ICCM-2004 WORKSHOPS P. 272–279

Quantified Set Inversion Algorithm

Pau Herrero, Miguel A. Sainz, Josep Vehí, Luc Jaulin

Introduction

Real physical world problems frequently involve non-linear constraints over real numbers, uncertain data and badly defined parameters. This kind of problems can be expressed in terms of numerical Constraints Satisfaction Problems (CSP) [1].

Usually, quantifiers ($\forall$, $\exists$) arise in certain situations of uncertainty in the parameters of CSP. For example, universal quantification, models situations where some parameters are unknown and the mathematical model has to hold for every possible choice of these parameters. On the other hand, existential quantification models situations where some parameters can be chosen by the designer. When this quantification appears on a CSP, it can be expressed in terms of numerical Quantified Constraints Satisfaction Problems (QCSP).

The importance of solving QCSP lies on the fact that many physical problems, for example in control engineering [2–5], electrical engineering [6], mechanical engineering [7, 8], biology [9] and various others [10, 11], can be expressed under this paradigm.

Up to now, Cylindrical Algebraic Decomposition [12–14], for which a practical implementation exists [15], has been the most extended method to solve this type of problems. However, this technique is only well suited for small or middle-size problems because of its computational complexity. Moreover, it often generates huge output consisting on highly complicated algebraic expressions which are not useful for many applications and it does not provide partial information before computing the total result.

Methods that appear lately [16, 17] try to avoid some of these problems restricting oneself to approximate instead of exact solutions, using solvers based on numerical methods. However, these algorithms are also restricted to very special cases (e. g. quantified variables only occur once, only one quantifier, etc.). Recently, some of these deficiencies have been partially removed by Ratschan [18–20] but, a lot of work remains to be done before obtaining an efficient and general method. It is important to remark the important contribution on the mathematical foundations of the problem recently done by Shary in [21].

This paper describes a new reliable an efficient method, based on Modal Interval Analysis [22, 23], Set Inversion techniques [24], for the characterization of solution sets defined by numerical Quantified Constraints Satisfaction Problems (QCSP).

1. Problem Statement

A Quantified Constraint (QC) is an algebraic expression over the reals which contains quantifiers ($\exists$, $\forall$), predicate symbols (e. g., $=$, $<$, $\leq$), function symbols (e. g., $+$, $-$, $\times$, sin, exp), rational constants and variables $\boldsymbol{x} = \{x_1, \dots, x_n\}$ ranging over reals domains $\boldsymbol{D} = \{D_1, \dots, D_n\}$.

An example of a QC is the following one,

$$\forall x \in \mathbb{R} \quad x^4 + px^2 + qx + r \geq 0, \tag{1}$$

where x is a universally ($\forall$) quantified variable and p and r are free variable.

As defined in [21], a numerical constraint satisfaction problem, is a triple CSP $= (\boldsymbol{x}, \boldsymbol{D}, \boldsymbol{\mathcal{C}}(\boldsymbol{x}))$ defined by

(i) a set of numeric variables $\boldsymbol{x} = \{x_1, \ldots, x_n\}$,

(ii) a set of domains $\boldsymbol{D} = \{D_1, \ldots, D_n\}$ where D_i, a set of numeric values, is the domain associated with the variable x_i.

(iii) a set of constraints $\boldsymbol{\mathcal{C}}(\boldsymbol{x}) = \{\mathcal{C}_1(\boldsymbol{x}), \ldots, \mathcal{C}_m(\boldsymbol{x})\}$ where a constraint $\mathcal{C}_i(\boldsymbol{x})$ is determined by any numeric relation (equation, inequality, inclusion, etc.) linking a set of variables under consideration.

