sPREAd : A Balancing Constraint Based on Statistics

Gilles Pesant[1] and Jean-Charles Régin[2]

[1] ILOG, 1681 route des Dolines, 06560 Valbonne, France
[2] Computing and Information Science, Cornell University, Ithaca NY 14850 USA *
pesant@crt.umontreal.ca, jcregin@cs.cornell.edu

Abstract. Many combinatorial problems require of their solutions that they achieve a certain balance of given features. In the constraint programming literature, little has been written to specifically address this issue, particularly at the modeling level. We propose a new constraint dedicated to balancing, based on well-known and well-understood concepts in statistics. We show how it can be used to model different situations in which balance is important. We also design efficient filtering algorithms to guide the search towards balanced solutions.

1 Introduction

We have seen many advances in CP modeling in recent years. Useful problem substructures have been identified, leading to new constraints with efficient filtering algorithms. Soft constraints have been introduced to handle over-constrained problems. Lexicographic constraints have been designed to break problem symmetries. Efforts to automate the modeling process have also been made.

One aspect that has lacked a truly satisfying approach to date is the ability to balance certain features of a solution. Take for example the balanced academic curriculum problem [1], in which courses are assigned to periods so as to balance the academic load between periods. Because of additional constraints (prerequisite courses, minimum and maximum number of courses per period) and a varying number of credits per course, reaching perfect balance is generally impossible. Given that, some common ways of encouraging balance at the modeling level are:

a) to set reasonable bounds on each load, tolerating a certain deviation from the ideal value;

b) to minimize the greatest load, thus avoiding outliers (or at least those above the ideal value);

c) to take the least square error.

* Research conducted while the first author was on sabbatical leave from École Polytechnique de Montréal and while the second author was on sabbatical leave from ILOG.

The first two options both have the disadvantage of putting on an equal footing solutions with quite different balance:

a) If we require that loads belong to $[8, 12]$, aiming for an ideal load of 10, then sets of loads $\{10, 10, 10, 10, 9, 11\}$ and $\{8, 8, 8, 12, 12, 12\}$ both satisfy the restriction but the former is much more balanced. The situation could be corrected somewhat by only allowing deviations for a few of the loads, but which ones should it be? We run the risk of being unfair (ironically) or, even worse, of excluding legitimate solutions.

b) Loads $\{10, 10, 10, 10, 9, 11\}$ and $\{9, 9, 9, 11, 11, 11\}$ both have a greatest load of 11 but again the former is more balanced.

The last option corrects this by considering a combination of individual deviations. However it requires that we solve a discrete optimization problem, with a nonlinear objective as is the case for the second option. This may not be easy and, if other real objectives are present, we have to come up with suitable weights for the different terms of the objective function. Balance can also be dealt with in the search strategy: by keeping track of previous course assignments, we can favor certain future assignments that will improve the balance. This is helpful but using it on its own means that an important aspect of the problem is not present at all in the model itself.

Balance is often important in assignment problems or in problems with an assignment component. We give a few examples. In assembly line balancing the workload of the line operators must be balanced. In rostering we may talk of fairness instead of balance, because of the human factor. Here we want a fair distribution of weekends off or of night shifts among workers, for example. In vehicle routing one dimension of the problem is to partition the customers into the different routes — balancing the number of customers served on each route, the quantity of goods delivered, or the time required to complete the route may be of interest. In one of the few works specifically addressing balance in the context of constraint satisfaction, an earth observation satellite scheduling and sharing problem is used to investigate three ways of handling fairness among agents with competing observation requirements [3]. The first one applies a decomposition into individual problems, each with a fair share of observation windows, but overall efficiency suffers. The second one favors efficiency and sets a lower bound on individual shares for fairness (option *a)* above). The third one computes a set of Pareto-optimal solutions in the two-dimensional space of overall efficiency and fairness. To evaluate fairness, they use the Gini index, popular in microeconomics.

We could describe the balance we seek in the following way:

- the average value should be close to a given target, corresponding to the ideal value;
- there should be no outliers, as they would correspond to an unbalanced situation;
- values should be grouped around the average value.

We claim that statistics provide appropriate mathematical concepts to express this. We will propose a constraint expressing balance in a way similar to the method of least squares mentioned before but by setting limits on the deviation instead of minimizing an objective which must be weighted relative to other potential objectives.

The rest of this paper is organized as follows. Section 2 reviews some basic concepts and definitions in statistics. Section 3 defines the new constraint based on statistics that we propose and presents typical uses. Section 4 derives some inequalities bounding the number of variables taking extreme values and uses them to filter the domains of the variables. Section 5 builds up to another filtering algorithm, this one achieving bounds consistency.

2 Statistics Background

Given a collection of numbers, even simple summary statistics about them can be revealing. *Measures of location* tell us about the central tendency of the values. The most common such measures are the mode, the median, and the mean. The *mode* is the value(s) occurring most often in a given collection of numbers. The *median*, denoted $\tilde{x}$, is the smallest value such that at least half of the numbers are no larger than it, and at least half of the numbers are at least as large as it. We will prefer to use the mean because it is instrumental in telling us how many values may exceed a given threshold, which will prove useful to filter domains.

Definition 1 (mean). *The* (arithmetic) mean *of a collection of values* $\langle x_1, x_2, \ldots, x_n \rangle$*, denoted* μ*, is computed as*

$$\mu = \frac{1}{n} \sum_{i=1}^{n} x_i. \tag{1}$$

Measures of spread tell us whether the values tend to be bunched together or spread out. The most common measures are the range, the semi-quartile range, and the standard deviation. The *range* is the size of the smallest interval containing all the values. Unfortunately that measure is highly sensitive to outliers. The *semi-quartile range*, half the size of the smallest interval containing fifty percent of the most central values, partially overcomes that drawback. We favor the more familiar standard deviation, partly for the same reason as the mean: we will be able to limit the number of values straying away from the center.

Definition 2 (standard deviation). *The* standard (or root-mean-square) deviation *of a collection of values* $\langle x_1, x_2, \ldots, x_n \rangle$*, denoted* σ*, is computed as*

$$\sigma = (\frac{1}{n} \sum_{i=1}^{n} (x_i - \mu)^2)^{\frac{1}{2}}. \tag{2}$$

An alternate way of computing the standard deviation, which is more numerically stable, is obtained through the *Koenig-Huyghens relation*:

$$\sigma^2 = \frac{1}{n} \sum_{i=1}^{n} x_i^2 - \mu^2. \tag{3}$$

Measures of skewness tell us about the general shape of the distribution of values. Two collections of values with identical mean and standard deviation may nevertheless be significantly different. A perfectly symmetric *continuous* distribution will have its median and mean coincide. A distribution with a positive (resp. negative) bias will have $\tilde{x} < \mu$ (resp. $\tilde{x} > \mu$). The *Pearson coefficient*, computed as $3(\mu - \tilde{x})/\sigma$, is one way to measure skewness. The simpler form $\mu - \tilde{x}$ at least preserves the sign of the bias.

