Including Ordinary Differential Equations Based Constraints in the Standard CP Framework

Alexandre Goldsztejn[2], Olivier Mullier[1,3],
Damien Eveillard[1], and Hiroshi Hosobe[3]

[1] University of Nantes, CNRS, LINA (UMR 6241), Nantes, France
[2] CNRS, LINA (UMR 6241), Nantes, France
[3] National Institute of Informatics, Tokyo, Japan
Alexandre.Goldsztejn@univ-nantes.fr,
Olivier.Mullier@etu.univ-nantes.fr,
Damien.Eveillard@univ-nantes.fr,
Hosobe@nii.ac.jp

Abstract. Coupling constraints and ordinary differential equations has numerous applications. This paper shows how to introduce constraints involving ordinary differential equations into the numerical constraint satisfaction problem framework in a natural and efficient way. Slightly adapted standard filtering algorithms proposed in the numerical constraint satisfaction problem framework are applied to these constraints leading to a branch and prune algorithm that handles ordinary differential equations based constraints. Preliminary experiments are presented.

1 Introduction

Solving problems involving constraints and ordinary differential equations (ODE) allows numerous applications (e.g. in parameter estimation, control, design). A lot of work has been dedicated to solving such problems ([1,2,3,4,5,6] and references therein). Each of these works proposed a method dedicated to a specific problem, which somehow goes against the paradigm of CSPs that intends separating the problem declaration and the resolution process. However these problems all involve variables on continuous domains and some relations that constrain the variable values thus they should match naturally the numerical CSP framework (NCSP). Such a matching should allow cross-fertilization between the NCSP framework and these specific resolution methods, and expressing and solving more problems involving ODE based constraints. Furthermore, separating the problem declaration and the resolution process has many advantages which have been the basis of the success of the CP framework.

On the other hand, a specialized class of CSPs which includes ODE based constraints was proposed in [7,8], namely Constraint Satisfaction Differential Problems (CSDP). CSDPs include two different kinds of variables: Some functional variables whose values are functions $\mathbf{x}(t)$ from $\mathbb{R}$ to $\mathbb{R}^n$ (corresponding to the trajectories of the ODE) and some real variables, called restriction variables, that correspond to characteristic values of functions $\mathbf{x}(t)$ (for example $\mathbf{x}(0)$

D. Cohen (Ed.): CP 2010, LNCS 6308, pp. 221–235, 2010.

or $\max_{0\leq t\leq 1}\mathbf{x}(t)$). The constraints of the CSDP include constraints acting on the functional variables (corresponding to the ODE definition) and constraints acting on the restriction variables (like the Value Restriction constraints, e.g. $||\mathbf{x}(0)||\leq 1$, and the Maximum Restriction constraints, e.g. $\max_{0\leq t\leq 1}\mathbf{x}(t)\leq 1$). Thus, although CSDPs are a specific class of CSPs, they do not match the NCSP framework. In particular, they are more complex and less flexible than one could expect (for example restriction variables of CSDPs can only be directly related to the ODE solution through some Restriction constraints, see Subsection 3.2 for an example of problem that cannot be expressed in the CSDP framework).

In the present paper, we show that ODE based constraints can be naturally used in the NCSP framework. This is done by abstracting an ODE by its *solution operator* which is a function that maps an initial condition and a time to the corresponding final state (cf. Section 3). We show that in particular parameter estimation problems and two-point boundary value problems can be expressed homogeneously in this framework (cf. subsections 3.3 and 3.4). The only adaptation of the NCSP framework to solve such constraints involving ODEs is to tune the standard filtering algorithms in order to solve efficiently these constraints. In particular, solving these constraints involve evaluating the solution operator and its derivatives which is very expensive since each evaluation requires simulating the ODE. These solution operator evaluations are performed rigorously for interval inputs using standard methods from interval analysis for integrating ODEs. On the other hand, these methods usually evaluate both the solution operator and its derivatives for the same cost. Therefore, standard NCSPs filtering algorithms have to be tuned in order to evaluate the constraints as rarely as possible, while exploiting all information provided by these evaluations.

Notations. Intervals are denoted using brackets, e.g. $[x]$ is an interval, and the set of closed intervals by $\mathbb{IR}$. Vectors are denoted by bold symbols, so $[\mathbf{x}]$ is an interval vector (also called a box). Also, $\mathbf{f}:\mathbb{R}^n\to\mathbb{R}^m$ is a vector valued function, which can be viewed as a vector of real valued function, i.e. $\mathbf{f}(\mathbf{x})=(f_1(\mathbf{x}),\ldots,f_m(\mathbf{x}))$. Its Jacobian is denoted by $D\mathbf{f}(\mathbf{x})=(\frac{\partial f_i}{\partial x_j}(\mathbf{x}))_{ij}\in\mathbb{R}^{m\times n}$. The lower and upper bounds of an interval $[x]$ are denoted by $\inf[x]\in\mathbb{R}$ and $\sup[x]\in\mathbb{R}$ respectively. When not confusing, the lower and upper bounds of an interval $[x]$ will be denoted respectively by $\underline{x}\in\mathbb{R}$ and $\overline{x}\in\mathbb{R}$. Inequality for vectors being understood componentwise, $\sup[\mathbf{x}]\leq 0$ means that $[\mathbf{x}]$ contains vectors whose components are all nonpositive. In the case of interval vectors, $\inf[\mathbf{x}]\in\mathbb{R}^n$ and $\sup[\mathbf{x}]\in\mathbb{R}^n$. The width $\mathrm{wid}[x]$ of an interval is $\overline{x}-\underline{x}$, while the width of an interval vector is the maximum of the widths of it components.

2 Interval Analysis for Numerical CSPs

A constraint satisfaction problem (CSP) is a triplet made of a list of variables, a list of domains corresponding to these variables, and a set of constraints. Numerical CSPs are simply characterized by continuous domains, although issues and technics involved in the resolution of NCSPs and discrete domain CSP are

different. This section presents the basics of NCSPs resolution for constraints that do not involve any ODE.