A solution to a numeric constraint satisfaction problem CSP $= (\boldsymbol{x}, \boldsymbol{D}, \boldsymbol{\mathcal{C}}(\boldsymbol{x}))$ is an instantiation of the variables of $\boldsymbol{x}$ for which both inclusion in the associated domains and all the constraints of $\boldsymbol{\mathcal{C}}(\boldsymbol{x})$ are satisfied. All the solutions of a constraint satisfaction problem thus constitute the set

$$\Sigma = \{\boldsymbol{x} \in \boldsymbol{D} \mid \boldsymbol{\mathcal{C}}(\boldsymbol{x}) \text{ is satisfied}\}. \tag{2}$$

Now suppose that the constraints $\boldsymbol{\mathcal{C}}(\boldsymbol{x})$ depend on some parameters $p_1, p_2, \ldots, p_l$ about which we only know that they belong to some intervals $P_1, P_2, \ldots, P_l$. Moreover, these parameters have an associated quantifier $Q \in \{\forall, \exists\}$. Taking into account the dual character of interval uncertainty, the most general definition of the set of solutions to such Quantified Constraint Satisfaction problem QCSP should have the form

$$\Sigma = \{\boldsymbol{x} \in \boldsymbol{D} \mid Q_1(p_{\sigma_1}, P_{\sigma_1}) \ldots Q_l(p_{\sigma_l}, P_{\sigma_l}) \boldsymbol{\mathcal{C}}(\boldsymbol{x})\}, \tag{3}$$

where

- Q_i are logical quantifiers $\forall$ or $\exists$ (in this paper, only the case of universal quantifiers preceding the existential ones will be dealt),
- $\{p_1, p_2, \ldots, p_l\}$ is the set of parameters of the constraints system considered,
- $\{P_1, P_2, \ldots, P_l\}$ is a set of intervals containing the possible values of these parameters, and
- $\sigma_i \in \Sigma_l$ is a permutation of the numbers $1, \ldots, l$.

The sets of the form (3) will be referred to as quantified solutions sets to the numerical quantified constraints satisfaction problem QCSP $= (\boldsymbol{x}, \boldsymbol{D}, \boldsymbol{\mathcal{C}}(\boldsymbol{x}))$.

2. Methodology

2.1. Set Inversion. One way of solving a CSP is through the characterization of its solution set by means of the Set Inversion (SI) approach.

Let CSP be a constraint satisfaction problem CSP $= (\boldsymbol{x}, \boldsymbol{D}, \boldsymbol{\mathcal{C}}(\boldsymbol{x}))$. Set inversion aims at characterizing the set Σ of all $\boldsymbol{x}$ such that $\boldsymbol{\mathcal{C}}$ is satisfied.

Remark. All constraints are considered under the form $\boldsymbol{\mathcal{C}}(\boldsymbol{x}) := \boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{y}$, where $\boldsymbol{f}$ a continuous function from $\mathbb{R}^n$ to $\mathbb{R}^m$.

Given a box $\boldsymbol{X}$ (cartesian product of intervals), an algorithm which does set inversion is based on a branch-and-bound technique and the 3 followings set of rules.

Rule 1: $\forall(\boldsymbol{x}, \boldsymbol{X})\boldsymbol{\mathcal{C}}(\boldsymbol{x}) \Leftrightarrow \boldsymbol{X} \subseteq \Sigma$.

This logic formula, used to prove that a box $\boldsymbol{X}$ is contained in the solution set, is equivalent to the following interval computation and interval inclusions

$$\mathrm{Out}\big(f(\boldsymbol{X})\big) \subseteq \boldsymbol{Y},$$

where $\boldsymbol{f}(\boldsymbol{X})$ are the ranges of the function components over the interval vector $\boldsymbol{X}$ and $\mathrm{Out}(\boldsymbol{f}(\boldsymbol{X}))$ are outer approximations of $\boldsymbol{f}(\boldsymbol{X})$

Rule 2: $\forall(\boldsymbol{x}, \boldsymbol{X})\neg\boldsymbol{\mathcal{C}}(\boldsymbol{x}) \Leftrightarrow \boldsymbol{X} \subseteq \overline{\Sigma}$.

