Revisiting the soft global cardinality constraint

Pierre Schaus[1], Pascal Van Hentenryck[1,2], Alessandro Zanarini[1]

[1]Dynadec, [2]Brown University
{pschaus,pvh,alessandro.zanarini}@dynadec.com

Abstract. The Soft Global Cardinality Constraint (SOFTGGC) relaxes the Global Cardinality Constraint (gcc) by introducing a violation variable representing unmet requirements on the number of value occurrences. A first domain consistent filtering algorithm was introduced by Van Hoeve et al. in 2004 using a minimum cost flow algorithm. A simpler and more efficient filtering algorithm was introduced in 2006 by Zanarini et al. using matchings in bipartite graphs. While the consistency check introduced in the second algorithm is correct, we show that the algorithm may not achieve domain consistency when cardinality requirements contain zeroes. We give new domain consistent conditions and show how to achieve domain consistency within the same time bounds. The SOFTGGC constraint was implemented in COMET.

1 Introduction

Régin et al. [3] suggested to soften global constraints by introducing a cost variable measuring the violation of the constraints. This has the advantage that over-constrained satisfaction problems can be turned into a constrained optimization problem solvable by traditional CP solvers; furthermore specialized filtering algorithms can be employed to filter the variables involved in soft constraints. One of the most used global constraints to solve practical problems is the Global Cardinality Constraint (gcc) introduced in [2]:

Definition 1.

$$gcc(X, l, u) = \{(d_1, \ldots, d_n) \mid d_i \in D_i, l_d \leq |\{d_i \mid d_i = d\}| \leq u_d \ , \forall d \ \in D_X\}$$

*A soft version of this constraint (*SOFTGGC*) was introduced in [4]:*

$$softggc(X, l, u, Z) = \{(d_1, \ldots, d_n) \mid d_i \in D_i, d_z \in D_Z, viol(d_1, \ldots, d_n) \leq d_z\} \ where$$
$$viol(d_1, \ldots, d_n) = \sum_{d \in D_X} \max(0, |\{d_i \mid d_i = d\}| - u_d, l_d - |\{d_i \mid d_i = d\}|).$$

The violation represents the sum of excess or shortage for each value. For space reasons, we only consider the *value-based violation* version of the constraint in this paper as the extension to *variable violation* follows directly (as in [5]).

The domain filtering algorithm for SOFTGGC introduced in [5] exploits matching theory and we use the same notation for consistency and clarity. The consistency check and the filtering of the violation variable Z are briefly summarized

in Section 2. The main contribution of the paper is in Section 3 where the corrected filtering algorithm for the X variables is presented. Please refer to [5] for some basic notions about matching theory.

2 Consistency Checking and Filtering of $Z^{\min}$ [5]

Let $G(X \cup D, E)$ be an undirected bipartite graph (also known as the value graph) such that one partition represents the variable set and the other one the value set. There is an edge $\{x_i, d\} \in E$ if and only if $d \in D_i$. Two specialized versions of G were introduced in [5]: They differ only by the capacity of value vertices, where vertex capacity is the maximum number of edges belonging to the matching that share the vertex.

Definition 2. *Let G_o (the overflow graph) be a value graph such that the capacities of value-vertices are set to $c(d) = u_d$. Analogously let G_u (the underflow graph) be a value graph such that the capacities of value vertices are set to $c(d) = l_d$. In both G_o and G_u, variable vertices have unit capacities.*

The violation is expressed in terms of overflows and underflows. They are are characterized by Theorem 1 which specifies how to find

- a valid assignment $(d_1, \ldots, d_n)$ that minimizes the total *overflow* :

$$\sum_{d \in D_X} \max(0, |\{d_i \mid d_i = d\}| - u_d),$$

- a valid assignment $(d_1, \ldots, d_n)$ that minimizes the total *underflow*:

$$\sum_{d \in D_X} \max(0, l_d - |\{d_i \mid d_i = d\}|).$$

Theorem 1 ([5]). *Given a maximum matching M_o in the graph G_o, it is not possible to find an assignment with a total overflow less than $BOF = |X| - |M_o|$ (best overflow). Given a maximum matching M_u in the graph G_u, it is not possible to find an assignment with a total underflow less than $BUF = \sum_{d \in D} l_d - |M_u|$ (best underflow).*

Theorem 2 ([5]). *Given a* SOFTGGC *constraint and two maximum matchings M_o and M_u , respectively in G_o and G_u , it is possible to build a class of assignments with overflow equal to $BOF = |X| - |M_o|$ (best overflow) and $BUF = \sum_{d \in D} l_d - |M_u|$ (best underflow).*

In other words, Theorem 2 tells us that it is possible to find an assignment with a violation equal to $BOF + BUF$. The violation variable can be filtered as $Z^{\min} \leftarrow \max(Z^{\min}, BOF + BUF)$. If $BOF + BUF > Z^{\max}$, then the constraint is inconsistent. The algorithm to find an assignment with a violation equal to $BOF + BUF$ starts from a matching M_u in G_u (having by definition a underflow

of BUF and an overflow of 0 but not representing a complete assignment). This matching is then increased in G_o using a classical augmenting-path algorithm. Since the augmenting-path algorithm never decreases the degree of a vertex, the underflow cannot increase (it remains constant). At the end, the overflow is equal to BOF. The final assignment has a violation equal to $BOF + BUF$. See [5] for further details.

