INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Multi-Agent Electro-Location and the "Among" Constraint

Gilles Chabert — Frédéric Boyer — Sophie Demassey

N° 7640

May 2011

Programs, Verification and Proofs

ISSN 0249-6399 ISRN INRIA/RR--7640--FR+ENG

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

centre de recherche
RENNES - BRETAGNE ATLANTIQUE

Multi-Agent Electro-Location and the "Among" Constraint

Gilles Chabert*† , Frédéric Boyer*‡, Sophie Demassey*†

Theme : Programs, Verification and Proofs
Équipe-Projet TASC

Rapport de recherche n° 7640 — May 2011 — 17 pages

Abstract: In this paper, we give a new approach for the localization of several autonomous fish robots equipped with the *electric sense*. The approach is based on interval arithmetic and constraint programming. It introduces a generalization of the `among` constraint which was used so far in the different context of resource allocation in logistics. We prove that the bound consistency for a conjunction of `among` constraints with interval domains is NP-complete in the vector case and polynomial in the one-dimensional case.

Although it was designed for electric fish robots, this work is not restricted to this type of systems. The approach can be easily extended to other multi-agent systems with low-range sensing.

Key-words: Artificial intelligence, autonomous robotics, constraint programming, interval arithmetic, continuous global constraints

* Mines de Nantes, 4 rue Alfred Kastler 44300 Nantes, France
† Equipe-Projet TASC, LINA, CNRS UMR 6241
‡ Equipe Robotique, IRCCYN, CNRS UMR 6597

Centre de recherche INRIA Rennes – Bretagne Atlantique
IRISA, Campus universitaire de Beaulieu, 35042 Rennes Cedex
Téléphone : +33 2 99 84 71 00 — Télécopie : +33 2 99 84 71 71

Electro-Localisation Multi-Agent et la Contrainte "Among"

Résumé : Cet article propose une nouvelle approche pour le problème de la localisation de robots autonomes sous-marins dotés du sens électrique. Basée sur l'arithmétique d'intervalles et la programmation par contraintes, elle considère une généralisation de la contrainte `among`, jusqu'ici utilisée principalement dans un contexte logistique (allocation de ressources).

Nous prouvons que le calcul de la cohérence aux bornes pour une conjonction de contraintes `among` dont les domaines des valeurs forment des intervalles est NP-complet dans le cas général et polynomial en une dimension.

Bien que ce travail ait été conduit pour des robots sous-marin éléctriques, son cadre d'application dépasse ce type de technologie. L'approche peut en effet être facilement étendue à d'autres systèmes multi-agents dont les capteurs sont à faible portée.

Mots-clés : Intelligence artificielle, robotique autonome, programmation par contraintes, arithmétique d'intervalles, contraintes globales continues

1 Introduction

This work has been conducted in the ANGELS[1] research project on bio-inspired anguiliform robots with electric sense [12]. A typical mission for these robots would be to explore inundated underground cavities that are unreachable by humans.

Three important features have been developed to this aim.

First, their eel-like shape and ability to swim make them possible to travel through the galleries, even in case of muddy water.

Second, robots are equipped with electric sense. The electric sense means the ability to emit an electric field inside the water and to measure the intensity (or voltage) at different points on the body. This intensity varies when surrounding objects distort the electric field. So this variation can be used to feel the presence of objects. The electric sense is perhaps the only sense adapted to confined and dark underwater environments, sonar technologies being only adequate in large spaces with few unevenness.

Third, a big robot can split into smaller agents that can move and feel the environment independently. Small agents are useful for locomotion to cross obstacles and also for perception, as the following scenario illustrates.

Figure 1 depicts a typical scenario: the robot dives into the main hole. It swims across the galleries until it falls on a cavity. It then splits into small robots that explore the cavity for a fixed period of time. Once the time is out, the exploration is over and the robot has to return. Hence, everytime two agents meet, they dock to each other and once they are all pieced together again, the robot starts looking for the way out. It then swims back to the surface.

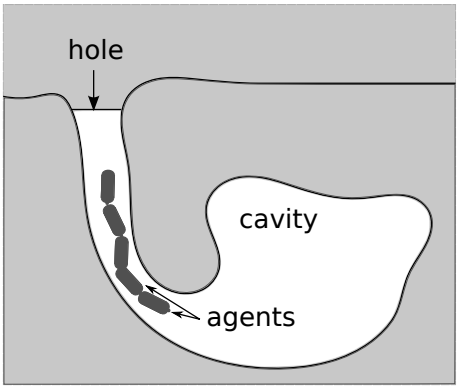

Figure 1: **Scenario for Electro-Location.**

Of course, there are many different issues (in mechatronics, fluid mechanics, control engineering, etc.) involved in such a scenario. We will focus here on one of them: the *multi-agent electro-location*.

[1] http://www.theangelsproject.eu.

1.1 Objective and Main Difficulties

After the mission, all the data collected by the sensors are processed. Data include, at each time step, the electric intensity as well as proprioceptive data (the velocity and angular speed of the agent measured by a gyrocompass).

The goal is typically to see if some object of interest (say, a corpse) is lying somewhere in the cavity.

But locating an object requires first to localize the robots themselves. So a big concern – called *electro-location* – is to localize the robots and this is the subject of this paper.