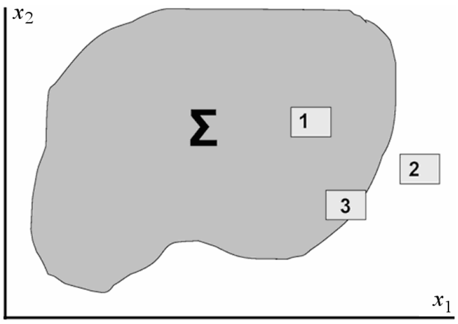

Figure 1. Solution set: 1 – solution, 2 – non-solution, 3 – undefined

This logic formula, used to prove that a box $\boldsymbol{X}$ does not belongs to the solution set, is easily proved by means of the following interval computation and interval inclusions

$$\mathrm{Out}\big(f(\boldsymbol{X})\big) \subseteq \overline{\boldsymbol{Y}}.$$

Finally, if Rule 1 and 2 are not accomplished the position of the box $\boldsymbol{X}$ is undefined

Rule 3: Otherwise, $\boldsymbol{X}$ is undefined.

Figure 1 shows a two dimensional example of the three possible situations corresponding to the 3 rules.

Then the algorithm which does set inversion is as follows

Algorithm SI (In: $\boldsymbol{\mathcal{C}}, \boldsymbol{X}, \epsilon$, Out: $\mathrm{Inn}(\Sigma)$, $\mathrm{Out}(\Sigma)$)

1. **if** With $(\boldsymbol{X}) \leq \epsilon$ **then** $\boldsymbol{X}$ is undefined
2. **else if** Rule 1 is satisfied **then** $\boldsymbol{X}$ is solution
3. **else if** Rule 2 is satisfied **then** $\boldsymbol{X}$ is non solution
4. **else** $\mathrm{Branch}(\boldsymbol{X}, \boldsymbol{X}_1, \boldsymbol{X}_2)$
5. $\mathrm{SI}(\boldsymbol{\mathcal{C}}, \boldsymbol{X}_1, \epsilon)$
6. $\mathrm{SI}(\boldsymbol{\mathcal{C}}, \boldsymbol{X}_2, \epsilon)$

where

- ϵ: SI stops the branching procedure over $\boldsymbol{X}$ when this precision is reached,
- $\mathrm{Inn}(\Sigma)$: Inner approximation of the solution set,
- $\mathrm{Out}(\Sigma)$: Outer approximation of the solution set.

2.2. Quantified Set Inversion via Modal Interval Analysis. Classical Set Inversion is well suited characterizing solution sets of the form (2). The problem arises when the sets are of the form (3). Then, a new algorithm for the characterization of quantified solution sets is needed. This algorithm will be referred to as Quantified Set Inversion (QSI).

Let us consider the case when the constraints are under the form $\boldsymbol{\mathcal{C}}(\boldsymbol{x}) := f(\boldsymbol{x}) \substack{<\\=\\>} 0$, with f a continuous function from $\mathbb{R}^n$ to $\mathbb{R}$.

The main difference between the classical Set Inversion Algorithm and the quantified one lies on the used set of rules. For the proposed algorithm the following rules will be used:

Rule 1: $\forall(\boldsymbol{x}, \boldsymbol{X})\forall(\boldsymbol{p_U}, \boldsymbol{P_U})\exists(\boldsymbol{p_E}, \boldsymbol{P_E})\boldsymbol{\mathcal{C}}(\boldsymbol{x}) \Leftrightarrow \boldsymbol{X} \subseteq \Sigma.$

