Büchi Store: An Open Repository of Büchi Automata*

Yih-Kuen Tsay, Ming-Hsien Tsai, Jinn-Shu Chang, and Yi-Wen Chang

Department of Information Management, National Taiwan University, Taiwan

Abstract. We introduce the Büchi Store, an open repository of Büchi automata for model-checking practice, research, and education. The repository contains Büchi automata and their complements for common specification patterns and numerous temporal formulae. These automata are made as small as possible by various construction techniques, in view that smaller automata are easier to understand and often help in speeding up the model-checking process. The repository is open, allowing the user to add new automata or smaller ones that are equivalent to some existing automaton. Such a collection of Büchi automata is also useful as a benchmark for evaluating complementation or translation algorithms and as examples for studying Büchi automata and temporal logic.

1 Introduction

Büchi automata [1] are finite automata operating on infinite words. They play a fundamental role in the automata-theoretic approach to linear-time model checking [20]. In the approach, model checking boils down to testing the emptiness of an intersection automaton $A \cap B_{\neg\varphi}$, where A is a Büchi automaton modeling the system and $B_{\neg\varphi}$ is another Büchi automaton representing all behaviors not permitted by a temporal specification formula φ. In general, for a given system, the smaller $B_{\neg\varphi}$ is, the faster the model-checking process may be completed.

To apply the automata-theoretic approach, an algorithm for translating a temporal formula into an equivalent Büchi automaton is essential. There has been a long line of research on such translation algorithms, aiming to produce smaller automata. According to our experiments, none of the proposed algorithms outperforms the others for every temporal formula tested. The table below shows a comparison of some of the algorithms for three selected cases.

Formula	LTL2AUT[4]		Couvreur[3]		LTL2BA[5]		LTL2Buchi[6]		Spin[7]	
	state	tran.	state	tran.	state	tran.	state	tran.	state	tran.
$\neg p \,\mathcal{W}\, q$	4	16	3	12	3	12	3	12	4	16
$\Box(p \rightarrow \Diamond q)$	4	30	3	20	6	41	3	20	4	28
$\Diamond\Box p \vee \Diamond\Box q$	8	38	5	28	5	28	5	28	3	12

* This work was supported in part by National Science Council, Taiwan, under the grant NSC97-2221-E-002-074-MY3.

P.A. Abdulla and K.R.M. Leino (Eds.): TACAS 2011, LNCS 6605, pp. 262–266, 2011.

Given that smaller automata usually expedite the model-checking process, it is certainly desirable that one is always guaranteed to get the smallest possible automaton for (the negation of) a specification formula. One way to provide the guarantee is to try all algorithms or even manual construction and take the best result. This simple-minded technique turns out to be feasible, as most specifications use formulae of the same patterns and the tedious work of trying all alternatives needs only be done once for a particular pattern instance.

To give the specification as a temporal formula sometimes may not be practical, if not impossible (using quantification over propositions). When the specification is given directly as an automaton, taking the complement of the specification automaton becomes necessary. Consequently, in parallel with the research on translation algorithms, there has also been substantial research on algorithms for Büchi complementation. The aim again is to produce smaller automata.

Several Büchi complementation algorithms have been proposed that achieve the lower bound of $2^{\Omega(n \log n)}$ [11]. However, the performances of these "optimal" algorithms differ from case to case, sometimes quite dramatically. The table below shows a comparison of some of the complementation algorithms for four selected Büchi automata (identified by equivalent temporal formulae). In the literature, evaluations of these algorithms usually stop at a theoretical-analysis level, partly due to the lack of or inaccessibility to actual implementations. This may be remedied if a suitable set of benchmark cases becomes available and subsequent evaluations are conducted using the same benchmark.