A solution of a NCSP is often defined as a real vector $\mathbf{x}$ that belongs to the Cartesian product $[\mathbf{x}]$ of the variable domains (i.e. $[\mathbf{x}] = ([x_1], \ldots, [x_n])$), whose components correspond to the values of the variables (i.e. $\mathbf{x} = (x_1, \ldots, x_n)$) and that satisfies the constraints. The set of its solutions is denoted by $\mathrm{Sol}(\mathcal{P}) \subseteq [\mathbf{x}]$.

2.1 Interval Analysis

The modern interval analysis was born in the 60's with [9] (see [10,11] and extensive references). It allows dealing rigorously with real numbers and functions using machine representable numbers. It therefore plays a key role in the resolution of NCSPs.

The most basic operation of interval analysis is to enclose the image of intervals by a real function. First, arithmetic operations ($+$, $-$, $\times$ and $\div$) and elementary functions (exp, ln, sin, cos, etc.) are extended to intervals by $[x] \circ [y] = \{x \circ y : x \in [x], y \in [y]\}$ and $f([x]) = \{f(x) : x \in [x]\}$ respectively. For example, $[\underline{x}, \overline{x}] + [\underline{y}, \overline{y}] = [\underline{x} + \underline{y}, \overline{x} + \overline{y}]$ and $\exp([\underline{x}, \overline{x}]) = [\exp(\underline{x}), \exp(\overline{x})]$ (non monotonous functions require more complicated but still simple algorithms). Then, expression compound of these elementary operations can be evaluated to intervals replacing real operations by their interval counterparts. The fundamental theorem of interval analysis states that the interval evaluation of an expression gives rise to an interval enclosure of this function (see e.g. [10]). For example, the interval evaluation of $f(x, y) = xy + \exp(x + y)$ for interval arguments $[x] = [-1, 1]$ and $[y] = [3, 4]$ gives rise to $[3.38, 152.42]$ (rounded to two decimals) which is a superset of $\{f(x, y) : x \in [x], y \in [y]\}$.

The interval evaluation of an expression is the basis for dealing with constraints. For example consider the constraint $f(x, y) = 0$ where f and the domains $[x]$ and $[y]$ are defined as above. The interval evaluation of the expression of f gave rise to an interval that does not contain 0, and therefore proves that the constraint has no solution in the domain hence providing a rigorous filtering process. More elaborated filtering algorithms are presented in Subsection 2.3.

2.2 The Branch and Prune Algorithm

The branch and prune algorithm [12,13] alternates branching and pruning in order to produce two pavings[1] $\mathcal{B}$ and $\mathcal{S}$, called respectively *boundary boxes* and *solution boxes*. It is described in Algorithm 1 where the pruning is performed by the function $\mathrm{Contract}_{\mathcal{C}}$ whose semantic is

$$\mathbf{x} \in [\mathbf{x}] \wedge \big(\forall c \in \mathcal{C}, c(\mathbf{x})\big) \Longrightarrow \mathbf{x} \in \mathrm{Contract}_{\mathcal{C}}([\mathbf{x}]). \tag{1}$$

So $\mathrm{Contract}_{\mathcal{C}}$ contracts a box in such a way that no NCSP solution is lost, and therefore all the solutions are eventually contained in one box from $\mathcal{B}$ or $\mathcal{S}$:

$$\mathrm{Sol}(\mathcal{P}) \ \subseteq \ (\cup\mathcal{S}) \cup (\cup\mathcal{B}). \tag{2}$$

[1] I.e. sets of boxes which overlap only on their boundaries.

Algorithm 1. Branch and Prune Algorithm.

Input: $\mathcal{C} = \{c_1, \ldots, c_m\}$, $[\mathbf{x}] \in \mathbb{IR}^n$, $\epsilon > 0$
Output: $\mathcal{S} \subseteq \mathbb{IR}^n$, $\mathcal{B} \subseteq \mathbb{IR}^n$
$\mathcal{L} \leftarrow \{[\mathbf{x}]\}$; $\mathcal{B} \leftarrow \emptyset$; $\mathcal{S} \leftarrow \emptyset$;
while $\mathcal{L} \neq \emptyset$ **do**
  $([\mathbf{x}], \mathcal{L}) \leftarrow \text{Extract}(\mathcal{L})$;
  if $\text{IsSolution}_{\mathcal{C}}([\mathbf{x}])$ **then** $\mathcal{S} \leftarrow \mathcal{S} \cup \{[\mathbf{x}]\}$;
  else if $\text{wid}([\mathbf{x}]) < \epsilon$ **then** $\mathcal{B} \leftarrow \mathcal{B} \cup \{[\mathbf{x}]\}$;
  else
    $[\mathbf{x}] \leftarrow \text{Contract}_{\mathcal{C}}([\mathbf{x}])$;
    if $[\mathbf{x}] \neq \emptyset$ **then**
      $\{[\mathbf{x}'], [\mathbf{x}'']\} \leftarrow \text{Split}([\mathbf{x}])$;
      $\mathcal{L} \leftarrow \mathcal{L} \cup \{[\mathbf{x}'], [\mathbf{x}'']\}$;
    end
  end
end
return $(\mathcal{S}, \mathcal{B})$;

The solution boxes carry the additional information that the function $\text{IsSolution}_{\mathcal{C}}$, which returns a boolean, evaluates to True on them. The semantic of this function depends on the type of constraints involved in the NCSP. In this paper, we are interested in two kinds of NCSPs that are detailed below.