Two well-known inequalities for random variables can be recast for our purpose. They can be useful to derive filtering algorithms.

Theorem 1 (Markov's Inequality). *Consider a collection of non-negative values $\langle x_1, x_2, \dots, x_n \rangle$ with mean μ, and some threshold τ. Then the fraction of these values that are greater or equal to τ is at most $\frac{\mu}{\tau}$.*

For example, if the threshold selected is three times the mean then at most one third of the values are no smaller than that threshold.

Theorem 2 (Bienaymé-Chebychev's Inequality). *Consider a collection of values $\langle x_1, x_2, \dots, x_n \rangle$ with mean μ and standard deviation σ, and some positive number k. Then the fraction of these values that are $k\sigma$ or further from μ is at most $\frac{1}{k^2}$.*

This important result bounds the number of values that can be far from the mean. For example, at most 25% of the values may be two standard deviations away from the mean and at most 4% may be five standard deviations away.

3 The Spread Constraint

This section defines and discusses the constraint we propose. We first give some basic definitions and notation in constraint programming.

Definition 3 (finite-domain (discrete) variable). *A* finite-domain (discrete) variable *x takes a value in $D(x)$, a finite set called its* domain. *Whenever there is a total order defined on that set (e.g. when it is a subset of $\mathbb{N}$), we denote the smallest (resp. largest) value x may take as $x^{\min}$ (resp. $x^{\max}$).*

Definition 4 (bounded-domain (continuous) variable). *A* bounded-domain (continuous) variable *y takes a value in $I_D(y) = [y^{\min}, y^{\max}]$, an interval on $\mathbb{R}$ called its* domain *as well.*

Definition 5 (relaxed domain). *Given finite-domain variable x, we denote by $I_D(x)$ its domain relaxed to the continuous interval $[x^{\min}, x^{\max}]$. By extension for a union of domains $\mathcal{D} = \bigcup_{i=1}^{n} D(x_i)$, let $I_{\mathcal{D}}$ represent the continuous interval* $[\min_{i=1}^{n} x_i^{\min}, \max_{i=1}^{n} x_i^{\max}]$.

We are now ready to state the constraint:

Definition 6 (spread constraint). *Given a set of finite-domain variables* $X = \{x_1, x_2, \ldots, x_n\}$ *and bounded-domain continuous variables* μ*,* σ*, and* $\tilde{x}$*, constraint* `spread`$(X, \mu, \sigma, \tilde{x})$ *states that the collection of values taken by the variables of* X *exhibits an arithmetic mean* μ*, a standard deviation* σ*, and a median* $\tilde{x}$*.*

There are clear advantages to this formulation. First, it is not affected by a permutation of the values given to the x_i's. No particular variable or subset of variables is a priori identified as taking a lower value than others, for example, which might be necessary with other approaches to fairness, ironically introducing a bias. Second, it is not affected by the sign of the deviation. The impact on the standard deviation of a value away from the mean is the same whether the value is above or below the mean. Finally, it is based on well-established concepts in statistics.

3.1 Typical Uses

We outline some typical uses of the constraint by focusing on how μ is constrained and illustrate them with examples taken from rostering. First note that if we set σ to 0, we are asking for perfect fairness: every x_i should be identical. If in essence we have a fixed number of balls to distribute as evenly as possible into a fixed number of boxes, μ is fixed since it corresponds to the ratio of the number of balls to the number of boxes. We constrain the variables by limiting σ. This situation occurs, for example, when night shifts should be evenly distributed among 10 staff members and we know that there are exactly 200 night shifts to cover:

$$\texttt{spread}(X, 20, [0, 1], \tilde{x})$$

If on the contrary the number of balls is unknown, μ is not fixed. We may have some approximate idea of what the mean should be and in this case μ is constrained around that approximation. For example, weekends off should be evenly spread over the whole planning horizon in an individual schedule. Taking our variables to be the size of the gaps between such weekends and even given the number of them, the mean may not be known because of the uncertainty as to where the last weekend off falls. Nevertheless, we may wish for a typical gap of 3:

$$\texttt{spread}(X, [2.8, 3.2], [0, 0.5], \tilde{x})$$

Other times we have no idea what the mean could be and μ is left free. For example, a weekend on which one day is worked and the other not is called a "broken" weekend, a generally undesirable feature. We often do not know in advance how many such weekends will occur in a schedule but we nevertheless wish the number of broken weekends to be evenly distributed among all staff members. Considering a roster over w weeks, we could state:

$$\texttt{spread}(X, [0, w], [0, \frac{w}{3}], \tilde{x})$$

If two of the staff members have more seniority, their number of broken weekends should be about half that of the others:

$$y_1 = 2x_1,\ y_2 = 2x_2,\ y_i = x_i\ (3 \le i \le n), \quad \texttt{spread}(Y, [0, w], [0, \frac{w}{3}], \tilde{x})$$

Since broken weekends are undesirable, we could prefer instead that the distribution of values does not show a negative bias, i.e. there should not be a majority of staff members with an above-average number of such weekends:

$$0 \le \tilde{x} \le \mu \le w, \quad \texttt{spread}(X, \mu, [0, \frac{w}{3}], \tilde{x})$$

4 Fast Filtering

It would be difficult to efficiently achieve domain consistency on the `spread` constraint because even in the special case where μ is fixed, we are left with a linear Diophantine equation originating from Definition 1. In the case of Definition 2, it is not even linear. Bounds consistency is a common compromise in such a case. At a minimum, we can apply bounds consistency on (1) and (3).

Example 1. Consider a set of ten variables required to take integer values from $\{7, 8, \dots, 13\}$ such that $\mu \in [9.5, 10.5]$. Suppose that at some point five of the variables are fixed to value 13. Bounds consistency on (1) alone will remove 13 from the domain of every other variable as there is no support for a sixth variable taking that value: $[9.5, 10.5] \cap (6 \times 13 + 4 \times [7, 13])/10 = [9.5, 10.5] \cap [10.6, 13] = \emptyset$.