This challenge has specific aspects due to our context. First, there are significant uncertainties on all the measurements, including the proprioceptive ones. Odometry is therefore limited and robots get quickly lost. Second, the electric sense gives only a very indirect information of the environment, namely, variation of intensities (see next section). We cannot easily extract positions of surrounding object as we do when looking at a camera image or sonar waterfall. Finally, there are several agents so we have to cross the information retrieved by the different agents to improve globally our knowledge of the trajectories.

In this paper, we propose a new approach that integrates all these aspects. The main innovation is in the way we make agents interact via a *global constraint*.

2 Electro-Location

Localization based on perception is traditionally addressed by two classes of methods: *parametric methods* and *machine learning methods*. We survey both of them in this section and give their advantages and limitations:

2.1 Parametric methods

They are based on the following principles. First, we define a vector $\vec{x}$, the state of an agent. This vector typically includes the Cartesian coordinates and the orientation. The dimension d of $\vec{x}$ usually corresponds to the number of degrees of freedom.

Second, we have to extract from the signals the states of the observed objects relative to that of the sensor. For instance, if an agent A detects two objects O and O', the vector of relative states is $\vec{p} = (\vec{x}_O - \vec{x}_A, \vec{x}_{O'} - \vec{x}_A)$. It has $2d$ components. We call $\vec{p}$ a vector of *parameters*. Third, we have to process all these vectors of parameters obtained from the signals in order to get relevant estimations of the different states.

Parametric methods used for electro-location include the Kalman filter [2, 14], particle filters [21]. One could also resort to meta-heuristics or interval-based constraint methods (see §3.1). Bayesian filters are usually very good for the second step, for obtaining an accurate estimation of parameters in case of repeated detections (e.g, tracking) while interval methods are rather good for the third step, that is, for propagating estimations of parameters globally.

In the context of electrolocation, all these approaches suffer from important limitations.

1. To extract the vector $\vec{p}$ from the measured signal I, we need explicit relations between them. The theory of electrostatics can provide formulas $I = f(\vec{p})$ but only in specific cases. For instance, Rasnow [16] gave formula for the

electric response of a sphere. More recently, combinations of spheres and walls have been taken into account [2]. But there is no such model for more complex objects.

2. Even in the case of simple object types, models are nonlinear and get quickly intractable as the number of objects increase. Unfortunately, Baysian filters need initial estimations of parameters each time new objects appear. They are therefore very hard to obtain, especially when multiple objects are involved in the scene, and require heavy optimization procedures. Interval methods can be applied but they are unable to provide sharp enclosures of the estimates. Introducing branching does not help much and leads to desastrous computation time.[2]

3. Even if models are known and presumably easy to initialize, we still have to know *which* model applies. Indeed, models change with time as the agent meets new objects and we cannot reasonably run in parallel the models obtained for all possible combinations of objects in the scene. In the publications on electro-location cited above, models are forced manually.

2.2 Non-parametric methods

Non-parametric methods follow a radically different principle. They do not handle explicitly a state $\vec{x}$ as above and – as their name tells – they do not require a model. Let us take a simple example. Assume we want to distinguish a cube from an hexagon using the electric sense of an agent. The idea is to use machine learning algorithms in order to *train* a program that will have to make the right choice.

Non-parametric methods differ from the type of algorithms used to train and the decision procedure. *Classification* techniques [1] are based on raw flat data while *artificial neural networks* [5] require some structure on the measurements. The classification approach applies sophisticated deduction schemes while building a neural network gives directly the desired program. To some extent, non-parametric methods are more bio-inspired since animals do not have models but a long-time training instead.

We easily see that the two first limitations mentioned for parametric methods do not carry over the non-parametric ones. On the other hand, the latters can only make decision between a very limited number of scenarios obtained after training campains. Furthermore, if they are the best to make qualitative choice as above, they are clearly not adequate for providing accurate or reliable numerics.

2.3 A compromise

The approach we decided to adopt is somehow a compromise. We still have a state $\vec{x}$ but do not consider *exact* models. We have said that applying the laws of electrostatics yields complex formulas when considering a specific scene. However, there exists coarse –but general– rules that can be applied everywhere. Let A be an agent. We know that the variation of the signal measured by A and due to the presence of a remote object is *in the order of* $1/d^k$ with $k = 3$ and where d is the distance separating the object from A.

[2] Even two walls are enough for this to happen (this has been tested experimentally).

There is also an additive rule. If A is surrounded by n other objects at distances $d_1, \ldots, d_n$ from A respectively, then the current I measured by A will satisfy

$$I \in \Big(\sum_{i=1}^{n} \frac{1}{d_i^k}\Big)[\underline{\alpha}, \overline{\alpha}] \tag{1}$$

where $[\underline{\alpha}, \overline{\alpha}]$ is an interval of coefficients (in $\mathbb{R}$) that basically encompasses all the information we want to ignore, that is, the specific shape of an object, the conductivity of water, the orientation of the object with respect to A, etc. Of course, this interval coefficient has to be calculated experimentally. Should it be large, it becomes small as compared to $1/d^k$ when d approaches zero. That is our point: an agent can *feel* an approaching object while being unable to determine its shape and the more the exponent k, the less significant the coefficient. That is why this approach works better for low-range sensors.

