Proposal for M2 Research Internship for ENS Lyon for 2024

Programming with Ordinary Differential Equations and Continuous Time: Towards a programming language or pseudo-programming language.

Olivier Bournez

Keywords: Continuous time models of computation. Analog machines and models. Ordinary Differential Equations. Computability. Complexity.

Supervisor: Olivier Bournez.

Administrative Position: Professor (of Computer Science) at Ecole Polytechnique.

Location: Laboratoire d’Informatique de l’X, LIX, Ecole Polytechnique, 91128 Palaiseau Cedex

Phone: +33 (0)1 77 57 80 78

Email: olivier.bournez@lix.polytechnique.fr.

Website: http://www.lix.polytechnique.fr/~bournez.

Language: French or English. This proposal is intentionally written in English for Non-French speaking students who may be interested.

General Introduction

It has been understood quite recently that it is possible to program with Ordinary Differential Equations (ODEs). This actually was obtained as a side effect from attempts to relate the computational power of analog computational models to classical computability. The former are working over continuous data (for example voltage, or concentrations) with a continuous time (ex: according to a system of ordinary differential equations corresponding to some analog electronic circuits, or the kinetic of reactions), unlike the latter, such as Turing machines, working on discrete entities such as words, with a discrete time.

Refer to [5, 3, 14] for surveys on analog computation with a point of view based on computation theory aspects, relating these models to discrete models, or to [17, 16, 12] for surveys discussing historical and technological aspects related to analog models of computation. Analog models of computation include in particular the differential analyzers.

This has been obtained by realizing that continuous time processes defined by ODEs, and even defined by polynomial ODEs, can simulate various discrete time processes. They hence can be used to simulate models such as Turing machines [6, 10], and even some more exotic models working with a discrete time but over continuous data. This is based on various refinements of constructions done in [6, 10, 11, 15, 8]. We call this ODE programming, as this is indeed some kind of programming with various continuous constructions.

Forgetting analog machines or models of computation, it is important to realize that ODEs is a kind of universal language of mathematics that is used in many, if not all, experimental sciences: Physics, Biology, Chemistry, . . . . Consequently, once it is known that one can program with ODEs, many questions asking about universality, or computations, in experimental contexts can be solved. This is exactly what has been done by several authors, including ourselves, to solve various open problems, in various contexts such as applied maths, computer algebra, biocomputing, . . .

Applications include: Characterization of computability and complexity classes using ODEs [11, 10, 15, 8]; Proof of the existence of a universal (in the sense of Rubel) ODE [4]; Proof of the strong Turing completeness of biochemical reactions [7], or more generally various statements about the completeness of reachability problems (e.g. PTIME-completeness of bounded reachability) for ODEs [13].

Description of the work

However, when going to say more about proofs, their authors including ourselves, often feel frustrated: Currently, the proofs are mostly based on technical lemmas and constructions done with ODEs, often mixing both the
ideas behind these constructions, with numerical analysis considerations about errors and error propagation in the equations. We believe this is one factor hampering a more widespread use of this technology in other contexts.

The article [2, 1] are born from an attempt to popularize this ODE programming technology to a more general public, and in particular master and even undergraduate students. We show how some constructions can be reformulated using some notations, that can be seen as a pseudo programming language. This provides a way to explain in an easier and modular way the main intuitions behind some of the constructions, focusing on the algorithm design part. We focused here, as an example, on how the proof of the universality of polynomial ODEs (a result due to [9], and fully developed in [10]) can be reformulated and presented.

The purpose of the internship will be to develop this approach, in order to reformulate or improve some of the constructions. New results include the characterization of complexity classes with ordinary differential equations, or applications in experimental sciences.

Comment

The actual topic of the work is related to computability theory. This requires only common and basic knowledge in ordinary differential equations. Most of the intuitions of our today’s constructions come from classical computability.

There is no specific prerequisite for this internship, except some knowledge about computability theory. This subject can be extended to a PhD. Possibilities of funding according to the administrative situation of candidates.

The subject can also be adapted according to requests, knowledge and skills of candidates. Please contact me if interested or in case of questions.

References