Theorem 1 bounds the number of occurrences of values far from the mean. We could extend this result to filter the domains of the variables in X but it would not give us more than bounds consistency on (1). However Theorem 2 also bounds the number of occurrences of values far from the mean and we will show that it can lead to more filtering than bounds consistency on (1) and (3).

4.1 Exploiting Bienaymé-Chebychev's Inequality

We wish to derive a family of inequalities for consecutive integer thresholds away from the mean. The key observation is that the value k in the theorem need not be integer: we therefore use appropriate values of k that will provide the integer thresholds we need. By seeking these exact values we obtain the strongest possible bounds from the theorem.

Let $\mathcal{D} = \bigcup_{i=1}^{n} D(x_i)$ and $I_{\mathcal{D}} = [a, b]$. Define variables $c_\ell,\ \ell \in \{a, a+1, \dots, b\}$ as the number of times a variable from X takes value ℓ. First consider the case $\mu - a \ge b - \mu$. Since this means there is at least as much slack below μ as there is above, we focus on threshold values below, that is $a + j$ for $0 \le j < \mu - a$. For each threshold $a + j$ we seek k such that $\mu - k\sigma = a + j$, in order to get the smallest bound $\frac{1}{k^2}$. Solving for k we obtain $k = \frac{\mu - a - j}{\sigma}$, yielding:

$$\sum_{\ell=a}^{a+j} c_\ell + \sum_{\ell=\mu^{\max}+\mu^{\min}-(a+j)}^{b} c_\ell \le \lfloor \frac{\sigma^2}{(\mu - a - j)^2} \cdot n \rfloor \qquad 0 \le j < \mu - a \quad (4)$$

The lower limit in the second sum ensures that it only considers values that are at least $\mu - (a + j)$ away from the mean, as in the first sum.

Similarly when $\mu - a < b - \mu$, for each threshold $b - j$ we seek k such that $\mu + k\sigma = b - j$, yielding:

$$\sum_{\ell=a}^{\mu^{\min}-((b-j)-\mu^{\max})} c_\ell + \sum_{\ell=b-j}^{b} c_\ell \le \lfloor \frac{\sigma^2}{(b-\mu-j)^2} \cdot n \rfloor \qquad 0 \le j < b - \mu \quad (5)$$

These inequalities can lead to better filtering than bounds consistency on (1) and (3) because they simultaneously take into account μ and σ, as illustrated in the following example.

Example 2. Consider again the situation depicted in Example 1 with the additional restriction that $\sigma \in [0, 0.4]$. The left-hand side of (3) consequently lies in $[0, 0.16]$, which has plenty of overlap with the right-hand side $(10 \times [7^2, 13^2] - [9.5^2, 10.5^2])/10 \subset [-61.25, 78.75]$. It is easy to verify that (3) is bounds consistent: for example, checking value 7 only shrinks the right-hand side to $[-61.25, 66.75]$. Equation (1) is bounds consistent as well. However inequality (4) for $j = 1$ gives $c_7 + c_8 + c_{12} + c_{13} \le \lfloor 0.7\bar{1} \rfloor = 0$. In other words, the domain of each x_i can be narrowed to $\{9, 10, 11\}$.

The c_ℓ variables are the same we would use in a global cardinality constraint except that here we do not bound them individually but in telescoping sums. We can maintain bounds consistency on each inequality in $\mathcal{O}(b-a)$ time and use an *upper bound constraint* (half of a `gcc`) between the c_ℓ's and the x_i's on which we maintain bounds consistency in $\mathcal{O}(n + t)$ time where t is the time required to sort the x_i's by their bounds [4]. (Note however that we do not necessarily achieve bounds consistency on the `spread` constraint as Example 3 will show.) The overall time complexity is $\mathcal{O}(n+t+(b-a)^2)$. Since n, the number of variables, is typically much larger than $b - a$, the span of the values, the algorithm runs in linear time under the reasonable assumption that $b - a$ is a small constant.

4.2 Median

Simple inequalities follow from the definition of the median:

$$\sum_{i=a}^{\tilde{x}^{\min}-1} c_\ell < \lfloor \frac{n}{2} \rfloor, \qquad \sum_{i=a}^{\tilde{x}^{\max}} c_\ell \ge \lceil \frac{n}{2} \rceil \quad (6)$$

$$\sum_{i=\tilde{x}^{\max}+1}^{b} c_\ell < \lfloor \frac{n}{2} \rfloor, \qquad \sum_{i=\tilde{x}^{\min}}^{b} c_\ell \ge \lceil \frac{n}{2} \rceil \quad (7)$$

We can maintain bounds consistency on them as well but here we should combine them with a (full) bounds consistent `gcc` constraint [4][2]. To filter on $\tilde{x}$, we can use the fact that $\tilde{x} = \min\{k \mid \sum_{i=a}^{k} c_\ell \ge \lceil \frac{n}{2} \rceil\} = \max\{k \mid \sum_{i=k}^{b} c_\ell \ge \lceil \frac{n}{2} \rceil\}$.

5 A Bounds Consistency Algorithm

The algorithm of the previous section did not consider the individual domains of the x_i's but worked instead from the smallest interval containing all of them. Even a very simple example like Example 3 is enough to show that some filtering may be missed when the domains are significantly different. This section describes an algorithm that takes into account the span of each individual domain of the x_i's and that achieves bounds consistency for the `spread` constraint.

Example 3. Consider two variables with respective domains $\{7, 8\}$ and $\{12, 13\}$ such that $\mu \in [9.5, 10.5]$ and $\sigma \in [0, 2]$. Equations (1) and (3) are bounds consistent and inequality (4) for $j = 0$ gives $c_7 + c_{13} \leq \lfloor 1.28 \rfloor = 1$ but there is clearly no solution with 7 or 13.