This principle is not a novelty and is already typically used for control [2]. Indeed, it is clear that if we ask robots to avoid walls of the cavity, we do not ask them to identify their shape. So obstacle avoidance commands work in this way.

It is less evident, however, that localization methods can also be built upon the same basis. Jaulin *and al.* has already pioneered localization algorithms with this type of rules [9]. Note however that a relation like (1) was used to create a map, not for the inter-location of agents. The originality of the present work is in the introduction of global constraints for this purpose. We need now to place ourselves in the setting of *constraint programming*. The key idea of the paper will be exposed in §3.3.

3 Constraints

3.1 The Constraint Paradigm

Constraint programming [7, 19] is a generic framework for solving combinatorial problems that can be modeled as a set of variables taking their values inside given domains and linked by constraints. We try, in this section, to give the intuition of how it works in a few lines. More basic material can be found in one of the references given above.

In this paradigm, a problem is viewed as an hypergraph where nodes represent variables and constraints hyperedges. Each contraint c is associated to a filtering algorithm that removes values in the domains of the adjacent variables. A value v is removed from the domain of a variable x only if c is not satisfied when $x = v$ whatever value the other variables take in their respective domains. Hence, removing v from the domain is safe with respect to the overall problem. Enforcing *generalized arc consistency* (`GAC`) on constraint c is to remove all infeasible values from the domains of the constraint variables. As a cheaper alternative, enforcing *bound consisteny* is to check and potentially filter only the bounds of the variable domains.

The idea is then to propagate in the hypergraph the domain reductions made on some constraint by enforcing consistency on the other constraints adjacent to the impacted variables.

Filtering algorithms exist for isolated mathematical relations but also for conjunctions that form specific patterns commonly found in problems. For

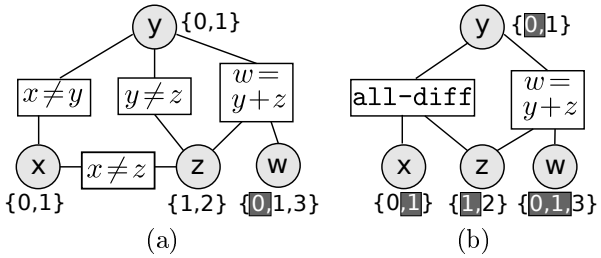

Figure 2: **`GAC` and propagation with 2 different models of the same problem.** Rectangles represent the hyperedges and domains are in braces. We enforce `GAC` on each constraint and loop until the fixpoint is reached. Values removed by this process are highlighted. (a) Only 0 is removed from the domain of w. (b) A significantly better filtering is gained by introducing the `all-diff` global constraint.

instance, the conjunction of $x \neq y$, $y \neq z$ and $z \neq x$ can be modeled using one *global constraint* named `all-diff` [17]. Enforcing `GAC` on `all-diff`(x, y, z) requires a more sophisticated algorithm than propagating on the inequalities individually. However, it may offer a deeper reduction, as shows example of Figure 2.

Initially devoted to discrete optimization, constraint programming has been extended successfully to nonlinear constraints over the reals thanks to the integration of symbolic processing and interval analysis (see [6, 10] or [19], chap.16).

3.2 Notations

Small letters like x and $\vec{x}$ denote real numbers and vectors respectively.

Capital letters like X or $\vec{X}$ denote intervals in $\mathbb{R}$ and boxes in $\mathbb{R}^d$ respectively. The lower and upper bound of an interval X are $\underline{x}$ and $\overline{x}$. A box may correspond to the domain of a variable (e.g., $\vec{X}_1$) or a *value domain* V (see Definition 1).

Note that indices are used for enumerating in a collection of variables and domains, not for identifying a component. Hence, $\vec{X}_j$ is the j^{th} box in a collection of boxes, not the j^{th} component of a box $\vec{X}$ (we shall not need a notation to designate components).

In a multi-agent scenario as above, we have N agents with d coordinates. Thus, at each time t, we consider an array of d-dimensional variables $\vec{x}_1(t)$, ..., $\vec{x}_N(t)$ with domains $\vec{X}_1(t), \ldots, \vec{X}_N(t)$.

3.3 Main Idea

Let us start by a trivial fact: if an agent does not move, it cannot get lost.[3]

Imagine now that when entering the cavity, some agents halt while the others start exploring the area. We will see that stationary agents can serve as beacons for the moving ones. Let us call the formers *beacons* and the latters *explorers*.

[3]This sounds obvious at least for ground robots. Underwater, the robot may be carried by the current. Fortunately, there is no current in the type of environment where the electric sense is needed. Furthermore, the agent can adopt a looping behavior with all its actuators blocked. It is then guaranteed to describe circles with almost no drifting.

When an explorer is detected by a beacon[4], we know that the position of the explorer is in the range of the beacon so we can refine the estimation of its position. However, things are not so easy because when a beacon detects something, we don't know what it is. And since explorers get quickly lost, their estimated positions do not help much neither.

Worse, if a beacon is stuck to a wall or to another beacon, it will constantly receive a high noise that will make it just blind. So, for our idea to be applicable, beacons must actually not immediatly stop when entering the cavity. They have to move randomly during a short lapse of time, until they get far enough from each others, that is, until they are *electrically isolated*, that is, they measure a signal equal to the ambiant noise. Again, the low range of the electric sense makes this strategy realistic.