The first involves only inequality constraints. It will be convenient to group the conjunction of several inequality constraints to one inequality constraint involving a vector valued function: $\mathbf{f}(\mathbf{x}) \leq \mathbf{0}$ with $\mathbf{f} : \mathbb{R}^n \longrightarrow \mathbb{R}^m$. In this case, the function $\text{IsSolution}_{\{\mathbf{f}(\mathbf{x}) \leq \mathbf{0}\}}([\mathbf{x}])$ returns True if $[\mathbf{f}]([\mathbf{x}]) \leq 0$ and False otherwise. Thus, $\text{IsSolution}_{\{\mathbf{f}(\mathbf{x}) \leq 0\}}([\mathbf{x}])$ is true only if the box $[\mathbf{x}]$ contains only solutions of the constraint. If several such constraints are involved then the conjunction of each constraint solution test can obviously be used. Finally in this case, the branch and prune algorithms outputs pavings that satisfies $\cup\mathcal{S} \subseteq \text{Sol}(\mathcal{P})$ in addition to (2). Such pavings are shown in Section 4 (see e.g. Figure 2 page 230).

The second kind of NCSPs involves n equations and n variables (which corresponds to a well constrained system of equations) and possibly additional inequalities. In this case, the NCSP typically has a finite number of solutions and the function $\text{IsSolution}_{\mathcal{C}}([\mathbf{x}])$ returns true only if the existence of one solution in $[\mathbf{x}]$ is proved. This existence proof is carried out using the multidimensional interval Newton operator (4) which is introduced in Subsection 2.3. Therefore uniqueness of this solution in $[\mathbf{x}]$ is also proved.

2.3 Filtering Algorithms for NCSP

Standard filtering algorithms used in the NCSP framework must be slightly adapted in order to be efficiently used for constraints involving ODEs. These constraints are characterized by the fact that evaluating a function involving an ODE is very expensive, but only a small extra work is necessary to evaluate the

Jacobian of this function (see Section 3 for details). Thus the filtering algorithms presented in this section also classically used in the context of NCSP may turn out not to be the most efficient ones when constraints involve no ODEs.

Conjunction of inequality constraints. Two filtering algorithms will be used for such constraints $\mathbf{f}(\mathbf{x}) \leq \mathbf{0}$: The simple interval enclosure test and the unidimensional interval Newton[2]. The interval enclosure test simply consists of computing an interval enclosure of $\mathbf{f}$ and checking whether or not it allows rejecting the whole box: If $\inf[\mathbf{f}]([\mathbf{x}]) \leq \mathbf{0}$ is not satisfied then the whole box can be rejected. The unidimensional interval Newton allows removing a slice of $[\mathbf{x}]$ as follows: The domain $[x_j]$ of the variable x_j is contracted to the new domain $[x'_j]$ applying the unidimensional interval Newton for each component of $\mathbf{f}(\mathbf{x}) \leq \mathbf{0}$:

$$[x'_j] = \bigcap_{i \in \{1,\ldots,m\}} \left(\tilde{x}_j - \frac{1}{[a_{ij}]}\left([b_i] + [0,\infty] + \sum_{k \neq j} [a_{ik}]([x_k] - \tilde{x}_k)\right)\right) \cap [x_j] \qquad (3)$$

where $[A] = [D\mathbf{f}]([\mathbf{x}])$, $[\mathbf{b}] = [\mathbf{f}](\tilde{\mathbf{x}})$ for some $\tilde{\mathbf{x}} \in [\mathbf{x}]$, usually the midpoint of $[\mathbf{x}]$. The addition of the interval $[0,\infty]$ allows applying the interval Newton, which is originally defined for equality constraints, to inequality constraints: Indeed, $f(x) \leq 0$ is equivalent to $0 \in f(x) + [0,\infty]$ and the interval Newton for equality constraint is actually applied to the latter.

It is worth noting that in order to apply these two filtering algorithms, only interval enclosures of $\mathbf{f}$ over $[\mathbf{x}]$ and over $\tilde{\mathbf{x}}$, and of $D\mathbf{f}$ over the whole domain are required.

Well constrained systems of equations. The simple interval enclosure test is also applied to constraints $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ by rejecting the whole box if $0 \in [\mathbf{f}]([\mathbf{x}])$ is not satisfied. We will also use the Krawczyk version of the multidimensional interval Newton (see e.g. [10]) that allows contracting the whole domain $[\mathbf{x}]$ to the new domain $[\mathbf{x}']$ as follows:

$$[\mathbf{x}'] = \big((I - C[A])[\mathbf{x}] + C[\mathbf{b}]\big) \cap [\mathbf{x}] \qquad (4)$$

where $[A]$ and $[\mathbf{b}]$ are defined like in (3), and $C = (\mathrm{mid}[A])^{-1}$.

Remark 1. The multidimensional interval Newton operator is used for both filtering non consistent vectors and proving the existence of solutions (see [10] for details). Thus only one application of the multidimensional interval Newton is performed for both purposes, while this optimization is not explicitly shown in Algorithm 1 for clarity.

It is worth noting that again in order to apply the filtering algorithms dedicated to well constrained systems of equations, only interval enclosures of $\mathbf{f}$ over

[2] Usually, the unidimensional interval Newton is improved encapsulating it into the box-consistency filtering (cf. [14] and references therein). However, the box-consistency filtering requires many function evaluations, which makes it too expensive when ODEs are involved.

$[\mathbf{x}]$ and over $\tilde{\mathbf{x}}$, and of $D\mathbf{f}$ over the whole domain are required. Therefore, all we will need to be able to apply these filtering algorithms to ODE based constraints is to be able to compute these interval enclosures in the case of constraints involving ODEs. This is done by abstracting the ODEs by their solution operator, as shown in the next section.

3 Including ODE Based Constraints in CSPs

We consider herein an ODE $\mathbf{x}'(t) = \mathbf{h}(\mathbf{x}(t))$ where $\mathbf{h} : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ is supposed to be enough differentiable so that interval integrator can solve it (see e.g. [15] for an introduction to ODEs). The *solution operator* of this ODE is introduced in Subsection 3.1, while the remaining subsections show how to use this solution operator to model different problems.