5.1 Establishing the Optimal Value

Definition 7. *Let $X = \{x_1, x_2, \ldots, x_n\}$ as before and define the following problem $\Pi(X, q)$ for some fixed number q:*

$$\min \sum_{i=1}^{n} (x_i - \frac{q}{n})^2 \text{ such that } \sum_{i=1}^{n} x_i = q, \quad x_i \in I_D(x_i)\ 1 \leq i \leq n.$$

We also define the more general problem $\Pi(X, [\ell_q, u_q])$:

$$\min \sum_{i=1}^{n} (x_i - \frac{q}{n})^2 \text{ such that } \sum_{i=1}^{n} x_i = q, \quad x_i \in I_D(x_i)\ 1 \leq i \leq n, \quad q \in [\ell_q, u_q].$$

We will denote by opt(Π) *the optimal value of the problem Π.*

Definition 8. *An assignment $A : x \mapsto I_D(x)$ over X is said to be a ν-*centered assignment *when*

$$A(x) = \begin{cases} x^{\max}, & \text{if } x^{\max} \leq \nu \\ x^{\min}, & \text{if } x^{\min} \geq \nu \\ \nu, & \text{otherwise} \end{cases}$$

Lemma 1 *Any optimal solution to $\Pi(X, q)$ must be a ν-centered assignment.*

Proof. The objective function of $\Pi(X, q)$ can be rewritten as follows: $\sum(x_i - \frac{q}{n})^2 = (\sum x_i^2) - \frac{q^2}{n}$ because $\sum x_i = q$. Thus, for a given q the minimum value of $\sum(x_i - \frac{q}{n})^2$ can be deduced from the minimum value of $\sum x_i^2$. Consider an assignment A on X which is a solution to $\Pi(X, q)$ but not a ν-centered assignment. We prove that $\sum(A(x_i))^2$ is not optimal by constructing another assignment B such that $\sum(B(x_i))^2 < \sum(A(x_i))^2$. There are three ways in which A may fail to be a ν-centered assignment:

• $\exists\, i, j$ s.t. $x_i^{\min} < A(x_i) < x_i^{\max}$, $x_j^{\min} < A(x_j) < x_j^{\max}$, and $A(x_i) > A(x_j)$. Define B as $B(x_i) = A(x_i) - d$, $B(x_j) = A(x_j) + d$, $B(x_k) = A(x_k)$ for $k \neq i, j$, where $d = \frac{1}{2} \min((x_j^{\max} - A(x_j)), (A(x_i) - A(x_j)), (A(x_i) - x_i^{\min}))$. Then B is

I	$ES(I)$	$\lvert M(I)\rvert$	$V(I)$	$GC(I)$	q-opt($\Pi(X, I)$)	opt($\Pi(X, I)$)
$[0, 1]$	13	2	$[13, 15]$	18	15	19.5
$[1, 2]$	12	3	$[15, 18]$	24	18	12.0
$[2, 3]$	14	2	$[18, 20]$	21	20	$8.\bar{3}$
$[3, 4]$	8	4	$[20, 24]$	24	24	8.0
$[4, 5]$	16	2	$[24, 26]$	24	24	8.0
$[5, 6]$	26	0	$[26, 26]$	26	26	$9.\bar{3}$
$[6, 9]$	20	1	$[26, 29]$	24	26	$9.\bar{3}$

Table 1. Relevant values computed from Example 4.

also a solution to $\Pi(X, q)$ from the definition of B and the choice of d. Now $(B(x_i))^2 + (B(x_j))^2 = (A(x_i) - d)^2 + (A(x_j) + d)^2 = (A(x_i))^2 + (A(x_j))^2 + 2d(A(x_j) - A(x_i) + d)$. Since $A(x_j) < A(x_i)$, $d > 0$ and $d \le (A(x_i) - A(x_j))/2$ we have that $2d(A(x_j) - A(x_i) + d) < 0$. Thus $(B(x_i))^2 + (B(x_j))^2 < (A(x_i))^2 + (A(x_j))^2$ and B is a better assignment.

- $\exists\ i$ s.t. $A(x_i) = x_i^{\max} > \nu$ (and the symmetric case $A(x_i) = x_i^{\min} < \nu$). Take j s.t. $A(x_j) = \nu < x_j^{\max}$ (if we cannot find such a j then we are in the third case below). Build B as in the first case.
- $\exists\ i, j$ s.t. $A(x_i) = x_i^{\max}$, $A(x_j) = x_j^{\min}$, and $A(x_i) > A(x_j)$ (i.e. the two groups overlap). Build B as in the first case. □

To simplify the analysis, we first partition $I_{\mathcal{D}}$ into intervals in which the status of the relaxed domains of the x_i's does not vary: each either completely lies to the left or right, or completely contains the interval. We then exhibit a particular ν-centered assignment and show that it is an optimal solution to $\Pi(X, q)$. Finally we generalize for an unspecified value $q \in [\ell_q, u_q]$.

Definition 9. *Let $B(X)$ be the sorted sequence of bounds of the relaxed domains of the variables of X, in non-decreasing order and with duplicates removed. Define $\mathcal{I}(X)$ as the set of intervals defined by a pair of two consecutive elements of $B(X)$. The k^{th} interval of $\mathcal{I}(X)$ is denoted by I_k.*

Example 4. Let $D(x_1) = [0, 2]$, $D(x_2) = [1, 4]$, $D(x_3) = [0, 5]$, $D(x_4) = [3, 5]$, $D(x_5) = [3, 4]$, $D(x_6) = [6, 9]$. Then $I_1 = [0, 1]$, $I_2 = [1, 2]$, $I_3 = [2, 3]$, $I_4 = [3, 4]$, $I_5 = [4, 5]$, $I_6 = [5, 6]$, $I_7 = [6, 9]$.

Definition 10. $\underline{S}(X) = \sum_{x \in X} x^{\min}$ *and* $\overline{S}(X) = \sum_{x \in X} x^{\max}$.
Let I be an interval of $\mathcal{I}(X)$. Then

- $R(I) = \{x \mid x^{\min} \ge \max(I)\}$*, the variables lying to the right of I*
- $L(I) = \{x \mid x^{\max} \le \min(I)\}$*, the variables lying to the left of I*
- $M(I) = \{x \mid I \subseteq I_D(x)\}$*, the variables overlapping I*
- $ES(I) = \sum_{x \in L(I)} x^{\max} + \sum_{x \in R(I)} x^{\min}$
- $V(I) = [ES(I) + \min(I) \times |M(I)|, ES(I) + \max(I) \times |M(I)|]$

Lemma 2 *$ES(I_{k+1}) = ES(I_k) + (p_{k+1} - q_{k+1}) \times \max(I_k)$, where $p_{k+1} = |L(I_{k+1})| - |L(I_k)|$ and $q_{k+1} = |R(I_k)| - |R(I_{k+1})|$.*

Proof. $\forall x \in (R(I_k) - R(I_{k+1}))$ $x^{\min} = \min(I_{k+1})$ and $\forall x \in (L(I_{k+1}) - L(I_k))$ $x^{\max} = \max(I_k)$. From Def. 9 $\max(I_k) = \min(I_{k+1})$. □