This logic formula, used to prove that a box $\boldsymbol{X}$ belongs to the solution set, can not be easily proved by means of classical interval computations. For this reason, Modal Interval Analysis is proposed (MIA). MIA is a powerful mathematical tool which allows the evaluation of quantified interval formulas by means of interval computations. Concretely, to evaluate the set of logic formulas, the *-semantic theorem given by MIA is used to reduce equivalently the logical formula to the interval inclusion

$$\mathrm{Out}(f^*(\boldsymbol{X}, \boldsymbol{P_U}, \boldsymbol{P_E})) \subseteq Z,$$

where $\boldsymbol{X}, \boldsymbol{P_U}$ are proper intervals, $\boldsymbol{P_E}$ improper one, $\mathrm{Out}(f^*(\boldsymbol{X}, \boldsymbol{P_U}, \boldsymbol{P_E}))$ is an outer approximation of the the *-semantic extension of the continuous function f and $Z = [0, 0]$, $Z = [-\infty, 0]$ or $Z = [0, \infty]$ depending on if the constraints are under the form $\boldsymbol{\mathcal{C}}(\boldsymbol{x}) := f(\boldsymbol{x}) = 0$, $\boldsymbol{\mathcal{C}}(\boldsymbol{x}) := f(\boldsymbol{x}) < 0$ or $\boldsymbol{\mathcal{C}}(\boldsymbol{x}) := f(\boldsymbol{x}) > 0$, respectively.

In order to obtain the second rule, used to prove that a box $\boldsymbol{X}$ does not belongs to the solution set, the following implication is used:

Rule 2: $\neg(\forall(\boldsymbol{p_U}, \boldsymbol{P_U})\exists(\boldsymbol{p_E}, \boldsymbol{P_E})\exists(\boldsymbol{x}, \boldsymbol{X})\ \boldsymbol{\mathcal{C}}(\boldsymbol{x})) \Rightarrow \boldsymbol{X} \subseteq \overline{\Sigma}.$

This logical formula is, analogously, equivalent to the following interval exclusion:

$$\mathrm{Inn}\big(f^*(\boldsymbol{X}, \boldsymbol{P_U}, \boldsymbol{P_E})\big) \nsubseteq Z,$$

where $\boldsymbol{P_U}$ is a proper interval, $\boldsymbol{X}$, $\boldsymbol{P_E}$ improper ones, $Inn(f^*(\boldsymbol{X}, \boldsymbol{P_U}, \boldsymbol{P_E}))$ is an inner approximation of the the *-semantic extension of the continuous function f. and $Z = [0, 0]$, $Z = [-\infty, 0]$ or $Z = [0, \infty]$ depending on if the constraints are under the form $\boldsymbol{\mathcal{C}}(\boldsymbol{x}) := f(\boldsymbol{x}) = 0$, $\boldsymbol{\mathcal{C}}(\boldsymbol{x}) := f(\boldsymbol{x}) < 0$ or $\boldsymbol{\mathcal{C}}(\boldsymbol{x}) := f(\boldsymbol{x}) > 0$, respectively.

Finally, if none of these rules are accomplished, the box $\boldsymbol{X}$ is undefined.

Rule 3: otherwise, $\boldsymbol{X}$ is undefined.

Computing the semantic extension of a continuous function f by means of any of their interpretable rational extensions provokes an overestimation of the interval evaluation, due to the multi-occurrences of variable, when the rational computations is not optimal. An algorithm, based on results of Modal Interval Analysis and branch-and-bound techniques which allows to efficiently compute an inner and an outer approximation of f^* has been recently built.

When the constraints are under the form $C(\boldsymbol{x}) := \boldsymbol{f}(\boldsymbol{x}) \lesseqqgtr 0$, with $\boldsymbol{f}$ a continuous function from $\mathbb{R}^n$ to $\mathbb{R}^m$ and each variable existentially quantified appears in only a component function, the problem is reduced to m different problems, one for each component function.