3 Filtering of X

While the consistency check is correct, the original paper [5] overlooked the case in which the lower or upper bounds of the value occurrences are zeroes and it does not characterize the conditions to achieve domain consistency in such cases (see Example 1 below). In this section, we review and correct the theorems on which the filtering algorithm is built upon. Theorem 3 shows the properties for which a vertex x is matched in every maximum matching.

Theorem 3. *A variable vertex x is matched in every maximum matching of G_u (G_o) iff it is matched in a maximum matching M_u (M_o) and there does not exist an M-alternating path starting from a free variable vertex and finishing in x.*

Proof. $\Rightarrow$ Suppose there exists an even M-alternating path P starting from a free variable vertex x' such that $P = \{x', \ldots, x\}$; the alternating path is even as x and x' belong to the same vertex partition furthermore x must be matched as P is an alternating path and x' is free. Then $M' = M \oplus P'$ is still a maximum matching in which x is free.
$\Leftarrow$ Suppose there exists a maximum matching M' in which x is free. Any of the adjacent vertices of x are matched otherwise M' is not maximum. We can build an even alternating path starting from x by choosing one of the adjacent vertices of x and then by following the edge belonging to M'. By using such an even alternating path, it is possible to build a new maximum matching in which x is matched and there exists an M-alternating path starting from a free variable-vertex.

Theorem 4 gives the conditions under which forcing an assignment $x \leftarrow v$ leads to a unit increase in the best underflow.

Theorem 4. *Forcing the assignment $x \leftarrow v$ leads to decrease the size of the maximum matching in G_u by one and thus increasing the underflow by one if and only if $l_v > 0$ and $e = \{x, v\}$ does not belong to a maximum matching in G_u, or $l_v = 0$ and the vertex x is matched in every maximum matching in G_u.*

Proof. If $l_v > 0$, then there exists a matching (maybe not maximum) using the edge $x \leftarrow v$. Consequently, if $e = \{x, v\}$ does not belong to any maximum matching of G_u, the size of a maximum matching would decrease by one forcing the assignment $x \leftarrow v$. Now if $l_v = 0$, then the edge $x \leftarrow v$ belongs to no matching of G_u so the size of the maximum matching decreases only if the vertex x is matched in every maximum matching. □

Example 1. Assume $D_1 = \{1, 2\}$, $D_2 = \{1, 3\}$, $D_3 = \{1, 3\}$ with $l_1 = 0, l_2 = 1, l_3 = 1$ (upper bounds are all equal to $|X|$). A maximum underflow matching is $M = \{\{X_1, 2\}, \{X_2, 3\}\}$ that does not incur in any violation. The wrong assumption made in [5] was to consider that values with capacity equal to zero would not cause any sort of underflow increase. However, in some cases, this assumption is incorrect as we now show.

Forcing the assignment $X_2 = 1$ (or equivalently $X_3 = 1$) does not cause an increment of violation; X_2 belongs to an M-alternating path starting from a free variable vertex, i.e., $P = (X_3, 3, X_2)$ therefore there exists a maximum matching M' in which X_2 is free ($M' = \{\{X_1, 2\}, \{X_3, 3\}\}$). Then, X_2 can take the value 1 without increasing the underflow. The assignment $X = (2, 1, 3)$ has total violation equal to zero. However, forcing the assignment $X_1 = 1$ causes an increment of violation; there is no M-alternating path starting from a free variable vertex and ending in X_1 thus X_1 is matched in every maximum matching and it is not free to take the value 1 without increasing the underflow. The best assignment would then have a unit underflow (e.g., $X = (1, 3, 3)$) as only X_1 can take the value 2.

Similarly, Theorem 5 gives the conditions under which forcing an assignment $x \leftarrow v$ leads to a unit increase to the best overflow.