Proposition 1 *$\forall a \in [\underline{S}(X), \overline{S}(X)]$ there exists $I \in \mathcal{I}(X)$ such that $a \in V(I)$.*

Proof. We already have $\min(V(I_1)) = \underline{S}(X)$ and $\max(V(I_{|\mathcal{I}(X)|})) = \overline{S}(X)$. It is therefore sufficient to show that for any two consecutive intervals I_k, I_{k+1} from $\mathcal{I}(X)$, we have $\min(V(I_{k+1})) = \max(V(I_k))$, thus leaving no gaps. Let $m_k = |M(I_k)|$ and $m_{k+1} = |M(I_{k+1})|$. From Lemma 2, $\min(V(I_{k+1})) = ES(I_{k+1}) + m_{k+1}\min(I_{k+1}) = ES(I_k) + (p_{k+1} - q_{k+1})\max(I_k) + m_{k+1}\min(I_{k+1})$. In addition, $m_{k+1} = m_k - p_{k+1} + q_{k+1}$ and $\min(I_{k+1}) = \max(I_k)$. Therefore $\min(V(I_{k+1})) = ES(I_k) + \max(I_k) \times |M(I_k)| = \max(V(I_k))$. □

Definition 11. *Given a value q such that $q \in [\underline{S}(X), \overline{S}(X)]$ and I such that $q \in V(I)$, define the following assignment $A_{q,I}$ on X:*

$$A_{q,I}(x) = \begin{cases} x^{\max}, & x \in L(I) \\ x^{\min}, & x \in R(I) \\ (q - ES(I))/|M(I)|, & x \in M(I) \end{cases}$$

Lemma 3 *Assignment $A_{q,I}$ is a feasible solution to $\Pi(X, q)$ and is ν-centered.*

Proof. We first have to show that every variable is assigned a value within its relaxed domain. It is immediate in the first two cases but not so for $x \in M(I)$. Since $q \in V(I)$, we have $q - ES(I) \in [\min(I) \times |M(I)|, \max(I) \times |M(I)|]$ and so $(q - ES(I))/|M(I)| \in [\min(I), \max(I)] = I \subseteq I_D(x)$, by definition of $M(I)$. This also shows that $A_{q,I}$ is ν-centered with $\nu = (q - ES(I))/|M(I)|$. As for the sum, $\sum_{i=1}^{n} A_{q,I}(x) = ES(I) + |M(I)|(q - ES(I))/|M(I)| = q$. □

Theorem 3. *$A_{q,I}$ is an optimal solution to $\Pi(X, q)$.*

Proof. Given lemmas 1 and 3, it is sufficient to show that $A_{q,I}$ is the unique feasible ν-centered assignment for $\Pi(X, q)$. Suppose A' is another such assignment. There is at least one variable x_j such that $A'(x_j) > A_{q,I}(x_j)$ because $A_{q,I}$ and A' are not equal but have the same sum. So, $A'(x_j) > x_j^{\min}$ and from Def. 8 for A' we have $\forall i$ s.t. $A'(x_i) < A'(x_j) : A'(x_i) = x_i^{\max} \geq A_{q,I}(x_i)$. Then, consider any variable x_i with $A'(x_i) \geq A'(x_j)$ and assume that $A_{q,I}(x_i) > A'(x_i)$. In this case, $A_{q,I}(x_i) > A_{q,I}(x_j)$ and since $x_j^{\max} \geq A'(x_j) > A_{q,I}(x_j)$ Def. 8 for $A_{q,I}$ implies that $A_{q,I}(x_i) = x_i^{\min}$ which is not possible because $A_{q,I}(x_i) > A'(x_i)$. Therefore $\forall i$ s.t. $A'(x_i) \geq A'(x_j)$: $A'(x_i) \geq A_{q,I}(x_i)$. Thus, $\forall i = 1..n, i \neq j$: $A'(x_i) \geq A_{q,I}(x_i)$ and $A'(x_j) > A_{q,I}(x_j)$ so the sum of the elements of $A_{q,I}$ cannot be equal to the sum of the elements of A'. □

Next we propose to do the same thing for the more general problem $\Pi(X, [\ell_q, u_q])$.

Theorem 4. *Given $I \in \mathcal{I}(X)$ and $GC(I) = n \times ES(I)/(n - |M(I)|)$. We will denote by q-opt($\Pi(X, V(I))$) the value of $q \in V(I)$ for which the objective value of $\Pi(X, V(I))$ is optimal. Then*

(i) If $GC(I) \in V(I)$ then q-opt($\Pi(X, V(I))$) $= GC(I)$.

(ii) If $GC(I) > \max(V(I))$ then q-opt($\Pi(X, V(I))$) $= \max(V(I))$.

(iii) If $GC(I) < \min(V(I))$ then q-opt($\Pi(X, V(I))$) $= \min(V(I))$.

Proof. Consider $q \in V(I)$ and $q' \in V(I)$ with $q \neq q'$. From Theorem 3 $A_{q,I}$ is an optimal solution of $\Pi(X, q)$, $A_{q',I}$ is an optimal solution of $\Pi(X, q')$. We have $\sum(x_i - \frac{q}{n})^2 = (\sum x_i^2) - \frac{q^2}{n}$ and let $D(q, q') = \sum(A_{q,I}(x) - \frac{q}{n})^2 - \sum(A_{q',I}(x) - \frac{q'}{n})^2$. The values q and q' belong to $V(I)$ so $\forall x \in (L(I) \cup R(I))$: $A_{q,I}(x) = A_{q',I}(x)$. Therefore the sums of the squares for $A_{q,I}$ and $A_{q',I}$ differ only for the elements of $M(I)$. If $M(I) = \emptyset$ then $\min(V(I)) = \max(V(I))$ so $q' \neq q$ does not exist. Let $m = |M(I)|$ and $e = ES(I)$. For $A_{q,I}$ we have $x \in M(I) \Rightarrow A_{q,I}(x) = (q - e)/m$, so $\sum_{x \in M(I)} A_{q,I}(x)^2 = \sum_{x \in M(I)}((q - e)/m)^2 = \frac{1}{m}(q^2 + e^2 - 2qe)$. Thus $D(q, q') = \frac{1}{m}(q^2 + e^2 - 2qe) - \frac{q^2}{n} - \frac{1}{m}((q')^2 + e^2 - 2q'e) + \frac{(q')^2}{n}$ or $D(q, q') = \frac{1}{nm}((n - m)(q^2 - (q')^2) - 2ne(q - q')$. Let $q' = q - \alpha$ for some $\alpha \neq 0$ then $D(q, q - \alpha) = \frac{\alpha}{nm}[(n - m)(2q - \alpha) - 2ne] = \frac{2\alpha q(n-m)}{nm} - \frac{\alpha^2(n-m)}{nm} - \frac{2\alpha ne}{nm}$. Now, we can use this property to prove the theorem:

(*i*) Let $q = GC(I) = \frac{ne}{n-m}$ then $D(q, q - \alpha) = \frac{-\alpha^2(n-m)}{nm}$ therefore since $n > m$ for all α such that $(q - \alpha) \in V(I)$, $D(q, q - \alpha) < 0$ so q-opt($\Pi(X, V(I))$) $= q$.