3.1 The ODE Solution Operator and Its Derivatives

The solution operator is introduced noting that the ODE maps an initial condition $\mathbf{x}(t_0) \in \mathbb{R}^n$ and a duration $t \in \mathbb{R}$ to an unique vector[3] $\mathbf{x}(t)$. Therefore, the ODE defines an operator $\Phi : \mathbb{R}^n \times \mathbb{R} \longrightarrow \mathbb{R}^n$ characterized by

$$\Phi\big(\mathbf{x}(t_0), t\big) = \mathbf{x}(t_0 + t) \tag{5}$$

called the ODE solution operator. It allows to abstract the simulation of the ODE into a simple function evaluation. On the other hand, the evaluation of the solution operator of course requires to integrate the ODE, thus each evaluation of Φ is computationally very expensive.

The Jacobian $D\Phi : \mathbb{R}^n \times \mathbb{R} \longrightarrow \mathbb{R}^{n\times(n+1)}$ of the solution operator is also very useful as it allows to quantify the sensitivity of Φ. For convenience, it is split into two submatrices $D_{\mathbf{x}}\Phi : \mathbb{R}^n \times \mathbb{R} \longrightarrow \mathbb{R}^{n\times n}$ and $D_t\Phi : \mathbb{R}^n \times \mathbb{R} \longrightarrow \mathbb{R}^{1\times n}$, the latter being actually equal to $\mathbf{h}(\Phi(\mathbf{x}, t))$. The former is usually computed solving the ODE first variational equation, which is a linear non-autonomous ODE of dimension n^2. Therefore, evaluating Φ and $D\Phi$ turns out to solving an ODE of dimension $n^2 + n$.

Standard integrators coming from numerical analysis (e.g. Runge-Kuntta, Adams methods, etc., see e.g. [15] for details) compute approximations of the ODE solution. Therefore, they can be used to evaluate approximately Φ and $D\Phi$. On the other hand, interval integrators (see [16] for review and references therein for the theory of interval integrators) allow enclosing rigorously the solution of the ODE for interval initial conditions. Therefore interval integrators give rise to interval enclosures of Φ and $D\Phi$. This is the key point of our approach: In order to apply the NCSP framework to ODE based constraints, we need interval enclosures of the solution operator and its Jacobian, these interval enclosures being computed by interval ODE integrators.

[3] Existence and uniqueness are assumed here, and follow from some hypothesis on the function ϕ that are usually verified. While the concept of solution operator can be generalized considering a domain different than $\mathbb{R}^n$, the interval integrators used in the sequel rigorously prove existence and uniqueness.

Remark 2. The work [17] proposes to use constraints techniques to help integrating ODE. However, this kind of techniques cannot be used directly in our framework since it does not provide any enclosure of the solution operator derivatives.

In theory an interval ODE integrator can compute $[\Phi]([\mathbf{x}],[t])$, $[\Phi](\text{mid}[\mathbf{x}],[t])$ and $[D\Phi]([\mathbf{x}],[t])$ in simulation for approximately the same cost than computing only $[\Phi]([\mathbf{x}],[t])$ (see [18,19]). This is perfectly fitted to the application of the different contractors used in NCSP resolution presented in Section 2.3.

In the next subsections, we show how the solution operator allows expressing ODE based constraints modeling different problems. It can be note already that any problem that can be expressed in the CSDP framework can also be expressed using the solution operator (this is trivial since the solution operator is actually equivalent to the ODE itself so any restriction acting on the ODE can be expressed using the solution operator).

3.2 An Academic Design Problem

A simple academic design problem first shows the expressiveness of the NCSP framework with ODE based constraints. We consider for that a 2D ODE $\mathbf{x}'(t) = \mathbf{h}(\mathbf{x}(t))$. We are looking for the points $\mathbf{a}$ and $\mathbf{b}$ belonging to the radius 1 circle centered on $(1,0)$ such that the state $\mathbf{x}(t_f)$ at a fixed time t_f of the ODE starting at $\mathbf{a}$ (i.e. $\mathbf{x}(0) = \mathbf{a}$), and the points $\mathbf{a}$ and $\mathbf{b}$ form an equilateral triangle. In the case where the ODE is $\mathbf{x}'(t) = A\mathbf{x}(t)$ with

$$A = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \tag{6}$$

and thus the solution operator $\Phi(\mathbf{x},t) = \exp(tA)$ is a rotation of angle t, and $t_f = \frac{\pi}{2}$, the solution can be found graphically as depicted in Figure 1.

This problem is expressed by the following simple NCSP: Variables are two angles θ and τ that define the position of $\mathbf{a}$ and $\mathbf{b}$ on the circle (i.e. $\mathbf{a}_\theta = (\cos(\theta),\sin(\theta))$ and $\mathbf{b}_\tau = (\cos(\tau),\sin(\tau))$) and the domains of these variables are $[-\pi,\pi]$. There are two constraints relating the distances between the three points :

$$||\mathbf{a}_\theta - \Phi(\mathbf{a}_\theta, t_f)||^2 - ||\mathbf{a}_\theta - \mathbf{b}_\tau||^2 = 0 \tag{7}$$

$$||\mathbf{b}_\tau - \Phi(\mathbf{a}_\theta, t_f)||^2 - ||\mathbf{a}_\theta - \mathbf{b}_\tau||^2 = 0 \tag{8}$$

where $\Phi : \mathbb{R}^2 \times \mathbb{R} \longrightarrow \mathbb{R}^2$ is the solution operator of the ODE. The expression of this geometric problem as a NCSP is really clear (even clearer than its original English description). Note that this problem cannot be expressed in the CSDP framework as variables are neither CSDP solution variables nor restriction variables[4]. Using the chain rule the constraints can be automatically differentiated,

[4] It could be possible to add a third kind of variables in the CSDP framework to handle this NCSP. However, the CSDP framework is already made quite complex because of these two kinds of variables and introducing a third kind of variables would make it even more complex.

