An automaton over data words that captures EMSO logic

Benedikt Bollig

LSV, ENS Cachan, CNRS, France
Email: bollig@lsv.ens-cachan.fr

A recent research stream considers data words and data trees, which are structures over an infinite alphabet. The alphabet is the cartesian product of a finite supply of labels and an infinite supply of (multiple) data values. Data words and trees are motivated from models in XML database theory, but they actually enjoy a wide range of applications and serve, e.g., as behavioral models of timed systems and dynamic systems with an unbounded number of processes. Of particular interest have been connections between automata and logic that come with mechanisms for testing data equality. A first automata model was introduced by Kaminski and Francez [5]. Their register automata are equipped with finitely many registers to store current data values and compare them with values that are read later. In search of the "right" automata class, many more models followed, among them two-way devices and automata with pebbles [6]. In [3], Bojańczyk et al. introduce data automata, which have a decidable emptiness problem and are used to check satisfiability of first-order formulas with two variables. Data automata were later shown to be equivalent to class memory automata [1], which are conceptually slightly different. They read a data word from left to right, but can access certain states from the past. When data values model process identifiers, this can indeed be interpreted as reading the current local state of a process. In [2], Bojańczyk and Lasota introduce class automata, a generalization of data automata that captures XPath. Though class automata have an undecidable emptiness problem, they prove useful for partial satisfiability checking as well as exploring the limits of XPath wrt. expressive power.

In this work, we consider an unrestricted existential monadic second-order (EMSO) logic over words with multiple data values. In particular, an unlimited number of variables is permitted. Our logic is strictly more expressive than the two-variable logic from [3] and at least incomparable with XPath. Our main result states that every EMSO formula can be compiled into a new automata model, which we call class register automaton. This model combines, in a natural manner, register automata [5] and class memory automata [1], and it uses the element of guessing a data value [4]. A class register automaton is a one-way device (unlike the pebble automata from [6]). Like a class memory automaton, it can access certain configurations in the past. However, we extend the notion of a configuration, which is no longer a simple state but composed of a state and a register assignment. This has meaningful interpretations such as "read current state of a process" or "send process identity from one to another process". Our automata have various potential applications, though their emptiness problem is undecidable. First, they may provide a framework for synthesizing system design models from logical specifications. Second, they are useful in the realm of satisfiability checking, as they come with the notion of abstract configurations that allows for the definition of an equivalent finitely branching infinite transition system. Third, our automaton unifies existing concepts and provides insight into their combined expressive power.

1. H. Björklund and Th. Schwentick. On notions of regularity for data languages. *Theoretical Computer Science*, 411(4-5):702–715, 2010.
2. M. Bojańczyk and S. Lasota. An extension of data automata that captures XPath. In *Proceedings of LICS'10*, pages 243–252. IEEE Computer Society, 2010.
3. M. Bojańczyk, A. Muscholl, Th. Schwentick, L. Segoufin, and C. David. Two-variable logic on words with data. In *Proceedings of LICS'06*, pages 7–16. IEEE Computer Society, 2006.
4. D. Figueira. Forward-XPath and extended register automata on data-trees. In *Proceedings of the 13th International Conference on Database Theory (ICDT'10)*, pages 230–240. ACM Press, 2010.
5. M. Kaminski and N. Francez. Finite-memory automata. *Theoretical Computer Science*, 134(2):329–363, 1994.
6. F. Neven, Th. Schwentick, and V. Vianu. Finite state machines for strings over infinite alphabets. *ACM Transactions on Computational Logic*, 5(3):403–435, 2004.