Information and Computation ••• (••••) •••–•••

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Automata for unordered trees[☆], [☆☆]

Adrien Boiret[a,b], Vincent Hugot[c,b,*], Joachim Niehren[c,b], Ralf Treinen[d]

[a] *University of Lille 1, France*
[b] *Links (Inria Lille & Cristal, UMR CNRS 9189), France*
[c] *Inria, France*
[d] *Univ Paris Diderot, Sorbonne Paris Cité, IRIF, UMR 8243, CNRS, F-75205 Paris, France*

ARTICLE INFO

Article history:
Received 19 April 2015
Available online xxxx

ABSTRACT

We present a framework for defining automata for unordered data trees that is parametrised by the way in which multisets of children nodes are described. Presburger tree automata and alternating Presburger tree automata are particular instances. We establish the usual equivalence in expressiveness of tree automata and MSO for our framework. We then investigate subclasses of automata for unordered trees for which testing language equivalence is in P-time. Starting from automata in our framework that describe multisets of children by finite automata, we propose two approaches to do this deterministically. We show that confluent horizontal evaluation leads to polynomial-time emptiness and universality, but coNP-complete emptiness and intersection. Finally, efficient algorithms can be obtained by imposing an order of horizontal evaluation globally for all automata in the class. Depending on the choice of the order, we obtain different classes of automata, each of which has the same expressiveness as Counting Mso.

1. Introduction

Data trees are a general and common model of hierarchical data structures, used in programming languages, structured documents, and in databases with nested relations. They are finite trees in which the edges or the nodes are labelled by data values from an infinite alphabet, most typically strings over some finite alphabet.

Unordered data trees are data trees for which there is, a priori, neither an order on the children of a tree node, nor a bound on their number. In the case where only the edges are labelled by data values, unordered data trees can be naturally represented in the JSON format (the JavaScript Object Notation [2]) as illustrated in Fig. 1$_{[p2]}$. JSON is a recent language-independent format for nested key-value stores, which has already found much interest in Web browsers and for NoSQL databases such as IBM's JAQL [3]. The unordered data tree of Fig. 1$_{[p2]}$ represents a part of a file system as it can be found on any modern operating system. As stated by the POSIX standard, there is no a priori order on the elements of a directory, so directory file trees are unordered data trees.

☆ Extended version of [1].

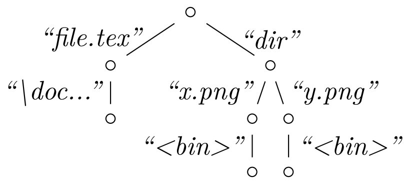
☆☆ This work has been partially supported by CPER Nord-Pas de Calais/FEDER DATA Advanced Data Science and Technologies 2015–2020 and the ANR project Colis, contract number ANR-15-CE25-0001-01.

* Corresponding author at: Inria, France.

E-mail addresses: adrien.boiret@inria.fr (A. Boiret), vincent.hugot@inria.fr (V. Hugot), joachim.niehren@inria.fr (J. Niehren), treinen@pps.univ-paris-diderot.fr (R. Treinen).

http://dx.doi.org/10.1016/j.ic.2016.07.012
0890-5401/© 2016 Elsevier Inc. All rights reserved.

```
{ "file.tex" : {"\documentclass...":{}},
  "dir" : {
    "x.png" : {"<bin>":{}},
    "y.png" : {"<bin>":{}},
  ...} ... }
```



(1.1)

Fig. 1. Unordered data trees in Json format, describing a typical file tree.

Logics and automata for unordered data trees were studied in the last twenty years mostly for querying Xml documents [4–6] and more recently for querying NoSql databases [7]. They were already studied earlier, for modelling feature structures in computational linguistics [8] and records in programming languages [9–11]. In particular, it was shown that Presburger tree automata can define the same languages of unordered data trees as Presburger Mso [4,5]. It is also folklore that the feature automata from [11] have the same expressiveness as Counting MSO, i.e., the restriction of Presburger Mso to counting constraints.