(*ii*) Let $q = max(V(I))$ then $D(q, q - \alpha) = \frac{2\alpha max(V(I))(n-m)}{nm} - \frac{\alpha^2(n-m)}{nm} - \frac{2\alpha ne}{nm}$. We have $GC(I) = \frac{ne}{n-m} > max(V(I))$ and $n > m$ and $q - \alpha < max(V(I)) \Rightarrow \alpha > 0$ so $\frac{2\alpha max(V(I))(n-m)}{nm} < \frac{2\alpha(ne/(n-m))(n-m)}{nm} = \frac{2\alpha ne}{nm}$. Therefore $D(q, q - \alpha) < \frac{-\alpha^2(n-m)}{nm} < 0$ because $n > m$. So q-opt($\Pi(X, V(I))$) $= q = max(V(I))$.

(*iii*) Let $q = min(V(I))$ then $D(q, q - \alpha) = \frac{2\alpha min(V(I))(n-m)}{nm} - \frac{\alpha^2(n-m)}{nm} - \frac{2\alpha ne}{nm}$. We have $GC(I) = ne/(n - m) < min(V(I))$ and $(n - m) > 0$ and $q - \alpha > min(V(I)) \Rightarrow \alpha < 0$ so $\alpha(n - m) < 0$ and $\frac{2\alpha min(V(I))(n-m)}{nm} < \frac{2\alpha(ne/(n-m))(n-m)}{nm} = \frac{2\alpha ne}{nm}$. Therefore $D(q, q - \alpha) < \frac{-\alpha^2(n-m)}{nm} < 0$ because $n > m$. So q-opt($\Pi(X, V(I))$) $= q = min(V(I))$. □

Corollary 1 *Theorem 4 holds if $V(I)$ is replaced by $V(I) \cap [\ell_q, u_q]$ provided $V(I) \cap [\ell_q, u_q] \neq \emptyset$.*

5.2 Computing the Optimal Value

Given $\mathcal{I}$ and the x_i's sorted according to their bounds, Algorithm 1 computes q-opt($\Pi(X, V(I))$) and opt($\Pi(X, V(I))$) for all $I \in \mathcal{I}$ in linear time. Following the notation of Lemma 2 we have $p_k = |L(I_k)| - |L(I_{k-1})|$ and $q_k = |R(I_{k-1})| - |R(I_k)|$.

The two steps of the algorithm before the loop can certainly be performed in $O(n)$. We argue that each iteration of the loop can be computed in $O(p_k + q_k)$ time. Sets $L(I_k)$, $R(I_k)$, and $M(I_k)$ are obtained in p_k, q_k, and $p_k + q_k$ steps

Compute $L(I_1)$, $R(I_1)$, $M(I_1)$, and $ES(I_1)$;
Compute q-opt($\Pi(X, V(I_1))$) and opt($\Pi(X, V(I_1))$) using Th. 4 and Def. 11 and 7;
for $k = 2$ to $|\mathcal{I}|$ **do**
  $L(I_k) \leftarrow L(I_{k-1}) \cup \{x \mid x^{\max} = \max(I_{k-1})\}$;
  $R(I_k) \leftarrow R(I_{k-1}) \setminus \{x \mid x^{\min} = \max(I_{k-1})\}$;
  $M(I_k) \leftarrow M(I_{k-1}) \setminus \{x \mid x^{\max} = \max(I_{k-1})\} \cup \{x \mid x^{\min} = \max(I_{k-1})\}$;
  $ES(I_k) \leftarrow ES(I_{k-1}) + (p_k - q_k) \times \max(I_{k-1})$;
  $V(I_k) \leftarrow [ES(I_k) + \min(I_k) \times |M(I_k)|, ES(I_k) + \max(I_k) \times |M(I_k)|]$;
  $GC(I_k) \leftarrow n \times ES(I_k)/(n - |M(I_k)|)$;
  Compute opt($\Pi(X, V(I_k))$) and opt($\Pi(X, V(I_k))$) using Th. 4, Def. 11 and 7;

Algorithm 1: Computing q-opt($\Pi(X, V(I))$) and opt($\Pi(X, V(I))$) for all $I \in \mathcal{I}$.

respectively, which correspond to the number of elements added or deleted (each is obtained in constant time since the x_i's are sorted). From Lemma 2, $ES(I_k)$ can also be computed in $p_k + q_k$ steps. When all these values are known, $V(I_k)$ and $GC(I_k)$ can be computed in $O(1)$ so from Theorem 4 q-opt($\Pi(X, V(I_k))$) can be computed in $O(1)$. In addition, opt($\Pi(X, V(I_{k-1}))$) is known and $A_{q,I_{k-1}}$ has $p_k + q_k$ values different from A_{q',I_k} so opt($\Pi(X, V(I_k))$) can be computed with $O(p_k + q_k)$ operations using the formula $\sum(x_i - q)^2 = \sum(x_i)^2 - \frac{q^2}{n}$. Since $\sum_{k=1}^{n} p_k = n$ and $\sum_{k=1}^{n} q_k = n$, the total amount of time to compute q-opt($\Pi(X, V(I))$) and opt($\Pi(X, V(I))$) for all $I \in \mathcal{I}$ is in $O(n)$.

Therefore, if we are provided with a maximum value $\pi^{\max}$ for $\Pi(X, [\ell_q, u_q])$ then we can reduce the interval $[\mu^{\min}, \mu^{\max}]$ for μ since $q = n\mu$. Such a value can be easily obtained from σ^{max} because from Def. 2 and Def. 7 we have the relation $n(\sigma^{max})^2 = \pi^{max}$.