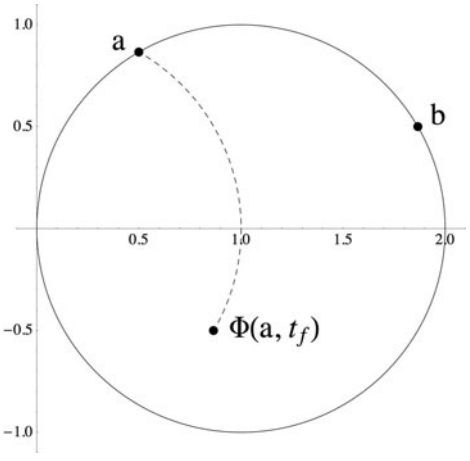

Fig. 1. Graphical solution of the NCSP of Subsection 3.2. The dashed line shows the ODE flow starting at **a** which is abstracted by the ODE solution operator in the NCSP definition.

hence allowing applying the contractors based on the interval Newton operator. Thus this problem can now be solved using the standard methods dedicated to NCSPs.

3.3 Parameter Estimation Problems

The parameter estimation problem represents a special case of finding the initial conditions that satisfy some constraints (usually coming from measurements or observations). Formally, given an ODE $\mathbf{x}'(t) = \mathbf{h}(\mathbf{x}(t))$ and some measurements $(t_i, [\mathbf{m}_i])$ for $i \in \{1, \ldots, p\}$, the problem consists of finding the initial values such that the state satisfies $\mathbf{x}(t_i) \in [\mathbf{m}_i]$ for $i \in \{1, \ldots, p\}$. Using the solution operator, this problem is easily cast to a standard NCSP (V, D, C) where: $V = \mathbf{x} = (x_1, \ldots, x_n)$ are the initial conditions we search, $D = [\mathbf{x}] = ([\mathbf{x}_1], \ldots, [\mathbf{x}_n])$ is the initial search region for the initial conditions and

$$C = \{\Phi(\mathbf{x}, t_1) \in [\mathbf{m}_1], \ldots, \Phi(\mathbf{x}, t_p) \in [\mathbf{m}_p]\}. \tag{9}$$

Each constraint represents two vectorial inequalities which can be handled using the evaluation test and the unidimensional interval operator as explained in Section 2.

However, an efficient handling of the conjunction of constraints C require an important optimization: When evaluating the solution operator for $\mathbf{x}$ and t_i, the interval integrator performs a simulation from 0 to t_i. Therefore, if the interval evaluation of the solution operator for different times are performed independently then simulations are unnecessarily repeated. Therefore, only one simulation from 0 to t_p must be performed to evaluate the solution operator and its Jacobian for all time measures.

3.4 Two-Point Boundary Value Problems

A two-point boundary value problem (TPBVP) consists of an ODE $\mathbf{x}'(t) = \mathbf{h}(\mathbf{x}(t))$ and n equality constraints g_i that relate $\mathbf{x}(t_0)$ and $\mathbf{x}(t_1)$. Solving the TPBVP consists in finding trajectories $\mathbf{x}(t)$ satisfying the ODE and the constraints $g_i(\mathbf{x}(t_0), \mathbf{x}(t_1)) = 0$. As the trajectory is completely defined by its initial condition, the TPBVP actually consists in finding the initial conditions that give rise to such trajectories. Such a problem perfectly fits the NCSP framework since these initial conditions are the solutions of the NCSP (V, D, C) where: $V = \mathbf{x} = (x_1, \ldots, x_n)$ are the initial conditions we search, $D = [\mathbf{x}] = ([\mathbf{x}_1], \ldots, [\mathbf{x}_n])$ is the initial search region for the initial conditions and $C = \{c_1(\mathbf{x}) = 0, \ldots, c_n(\mathbf{x}) = 0\}$ with

$$c_i(\mathbf{x}) = g_i(\mathbf{x}, \Phi(\mathbf{x}, t_1 - t_0)). \tag{10}$$

Note that once $[\Phi]([\mathbf{x}], t_1 - t_0)$, $[\Phi](\text{mid}[\mathbf{x}], t_1 - t_0)$ and $[D\Phi]([\mathbf{x}], t_1 - t_0)$ are evaluated then $[c_i]([\mathbf{x}])$, $[c_i](\text{mid}[\mathbf{x}])$ and $[Dc_i]([\mathbf{x}])$ directly follow from (10). In particular using the chain rule we obtain

$$Dc_i(\mathbf{x}) = \left.\frac{\partial g(\mathbf{u}, \mathbf{v})}{\partial \mathbf{u}}\right|_{(\mathbf{x},\mathbf{y})} + \left.\frac{\partial g(\mathbf{u}, \mathbf{v})}{\partial \mathbf{v}}\right|_{(\mathbf{x},\mathbf{y})} D_{\mathbf{x}}\Phi(\mathbf{x}, t_1 - t_0) \tag{11}$$

with $\mathbf{y} = \Phi(\mathbf{x}, t_1 - t_0)$. The expression (11) can be evaluated for interval arguments using $[\Phi]([\mathbf{x}])$ and $[D\Phi]([\mathbf{x}])$.

Remark 3. In some cases the number of variables of the problem can be reduced, which ease its resolution by a branch and prune algorithm. This is the case typically when one constraint g_i involves only the state at $t = t_0$ and can be solved formally. For example let $g_1(\mathbf{u}, \mathbf{v}) = u_1 - u_2$ and $g_2(\mathbf{u}, \mathbf{v}) = v_1 - v_2$ so the TPBVP to be solved is $x_1(t_0) = x_2(t_0)$ and $x_1(t_1) = x_2(t_1)$. So we need only to perform the search on $x_1(0)$ since $x_2(0)$ is a function of the former. So the NCSP we solve will actually be: $V = \{x\}$, $D = \{[x]\}$ and $C = \{\Phi((x, x), t_t)_1 - \Phi((x, x), t_t)_2 = 0\}$ which is simpler than the original NCSP involving the constraints (10).