3. Example

Given the intervals $T_1, \ldots, T_m$, $Y_1, \ldots, Y_m$, to find the inner estimation of the sets:

$$\begin{aligned}\Sigma_E = \big\{&\boldsymbol{p} \in \mathbb{R}^2 \mid \exists(t_1, T_1')\exists(y_1, Y_1') \cdots \exists(t_n, T_n')\exists(y_n, Y_n')\\ &(p_1 e^{-p_2 t_1} = y_1, \ \ldots, \ p_1 e^{-p_2 t_n} = y_n)\big\},\\ \Sigma_U = \big\{&\boldsymbol{p} \in \mathbb{R}^2 \mid \forall(t_1, T_1')\exists(y_1, Y_1') \cdots \forall(t_n, T_n')\exists(y_n, Y_n')\\ &(p_1 e^{-p_2 t_1} = y_1, \ \ldots, \ p_1 e^{-p_2 t_n} = y_n)\big\}.\end{aligned}$$

We have

$$\Sigma_E = \Sigma_{E_1} \cap \ldots \cap \Sigma_{E_n}, \qquad \Sigma_U = \Sigma_{U_1} \cap \ldots \cap \Sigma_{U_n},$$

where

$$\begin{aligned}\Sigma_{E_i} &:= \big\{\boldsymbol{p} \in \mathbb{R}^2 \mid \exists(t_i, T_i')\exists(y_i, Y_i')p_1 e^{-p_2 t_i} = y_i\big\},\\ \Sigma_{U_i} &:= \big\{\boldsymbol{p} \in \mathbb{R}^2 \mid \forall(t_i, T_i')\exists(y_i, Y_i')p_1 e^{-p_2 t_i} = y_i\big\}\end{aligned}$$

with $i = 1, \ldots, n$.

3.1. Characterizing $\boldsymbol{\Sigma_E}$. The logic formula which fulfils the points belonging to the solution set Σ_{E_i} is

$$\forall(p_1, P_1')\forall(p_2, P_2')\exists(t_i, T_i')\exists(y_i, Y_i') \quad p_1 e^{-p_2 t_i} - y_i = 0$$

which is equivalent to the following inclusion test

$$\mathrm{Out}(f_i^*(P_1, P_2, T_i, Y_i)) \subseteq [0, 0],$$

with P_1 and P_2 proper intervals and T_i and Y_i improper ones.

The logic formula which fulfils the points not belonging to the solution set Σ_{E_i} is

$$\neg(\exists(p_1, P_1')\exists(p_2, P_2')\exists(t_i, T_i')\exists(y_i, Y_i') \ p_1 e^{-p_2 t_i} - y_i = 0)$$

which is implied by the following exclusion test

$$\mathrm{Inn}(f_i^*(P_1, P_2, T_i, Y_i)) \nsubseteq [0, 0],$$

with P_1, P_2, T_i and Y_i improper intervals.

Then, $\Sigma_E = \Sigma_{E_1} \cap \cdots \cap \Sigma_{E_n}$.

In less than 4 seconds on a Pentium III 1 GHz, for $n = 2$, an $\epsilon = 0.05$ and the following interval domains: $\boldsymbol{X} = (P_1, P_2) = ([-1, 4], [-1, 1])$, $Y_1' = [1.3, 3.3]$, $Y_2' = [0.3, 2.3]$, $T_1' = [2, 3]$ and $T_2' = [3.5, 4]$, QSI generates the paving represented in figure 2, where the darker region corresponds to the solution set Σ_E, the grey region corresponds to the non solution set $\overline{\Sigma}_E$ and the white region is undefined.