Consider now an isolated beacon A. As shown in Figure 3, we can easily derive from formula (1) a threshold $\hat{I}$ and an interval $D = [\underline{D}, \overline{D}]$ that fulfill the following property: when the signal I received by A is greater than $\hat{I}$, there is at least one object whose distance from A is less than $\overline{D}$. Conversely, when $I < \hat{I}$, there is no object whose distance from A is less than $\underline{D}$.

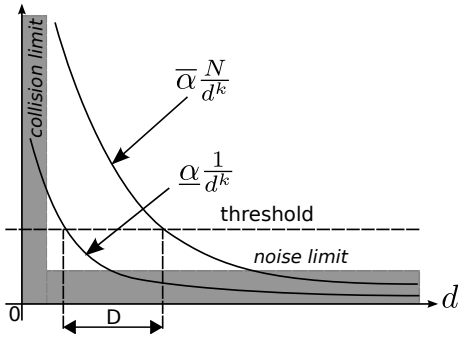

Figure 3: **Detection threshold**. The threshold is a value of the signal that fulfills two conditions: (1) to be significantly greater than the ambiant noise and (2) to be sufficiently low so that a single explorer can be detected before it goes away (to avoid the collision with the beacon). The distance interval D is then derived by considering the two extreme cases (1 or N agents approaching).

We need D to be calculated for the infinity norm distance $\|.\|_\infty$. In this way, we can build for each beacon A two boxes A^- and A^+ centered on the beacon position $\vec{x}_A$ and whose radii on each dimension are $\underline{D}$ and $\overline{D}$ respectively (see Figure 4 below). The previous rule then reads as follows: if $I > \hat{I}$ then at least one explorer is in A^+ and if $I < \hat{I}$ no explorer is in A^-. It turns out that a global constraint named `among` exactly matches such purpose. This constraint was used so far in a radicaly different context, namely, *resource allocation* in logicistics and the first contribution of this paper is to make connection between apparently unrelated fields.

Before getting into the details of the `among` constraint, we show an example of what we can infer on the trajectories.

[4] And not in the other way around; so the role are somehow inversed.

3.4 A Simulated Example

We consider the snapshot of a mission with 3 beacons and 2 explorers depicted in Figure 4. On the left side, the areas calculated from the detection threshold are represented. We assume that beacons 1 and 3 have detected something and not beacon 2. On the right side, the boxes with dashed contours represent the domains of the two explorers. For convenience, the boxes $\vec{A}_1^+$, $\vec{A}_2^-$ and $\vec{A}_3^+$ involved in the `among` constraints are also duplicated. The result of the filtering we can obtain on the positions of the explorers is hatched.

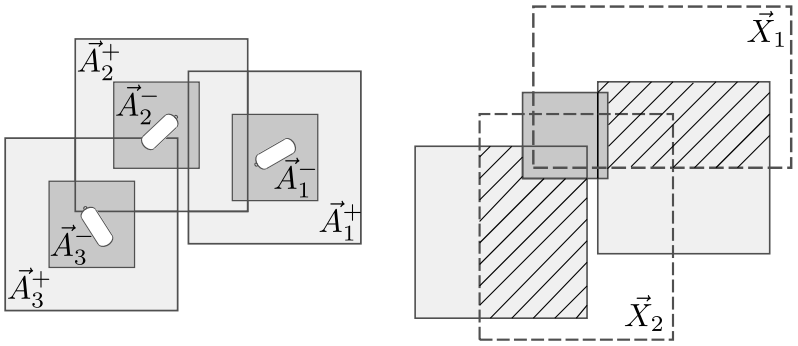

Figure 4: **Three beacons and two explorers.** (a) The areas calculated from the detection threshold. (b) The result of filtering on the domains of the two explorers (hatched boxes).

We reason as follows. Initially, both $\vec{X}_1$ and $\vec{X}_2$ intersect $\vec{A}_1^+$ and $\vec{A}_3^+$. However, since no point is allowed inside $\vec{A}_2^-$ we can narrow $\vec{X}_1$ accordingly. The new domain does not intersect anymore $\vec{A}_3^+$. Since at least one of the explorer is inside $\vec{A}_3^+$ it is necessarily $\vec{x}_2$ and the box $\vec{X}_2$ can be reduced to the hatched area on bottom left. Finally, since this new subbox does not intersect $\vec{A}_1^+$, we know that only $\vec{x}_1$ can be inside $\vec{A}_1^+$, which gives the top-right hatched area.

3.5 The model

The `among` constraint was first introduced in [3]. We give a definition of the `among` constraint directly in the vector case and with interval value domains.

Definition 1 Interval Among Constraint.

