Visibly Pushdown Automata vs. Bottom-up Tree Automata for XML

Alex Thomo

Both Visibly Pushdown Automata (VPAs) and bottom-Up Tree Automata (TA) have been used to represent XML specifications. In this short note, I would like to point out some facts about VPAs and TAs as they pertain to XML.

1. Representing extended DTDs (EDTDs [8]), using VPAs or TAs is theoretically the same. Both VPAs and TAs fully capture EDTDs (see [2, 6]), and furthermore the complexity of important decision problems (such as inclusion) for both VPAs and TAs is the same.

2. A reason for possibly preferring VPAs over TAs for XML is that VPAs are often more natural and exponentially more succinct than TAs when it comes to "semi-formally" specify documents using pattern-based conditions on the global linear order of XML. Fleshing out the example of Alur in PODS 07 [1], to express that we want properly nested XML documents which contain elements $a_1, \ldots, a_n$ (in this order) we can specify the word language

   $$L(\Sigma^* \langle a_1 \rangle \Sigma^* \langle /a_1 \rangle \Sigma^* \ldots \Sigma^* \langle a_n \rangle \Sigma^* \langle /a_n \rangle \Sigma^*) \cap PN,$$

   where PN is the language of all properly nested words on Σ. This specification compiles into a deterministic VPA of linear size, but standard deterministic bottom-up tree automata for this specification must be of size exponential in n. Since for deciding inclusion and equivalence of specifications, we need deterministic automata, these problems can be (often) decided by using VPAs in exponentially better time than by using TAs. We made use of this property in [10] to come up with a method for integrating XML data-sources (represented by VPAs).

   It seems that approaches as the above for specifying wanted documents are not uncommon or contrived examples in favor of VPAs. Rather, they seem to be practical and popular among users accustomed with regular expressions, (cf. Friedl, Mastering Regular Expressions. O'Reilly, 2006. and .NET platform of Microsoft).

3. VPAs allow for a natural extension of word deletions and insertions in the spirit of Lila Kari's thesis for regular languages [5] (see also [3, 4]). This is very useful in modeling XML schema evolution and performing data exchange (cf. [9, 11, 12]).

4. TAs, on the other hand, are quite nice when it comes to partially rewriting a regular tree language given a view language [7]. The bottom line is that, depending on the case, sometimes VPAs come in handy and some other times, TAs do a better job. Both of them are important tools in the repertoire of tools for reasoning on XML and other forms of tree-structured data.

References

[1] R. Alur. Marrying words and trees. In *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 233–242. ACM, 2007.

[2] H. Comon, M. Dauchet, R. Gilleron, C. Löding, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. 2007.

[3] G. Grahne and A. Thomo. Algebraic rewritings for optimizing regular path queries. In *International Conference on Database Theory*, pages 301–315. Springer, 2001.

[4] G. Grahne and A. Thomo. New rewritings and optimizations for regular path queries. In *International Conference on Database Theory*, pages 242–258. Springer, 2003.

[5] L. Kari. *On insertion and deletion in formal languages.* University of Turku Department of mathematics, 1991.

[6] V. Kumar, P. Madhusudan, and M. Viswanathan. Visibly pushdown automata for streaming xml. In *Proceedings of the 16th international conference on World Wide Web*, pages 1053–1062. ACM, 2007.

[7] L. V. Lakshmanan and A. Thomo. View-based tree-language rewritings for xml. In *International Symposium on Foundations of Information and Knowledge Systems*, pages 270–289. Springer, 2014.

[8] L. Segoufin and V. Vianu. Validating streaming xml documents. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 53–64. ACM, 2002.

[9] M. Shoaran and A. Thomo. Evolving schemas for streaming xml. *Theoretical Computer Science*, 412(35):4545–4557, 2011.

[10] A. Thomo and S. Venkatesh. Rewriting of visibly pushdown languages for xml data integration. *Theoretical Computer Science*, 412(39):5285–5297, 2011.

[11] A. Thomo, S. Venkatesh, and Y. Y. Ye. Visibly pushdown transducers for approximate validation of streaming xml. In *International Symposium on Foundations of Information and Knowledge Systems*, pages 219–238. Springer, 2008.

[12] Y. Y. F. Ye. Visibly pushdown transducers for approximate validation of streaming xml, 2008.