Automata and Logics over Nested Data

Adriana Baldacchino[a]

[a] *University of Oxford*

Abstract

Automata and logics over infinite data are widely studied due to their effectiveness as tools for verification, database theory and programming language semantics. We focus on automata and logics over *nested data*, which additionally model parent-child relationships between data values. Existing models focus on bounded depth nested data. We extend the ideas to unbounded depth nested data, introducing two new models of nested data automata. Furthermore, we relate these models to well-structured transition systems to prove decidability of emptiness of these automata, and explore their closure properties.

1 Introduction

Automata over infinite alphabets are extensions of classical automata, whose input consists of *data words*. These words consist of symbols (a, d) where $a \in \Sigma$ is a letter from a finite set, and d is the data value, which comes from some infinite set $\mathcal{D}$. These automata and their corresponding logics are widely studied in computer science, as the unbounded set $\mathcal{D}$ allows us to model various sets that arise naturally. This includes modelling attribute values in database theory, unbounded agents in verification and fresh variables in programming language semantics.

Our focus is on *class memory automata* (CMA) [2], which for each data value $d \in \mathcal{D}$, keeps track of the state at which it last saw d. This extra information is used when deciding the next transition. Furthermore, a CMA has a set of locally and globally accepting states, and it accepts a word if and only if the current state is globally accepting, and the data was last seen in locally accepting states. We further study *nested data class memory automata* (NDCMA) [5], which are a refinement of the class memory automata described above. These operate over *nested data*, first introduced in [1], which is data with an additional hierarchical structure. We consider the *weak* variant of these automata, where the set of locally accepting states is the set of all states, as this is necessary for decidability of emptiness.

Nested data can be expressed in different ways. One way (as used in [1,6]) is to consider words over the alphabet $\Sigma \times \mathcal{D}^k$ for a fixed k. Another interpretation (used in [5]) is to endow $\mathcal{D}$ itself with a forest structure by considering a predecessor relation $\text{pred} : \mathcal{D} \rightharpoonup \mathcal{D}$. These representations are equivalent as there are effective translations from one representation to the other, as shown in [5]. In our work we make use of the forest presentation, as our techniques follow those in [5].

Logics over nested data quickly become undecidable. Even with only two layers of nested data, two-variable first-order logic equipped with $x = y + 1$ and $x < y$ is undecidable. This was shown by a translation to multicounter automata [1]. In this same paper, it was shown that restricting to only the successor function results in a decidable logic, by a translation to shuffle expressions. A similar positive result was established in [6], where the authors established ND-LTL, a temporal logic on nested data words extending LTL by navigation along data values, which has a decidable satisfiability question. To do this, they introduce *prefix-closed nested data automata (pNDA)*, which turn out to be equivalent to weak NDCMA [5].

A key limitation to the existing models of automata over nested data is that the nesting is limited by some bound k. In the tuple representation, this is because each tuple is of size k, and for the forest representation this limitation arises as a bound for the depth of the forest structure of $\mathcal{D}$. Therefore, an interesting question to ask is whether this limitation can be lifted without introducing undecidability to the automata. We present new models on nested data extending NDCMA, which allow for parsing of data sets of unbounded depth and additionally have a decidable non-emptiness problem.

Given that NDCMA have been applied to obtain decidability results in the context of programming languages [3,4], our hope is that our extension could unlock further decidability results in the study of programming languages, as it would enable us to model unbounded creation of names, such as in loops. To aid in this pursuit, and to understand these models further, we study further closure properties of these automata, such as closure under complement, intersection and union. The talk will summarise the results of the author's master's project, which is supervised by Andrzej Murawski.

2 Defining Automata Models over Unbounded Nested Data

We now present the ideas behind two new models of unbounded-depth NDCMA. Note that to allow new data values to be available at any level, we require our forest to be infinitely full, that is there are infinite roots, and each node has an infinite number of children. Our automata need a finite description, so given some data value $d \in \mathcal{D}$ with an arbitrary number of predecessors $\text{pred}(d), \text{pred}^2(d), \ldots, \text{pred}^n(d)$ we need to find a way to only consider some subset of a fixed size k of these predecessors.

One way we can do this is by matching with any subsequence of predecessors of d. Essentially, our transition function is of the form $\Delta = \bigcup_{i=0}^{k} \Delta_i$ where

$$\Delta_i : Q \times \Sigma \times Q \times (Q \cup \{\bot\})^i \rightharpoonup \mathcal{P}(Q).$$

Then, given a data value d and the current labelling function $f : \mathcal{D} \to Q \cup \{\bot\}$, a transition $q \xrightarrow{a,p,p_1,\ldots,p_n} q'$ is possible if d is labelled with p, and the path $d = d_0, d_1, \ldots, d_k = r$ from d to the root contains a subsequence $d_{m_1}, d_{m_2}, \ldots, d_{m_n}$ which has the labels $p_1, \ldots, p_n$. We call such NDCMA *memoryless NDCMA*, as a node does not discriminate between any of its predecessors.

While memoryless NDCMA are well-defined models for handling unbounded nested data, they are a bit unwieldy to use. This is mainly because even if the automata is deterministic in the 'usual' sense, meaning the image of Δ_i is made up of singletons, a word can still have multiple possible runs due to the nondeterminism in picking the subsequence of predecessors.

