Task 5 : Tools
Deliverable 1: Choice of the architecture and first implementation of the core

April 2018

Involved DELTA members. Olivier Carton (IRIF), Olivier Gauwin (LaBRI), Benjamin Monmege (Task leader, LIS), Pierre-Alain Reynier (LIS), Jean-Marc Talbot (LIS), Didier Villevalois (LIS), Marc Zeitoun (LaBRI).

Recall of the context and motivation of the task. Several tools implement the Delta relationship between automata, logics and algebra into applicable software. As examples, we can cite the long list of libraries dealing with the efficient treatment of regular expressions, in various programming languages. Efficient pattern matching heavily relies on the use of finite state automata.

With respect to the extensions considered in this project, several tools have also been developed: Vaucanson [LRGS04] (that lifts automata and regular expressions in a wide context of transducers and weighted automata), OpenFST [All+07] (that constructs, combines, optimises, and searches weighted finite-state transducers), DReX [ADR15] (that proposes an efficient evaluation mechanism based on streaming string transducers).

Apart from tools, various logical specification languages have been proposed to ease the process of writing the desired properties of a system. Starting from linear temporal logic (LTL), which is widely used in the research community and for industrial purposes, several extensions have been designed to enhance the expressive power: Regular LTL [LS07; LS10], propositional dynamic logic (PDL) [FL79] now widely known as the Propositional Specification Language (PSL), governed by the 1.1 IEEE Standard 1850-2005.

Such handy and powerful logical specification languages have been investigated only superficially for quantitative functions or transductions. Mean-payoff automata expressions [Cha+10] have been shown to enjoy nice closure and decision properties, and other formalisms, with a more logical flavor, have been proposed by members of the project consortium in more general semirings (able to deal both with quantitative functions and transductions): weighted monadic second order logic (WMSO) [DG07], weighted first-order logic with transitive closure (WFOTC) [Bol+14; Bol+10], hybrid weighted expressions or hybrid weighted PDL [Mon13] that have been partially handled in a tool called QuantiS.

The objective of the task is to build, and disseminate, a **core tool for modeling and reasoning about transductions and quantitative functions**, with frontends able to specify them with a **variety of logical languages (or regular expressions)**. It must provide a toolbox of procedures, extracted from other tasks of the project: test for equivalence, evaluation, type-checking, simplification, translation from a formalism to another... The main goal is to obtain a tool with a simple and scalable interface, yet with a stable core.

Methodology. In the first 18 months of the project, we have worked towards the objectives of this task as follows:

- around the task leader, several meetings have been organised to draw the contour of the core data structures and algorithms that should be preferably implemented at first;
- members of the project present at the first consortium meeting (from 17 to 20 October 2017, on Porquerolles Island) have thoroughly discussed the objectives (going from an internal tool for the DELTA consortium or the nearby community only, up to a way to host other tools of the community with designated benchmarks, going through a pedagogical tool for Master's students) and ways to disseminate the tool (independant tool, or library integrated in the SageMath open-source mathematics software, extending the capabilities of the existing library for finite state machines).

Achievements. We have chosen to concentrate our first study to a relatively restricted model of automata in the core. Considering the consortium, we opted for a transducer/weighted automaton model, based on input/output transitions: this temporarily rules out streaming string transducers or cost-register automata, based on registers. We also concentrated ourselves to classical models, without stacks (for which the consortium has nevertheless worked in the last years [Dar+16; Fil+10]), and for which input structures are words (and not trees or nested words). Therefore, the model of interest which we focus on in our first implementation is **one-way or two-way finite state automata with output labels that are either words** (to model transducers) **or elements of a semiring** (like real numbers, integers, matrices, or languages).