Given a box $\vec{V} \subset \mathbb{R}^d$ (the "value domain"), an integer interval $K = [\underline{k}, \overline{k}] \subset \mathbb{Z}^+$ a family of variables $(\vec{x}_j)_{j\in J}$, we say that

$$\texttt{among}(\ (\vec{x}_j)_{j\in J}, \vec{V}, K\ ) \quad \textit{holds iff}$$

$$\underline{k} \leq \left|\{j \in J, \vec{x}_j \in \vec{V}\}\right| \leq \overline{k}.$$

Now, at each time step t and for each beacon, one of the two following relations will be entered in our model. If the current measured by the beacon

is less than $\hat{I}$, then we have:

$$\texttt{among}\Big((\vec{x}_1(t),\ldots,\vec{x}_n(t)), A_i^+, [1,N]\Big),$$

where i is the number of the beacon, and otherwise:

$$\texttt{among}\Big((\vec{x}_1(t),\ldots,\vec{x}_n(t)), A_i^-, [0,0]\Big).$$

Of course, there is combinatory behind and constraint propagation will do part of the job. But we need to be more efficient by taking into account the fact that, at a given time step, all the beacons detect something or not. That is, the positions of the explorers are simultaneously constrained by m `among` constraints where m is the number of beacons. So we have to devise a filtering algorithm dedicated to a conjunction of `among` constraints. And since real domains are represented by intervals, our goal amounts therefore to calculate the bound consistency for a conjunction of `among` constraint with interval value domains.

This topic will be the subject of the next section, where the complexity of this problem is analyzed.

The question of the experimental validation of our model is discussed in the conclusion.

4 Complexity Analysis

The original among constraint was defined in the one-dimensional case $d = 1$, on variables with finite domains $X_j \subset \mathbb{Z}_+$, and on any finite value domain $V \subset \mathbb{Z}_+$, that is not necessarily an interval. Régin [18] proved that enforcing `GAC` on a conjunction of among constraints of this kind is NP-hard. In the polynomial transformation he proposed, the variables have interval domains. NP-hardness holds, *de facto*, also for the bound consistency.

Enforcing `GAC` becomes tractable on the `sequence` constraint [22] and some of its generalizations [15]. These constraints are particular conjunctions of overlapping `among` constraints sharing the same value domain V. In this case, variables domains can trivially be assimilated to $\{0,1\}$ (either $x_i \in V$ or $x_i \notin V$). Hence, `sequence` is a special case of ours.

In our context, we consider a conjunction of d-dimensional interval `among` constraints, and as our variables are real vectors, it is only required to enforce bound consistency on such a conjunction. We show then two results. First, the problem is untractable in the vector case, that is, when the dimension d of variables and values is at least two. Second, the problem is in $\mathcal{P}$ when $d = 1$.

4.1 The vector case, $d \geq 2$

Proposition 1 *The satisfiability of a conjunction of d-dimensional* `among` *constraints with interval domains for both variables and values is NP-complete for* $d \geq 2$.

Proof. A variable assignment is a certificate so the problem is in NP. We transform now the problem of the satisfiability of `atmost-nvector` with interval domains, which was proven to be NP-complete (see Section 4 in [4]). More precisely, let us consider the problem $\mathcal{P}_{atmost}$ defined as follows:

Input: n rectangles $\vec{X}_1, \ldots, \vec{X}_n$ and an integer k.
Question: Is there n 2-dimensional vectors $\vec{x}_1, \ldots, \vec{x}_n$ such that $\vec{x}_j \in \vec{X}_j$ for all j, $1 \leq j \leq n$, and $|\{\vec{x}_1, \ldots, \vec{x}_n\}| = k$?

We apply now the following transformation. Define k variables $\vec{y}_1, \ldots, \vec{y}_k$ with same domain $\vec{Y} \supseteq \bigcup_{j=1}^{n} \vec{X}_j$, and consider the new problem $\mathcal{P}_{among}$ obtained from the conjunction of $\texttt{among}((\vec{y}_1, \ldots, \vec{y}_k), \vec{X}_j, [1, k])$ for all j, $1 \leq j \leq n$. These constraints basically force the number of $\vec{y}_i$'s inside each of the $\vec{X}_j$'s to be at least one (i.e., the variable $\vec{x}_j$ to be instantiated in $\mathcal{P}_{atmost}$). It is clear that answering "yes" to the previous problem is equivalent to find a consistent instantiation of the $\vec{y}_i$'s with respect to this new problem. ■

4.2 The one-dimensional case, $d = 1$

We show now that checking – and then enforcing – bound consistency can be achieved in polynomial time in the one-dimensional case. Our reasoning is illustrated on the example depicted in Figure 5. It exploits the fact that both value domains $V_1, \ldots, V_m$ and variable domains $X_1, \ldots, X_n$ are intervals. This restriction allows a first simplification: we can discretize the problem by splitting up the real line on which variables take their values, as a sequence of disjoint elementary intervals such that each value domain V_i coincides exactly with the union of consecutive elementary intervals.

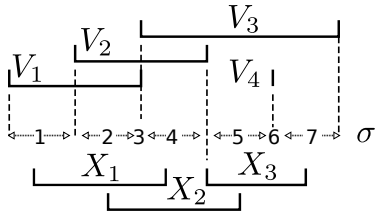

Figure 5: A conjunction of among constraints with interval domains for both values and variables.

Formally, consider T an interval of the real line which includes any value and variables domains, and for each $t \in T$, the maximal set $I_t \subseteq I$ of value domains covering t: $I_t = \{i \in I, t \in V_i\}$. As value domains are intervals, function $t \mapsto I_t$ is piecewise constant on T. Hence, T can be partitionned into the family of maximal intervals on which the function is locally constant. Let σ denote this partition and p its cardinality. First, σ is finite and has at most $2(m + 1)$ elements, since the bounds of the intervals in σ coincide with the bounds of T or that of the value domains $(V_i)_{i \in I}$. Second, the elements of σ are pairwise disjoint intervals, so it induces a natural linear order on σ. Last, each interval V_i spans one unique sub-sequence of elements in σ. In the example of Figure 5, there are $p = 7$ elementary intervals, and V_1 coincides with intervals $(1, 2, 3)$, V_2 with $(2, 4)$, V_3 with $(3, 4, 5, 6, 7)$, and V_4 with (6).