5.3 Bounds Reduction

We consider a variable x of X and we study the consequences of the modifications of the bounds of x. Of course if there is an interval I for which $A_{q,I}(x) = x^{\min}$ and opt($\Pi(X, V(I))$) is consistent with $\pi^{\max}$ (i.e. less than or equal to $\pi^{\max}$) then there is no need to consider any modification of the minimum, and the same reasoning can be applied to $x^{\max}$.

For a given interval I, we know how to compute efficiently the optimal solution opt($\Pi(X, V(I))$). Thus, we can study the consequences of the modification of the bounds of x for this interval, that is searching what are the minimum and the maximum values that x can take while opt($\Pi(X, V(I))$) $\leq \pi^{\max}$. Then, we can repeat this process for all the intervals. Efficiently computing the new possible bounds of x is not obvious because when x is changing the possible sum of the variables is also changing and this impacts the value $GC(I)$, and the optimal value of $\Pi(X, V(I))$ depends on it. The following propositions show how to compute them. For convenience let I be an interval, $m = |M(I)|$, $e = ES(I)$, $\delta = \pi^{\max} - \text{opt}(\Pi(X, V(I)))$. and $sol(a, b, c) = \frac{-b + \sqrt{b^2 - ac}}{a}$.

Proposition 2 *Given $x \in R(I)$, let $\Pi(X', V'(I))$ be the problem obtained by setting $x' = x + d$, $V'(I)$ and $GC'(I)$ be the corresponding quantities for X'.*

(i) If $GC(I) < \min(V(I))$ then
$GC'(I) < \min(V'(I))$ with $d < d_1 = \frac{n-m}{m}(\min(V(I)) - GC(I))$ and $\max(d) = sol(a_1, b_1, c_1)$, with $a_1 = 1 - \frac{1}{n}$, $b_1 = x - \frac{\min(V(I))}{n}$, $c_1 = -\delta$

(ii) If $\min(V(I)) \leq GC(I) < \max(V(I))$ then
$\min(V'(I)) \leq GC'(I) < \max(V'(I))$ with $d < d_2 = \frac{n-m}{m}(\max(V(I)) - GC(I))$ and $max(d) = sol(a_2, b_2, c_2)$, with $a_2 = 1 + \frac{m-n}{(n-m)^2}$, $b_2 = \frac{mES(I)}{n-m} + x - \frac{GC(I)}{n-m}$, $c_2 = -\delta$

(iii) If $GC(I) \geq \max(V(I))$ then
$GC'(I) \geq \max(V'(I))$ with $max(d) = sol(a_3, b_3, c_3)$, with $a_3 = a_1$, $c_3 = c_1$ and $b_3 = x - \frac{\max(V(I))}{n}$

Proof. *(i)* $x \in R(I)$, so $ES'(I) = ES(I) + d$, $V'(I) = V(I) + d$, $M'(I) = M(I)$ and $GC'(I) = GC(I) + nd/(n-m)$. Then $GC'(I) < \min(V'(I))$ if $GC(I) + nd/(n-m) < \min(V(I)) + d$ that is $d < \frac{n-m}{m}(\min(V(I)) - GC(I))$. The optimal value for q' is $\min(V'(I)) = q + d$. Consider $opt' = \Pi(X', V'(I))$. Then $opt' = \sum_{j \neq i}(x'_j)^2 + (x'_i)^2 - \frac{q'^2}{n}$. In addition, $\forall x'_j \in R(I) \cup L(I)$ with $x'_j \neq x$: $x'_j = x_j$ and $\forall x'_j \in M'(I)$: $x'_j = (q' - ES'(I))/m = q - ES(I)/m = x_j$. Then, $opt' = \sum_{j \neq i}(x_j)^2 + (x_i + d)^2 - \frac{(q+d)^2}{n}$. So, the value of d for which $opt' = \pi^{max}$ is a root of the equation: $(1 - \frac{1}{n})d^2 + 2d(x - \frac{q}{n}) - \delta = 0$, which has only one positive root.

(ii) similar as *(i)* excepted that the variables of $M(I)$ have no longer the same value. If $x_j \in M(I)$ then $x'_j = (GC'(I) - ES'(I))/m = x_j + d/(n-m)$.

(iii) similar as *(i)* excepted that $q = \max(V(I))$. □

From this proposition and for a given interval I and a given variable $x \in R(I)$, we can define Function *compute-d* (see Algorithm 2) which computes the greatest possible value of d. It is called with x^{min} as parameter for x. Its time complexity is in $O(1)$ because the recursive call in line ln1 does not satisfy the *(i)* and the recursive call in line ln2 does not satisfy neither *(i)* or *(ii)*.

The following proposition just mirrors the previous one and an algorithm similar to Algorithm 2 can be derived from it.

Proposition 3 *Given $x \in L(I)$, let $\Pi(X', V'(I))$ be the problem obtained by setting $x' = x - d$, $V'(I)$ and $GC'(I)$ be the corresponding quantities for X'.*

(i) If $GC(I) > \max(V(I))$ then
$GC'(I) > \max(V'(I))$ with $d < d_1 = \frac{n-m}{m}(GC(I) - \max(V(I)))$ and $max(d) = sol(a_1, -b_1, c_1)$, with a_1, b_1 and c_1 as defined in Prop.2.(i).

(ii) If $\max(V(I) \geq GC(I) > \min(V(I))$ then
$\max(V'(I)) \geq GC'(I) > \min(V'(I))$ with $d < d_2 = \frac{n-m}{m}(GC(I) - \min(V(I)))$ and $max(d) = sol(a_2, -b_2, c_2)$, with a_2, b_2 and c_2 as defined in Prop.2.(ii).

(iii) If $GC(I) \leq \min(V(I))$ then
$GC'(I) \leq \min(V'(I))$ with $max(d) = sol(a_3, -b_3, c_3)$, with a_3, b_3 and c_3 as defined in Prop.2.(iii).

