GRACeFUL

Global systems Rapid Assessment tools
through Constraint FUnctional Languages

FETPROACT-1-2014 Grant N° 640954

Constraints Composition Ops Library

D5.2

Lead Participant:	Armines (Nicolas Beldiceanu, Ekaterina Arafailova, Rémi Douence)
Partners Contributing:	UPC
Dissemination Level:	PU
Document Version:	Final
Date of Submission:	2016-09-30
Due Date of Delivery:	2016-09-30

1 Reading this document

All cyan links of this document can be clicked in order to access to the corresponding page on the web. This allows to put things in context.

In order to get the files attached to this document, you are advised to use **Adobe Reader** since we are using the LaTeX package attachfile and since many PDF viewers do not support attachments.

The source code of the **MiniZinc** implementation as well as the instructions for using it are provided as attached files in this document available by clicking on the **paper clip** icons located in Section 4. The same files are publicly available at https://github.com/GRACeFUL-project.

2 Introduction

This report itself is not the deliverable, but summarises the content of the deliverable, which consists of:

1. The research papers done for bridging the gap between the definition of constraints as composition of operators and the synthesis of the corresponding code for handling these constraints. The composition process is based on composing transducers, feature operators and aggregator operators.
2. An implementation in **MiniZinc** of these constraints based on this composition process.

Figure 1 illustrates the definition of constraints as multiple layers of operators, where the name of a constraint corresponds to the concatenation of the operators used for defining it.

Attached to this document it provides an open implementation of these constraints that uses the **MiniZinc** modelling language as well as a quick start how to use this implementation from existing solvers such as **Choco** or **SICStus Prolog**. The advantage of **MiniZinc** is to be interfaced with many solvers both from Constraint Programming, SAT, MIP and local search.

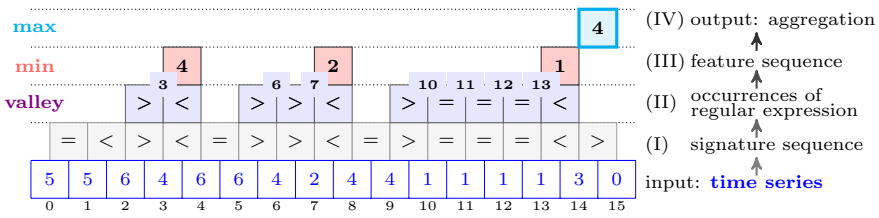

Figure 1: Compositional constraint definition as multiple layers of operators: pattern, feature and aggregation operators illustrating the constraint **max_min_valley**$(\langle 5,5,6,4,6,6,4,2,4,4,1,1,1,1,3,0\rangle, 4)$

3 Summary of scientific publications

The theory developed for synthesizing constraints from a functional specification is described in the following three scientific publications.

1. The way we describe constraints as a composition of operators and the way we synthesise automata constraints from transducers are described in this first paper [4]. The paper is freely accessible at https://hal.inria.fr/hal-01370322.

2. The way we optimise the corresponding automata in a mechanical way is described in this second paper [3]. The paper is freely accessible at https://hal.inria.fr/hal-01355262.

3. Finally the way we come up with combinatorial objects that are parametrised by the operators used in the constraint definition is described in this third paper [2]. These combinatorial objects correspond to parametrised bounds and parametrised glue matrices that are used for synthesising necessary conditions for the feasibility of each concrete constraint. The paper is freely accessible at https://hal.inria.fr/hal-01370317.

All three publications were presented at international conferences, **CP 2015**, **CPAIOR 2016** and **CP 2016**.

4 MiniZinc implementation

- A **MiniZinc** implementation of the time series constraints is available in this attached file  .
- Instructions how to use this **MiniZinc** implementation are given here  .
- Files referenced in the instructions are given here  and here  .

A detailed on-line synthesised catalogue explaining all the corresponding constraints and how they are synthesised with about 2000 illustrations is available as a **CoRR** report [1].

References

[1] E. Arafailova, N. Beldiceanu, R. Douence, M. Carlsson, P. Flener, M. A. F. Rodríguez, J. Pearson, and H. Simonis. Global constraint catalog, volume ii: Time-series constraints. *CoRR*, abs/1609.08925, 2016.

[2] Ekaterina Arafailova, Nicolas Beldiceanu, Mats Carlsson, Pierre Flener, María Andreína Francisco Rodríguez, Justin Pearson, and Helmut Simonis. Systematic derivation of bounds and glue constraints for time-series constraints. In Michel Rueher, editor, *Principles and Practice of Constraint Programming - 22nd International Conference, CP 2016, Toulouse, France, September 5-9, 2016, Proceedings*, volume 9892 of *Lecture Notes in Computer Science*, pages 13–29. Springer, 2016.

[3] Ekaterina Arafailova, Nicolas Beldiceanu, Rémi Douence, Pierre Flener, María Andreína Francisco Rodríguez, Justin Pearson, and Helmut Simonis. Time-series constraints: Improvements and application in CP and MIP contexts. In Claude-Guy Quimper, editor, *Integration of AI and OR Techniques in Constraint Programming - 13th International Conference, CPAIOR 2016, Banff, AB, Canada, May 29 - June 1, 2016, Proceedings*, volume 9676 of *Lecture Notes in Computer Science*, pages 18–34. Springer, 2016.

[4] Nicolas Beldiceanu, Mats Carlsson, Rémi Douence, and Helmut Simonis. Using finite transducers for describing and synthesising structural time-series constraints. *Constraints*, 21(1):22–40, 2016.