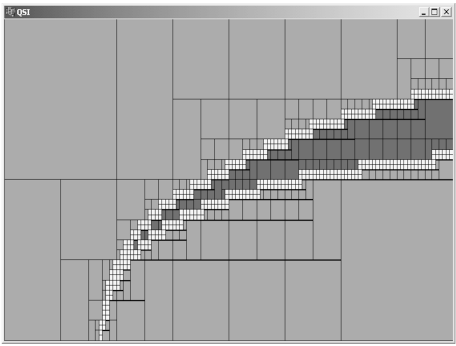

Figure 2. Σ_E paving

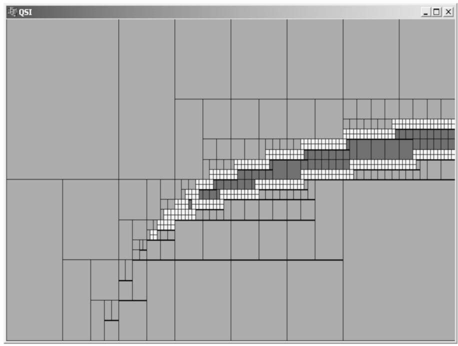

Figure 3. Σ_U paving

3.2. Characterizing Σ_U. The logic formula which fulfil the points belonging to the solution set Σ_{U_i} is

$$\forall(p_1, P_1')\forall(p_2, P_2')\forall(t_i, T_i')\exists(y_i, Y_i) \quad p_1 e^{-p_2 t_i} - y_i = 0$$

which is equivalent to the following inclusion test

$$\mathrm{Out}(f_i^*(P_1, P_2, T_i, Y_i)) \subset [0, 0], \tag{4}$$

where P_1, P_2 and T_i are proper intervals and Y_i is improper.

The logic formula which fulfil the points not belonging to the solution set Σ_{U_i} is

$$\neg(\forall(t_i, T_i')\exists(y_i, Y_i')\exists(p_1, P_1')\exists(p_2, P_2') \;\; p_1 e^{-p_2 t_i} - y_i = 0)$$

which is implied by the following exclusion test

$$\mathrm{Inn}(f_i^*(P_1, P_2, T_i, Y_i)) \nsubseteq [0, 0]$$

with P_1, P_2 and Y_i improper intervals and T_i proper.

Then, $\Sigma_U = \Sigma_{U_1} \cap \cdots \cap \Sigma_{U_m}$.

In less than 3 seconds on a Pentium III 1 GHz, for $n = 2$, an $\epsilon = 0.05$ and the same interval domains used for the previous example, QSI generates the paving represented in figure 3, where the darker region corresponds to the solution set Σ_U, the grey region corresponds to the non solution set $\overline{\Sigma}_U$ and the white region is undefined.

4. Future Work

4.1. Reducing the complexity via Constraint Propagation. In order to reduce the complexity of the set inversion algorithm due to the branching, a narrowing operator (a contractor) for quantified constraints will be provided. This contractor, based on *constraint propagation* techniques and Modal Interval Analysis, allows the contraction of an initial box $\boldsymbol{X}$ to another one $\boldsymbol{X}'$ such that $\boldsymbol{X}'$ still contains the solution set Σ.

The basic idea consists on decomposing the set of constraints into their primitive constraints and by means of Modal Interval Arithmetic to compute local approximations of the solution space for a given primitive constraint. These evaluation provokes domain reduction over $\boldsymbol{X}$ which are propagated through the whole set of constraints by a propagation engine.

4.2. Application on parameter identification. An application on parameter identification and its comparison with the classical interval approach used in [25] is under study.

References

[1] Lhomme O. Consistency techniques for numerical CSPs // Intern. Joint Conf. on Artificial Intelligence (IJCAI). – 1993. – Vol. 1. – P. 232–238.

[2] Abdallah C., Dorato P., Yang W., Liska R., Steinberg S. Applications of quantifier elimination theory to control system design // 4th IEEE Mediterranean Symp. on Control and Automation, Crete, Greece, 1996.

[3] Jaulin L., Walter É. Guaranteed tuning, with application to robust control and motion planning // Automatica. – 1996. – Vol. 32, No. 8. – P. 1217–1221.

[4] Jirstrand M. Nonlinear control system design by quantifier elimination // J. Symbolic Computation. – 1997. – Vol. 24, No. 2. – P. 137–152.

[5] Dorato P. Quantified multivariate polynomial inequalities // IEEE Control Systems Magazine. – 2000. – Vol. 20, No. 5. – P. 48–58.

