z_i = 1, \forall\, i \in M_{AC} \cup U_{AC} \tag{4}$$

$$z_i = 0, \forall\, i \in M_{NC} \cup U_{NC} \tag{5}$$

$$x_{ij} = 1, \forall\, (i,j) \in E_1 \tag{6}$$

$$x_{ii} = 1, \forall\, i \in V \tag{7}$$

$$x_{ij} \in \{0,1\}, \forall\, (i,j) \in E_2 \backslash E_1 \tag{8}$$

$$z_i \in \{0,1\}, \forall\, i \in V \tag{9}$$

The objective function (2) computes the sum of the weights of all vertices f-controlled by M. Inequality (3) guarantees that every time a vertex i is f-controlled by M, the left hand side will be greater or equal than 1. On the other hand, if the left hand side is less than 1, vertex i is not f-controlled by M and z_i will be settled to 0. The constants b_i are defined in order to maintain the difference between the two summutions and f_i/b_i always greater than -1, while guaranteing at the same time, a tight solution for the linear programming relaxation. Equalities (4) and (5) denotes, respectively, the set of vertices always f-controlled and never f-controlled by M. Equalities (6) and (7) are associated to the set of fixed edges. Finally, the linear programming relaxation (represented by $\bar{P}$) is obtained by replacing integrality constraints (8) and (9) by $x_{ij} \in [0,1]$ and $z_i \in [0,1]$, respectively.

Actually, we can prove through network flows arguments, that binary $0-1$ values (related to the $x's$ variables) are obtained after solving $\bar{P}$ (see [5]).

Therefore, in our *Based LP* procedure (Algorithm 2), a feasible solution for the GMCSP is constructed in the following way. Given a solution $(\tilde{x}, \tilde{z})$ of $\bar{P}$ with components $\tilde{x}_{ij} \in \{0,1\}, \forall\ (i,j) \in E_2$ and $\tilde{z}_i \in [0,1], \forall\ i \in V$, we simply define as f-controlled all vertices $i \in V$ with $\tilde{z}_i = 1$, and as non f-controlled the remaining vertices with $\tilde{z}_i < 1$. A new procedure may be constructed by simply choosing the best solution obtained in Algorithms 1 and 2. As discussed in [4], this procedure restricted to MCSP (particular instance of the GMCSP) has a performance ratio equal to $\frac{1}{2} + \frac{1+\sqrt{n}}{2(n-1)}$, $\forall\ n > 4$.

4 Tabu Search and Some Computational Results

Given a sandwich graph $G = (V, E)$ (associated to a current solution S), we denote by $N_0(G)$, our neighbourhood structure to be used within the TS framework [1]. In this neighbourhood, we hope to f-controll new vertices with positive weights in such way that all vertices already f-controlled by M in G remains f-controlled after the local search.

Assume, without loss of generality that $V = M_R \cup U_R$. Thus, given a sandwich graph G, consider $M_G \subseteq M_R$ and $U_G \subseteq U_R$ the subset of vertices f-controlled by M in G. In addition, consider: $L_G = M_G \cup U_G$. The tabu list T is constructed in the following way: given a sandwich graph G representing a local optimal, we choose an arbitrary vertex i (f-controlled by M) to be removed from L_G. If $i \in M_G$ (respectively $i \in U_G$) we define a new solution by adding (by removing) all edges incident to vertex i and updating all associated costs. This vertex remains in the tabu list by $|T|$ steps. Diversification strategies and Path Relinking where also implemented (see [5] for details).

In the computational tests, all vertices have associated weights and gaps defined at random within intervals established at hand. All parameters involved in the TS procedure were empirically tested. Table 1 present some results for graphs varing from 300 to 1000 vertices. The reduction rates are listed in the column *Reduction Rules*. The objective function values obtained by Algorithms 1 and 2 are presented, respectively, in columns *Based MYK* and *Based LP*. The best of both solutions (represented by boldface letters), is used as starting point in our TS procedure. For instances with 300 and 500 vertices, the TS was repeated 10 times in each case. The column *Best Value* shows the objective function value obtained at the best execution while the column *Average* exibits the medium performance after all repetions. The values between parentheses, indicates the number of times the best solution

Table 1
Results of TS for instances with 300, 500 and 1000 vertices.

Instance	Reduction Rules	Inicial Solution		Tabu		Path Relinking	Approx.	Time(s)
		Based MYK	Based LP	Best Value	Average			
G300-30-20-01	58,94%	2136	**2845**	2847 $_{(1)}$	2845,20	–	0,9904	5,09
G300-30-20-02	56,71%	3200	**4422**	4422 $_{(10)}$	4422	–	0,9966	5,23
G300-30-20-03	61,59%	2371	**2949**	2957 $_{(1)}$	2950,90	–	0,9796	5,84
G300-30-20-04	61,69%	3212	**3949**	3949 $_{(10)}$	3949	–	0,9465	2,37
G300-30-20-05	60,00%	3158	**3807**	3845 $_{(1)}$	3832,35	–	0,9462	3,05
G500-50-30-01	60,84%	8960	**10723**	10788 $_{(1)}$	10744,35	–	0,9487	1,57
G500-50-30-02	59,76%	8858	**11247**	11247 $_{(10)}$	11247	–	0,9822	3,58
G500-50-30-03	58,75%	9353	**12119**	12119 $_{(10)}$	12119	–	0,9842	4,15
G500-50-30-04	62,55%	9061	**10614**	10652 $_{(1)}$	10617,40	–	0,9460	2,21
G500-50-30-05	57,80%	8950	**12166**	12179 $_{(1)}$	12170,90	–	0,9879	5,01
G1000-100-10-1	59,87%	35765	**48231**	48316	–	48399	0,9975	9,99
G1000-100-10-2	60.05%	36961	**49942**	50148	–	50231	0,9961	9,21
G1000-100-10-3	62.04%	38858	**48218**	48801	–	48804	0,9902	12,38
G1000-100-10-4	60.12%	38229	**50984**	51072	–	51093	0,9959	10,24
G1000-100-10-5	60.85%	37131	**48319**	48669	–	48705	0,9933	9,94

were attained after 10 executions of the TS. For intances with 1000 vertices, we execute one iteration of the TS followed by the Path Relinking-PR procedure. The performance of the PR was better for some instances with up 1000 vertices. In the column *Approx.* we compute the approximation rates obtained through the average performance ratio of the TS and the bounds gathered by the linear programming relaxation. Note, for instance, that for all instances considered the approximation rates were within 6% of the optimum value. Finally, the column *Time* exibits the worst execution time (at all repetions) after the construction phase (in seconds), demanded by both TS and TS with PR.

References

[1] Glover, F., Laguna, M., "Tabu Search", Kluwer Acad. Publis., (1998).

[2] Golumbic, M. C., Kaplan, H. and Shamir, R., *Graph Sandwich Problems*, J. Algorithms, **19** (1995), 449–473,

[3] Makino, K., Yamashita, M. and Kameda, Tiko, "Max-and min-neighborhood monopolies", Algorithmica, 34 (2002), 240–260.

[4] Martinhon, C., Protti, F., "An Improved Derandomized Approximation Algorithm for the Max-Controlled Set Problem", WEA, LNCS 3059 (2004), 341–355.

[5] Santos, I. M., "Algoritmos Aproximados para o Problema do Maior Conjunto Controlado Generalizado", Master Dissertation - IC UFF, (2005).