Journal of Complexity 36 (2016) 106–140

Contents lists available at ScienceDirect

Journal of Complexity

journal homepage: www.elsevier.com/locate/jco

Computing with polynomial ordinary differential equations

Olivier Bournez [a,*], Daniel Graça [b,c], Amaury Pouly [a]

[a] École Polytechnique, LIX, 91128 Palaiseau Cedex, France

[b] CEDMES/FCT, Universidade do Algarve, C. Gambelas, 8005-139 Faro, Portugal

[c] SQIG/Instituto de Telecomunicações, Lisbon, Portugal

ARTICLE INFO

Article history:
Received 23 November 2015
Accepted 22 April 2016
Available online 13 May 2016

Keywords:
Analog computation
Continuous-time computations
General Purpose Analog Computer
Real computations

ABSTRACT

In 1941, Claude Shannon introduced the General Purpose Analog Computer (GPAC) as a mathematical model of Differential Analysers, that is to say as a model of continuous-time analog (mechanical, and later on electronic) machines of that time.

Following Shannon's arguments, functions generated by the GPAC must satisfy a polynomial differential algebraic equation (DAE). As it is known that some computable functions like Euler's $\Gamma(x) = \int_0^\infty t^{x-1}e^{-t}dt$ or Riemann's Zeta function $\zeta(x) = \sum_{k=0}^\infty \frac{1}{k^x}$ do not satisfy any polynomial DAE, this argument has often been used to demonstrate in the past that the GPAC is less powerful than digital computation.

It was proved in Bournez et al. (2007), that if a more modern notion of computation is considered, i.e. in particular if computability is not restricted to real-time generation of functions, the GPAC is actually equivalent to Turing machines.

Our purpose is first to discuss the robustness of the notion of computation involved in Bournez et al. (2007), by establishing that many natural variants of the notion of computation from this paper lead to the same computability result.

Second, to go from these computability results towards considerations about (time) complexity: we explore several natural variants for measuring time/space complexity of a computation.

Quite surprisingly, whereas defining a robust time complexity for general continuous time systems is a well known open problem,

* Corresponding author.
E-mail addresses: bournez@lix.polytechnique.fr (O. Bournez), dgraca@ualg.pt (D. Graça), pamaury@lix.polytechnique.fr (A. Pouly).

http://dx.doi.org/10.1016/j.jco.2016.05.002
0885-064X/

we prove that all variants are actually equivalent even at the complexity level. As a consequence, it seems that a robust and well defined notion of time complexity exists for the GPAC, or equivalently for computations by polynomial ordinary differential equations.

Another side effect of our proof is also that we show in some way that polynomial ordinary differential equations can actually be used as a kind of programming model, and that there is a rather nice and robust notion of ordinary differential equation (ODE) programming.

1. Introduction

Claude Shannon introduced in [20] the General Purpose Analog Computer (GPAC) as a model for Differential Analysers [7], which are mechanical (and later on electronic) continuous time analog machines, on which he worked as an operator. The model was later refined in [16,14]. It was originally presented by Shannon as a model based on circuits. Basically, a GPAC is any circuit (loops are allowed[1]) that can be built from the 4 basic units of Fig. 1, which implement constants, addition, multiplication and integration, all of them working over analog real quantities (that were corresponding to angles in the mechanical Differential Analysers, and later on to voltages in the electronic versions). Note that the set of allowed constants will generally be restricted, for example to rational numbers, to avoid pathological issues. Given such a circuit, the function which gives the value of every wire (or a subset of the wires) over time is said to be *generated* by the circuit. In Definition 11, we consider an extension of this notion.

An important aspect of this model is that despite the apparent simplicity of its basic blocks, sophisticated functions can easily be generated. Fig. 2 illustrates how the sine function can be generated using two integrators, with suitable initial states. Incidentally, the sine function is also the solution of a very simple ordinary differential equation. Shannon itself realized that functions generated by a GPAC are nothing more than solutions of a special class of polynomial differential equations. In particular it can be shown that a function $f : \mathbb{R} \to \mathbb{R}$ is generated by Shannon's model [20,14] if and only if it is a (component of the) solution of a polynomial initial value problem (PIVP) of the form:

$$\begin{cases} y'(t) = p(y(t)) \\ y(t_0) = y_0, \end{cases} \qquad t \in \mathbb{R} \tag{1}$$

where p is a vector of polynomials and $y(t)$ is vector. In other words, $f(t) = y_1(t)$, and $y_i'(t) = p_i(y(t))$ where p_i is a multivariate polynomial.

Intuitively, the link between a GPAC and a PIVP is the following: the idea is just to introduce a variable for each output of a basic unit, and write the corresponding ordinary differential equation (ODE), and observe that it can be written as an ODE with a polynomial right hand side.

While many of the usual real functions are known to be generated by a GPAC, a notable exception is Euler's Gamma function $\Gamma(x) = \int_0^{\infty} t^{x-1}e^{-t}dt$ function or Riemann's Zeta function $\zeta(x) = \sum_{k=0}^{\infty} \frac{1}{k^x}$ [20,17], which are known not to satisfy any polynomial DAE, i.e. they are not solutions of a system of the form (1). If we have in mind that these functions are known to be computable under the computable analysis framework [17,21] the previous result has long been interpreted as evidence that the GPAC is a somewhat weaker model than computable analysis.

In 2007, it was proved that this is more an artefact of the notion of real-time generation considered by Shannon than a true consideration about the computational power of the model. Indeed, Shannon

[1] There are some syntactic restrictions to avoid ill-defined circuits.