To allow for more control, we define a different model, referred to as k-memory NDCMA. As the name suggests, in this model each node has a bounded size memory. Whenever a new data value d is encountered, its memory is initialised based on the memory of its parent and the current state. Then, if we see d later on in the word, we can decide how to transition based on the label of d, and the label of the nodes in its memory. For this definition to work, we require that the memory of a new node is a suffix of the memory of the parent, and bound the maximum size of the memory by k.

3 Closure Properties of Unbounded NDCMA

These automata both admit a decidable non-emptiness problem. Our technique to prove this extends that of [5], which reduces non-emptiness of NDCMA to the coverability problem for *well-structured transition systems* (WSTS) [8]. These are sets, equipped additionally with a transition relation $\to$ and well-quasi order $\leq$ satisfying an *upward-compatibility* property.

If the order $\leq$ is decidable, and our WSTS additionally has an *effective pred-basis*, then the coverability problem is decidable. The original proof makes use of the tree homomorphism ordering, which is a well-quasi ordering over bounded depth trees. This is not a well-quasi ordering for unbounded trees, so it cannot be used analogously for our unbounded NDCMA. Our two automata models require different approaches. For memoryless NDCMA, we utilise the homeomorphic embedding ordering, which is a well-quasi order on all trees by Kruskal's tree theorem [7]. To handle k-memory NDCMA, we essentially extract a k-depth tree from the memory of the nodes – allowing us to use the tree homomorphism ordering.

Theorem 3.1 *The problems of emptiness of memoryless NDCMA and emptiness of k-memory NDCMA are reducible to the covering problem for a well-structured transition system with an effective pred-basis and decidable $\leq$, hence are decidable.*

Furthermore, we look at closure properties of these automata. The positive results about memoryless NDCMA can be obtained using the typical power construction. It follows that they are not closed under complement from the fact that CMA are not closed under complement.

Lemma 3.2 *Memoryless NDCMA are closed under union and intersection, but not closed under intersection.*

For k-memory NDCMA it is not as straightforward. These NDCMA are closed under union, using a non-deterministic construction, however they are not closed under intersection as we can in fact obtain the following undecidability result using a reduction to Turing machines.

Theorem 3.3 *Given two (deterministic) k-memory NDCMA $\mathcal{A}$, $\mathcal{B}$, the problem of deciding if $\mathcal{L}(\mathcal{A}) \cap \mathcal{L}(\mathcal{B}) = \varnothing$ is undecidable.*

The theorem has the following consequences:

Corollary 3.4 (i) *k-memory NDCMA are not closed under intersection.*

(ii) *Given two deterministic k-memory NDCMA $\mathcal{A}$, $\mathcal{B}$, it is undecidable whether $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{B})$.*

(iii) *Given two k-memory NDCMA $\mathcal{A}$, $\mathcal{B}$, it is undecidable whether $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{B})$.*

It is still open whether it is decidable if $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{B})$ given two *deterministic* k-memory NDCMA $\mathcal{A}$, $\mathcal{B}$. We summarise the closure results in the following table:

	memoryless NDCMA	k-memory NDCMA	det. k-memory NDCMA	det. NDCMA [5]
L^c	✗	✗	✓	✓
$L_1 \cap L_2$	✓	✗	✗	✓
$L_1 \cup L_2$	✓	✓	✗	✓
$L_1 \subseteq L_2$	✗	✗	✗	✓
$L_1 = L_2$	✗	✗	?	✓

References

[1] Björklund, H., Bojańczyk, M.: Shuffle Expressions and Words with Nested Data. Mathematical Foundations of Computer Science **4708**, 750–761 (2007), https://doi.org/10.1007/978-3-540-74456-6_66

[2] Björklund, H., Schwentick, T.: On Notions of Regularity for Data Languages. Theoretical Computer Science **411**(4), 702–715 (2007), https://doi.org/10.1016/j.tcs.2009.10.009

[3] Bunting, B. and Murawski, A. S.: Contextual Equivalence for State and Control via Nested Data. Proceedings of the 39th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '24. (19) (2024), https://doi.org/10.1145/3661814.3662109

[4] Cotton-Barratt, C. and Hopkins, D. and Murawski, A. S. and Ong, C.-H. L.: Fragments of ML decidable by nested data class memory automata. International Conference on Foundations of Software Science and Computation Structures. 249–263 (2015), https://doi.org/10.1007/978-3-662-46678-0_16

[5] Cotton-Barratt, C., Murawski, A. S., Ong, C.-H. L.: Weak and Nested Class Memory Automata. Language and Automata Theory and Applications **8977**(14), 188–199 (2015), https://doi.org/10.1007/978-3-319-15579-1_14

[6] Decker, N., Habermehl, P., Leucker, M., Thoma, D.: Ordered Navigation on Multi-attributed Data Words. CONCUR 2014. Lecture Notes in Computer Science, **8704**. 497–511 (2014), https://doi.org/10.1007/978-3-662-44584-6_34

[7] Kruskal, J. B.: Well-Quasi-Ordering, The Tree Theorem, and Vazsonyi's Conjecture. Transactions of the American Mathematical Society, **95**(2), 210–225 (1960), https://doi.org/10.2307/1993287

[8] Schmitz, S. and Schnoebelen, P.: Algorithmic Aspects of WQO Theory. Lecture Notes (2012), https://cel.hal.science/cel-00727025v2/file/lecturenotes.pdf