Function *compute*-d$(V(I), ES(I), GC(I), m, x)$: number
if $GC(I) < \min(V(I))$ **then**
  Compute $max(d)$ as indicated in Proposition 2(i);
  $d_1 \leftarrow \frac{n-m}{m}(\min(V(I)) - GC(I))$
  if $max(d) < d_1$ **then**
    Return $max(d)$;
  else
    $x' \leftarrow x + d_1$; $V'(I) \leftarrow V(I) + d_1$; $ES'(I) \leftarrow ES(I) + d_1$; $GC'(I) \leftarrow \min(V'(I))$
    ln1: Return d_1+ *compute*-d$(V'(I), ES'(I), GC'(I), m, x')$;
if $\min(V(I)) \leq GC(I) < \max(V(I))$ **then**
  Compute $max(d)$ as indicated in Proposition 2(ii);
  $d_2 \leftarrow \frac{n-m}{m}(\max(V(I)) - GC(I))$;
  if $max(d) < d_2$ **then**
    Return $max(d)$;
  else
    $x' \leftarrow x + d_2$; $V'(I) \leftarrow V(I) + d_2$; $ES'(I) \leftarrow ES(I) + d_2$; $GC'(I) \leftarrow \max(V'(I))$
    ln2: Return d_2+ *compute*-d$(V'(I), ES'(I), GC'(I), m, x')$;
Compute $max(d)$ as indicated in Proposition 2(iii) and Return $max(d)$;

Algorithm 2: Adjusting the upper bound of $x \in R(I)$.

When $x \in M(I)$ the problem is more complex because if x is modified then the number of variables in $M(I)$ is also modified:

Proposition 4 *Given $x \in M(I)$, let $\Pi(X', V'(I))$ be the problem obtained by setting $x' = x + d$, $V'(I)$ and $GC'(I)$ be the corresponding quantities for X'.*
(i) *If $GC(I) < \min(V(I))$ then*
$GC'(I) < \min(V'(I))$ *with* $d < d_1 = \frac{n-m+1}{m-1}[(\min(V(I)) - GC(I)) + \frac{ne}{(n-m)(m-1)}]$ *and* $max(d) = sol(a, b, c)$, *with* $q = \min(V(I))$ *and* $a = 1 - \frac{1}{n} + \frac{m}{(n-m+1)^2}$, $b = \frac{em}{(n-m)(n-m+1)} - \frac{em}{(n-m)(n-m+1)^2} - \frac{q}{n} + x$, $c = -\frac{m2e^2}{(n-m)^2(n-m+1)} + \frac{me^2}{(n-m)^2(n-m+1)^2} - \delta$
(ii) *If $\min(V(I)) \leq GC(I) < \max(V(I))$ then*
$\min(V'(I)) \leq GC'(I) < \max(V'(I))$ *with* $d < d_2 = \frac{n-m+1}{m-1}[(\max(V(I)) - GC(I)) + \frac{ne}{(n-m)(m-1)}]$ *and* $max(d) = sol(a, b, c)$, *with* $a = 1 + \frac{m-n}{(n-m+1)^2}$, $b = x + \frac{em}{(n-m)(n-m+1)} - \frac{GC(I)}{n-m+1} + \frac{e}{(n-m+1)^2}$, $c = -\frac{e^2}{(n-m)(n-m+1)^2} + \frac{2GC(I)e}{(n-m)(n-m+1)} - \frac{2me^2}{(n-m)^2(n-m+1)} - \delta$
(iii) *If $GC(I) \geq \max(V(I))$ then*
$GC'(I) \geq \max(V'(I))$ *with* $max(d) = sol(a, b, c)$ *of (i) with* $q = \max(V(I))$.

From this proposition we can derive a function which computes the maximum value of d. This function is slightly different from the one of Algorithm 2, because if $x \in M(I)$ then after modifying x we have $x \in R(I)$. So after a modification the proposed function directly calls Function *compute-d* of Algorithm 2.

Proposition 5 *Given $x \in M(I)$, let $\Pi(X', V'(I))$ be the problem obtained by setting $x' = x - d$, $V'(I)$ and $GC'(I)$ be the corresponding quantities for X'.*

(i) If $GC(I) > \max(V(I))$ *then*
$GC'(I) > \max(V'(I))$ *with* $d < d_1 = \frac{n-m+1}{m-1}[(GC(I) - \max(V(I))) + \frac{ne}{(n-m)(m-1)}]$ *and* $max(d) = sol(a_1, -b_1, c_1)$*, with* a_1, b_1 *and* c_1 *as defined in Prop.4.(i).*

(ii) If $GC(I) \in V(I)$ *then*
$GC'(I) \in V'(I)$ *with* $d < d_1 = \frac{n-m+1}{m-1}[(GC(I) - \min(V(I))) + \frac{ne}{(n-m)(m-1)}]$ *and* $max(d) = sol(a_2, -b_2, c_2)$*, with* a_2, b_2 *and* c_2 *as defined in Prop.4.(ii).*

(iii) If $GC(I) < \min(V(I))$ *then*
$GC'(I) < \min(V'(I))$ *with* $max(d) = sol(a_3, -b_3, c_3)$*, with* a_3, b_3 *and* c_3 *as defined in Prop.4.(iii).*

We can derive a similar algorithm from the previous propositions as we did from Proposition 4. Then, for each $x \in X$ we can compute for every interval $I \in \mathcal{I}$ the minimum and the maximum values of x denoted by $\min(x)$ and $\max(x)$ such that $\text{opt}(\Pi(X, V(I))) \leq \pi^{\max}$. By taking the minimum value of $\min(x)$ among the values computed for every interval we obtain the new minimum value of $D(x)$, and by taking the maximum value of $\max(x)$ among the values computed for every interval we obtain the new maximum value of $D(x)$. Since the number of intervals is at most n, this process takes $O(n)$ time per variable. So we can achieve bounds consistency on the variables of X is in $O(n^2)$.

6 Conclusion

This paper introduced a new constraint to express balance among n variables in constraint programming models. It is based on the notions of mean, median, and standard deviation from statistics. We gave several examples showing how balance can be formulated with this constraint. Two efficient filtering algorithms were given. The first one runs in $\mathcal{O}(n)$ time under a reasonable assumption. The second one achieves bounds consistency in $\mathcal{O}(n^2)$ time.

Acknowledgments

This research was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC), ILOG S.A., and the Intelligent Information Systems Institute at Cornell University.

References

1. Problem 30 of CSPLIB. (`www.csplib.org`).
2. I. Katriel and S Thiel. Fast Bound Consistency for the Global Cardinality Constraint. In *Proc. CP 2003*, pages 437–451. Springer-Verlag LNCS 2833, 2003.
3. M. Lemaître, G. Verfaillie, and N. Bataille. Exploiting a Common Property Resource under a Fairness Constraint: a Case Study. In *Proc. IJCAI*, Stockholm, Sweden, 1999.
4. C.-G. Quimper, P. van Beek, A. López-Ortiz, A. Golynski, and S. B. Sadjad. An Efficient Bounds Consistency Algorithm for the Global Cardinality Constraint. In *Proc. CP 2003*, pages 600–614. Springer-Verlag LNCS 2833, 2003.