Formula	Safra[13]		Piterman[12]		Rank-Based[9,14]		Slice-Based[8]	
	state	tran.	state	tran.	state	tran.	state	tran.
$\Box(p \rightarrow \Diamond(q \wedge \Diamond r))$	76	662	90	777	96	917	219	2836
$\Diamond\Box(\Diamond p \rightarrow q)$	35	188	13	62	13	72	24	119
$\Box(p \rightarrow p\,\mathcal{U}\,(q\,\mathcal{U}\,r))$	17	192	8	76	7	54	7	49
$p\,\mathcal{U}\,q \vee p\,\mathcal{U}\,r$	5	34	5	34	8	23	3	12

The Büchi Store was thus motivated and implemented as a website, accessible at `http://buchi.im.ntu.edu.tw`. One advantage for the Store to be on the Web is that the user always gets the most recent collection of automata. Another advantage is that it is easily made open for the user to contribute better (smaller) automata. The initial collection contains over six hundred Büchi automata. In the following sections we describe its implementation and main features, suggest three use cases, and then conclude by highlighting directions for improvement.

2 Implementation and Main Features

The basic client-server interactions in accessing the Büchi Store are realized by customizing the CodeIgniter [2], which is an open-source Web application framework. To perform automata and temporal formulae-related operations, such as equivalence checking and formula to automaton translation, the Store relies on the GOAL tool [19] and its recent extensions. One particularly important (and highly nontrivial) task is the classification of temporal formulae that identify

the Büchi automata in the Store into the Temporal Hierarchy of Manna and Pnueli [10]. To carry out the task automatically, we implemented the classification algorithm described in the same paper, which is based on characterization of a Streett automaton equivalent to the temporal formula being classified.

The main features of the current Büchi Store include:

- **Search:** Every automaton in the Store is identified by a temporal formula (in a variant of QPTL [15,16], which is expressively equivalent to Büchi automata). The user may find the automata that accept a particular language by posing a query with an equivalent temporal formula. Propositions are automatically renamed to increase matches (semantic matching between whole formulae is not attempted due to its high cost). This is like asking for a translation from the temporal formula into an equivalent Büchi automaton. A big difference is that the answer automata, if any, are the best among the results obtained from a large number of translation algorithms, enhanced with various optimization techniques such as simplification by simulation [17] or even manually optimized (and machine-checked for correctness).
- **Browse:** The user may browse the entire collection of Büchi automata by having the collection sorted according to temporal formula length, number of states, class in the Temporal Hierarchy, or class in the Spec Patterns [18]. While classification in the Temporal Hierarchy has been automated, the classification for the last sorting option has not. Rather, the Store relies on the user to provide suggestions, based on which a final classification could be made. This may be useful for educational purposes.
- **Upload:** The user may upload a Büchi automaton for a particular temporal formula. The automaton is checked for correctness, i.e., if it is indeed equivalent to the accompanying temporal formula. If it is correct and smaller than the automata for the formula in the Store, the repository is updated accordingly, keeping only the three smallest automata.

3 Use Cases

We describe three cases that we expect to represent typical usages of the Store.

- **Linear-time model checking:** The user may shop in the Store for the automata that are equivalent (with probable propositions renaming) to the negations of the temporal formulae which he wants to verify. The automata may be downloaded in the PROMELA format, for model checking using SPIN.
- **Benchmark cases for evaluating complementation algorithms:** Every Büchi automaton in the initial collection has a complement, which is reasonably well optimized. A subset of the collection could serve as a set of benchmark cases for evaluating Büchi complementation algorithms. This use case can certainly be adapted for evaluating translation algorithms.
- **Classification of temporal formulae:** The look of a temporal formula may not tell immediately to which class it belongs in the Temporal Hierarchy. It should be educational to practice on the cases that do not involve the

complication of going through Streett automata. For example, $\Box(p \rightarrow \Diamond q)$ is a recurrence formula because it is equivalent to $\Box\Diamond(\neg p \; \mathcal{B} \, q)$ (where $\mathcal{B}$ means "back-to", the past version of wait-for or weak until).