To choose the language in which we implement the core tool (this is a different question than the integration in SageMath as a library, that is possible whatever language we choose), Didier Villevalois produced a first proof-of-concept tool in the Scala programming language: Scala is a language that combines object-oriented and functional programming, and compiles itself to target the Java Virtual Machine, JavaScript, or LLVM, allowing for building a high-performance system with easy access to many ecosystems of libraries. In the proof-of-concept tool, a data structure for storing automata models has been written in two forms: an explicit storage of the automaton models (i.e. transitions are stored explicitly), and a symbolic storage where transitions are computed on-the-fly when needed. Then, basic algorithms have been implemented to evaluate the automaton (compute the output associated with a given input word) or check the membership of a pair input/output. The choice of a functional programming language has permitted to produce relatively concise pieces of code, that seem easily extendable to more general models later. Nevertheless, the compiling on a JVM makes the language Scala a good choice to distribute easily the tool afterwards. We plan to keep this choice in the future.

Next objectives. In the next months, we plan on quickly extending this first version with several other needed algorithms like some ones performing closure operations (concatenation, union, Kleene star, mirror...), to check the emptiness of an automaton, or the equivalence of two automata (in restricted cases to obtain decidability, and efficiency). We also need to define particular subclasses of the general model of automata we designed, like deterministic automata, one-way automata, which then allows one to implement known algorithms to check such subclasses (determinisation or one-way definability, for instance), or to wonder about other simplification of models (like minimisation of the number of states). A last short-term objective is to study more carefully the integration of our tool as a SageMath library.

Afterwards, our main objective will be to allow for other specification mechanisms to specify properties of transductions and weighted functions, e.g. the use of logic and not only automata and regular expressions. Various frontends will be developed for each logical formalisms: translation in-between different formalisms will be provided as much as possible. The DELTA postdoc involved in this task will have for mission to work on the developments of algorithms in the frontend related to visibly pushdown transducers: we plan to hire him to start in the last trimester of 2018. The software engineers team of LIS will also help us in the next months.

References

[ADR15] Rajeev Alur, Loris D'Antoni, and Mukund Raghothaman. "DReX: A Declarative Language for Efficiently Evaluating Regular String Transformations". In: *Proc. of POPL'15*. ACM, 2015.

[All+07] Cyril Allauzen et al. "OpenFst: A general and efficient weighted finite-state transducer library". In: *Proc. of CIAA'07*. Springer, 2007.

[Bol+10] Benedikt Bollig et al. "Pebble Weighted Automata and Transitive Closure Logics". In: *Proc. of ICALP'10*. Springer, 2010.

[Bol+14] Benedikt Bollig et al. "Pebble Weighted Automata and Weighted Logics". In: *ACM Transactions on Computational Logic* 15.2:15 (Apr. 2014).

[Cha+10] Krishnendu Chatterjee et al. "Mean-Payoff Automaton Expressions". In: *Proc. of CONCUR'10*. Springer, 2010.

[Dar+16] Luc Dartois et al. "Two-Way Visibly Pushdown Automata and Transducers". In: *Proc of LICS'16*. 2016.

[DG07] Manfred Droste and Paul Gastin. "Weighted Automata and Weighted Logics". In: *Theor. Comput. Sci.* 380.1-2 (2007).

[Fil+10] Emmanuel Filiot et al. "Properties of Visibly Pushdown Transducers". In: *Proc. of MFCS'10*. Springer, 2010.

[FL79] Michael J. Fischer and Richard E. Ladner. "Propositional Dynamic Logic of Regular Programs". In: *J. Comput. System Sci.* 18 (1979).

[LRGS04] Sylvain Lombardy, Yann Régis-Gianas, and Jacques Sakarovitch. "Introducing Vaucanson". In: *Theor. Comput. Sci.* 328.1-2 (2004).

[LS07] Martin Leucker and César Sánchez. "Regular Linear Temporal Logic". In: *Proc. of ICTAC'07*. Springer, 2007.

[LS10] Martin Leucker and César Sánchez. "Regular Linear Temporal Logic with Past". In: *Proceedings of VMCAI'10*. Springer, 2010.

[Mon13] Benjamin Monmege. "Specification and Verification of Quantitative Properties: Expressions, Logics, and Automata". PhD Thesis. Laboratoire Spécification et Vérification, ENS Cachan, France, Oct. 2013.