4 Experiments

Experiments have been carried out on a 2.53 GHz Intel Dual Core. We used the interval ODE integrator available as part of the CAPD library[5]. In theory $[\Phi]([\mathbf{x}], [t])$, $[D\Phi]([\mathbf{x}], [t])$ and $[\Phi](\text{mid}[\mathbf{x}], [t])$ can be computed during one single integration. However, the CAPD integrator does not offer this possibility so we performed two integrations independently, thus approximately doubling the computational time. Also, the CAPD integrator is optimized for small initial conditions (which is required for the usual usage of this integrator related to chaotic dynamical systems investigation) so we obtained very sharp enclosures of solutions, but poorer performances for exploring large domains.

[5] CAPD is available at `http://capd.ii.uj.edu.pl/`

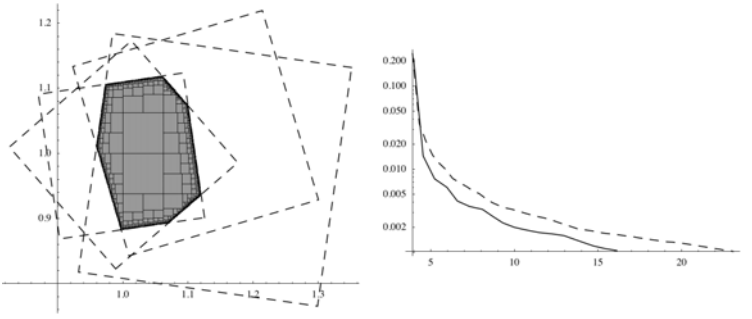

Fig. 2. Left: Paving obtained for the academic problem of Section 4.1 (solution and boundary boxes are in gray and black respectively). Right: Volume of the boundary boxes vs time for the interval enclosure test (dashed) and the Newton contractor (plain).

4.1 Parameter Estimation Problems

Academic Example. It is a simple problem for which we can formally compute the solution set. We look for the initial conditions $\mathbf{x}(0) \in ([0,2],[0,2])$ for which the trajectories of the ODE $\mathbf{x}'(t) = A\mathbf{x}(t)$ with

$$A = \begin{pmatrix} 0.1 & 1 \\ -1 & 0.1 \end{pmatrix} \tag{12}$$

(trajectories are spirals that unroll toward infinity) belong to the box $[\mathbf{m}_i] = \mathbf{m}_i \pm 0.25$ at time t_i. In that purpose, we dispose of the following values:

t_i	3	5	7	8
$\mathbf{m}_i$	$(-1.34,-1.52)$	$(-1.11,2.24)$	$(2.84,0.19)$	$(1.87,-2.52)$

As the ODE is linear, its trajectories can be formally computed using the matrix exponential: $\mathbf{x}(t) = \exp(tA)\mathbf{x}(0)$. Therefore, each constraint that forces the trajectory to be in a specific box at time t_i actually implies that the initial condition belongs to the parallelepiped $\{\exp(t_iA)^{-1}\mathbf{u} : \mathbf{u} \in \mathbf{m}_i \pm 0.25\}$. These parallelepipeds are represented in dashed lines on the left hand side diagram of Figure 2 together with the paving computed by our algorithm. The right hand side diagram of Figure 2 shows the total volume of the boundary boxes (vertical axis) that decreases with time (horizontal axis). The plain curve represents the timings obtained using both the interval enclosure test and the contractor based on the interval Newton, while the dashed curve corresponds to the interval enclosure test alone. We can see that the Newton based contractor improves the resolution, although it could be used more efficiently: Indeed, it could also be used to compute inner boxes (cf. [20]) or in conjunction with parallelotope domains (cf. [21]) although this framework needs to be extended to inequality constraints for this purpose.

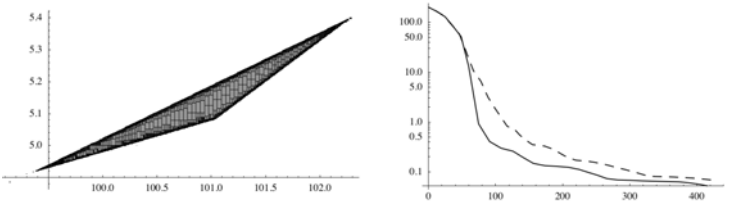

Fig. 3. Left: Paving obtained for the kinetic parameters estimation of an enzymatic reaction of Section 4.1 (solution and boundary boxes are in gray and blackrespectively). Right: Volume of the boundary boxes versus time for the interval enclosure test (dashed) and the Newton contractor (plain).

Estimating Kinetic Parameters of an Enzymatic Reaction. In [22] was reported the important potential interest of rigorously simulating the following kinetic enzymatic reaction:

$$\begin{aligned} s'(t) &= -\frac{V_{\max}\, s(t)}{k_s + s(t)} \\ p'(t) &= \frac{V_{\max}\, s(t)}{k_s + s(t)} \end{aligned} \tag{13}$$

However, the used implementation of HybridCC [23,24] could not integrate this ODE sharply enough to provide interesting numerical evidences. The NCSP framework proposed herein uses CAPD which is able to integrate sharply the ODE (13). Thus, we are in position to perform some parameter estimation of this model. Some measurements are provided for parameter estimation in [25,26], however the duration of the experiment is too long for CAPD to integrate it sharply enough for large initial conditions. Thus, we have prepared the following set of measurements by simulating the system for $s(0) = 25$, $p(0) = 0$, $V_{\max} = 100$ and $k_s = 5$ and added a random noise of amplitude 0.1. We obtained the following measurements:

t_i	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
p_i	8.01	15.32	21.01	23.92	24.74	24.96	24.97	25.02	24.95	24.91

The parameter estimation problem consists in finding the parameter values $(V_{\max}, k_s)$ in the initial domain $([90, 110], [0, 10])$ which satisfy $p(t_i) \in p_i \pm 0.1$. Results are shown in Figure 3. We see again that the interval Newton based contractor improves the resolution in spite of the possible improvements of its integration with the solution operator evaluation.