Theorem 5. *Forcing the assignment $x \leftarrow v$ leads to decrease the size of the maximum matching in G_o by one and thus increasing the overflow by one if and only if $u_v > 0$ and $e = \{x, v\}$ does not belong to a maximum matching in G_o, or $u_v = 0$ and the vertex x is matched in every maximum matching in G_o.*

Proof. Similar to proof of theorem 4

Example 2. Assume $D_1 = \{1, 4\}$, $D_2 = \{1, 2, 3\}$, $D_3 = \{1, 3\}$, $D_4 = \{3\}$, $D_5 = \{1, 2, 4\}$ with $u_1 = 1, u_2 = 2, u_3 = 1, u_4 = 0$ (all lower bounds are null). A maximum overflow assignment is $X = (1, 2, -, 3, 2)$ (unit violation). Variable X_1 belongs to an M-alternating path starting from a free variable-vertex $P = (X_3, 1, X_1)$, therefore there exists a maximum matching in which X_1 is free to take any value without increasing the best overflow (e.g., the full assignment $X = (4, 2, 1, 3, 2)$ has still an overflow equal to 1). For the arc $(X_5, 4)$, the situation is different: in fact X_5 is matched in every maximum matching. Therefore, forcing this assignment would increase the overflow to 2; the best assignment we could get is for instance $X = (1, 2, 1, 3, 4)$.

Theorem 6 gives the conditions to reach domain consistency.

Theorem 6. *Let G_o and G_u be the value graphs with respectively upper and lower bound capacities and let M_o and M_u be maximum matching respectively in G_o and G_u ; let BOF and BUF be respectively $BOF = |X| - |M_o|$ and $BUF = \sum_{d \in D} l_d - |M_u|$. The constraint $softgcc(X, l, u, Z)$ is domain consistent on X if and only if $\min D_Z \leq BOF + BUF$ and either:*

1. *$BOF + BUF < (\max D_Z - 1)$ or*

2. *if* $BOF + BUF = (\max D_Z - 1)$ *and for each edge* $e = \{x, v\}$ *either*
   - *it belongs to a maximum matching in* G_u*, or* $l_v = 0$ *and the vertex* x *is not matched in every maximum matching in* G_u ***or***
   - *it belongs to a maximum matching in* G_o*, or* $u_v = 0$ *and the vertex* x *is not matched in every maximum matching in* G_o *or*
3. *if* $BOF + BUF = \max D_Z$ *and for each edge* $e = \{x, v\}$ *we have that*
   - *it belongs to a maximum matching in* G_u*, or* $l_v = 0$ *and the vertex* x *is not matched in every maximum matching in* G_u ***and***
   - *it belongs to a maximum matching in* G_o*, or* $u_v = 0$ *and the vertex* x *is not matched in every maximum matching in* G_o *.*

Proof. Common to the proof of all the cases is the fact that there exists an assignment with violation $BOF + BUF$. If $BOF + BUF > \max D_Z$ then the constraint is inconsistent since $BOF + BUF$ is a lower bound on the violation.

1. Forcing the assignment of one variable cannot increase BUF by more than one and cannot increase BOF by more than one. So in the worst case the assignment of a variable to a value results in $BUF' = BUF + 1$ and $BOF' = BOF + 1$. By Theorem 2 it is possible to build an assignment for all the variables with violation $BUF' + BOF' = BUF + BOF + 2 \leq \max D_Z$ which is thus consistent.
2. Since $BOF + BUF = (\max D_Z - 1)$ and because of Theorem 2, at most one of BUF or BOF can increase by one by forcing the assignment $x \leftarrow v$. Theorems 4 and 5 tell us the conditions say $\mathcal{C}_u$ and $\mathcal{C}_o$ under which the assignment leads to increase respectively BUF and BOF by one. At most one of these conditions can be satisfied. Hence the condition expressed is nothing else than $\neg(\mathcal{C}_u \wedge \mathcal{C}_o) \equiv \neg\mathcal{C}_u \vee \neg\mathcal{C}_o$.
3. This is similar to previous point except that neither BUF nor BOF can increase by one. Hence the condition expressed is $\neg\mathcal{C}_u \wedge \neg\mathcal{C}_o$. □

Note that once we compute the alternating paths to detect edges belonging to a maximum matching, we get for free also the variable vertices that are matched in every maximum matching (Theorem 3). Therefore the complexity of the filtering algorithm remains unchanged w.r.t. [5].

The algorithm described in this paper is implemented in the constraints `softAtLeast`, `softAtMost` and `softCardinality` of COMET [1].

References

1. COMET 2.0. www.dynadec.com.
2. J-C. Régin. Generalized arc consistency for global cardinality constraint. *AAAI-96*, pages 209–215, 1996.
3. J.C. Régin, T. Petit, C. Bessière, and J.-F. Puget. An original constraint based approach for solving over constrained problems. *Sixth International Conference on Principles and Practice of Constraint Programming (CP 2000)*, 1894, 2000.
4. Willem Jan van Hoeve, Gilles Pesant, and Louis-Martin Rousseau. On global warming: Flow-based soft global constraints. *J. Heuristics*, 12(4-5):347–373, 2006.
5. Alessandro Zanarini, Michela Milano, and Gilles Pesant. Improved algorithm for the soft global cardinality constraint. In *CPAIOR*, pages 288–299, 2006.