Our discretization consists to identify each interval V_i to a discrete set $V'_i \subseteq \sigma$ by setting $s \subseteq V_i \leftrightarrow s \in V'_i$, $\forall s \in \sigma$. With a simple change of variables, let us identify each real variable x_j to a discrete variable x'_j with domain σ such that $x_j \in s \leftrightarrow x'_j = s$, $\forall s \in \sigma$. Intuitively, the among constraint just counts the

number of variables set in some interval, it does not impose a variable to take a specific value.

Therefore, checking the **SA***tisfiability* of a **C***onjunction of* **AM***ong Constraints with* **I***nterval Domains* can be formally stated as a discrete decision problem:

Definition 2 SACAMI.
Input: A linearly ordered finite set σ of cardinality $p > 0$, two families $X = (X_j)_{j\in J}$ and $V = (V_i)_{i\in I}$ of intervals of σ, of cardinalities $|J| = n$ and $|I| = m$, and for each interval $V_i \in V$, a minimum capacity $\underline{k}_i \in \mathbb{Z}_+$ and a maximum capacity $\overline{k}_i \in \mathbb{Z}_+$ with $\underline{k}_i \leq \overline{k}_i$.
Question: Is there an n-tuple $x \in \Pi_{j\in J} X_j \subseteq \sigma^n$ such that, for each interval $V_i \in V$, the number of entries of x belonging to interval V_i does not exceed its capacity bounds:

$$\underline{k}_i \leq \mid \{j \in J \mid x_j \in V_i\} \mid \leq \overline{k}_i, \quad \forall i \in I \ ? \tag{2}$$

Now, our key idea is to consider the following decomposition of

$$(\texttt{among}(\ (x_j)_{j\in J}, V_i, [\underline{k}_i, \overline{k}_i]\ ), \forall i \in I):$$

$$\underline{k}_i \leq \textstyle\sum_{s\in V_i} y_s \leq \overline{k}_i, \qquad \forall i \in I \tag{3}$$

$$\wedge \quad \texttt{among}(\ (x_j)_{j\in J}, \{s\}, y_s\ ), \qquad \forall s \in \sigma. \tag{4}$$

Note that this decomposition hinders the propagation. Note also that both the discretization and the decomposition remain valid when value and variable domains are not intervals. However, because the variable domains are intervals, we can reformulate (4) as a maximal flow problem, and because the value domains are intervals, we can reformulate (3) and (4) as a matrix of linear inequalities satisfying the integrality property. The following proposition presents this reformulation as an integer linear program, as well as the underlying flow problem.

Proposition 2 *An instance $(\sigma, X, V, \underline{k}, \overline{k})$ is a yes-instance of* SACAMI *if and only if the following integer linear program P is feasible.*

$$(P): \ \max \sum_{s\in\sigma} y_s$$

$$s.t. \quad \underline{k}_i \leq \sum_{s\in V_i} y_s \leq \overline{k}_i, \qquad \forall i \in I, \tag{3}$$

$$L_{[a,b]} \leq \sum_{s\in[a,b]} y_s \leq U_{[a,b]}, \qquad \forall a \leq b \in \sigma, \tag{4'}$$

$$y_s \geq 0, \qquad \forall s \in \sigma.$$

where, for each non-empty interval $[a, b]$ of σ, $L_{[a,b]}$ is the number of intervals in X which are included in interval $[a, b]$, and $U_{[a,b]}$ is the number of intervals in X which intersect interval $[a, b]$:

$$\begin{aligned} L_{[a,b]} &= |\{j \in J \mid X_j \subseteq [a, b]\}|, \\ U_{[a,b]} &= |\{j \in J \mid X_j \cap [a, b] \neq \emptyset\}|. \end{aligned}$$

Proof. We shall prove that there is a mapping between the feasible solutions y of P and the feasible solutions x of SACAMI. Assume there exists $x \in \Pi_{j \in J} X_j \subseteq \sigma^n$ satisfying (2) and let y_s denote, for each value $s \in \sigma$, the number of entries in x which are equal to s:

$$y_s = |\{j \in J \mid x_j = s\}|, \quad \forall s \in \sigma.$$

Then, y is a feasible solution of P, as the satisfaction of constraints (3) directly arise from (2), and constraints (4') from $X_j \subseteq [a,b] \implies x_j \in [a,b]$ and $x_j \in [a,b] \implies X_j \cap [a,b] \neq \emptyset$.