4.2 Two-Point BVPs

We have solved two boundary value problems that were studied in [6]. The method proposed in [6] consists of a bisection algorithm that implements a specific filtering algorithm on Taylor models (see [27] for details on the Taylor model method used in [6]). On the one hand, using Taylor models allows integrating

Table 1. Solutions for the catalytic reaction in a flat particle problem computed from the initial interval $[0, 1]$

λ	Solution enclosure	Width	Time (s)
0.05	0.97034556001404[65, 88]	2.2×10^{-15}	1.1
0.1	0.9226804137526[691, 733]	4.2×10^{-15}	3.7
	0.5058725840206[640, 786]	1.5×10^{-14}	
	0.06446821272269[81, 149]	6.7×10^{-15}	
0.15	0.01655884279376[13, 30]	1.6×10^{-15}	5.3

sharply larger initial conditions than the algorithm implemented in CAPD so larger initial domains can be used than in our implementation (cf. the tubular reactor model problem). On the other hand, the NCSP framework allows to compute sharper enclosures and proving rigorously the existence of solutions which is an important issue in BVP resolution. This detailed in the following.

Catalytic Reaction in a Flat Particle. Under some assumptions, the catalytic reaction of a particle can be modeled by the following BVP (cf. [28]):

$$\begin{aligned} x_1'(t) &= x_2(t) \\ x_2'(t) &= \lambda x_1(t) \exp\Big(\frac{\gamma\beta\big(1-x_1(t)\big)}{1+\beta\big(1-x_1(t)\big)}\Big) \\ x_2(0) &= 0 \\ x_1(1) &= 1. \end{aligned} \tag{14}$$

This is naturally expressed as a NCSP as shown in Section 3.4. As in [6], we solved the problem for $\gamma = 20$, $\beta = 0.4$, three values of λ and the initial search interval $x_1(0) \in [0, 1]$. Table 1 reports the results obtained. We have found the same solutions in approximately the same computation time as in [6]. However, the enclosures obtained by our algorithm are much sharper, while we have proved the existence of one unique solution in each of them. This is an important advantage of the usage of the interval Newton as usually implemented in the NCSP framework.

Tubular Reactor Model. The following simplified model of a tubular reactor was studied in [29,6]:

$$\begin{aligned} x_1'(t) &= x_2(t) \\ x_2'(t) &= 6\Big(x_2(t) - 0.05\big(1 - x_1(t)\big) \exp\big(\tfrac{10x_1(t)}{1+0.5x_1(t)}\big)\Big) \\ x_2(0) &= 6x_1(0) \\ x_2(1) &= 0. \end{aligned} \tag{15}$$

In [6] the initial search domain was $x_1(0) \in [0, 1]$. However, we found that a trajectory starting at $x_1(0) = 0$ and $x_2(0) = 0$ probably converges to a singularity in finite time, which prevents any further integration[6]. The bisection

[6] This behavior was pointed out by Daniel Wilczak in a personal communication.

Table 2. Solutions for the tubular reactor model computed in 30 seconds from the initial interval $[0.01, 0.04]$

Solution enclosure	Width
$0.010042462528303[8, 9]$	2.8×10^{-17}
$0.01844542857635[59, 68]$	7.6×10^{-16}
$0.03738712388979[57, 179]$	1.3×10^{-14}

algorithm proposed in [6] is able to get rid of this convergence to a singularity by enforcing the constraint $x_1(t) \in [0, 1]$ for all t, which is justified by physical considerations. This additional constraint strongly ease the resolution of the BVP, but we have not yet been able to integrate this constraint in our framework. This together with the fact that CAPD is optimized for small initial conditions (while the ODE integrator used in [6] better integrates larger initial conditions) restricted us to a smaller initial domain: Table 2 shows the enclosures computed by our algorithm starting from the initial domain $x_1(0) \in [0.01, 0.04]$ (which contains all the BVP solutions). We have found the same solutions as in [6]. However, five intervals were found in [6] and the authors conjectured that three of them were corresponding to the same solution (because they were very small neighbors). Our algorithm gave the rigorous proof that a unique solution lies in each intervals hence proving that this BVP actually has three solutions, and provided sharper enclosures of these solutions. These are the typical advantages of the NCSP framework.

5 Conclusion

We have shown that abstracting an ODE by its solution operator allows including naturally any ODE based constraint into the standard framework of numerical CSPs (NCSP). This permits the homogenizing and generalizing of several state of the art algorithms, like for example those dedicated to parameter estimation and boundary value problems. The expression of problems involving ODE in this framework presents many advantages: First it allows to separate the problem declaration and its resolution, which is a well known advantage of the CP framework. Second, the resolution of these problems can benefit from present and future efficient methods designed in the NCSP framework.

Our current implementation of this framework is not yet able to solve the hardest problems solved by dedicated state of the art methods. However, it has already shown some advantages with some very sharp BVP solution enclosures and their existence and uniqueness proof.

Tackling larger problems require strong optimization of the resolution process. For example, experiments have shown that at the beginning of the search, when intervals are large, the pruning is not efficient so reducing the order of the Taylor expansion would certainly pay off. Also, we are using today the interval integrator from the CAPD library but implementing a dedicated interval

integrator would allow integrating ODE more carefully in order to save computations that are not necessary to the CSP resolution process. Also, this interval integrator is optimized for small initial conditions, which is good for computing sharp enclosure of solutions at the end of the resolution process but which is not efficient at the beginning of the search process when domain to be proceeded are large.