[6] Sturm T. Reasoning over networks by symbolic methods // Applicable Algebra in Engineering Communication and Computing. – 2000. – Vol. 10, No. 1. – P. 79–96.

[7] Ioakimidis N.I. REDLOG-aided derivation of feasibility conditions in applied mechanics and engineering problems under simple inequality constraints // J. Mechanical Engineering (Strojnícky Casopis). – 1999. – Vol. 50, No. 1, – P. 58–69.

[8] Ioakimidis N.I. Quantifier elimination in applied mechanics problems with cylindrical algebraic decomposition // Intern. J. Solids and Structures. – 1997. – Vol. 34, No. 30. – P. 4037–4070.

[9] Chauvin C., Müller M., Weber A. An application of quantifier elimination to mathematical biology // Computer Algebra in Science and Engineering. – World Scientific, 1994. – P. 287–296.

[10] Benhamou F., Goualard F. Universally quantified interval constraints // Proc. of the Sixth Int. Conf. on Principles and Practice of Constraint Programming (CP'2000). – Singapore, 2000. – No. 1894.

[11] Gerber R., Pugh W., Saksena M. Parametric dispatching of hard real-time tasks // IEEE Transactions on Computers. – 1995. – Vol. 44, No. 3. – P. 471–479.

[12] Tarski A. A Decision Method for Elementary Algebra and Geometry. – Berkeley: Univ. of California Press, 1951.

[13] Collins G.E. Quantifier elimination for real closed fields by cylindrical algebraic decomposition // 2nd GI Conf. Automata Theory and Formal Languages. – 1975. – Vol. 33. – P. 134–189.

[14] Hong H. Simple solution formula construction in cylindrical algebraic decomposition based quantifier elimination // Proc. ISSAC'92, International Symposium on Symbolic and Algebraic Computation. – 1992. – P. 177–188.

[15] Brown Chr. W. Quantifier Elimination by Partial Cylindrical Algebraic Decomposition. – http://www.cs.usna.edu/~qepcad/B/QEPCAD.html.

[16] Garloff Ü., Graf B. Solving strict polynomial inequalities by Bernstein expansion // The Use of Symbolic Methods in Control System Analysis and Design. – London: The Institution of Electrical Engineers, 1999. – P. 339–352.

[17] Benhamou F., Goulard F. Universally quantified interval constraints // Sixth Intern. Conf. on Principles of Constraint Programming, 2000.

[18] Ratschan S. Quantified Constraints Under Perturbation // J. Symbolic Computation. – 2001. – Vol. 33, No. 4. – P. 493–505.

[19] Ratschan S. Efficient Solving of Quantified Inequality Constraints over the Real Numbers // Submitted for publication, 2003.

[20] Ratschan S. Approximate Quantified Constraint Solving. – http://www.mpi-sb.mpg.de/~ratschan/AQCS/AQCS.html.

[21] Shary S.P. A new technique in systems analysis under interval uncertainty and ambiguity // Reliable Computing. – 2002. – Vol. 8. – P. 321–418.

[22] Gardenyes E., Sainz M.Á., Jorba L., Calm R., Estela R., Mielgo H., Trepat A. Modal Intervals // Reliable Computing. – 2001. – Vol. 7, No. 2. – P. 77–111.

[23] Sainz M.Á., Gardenyes E., Jorba L. Interval Estimations of Solution Sets to Real-Valued Systems of Linear or Non-Linear Equations // Reliable Computing. – 2002. – Vol. 8, No. 4. – P. 283–305.

[24] Jaulin L., Walter E. Set inversion via interval analysis for nonlinear bounded-error estimation year // Automatica. – 1993. – Vol. 32, No. 8. – P. 1053–1064.

[25] Jaulin L., Walter E. Guaranteed Bounded-Error Parameter Estimation for Nonlinear Models with Uncertain Experimental Factors // Automatica. – 1999. – Vol. 35, No. 5. – P. 849–856.