Counting Mso is still sufficiently expressive for most applications, when extended with regular expressions for matching data values. For instance, it can be used to express and verify that a LaTeX repository contains exactly one main file, i.e. one file whose name matches the regular expression *"*.tex*" (where * matches any string) and whose content matches "\documentclass"*. This is a common type of requirement, for instance when uploading the sources of a paper for remote compilation. Only main files can be compiled; others do not describe complete documents and are just meant to be included by the main file. If there are several main files, it is not clear which one should be compiled.

The mentioned notions of tree automata for unordered data trees have the advantage that membership can be tested in polynomial time. But they also have at least two disadvantages. First, they are inconvenient for modelling, since the nondeterministic semantics may intervene in a surprising manner with counting children in a given state. And second, when it comes to static analysis problems such as satisfiability, inclusion, or equivalence checking, they lack relevant subclasses of automata for which these problems can be solved efficiently. For Presburger tree automata, the problem is that their notion of determinism is limited to vertical processing, so that none of the above problems can be solved efficiently even for flat unordered trees. Feature automata [11] have a satisfactory notion of determinism, which captures vertical and horizontal processing, so that the above problems can be solved in polynomial time. This does not help much, however, since feature automata may grow exponentially in size when expressing simple patterns such as $\{a_1 : \{\},\ a_2 : \{\}, \ldots, a_n : \{\}\}$ where all a_i are different data values. The problem here is that feature automata must be able to read the n different data values in all possible orders. This requires 2^n states in order to memorise which subset of the n data values has already been read.

The first contribution of this paper is a general framework for defining automata for unordered data trees. Similarly to the framework for hedge automata for unranked ordered trees in [12], we keep the way in which horizontal languages are specified as a parameter. The canonical choices are to use counting constraints, leading to counting tree automata having the same expressiveness as feature automata, or Presburger constraints leading to Presburger tree automata. Alternating automata for unordered data trees are also supported by our framework. In particular, we obtain definitions of alternating counting tree automata and of alternating Presburger tree automata, which have not been studied before. It turns out that the alternating models are indeed more natural for modelling properties of unordered data trees than their nondeterministic counterparts. The advantage is that any node of the tree can be assigned to several states at the same time, so that state counting cannot be compromised by nondeterministic state choices; Example 22[p13] provides a concrete case showing that this typically happens in practice.

We establish the classical equivalence between monadic second-order logic (Mso) and the automata notions introduced by our framework. Which variant of Mso is chosen depends on which descriptor class of horizontal languages is permitted by the automata. In order to make the equivalence work, we have to impose some restrictions on the descriptor classes that are satisfied by all usual choices. Under these restrictions, we can also show that alternating automata have the same expressiveness as nondeterministic automata. The equivalence proof between Mso and nondeterministic automata is based on the usual closure properties of tree languages, which we establish for the automata defined in our framework. In order to show closure under complement, we rely on the notion of vertical determinism known from previous work on hedge automata and Presburger tree automata.

The second contribution is a study of subclasses of automata for unordered trees for which the usual decision problems can be solved efficiently. The focus here is on finding a good notion of horizontal determinism in addition to the usual vertical determinism. In the spirit of stepwise tree automata for unranked ordered trees [13], we shall use a single finite automaton for defining the many horizontal languages in the rules of an automaton. The horizontal languages are languages of multisets, whose elements may be read nondeterministically in any order by the finite automaton. Unsurprisingly, the membership problem becomes NP-hard in this case, since all orders must be inspected in the worst case. A first notion of horizontal determinism can then be defined by a restriction to confluent horizontal rewriting, so that the order of rewriting becomes irrelevant. For instance, one can test the above arity in the order $a_1, \ldots, a_n$ or else in the inverse order (but not necessarily in all orders, in contrast to feature automata). Our first positive result is that the restriction to confluent rewriting leads to polynomial-time membership, emptiness, and universality, as one might have hoped. However, the emptiness of binary intersections as well as inclusion still suffers from coNP-completeness, which might appear a little surprising, so