Finally, some specific NCSP features like universally quantified constraints, parallelotopes domains, rigorous global optimization can naturally be used with ODE based constraints, which can lead to interesting developments.

Acknowledgments

This work was supported in part by the International Internship Program of the National Institute of Informatics, Japan. The authors are particularly grateful to Daniel Wilczak who helped them using the ODE integrator available in CAPD, and for his helpful comments on our simulations.

References

1. Raïssi, T., Ramdani, N., Candau, Y.: Set membership state and parameter estimation for systems described by nonlinear differential equations. Automatica 40(10), 1771–1777 (2004)
2. Granvilliers, L., Cruz, J., Barahona, P.: Parameter estimation using interval computations. SIAM J. Sci. Comput. 26(2), 591–612 (2005)
3. Kapela, T., Simó, C.: Computer assisted proofs for nonsymmetric planar choreographies and for stability of the eight. Nonlinearity 20(5), 1241 (2007)
4. Lin, Y., Stadtherr, M.A.: Guaranteed state and parameter estimation for nonlinear continuous-time systems with bounded-error measurements. Industrial & Engineering Chemistry Research 46(22), 7198–7207 (2007)
5. Johnson, T., Tucker, W.: Brief paper: Rigorous parameter reconstruction for differential equations with noisy data. Automatica 44(9), 2422–2426 (2008)
6. Lin, Y., Enszer, J.A., Stadtherr, M.A.: Enclosing all solutions of two-point boundary value problems for odes. Computers & Chemical Engineering 32(8), 1714–1725 (2008)
7. Cruz, J., Barahona, P.: Constraint Satisfaction Differential Problems. In: Rossi, F. (ed.) CP 2003. LNCS, vol. 2833, pp. 259–273. Springer, Heidelberg (2003)
8. Cruz, J., Barahona, P.: Constraint reasoning in deep biomedical models. Artificial Intelligence in Medicine 34(1), 77–88 (2005)
9. Moore, R.: Interval Analysis. Prentice-Hall, Englewood Cliffs (1966)
10. Neumaier, A.: Interval Methods for Systems of Equations. Cambridge Univ. Press, Cambridge (1990)
11. Benhamou, F., Older, W.: Applying Interval Arithmetic to Real, Integer and Boolean Constraints. Journal of Logic Programming 32(1), 1–24 (1997)
12. Van Hentenryck, P., McAllester, D., Kapur, D.: Solving polynomial systems using a branch and prune approach. SIAM J. Numer. Anal. 34(2), 797–827 (1997)
13. Granvilliers, L.: A symbolic-numerical branch and prune algorithm for solving nonlinear polynomial systems. Journal of Universal Computer Science 4(2), 125–146 (1998)

14. Goldsztejn, A., Goualard, F.: Box Consistency through Adaptive Shaving. In: Proc. of ACM SAC 2010, pp. 2049–2054 (2010)
15. Hairer, E., Nørsett, S.P., Wanner, G.: Solving Ordinary Differential Equations I. Springer, Heidelberg (2000)
16. Nedialkov, N.S., Jackson, K.R., Corliss, G.F.: Validated Solutions of Initial Value Problems for Ordinary Differential Equations. Applied Mathematics and Computation 105(1), 21–68 (1999)
17. Janssen, M., Deville, Y., Hentenryck, P.V.: Multistep filtering operators for ordinary differential equations. In: Jaffar, J. (ed.) CP 1999. LNCS, vol. 1713, pp. 246–260. Springer, Heidelberg (1999)
18. Zgliczynski, P.: C^1-Lohner Algorithm. Foundations of Computational Mathematics 2(4), 429–465 (2002)
19. Goldsztejn, A., Hayes, W.: Reliable Inner Approximation of the Solution Set to Initial Value Problems with Uncertain Initial Value. In: Proceedings of SCAN 2006, p. 19. IEEE Press, Los Alamitos (2006)
20. Goldsztejn, A., Michel, C., Rueher, M.: Efficient Handling of Universally Quantified Inequalities. Constraints 14(1), 117–135 (2008)
21. Goldsztejn, A., Granvilliers, L.: A New Framework for Sharp and Efficient Resolution of NCSP with Manifolds of Solutionss. Constraints 15(2), 190–212 (2010)
22. Eveillard, D., Ropers, D., de Jong, H., Branlant, C., Bockmayr, A.: A multi-scale constraint programming model of alternative splicing regulation. Theor. Comput. Sci. 325(1), 3–24 (2004)
23. Carlson, B., Gupta, V.: Hybrid cc with interval constraints. In: HSCC 1998: Proceedings of the First International Workshop on Hybrid Systems, pp. 80–95 (1998)
24. Gupta, V., Jagadeesan, R., Saraswat, V.A.: Computing with continuous change. Sci. Comput. Program. 30(1-2), 3–49 (1998)
25. Kuzmic, P.: Program DYNAFIT for the Analysis of Enzyme Kinetic Data: Application to HIV Proteinase. Analytical Biochemistry 237(2), 260–273 (1996)
26. Mendes, P., Kell, D.: Non-linear optimization of biochemical pathways: applications to metabolic engineering and parameter estimation. Bioinformatics 14(10), 869–883 (1998)
27. Lin, Y., Stadtherr, M.A.: Validated solutions of initial value problems for parametric odes. Applied Numerical Mathematics 57(10), 1145–1162 (2007)
28. Hlavácek, V., Marek, M., Kubícek, M.: Modelling of chemical reactors – X multiple solutions of enthalpy and mass balances for a catalytic reaction within a porous catalyst particle. Chemical Engineering Science 23(9), 1083–1097 (1968)
29. Chen, Y.: Dynamic systems optimization. Ph. D. Thesis, University of California (2006)