Conversely, let y be a feasible solution of P. Note first that its objective value $\sum_{s \in \sigma} y_s$ is equal to n, since $L_\sigma = U_\sigma = n$. Consider the capacitated directed bi-partite graph $G = (J \times \sigma, E, c)$ on the arc set $E = \{(j,s) \in J \times \sigma \mid s \in X_j\}$ with capacity $c_e = 1$ on each arc $e \in E$. We add to G a source u and an arc (u,j) of capacity 1, for all $j \in J$, a sink v and an arc (s,v) of capacity y_s for all $s \in \sigma$ (see Figure 6). It is easy to see that every feasible (u,v)-flow of value n defines a feasible solution x of SACAMI, by setting x_j the flow on arc (j,s), for all $j \in J$. To prove there exists such a flow, we use the theorem of Hoffman and show that the capacity of any (u,v)-cutset (U,V) of G is greater than or equal to n [13]. Let (U,V) be a cutset of G, $\sigma_U = \sigma \cap U$, $J_U = J \cap U$, and $J'_U = \{j \in J_U \mid X_j \subseteq \sigma_U\}$. By definition of G, the arcs in the cutset (U,V) are of the form, either (u,j) with $j \in J \setminus J_U$ and capacity 1, or $(j,s) \in E$ with $j \in J_U$, $s \in \sigma \setminus \sigma_U$ and capacity 1, or (s,v) with $s \in \sigma_U$ and capacity y_s. The total capacity of the first set of arcs is $|J \setminus J_U|$. The capacity of the second set is at least $|J_U \setminus J'_U|$ since, for all $j \in J_U \setminus J'_U$, $X_j \not\subseteq \sigma_U$, then there exists at least one arc $(j,s) \in E$ in the cutset. Last, the capacity of the third set is at least $|J'_U|$ since, according to (4'):

$$\begin{aligned} |J'_U| &\leq |\{j \in J \mid X_j \subseteq \sigma_U\}| \\ &\leq \sum_{[a,b] \subseteq \sigma_U} |\{j \in J \mid X_j \subseteq [a,b]\}| \\ &\leq \sum_{[a,b] \subseteq \sigma_U} \sum_{s \in [a,b]} y_s = \sum_{s \in \sigma_U} y_s. \end{aligned}$$

Hence, the capacity of the cutset is at least n and the result follows. ■

We remark that the proof remains true when relaxing in program P every constraint in (4') corresponding to some interval $[a,b]$ that does not include any variable domain X_j. We can still decrease the number of constraints in P by merging every constraint in (3) to the constraint in (4') corresponding to the same interval $[a,b] = V_i$, $i \in I$, as follows:

$$L'_{V_i} = \max(L_{V_i}, \underline{k}_i) \leq \sum_{s \in V_i} y_s \leq \min(U_{V_i}, \overline{k}_i) = U'_{V_i}.$$

After this simplification, the program P has then at most $(p(p+1)/2) = O(m^2)$ constraints and $p = O(m)$ variables.

Proposition 3 *The integer linear program P can be solved in polynomial time.*

Proof. By ordering the constraints, program P can be re-written in matrix notation as:

$$(P) : \max\{\ \vec{1}y \mid Ay \leq U, -Ay \leq L, y \geq 0\ \}$$

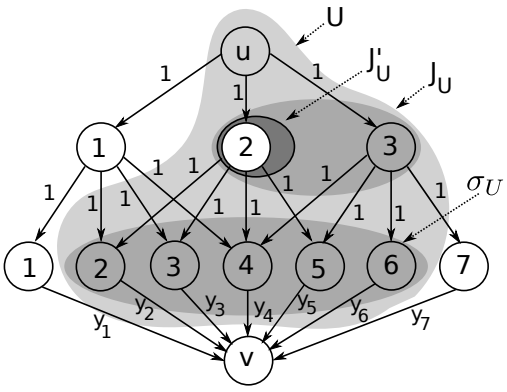

Figure 6: **The network flow model of constraints** (4) corresponding to the example of Figure 5, and an example of a cutsest used in the proof of Proposition 2. The cutset U is painted in light gray. The subset of nodes J_U and σ_U are in medium gray and J'_U in dark gray.

where A is an $(p(p+1)/2) \times p$ matrix, having the *consecutive ones property for rows*, since its entries are zeros and ones and the ones in each row occur consecutively. Therefore A is totally unimodular [20], so as the coefficient matrix $\begin{pmatrix} A \\ -A \end{pmatrix}$ of P. The theorem of Hoffman and Kruskal implies that the optimal solutions of P coincide with the optimal solutions of its LP-relaxation and this latter problem is solvable in polynomial time [20]. ■

Clearly our reformulation can be built in polynomial time, then:

Corollary 1 SACAMI *is in* $\mathcal{P}$.

Solving SACAMI is to check bound consistency. To enforce bound consistency, one can trivially apply SACAMI sequentially $O(mn)$ times when enumerating on the values of the bounds of the variables:

Corollary 2 *The bound consistency of a conjunction of* 1*-dimensional* `among` *constraints with interval value domains can be achieved in polynomial time.*

5 Conclusion

There is two contributions in this paper. First, we have proposed an approach based on the `among` global constraint for dealing with the inter-location of several autonomous robots equipped with the electric sense. Second, we have proven new theoretical results on this constraint: providing that domains are intervals, we have shown that the bound consistency for a conjunction of `among` constraints is NP-complete in the vector case and in $\mathcal{P}$ in the one-dimensional case. Note that it is easy to derive a filtering algorithm in several dimensions by decomposing the constraint into its projections (as done, e.g., with `atmost-nvector` [4]).