Concluding Remarks. To further improve the Store, first of all, we as the developers will continue to expand the collection, besides hoping for the user to do the same. Explanatory descriptions other than temporal formulae should be helpful additions for searching and understanding. Automatic classification of temporal formulae into the various specification patterns should also be useful.

References

1. Büchi, J.R.: On a decision method in restricted second-order arithmetic. In: Int'l Congress on Logic, Methodology and Philosophy of Science, pp. 1–11 (1962)
2. CodeIgniter, `http://codeigniter.com/`
3. Couvreur, J.M.: On-the-fly verification of linear temporal logic. In: Woodcock, J.C.P., Davies, J. (eds.) FM 1999. LNCS, vol. 1708, pp. 253–271. Springer, Heidelberg (1999)
4. Daniele, M., Giunchiglia, F., Vardi, M.Y.: Improved automata generation for linear temporal logic. In: Halbwachs, N., Peled, D.A. (eds.) CAV 1999. LNCS, vol. 1633, pp. 249–260. Springer, Heidelberg (1999)
5. Gastin, P., Oddoux, D.: Fast LTL to Büchi automata translation. In: Berry, G., Comon, H., Finkel, A. (eds.) CAV 2001. LNCS, vol. 2102, pp. 53–65. Springer, Heidelberg (2001)
6. Giannakopoulou, D., Lerda, F.: From states to transitions: Improving translation of LTL formulae to Büchi automata. In: Peled, D.A., Vardi, M.Y. (eds.) FORTE 2002. LNCS, vol. 2529, pp. 308–326. Springer, Heidelberg (2002)
7. Holzmann, G.J.: The SPIN Model Checker: Primer and Reference Manual. Addison-Wesley, Reading (2003)
8. Kähler, D., Wilke, T.: Complementation, disambiguation, and determinization of Büchi automata unified. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part I. LNCS, vol. 5125, pp. 724–735. Springer, Heidelberg (2008)
9. Kupferman, O., Vardi, M.Y.: Weak alternating automata are not that weak. ACM Transactions on Computational Logic 2(3), 408–429 (2001)
10. Manna, Z., Pnueli, A.: A hierarchy of temporal properties. In: PODC, pp. 377–408. ACM, New York (1990)
11. Michel, M.: Complementation is more difficult with automata on infinite words. In: CNET, Paris (1988)
12. Piterman, N.: From nondeterministic Büchi and Streett automata to deterministic parity automata. In: LICS, pp. 255–264. IEEE, Los Alamitos (2006)
13. Safra, S.: On the complexity of ω-automta. In: FOCS, pp. 319–327. IEEE, Los Alamitos (1988)
14. Schewe, S.: Büchi complementation made tight. In: STACS, pp. 661–672 (2009)
15. Sistla, A.P.: Theoretical Issues in the Design and Verification of Distributed Systems. PhD thesis, Harvard (1983)

16. Sistla, A.P., Vardi, M.Y., Wolper, P.: The complementation problem for Büchi automata with applications to temporal logic. TCS 49, 217–237 (1987)
17. Somenzi, F., Bloem, R.: Efficient Büchi automata from LTL formulae. In: Emerson, E.A., Sistla, A.P. (eds.) CAV 2000. LNCS, vol. 1855, pp. 248–263. Springer, Heidelberg (2000)
18. The Spec Patterns repository, `http://patterns.projects.cis.ksu.edu/`
19. Tsay, Y.-K., Chen, Y.-F., Tsai, M.-H., Chan, W.-C., Luo, C.-J.: GOAL extended: Towards a research tool for omega automata and temporal logic. In: Ramakrishnan, C.R., Rehof, J. (eds.) TACAS 2008. LNCS, vol. 4963, pp. 346–350. Springer, Heidelberg (2008)
20. Vardi, M.Y., Wolper, P.: An automata-theoretic approach to automatic program verification. In: LICS, pp. 332–344. IEEE, Los Alamitos (1986)