There is a real and upgrowing interest nowadays for bio-inspired architectures and a similar approach could be carried out for sensors that mimic, e.g., the fish lateral line [8] or the eye of a fly [11].

Of course, these new bio-inspired architectures (including ours) are not fully operationnal yet, even if great progresses are made everyday on many aspects. Fortunately, by the time this paper is submitted, a multi-agent simulator for the electric sense is about to be complete by our partners. Hence, it shall be soon possible to test the efficiency of our model based on the `among` constraint.

As we have justified, we are missing computation models for electro-location and the desired scenario seemed out of reach not that long ago. Often, the experimental setup is conditioned by our ability to handle computations whereas it is usually in the other way around. *If computer scientists can only localize robots in a rectangular environment, experiments will be made in a swimming pool.* Our `among` constraint, by definition, allows to infer considerable information from the signal as contrary to other methods. So, it may offer the possibility to consider more interesting scenarii in experiments than before, should we impose many beacons, small areas or perfectly insulating walls, etc.

The technical part of the paper also serves its own purpose. The algorithm we gave for enforcing the bound consistency in the case of interval domains can be used as a propagator for arbitrary value domains. One extension of this work is therefore to test and compare this new algorithm on standard benchmarks of the constraint programming community.

References

[1] M. Alamir, O. Omar, N. Servagent, and A. Girin. On Solving Inverse Problems for Electric Fish Like Robots. In *IEEE Conference on Robotics and Biomimetics*, 2010.

[2] G. Baffet, P.B. Gossiaux, M. Porez, and F. Boyer. Underwater Robotic: Localization with Electrolocation for Collision Avoidance. In *CIFA*, 2008.

[3] N. Beldiceanu and E. Contejean. Introducing Global Constraints in CHIP. *Jour. of Mathematical and Computer Moddeling*, 20(12):97–123, 1994.

[4] G. Chabert, L. Jaulin, and X. Lorca. A Constraint on the Number of Distinct Vectors with Application to Localization. In *CP'09*, volume 5732 of *LNCS*, pages 196–210, 2009.

[5] S. Chevallier, N. Cuperlier, and P. Gaussier. Efficient Neural Models for Visual Attention. In *Computer Vision and Graphics*, volume 6374 of *LNCS*, pages 257–264, 2010.

[6] E. Davis. Constraint Propagation with Interval Labels. *Artificial Intelligence*, 32(3):281–331, 1987.

[7] R. Dechter. *Constraint Processing*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[8] J. Engelmann, S. Kröther, H. Bleckmann, and J. Mogdans. Effects of Running Water on Lateral Line Responses to Moving Objects. *Brain, Behavior and Evolution*, 61:195–212, 2003.

[9] L. Jaulin, E. Walter, O. Lévêque, and D. Meizel. Set Inversion for Chi-Algorithms, with Application to Guaranteed Robot Localization. *Math. Comput. Simulation*, 52:197–210, 2000.

[10] R.B. Kearfott. *Rigorous Global Search: Continuous Problems*. Springer, 1996.

[11] L Kerhuel, S. Viollet, and N.H. Franceschini. Steering by Gazing: An Efficient Biomimetic Control Strategy for Visually Guided Micro Aerial Vehicles. *IEEE Transactions on Robotics*, 26(2):307–319, 2010.

[12] W. Khalil, G. Gallot, and F. Boyer. Dynamic Modeling and Simulation of a 3-D Serial Eel-Like Robot. *IEEE Transactions on Systems, Man, and Cybernetics*, C-37(6):1259–1268, 2007.

[13] E. Lawler. *"Combinatorial Optimization: Networks and Matroids"*. Saunders College Publishing, 1976.

[14] V. Lebastard, C. Chevallereau, A. Amrouche, and B. Jawad. Underwater Robot Navigation around a Sphere using Electrolocation Sense and Kalman Filter. In *IROS*, pages 4225–4230, 2010.

[15] M.J. Maher, N. Narodytska, C-J. Quimper, and T. Walsh. Flow-Based Propagators for the SEQUENCE and Related Global Constraints. In *CP'08*, volume 5202 of *LNCS*, pages 159–174, 2008.

[16] B. Rasnow. The Effects of Simple Objects on the Electric Field of Apteronotus. *Journal of Comparative Physiology, A*, 178(3):397–411, 1996.

[17] J-C. Régin. A Filtering Algorithm for Constraints of Difference in CSPs. In *AAAI*, pages 362–367, 1994.

[18] J-C. Régin. Combination of Among and Cardinality Constraints. In *CPAIOR*, pages 288–303, 2005.

[19] F. Rossi, P. van Beek, and T. Walsh. *Handbook of Constraint Programming (Foundations of Artificial Intelligence)*. Elsevier Science Inc., 2006.

[20] A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, Chichester, 1986.

[21] J. Solberg, K.M. Lynch, and M.A. MacIver. Active Electrolocation for Underwater Target Localization. *Int. Journal of Robotics Research*, 27(5):529–548, 2008.

[22] W-J. van Hoeve, G. Pesant, L-M. Rousseau, and A. Sabharwal. New filtering algorithms for combinations of among constraints. *Constraints*, 14(2):273–292, 2009.

Centre de recherche INRIA Rennes – Bretagne Atlantique
IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Grenoble – Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier
Centre de recherche INRIA Lille – Nord Europe : Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399