# GOAL for Games, Omega-Automata, and Logics

Ming-Hsien Tsai, Yih-Kuen Tsay, and Yu-Shiang Hwang

National Taiwan University

**Abstract.** This paper introduces the second generation of GOAL, which is a graphical interactive tool for games, $\omega$-automata, and logics. It is a complete redesign with an extensible architecture, many enhancements to existing functions, and new features. The extensible architecture allows easy integration of third-party plugins. The enhancements provide more automata conversion, complementation, simplification, and testing algorithms, translation of full QPTL formulae, and better automata navigation with more layout algorithms and utility functions. The new features include game solving, manipulation of two-way alternating automata, translation of ACTL formulae and $\omega$-regular expressions, test of language star-freeness, classification of $\omega$-regular languages into the temporal hierarchy of Manna and Pnueli, and a script interpreter.

## 1 Introduction

GOAL (http://goal.im.ntu.edu.tw) is a graphical interactive tool for defining and manipulating games, $\omega$-automata, and logic formulae. The first generation of GOAL (or simply the 1st gen) was formally introduced in [52] and later extended in [51]. It is implemented in Java and built upon the classic finite automata and graphical modules of JFLAP [40]. The main features of the 1st gen include (1) drawing games and $\omega$-automata, (2) translating quantified propositional temporal logic (QPTL, an extension of PTL and also LTL) formulae in prenex normal form to equivalent Büchi automata, (3) complementing Büchi automata, (4) testing containment and equivalence between two Büchi automata, and (5) exporting Büchi automata as Promela code. It also provides a command-line mode and utility functions for collecting statistic data and generating random automata and temporal formulae.

A typical usage of GOAL is for educational purposes, helping the user get a better understanding of Büchi automata and their relation to temporal logic. For example, the user may follow a translation algorithm step-by-step in the tool to see how a QPTL formula is translated to an equivalent Büchi automaton. GOAL may also be used for supplementing automata-theoretic model checkers such as SPIN [20]. For example, the user may construct a smaller specification automaton that is checked to be correct in that it is equivalent to a larger reference Büchi automaton or a QPTL formula. Moreover, GOAL has been used for supporting research and tools development [1,3,8,50,53].

This paper introduces the second generation of GOAL, which is a complete redesign with an extensible architecture, many enhancements to existing functions, and new features. The extensible architecture allows easy integration of

N. Sharygina and H. Veith (Eds.): CAV 2013, LNCS 8044, pp. 883–889, 2013.

**Table 1.** Major algorithms in GOAL. An * indicates that the implementation of the algorithm has already been reported in [52,51].

| **Translation of QPTL Formulae** |
|---|
| Tableau*, Incremental Tableau* [26], Temporal Tester* [27], GPVW* [16], GPVW+* [16], LTL2AUT* [16], LTL2AUT+* [49], LTL2BA* [14], PLTL2BA* [15], MoDeLLa [44,49], Couvreur's [10,49], LTL2BUCHI [17,49], KP02 [28], CCJ09 [9] |
| **Complementation of Büchi Automata** |
| Safra* [41], WAPA* [48], WAA* [30], Safra-Piterman* [39], Kurshan's [31], Ramsey-based [5,45], Muller-Schupp [37,2], Rank-based [30,43], Slice-based [25] |
| **Simplification of Automata** |
| Direct and Reverse Simulation* [47], Pruning Fair Sets* [47], Delayed Simulation [12], Fair Simulation [19], Parity Simplification [6] |
| **Parity Game Solving** |
| Recursive [32], McNaughton-Zielonka [36,54], Dominion Decomposition [24], Small Progress Measure [23], Big Steps [42], Global Optimization [13] |

third-party plugins. The enhancements provide many more operations and tests on automata, more translation algorithms for QPTL (one of which supports full QPTL formulae that are not required to be in prenex normal form), and better automata navigation with more layout algorithms and utility functions. The new features provide game solving and conversion, support of two-way alternating automata and $\forall$ computation tree logic (ACTL), and more classification information about $\omega$-regular languages such as star-freeness. Table 1 summarizes the major algorithms implemented in GOAL.

## 2 Enhanced Functions

We describe in this section how the functions of the 1st gen have been extended.

- **Translation of QPTL Formulae:** We have implemented five more translation algorithms for QPTL, namely KP02 [28], CCJ09 [9], MoDeLLa [44], Couvreur's [10], and LTL2BUCHI [17]. KP02 is the only one that supports full QPTL by an inductive construction, while CCJ09 can translate four specific patterns of QPTL formulae to minimal Büchi automata. We have also extended MoDeLLa, Couvreur's, and LTL2BUCHI to allow past operators.
- **Conversion between Automata:** Compared to the 1st gen, GOAL now supports many more conversions between automata, especially the conversion from nondeterministic Büchi automata, if possible, to deterministic automata with Büchi [33,4] or co-Büchi [4] acceptance conditions. Moreover, GOAL can automatically find and apply a sequence of chained conversions to convert an automaton to another type specified by the user.
- **Operation and Simplification on Automata:** GOAL has five more complementation constructions for Büchi automata, namely Kurshan's [31], Ramsey-based [5,45], Muller-Schupp [37,2], Rank-based [30,43], and Slice-based [25]. With chained conversions, several Boolean operations previously implemented only for Büchi automata can be applied to any automata

convertible to equivalent Büchi automata. We have also implemented simplification of Büchi automata with delayed simulation and fair simulation relations based on solving simulation games [12,19].
- **Test on Automata:** GOAL has five more containment testing algorithms, which are performed incrementally with a double depth-first search. Four of them are based on determinization-based complementation constructions [41,37,2,39], while one is based on slice-based constructions [25]. Such incremental algorithms usually can find counterexamples earlier. Same as in automata conversion, several tests implemented for Büchi automata can also be applied to other types of automata.
- **Automata Navigation:** Navigation of a large automaton can be cumbersome. To make it easier, GOAL has eleven new layout algorithms. After an automatic layout, the user can manually arrange the states with the help of guidelines, gridlines, and snapping to grids to make the layout even better. GOAL also allows the user to focus on a state and its neighbors such that the user can easily traverse a particular path of an automaton.

## 3 New Features

We now detail the new features of GOAL.

- **Extensibility:** GOAL can be easily extended with the help of Java Plugin Framework [22]. The user may add a new menu item, command, or algorithm with a plugin simply by extending the classes or interfaces of GOAL and then writing a configuration file for the plugin. During runtime, GOAL will read the configuration file and enable the third-party plugin. In fact, GOAL itself is composed of three plugins, namely CORE, UI, and CMD, where CORE provides basic data structures and implementations of algorithms, UI provides a graphical interface, and CMD provides a command-line interface.
- **Game Solving and Conversion:** We have implemented eight game solving algorithms of which one is for reachability games, one for Büchi games, and six for parity games. Winning regions and strategies in a solved game are highlighted and can be exported with the game to a file. Several conversions between games, including the conversion from a Muller game to a parity game [35], are implemented. To help experiments with games, the generation of random games is provided as well. GOAL can also take the product of a game arena (that describes the allowed interactions between a module and its environment) and a specification automaton (resulting in a game), and hence may be used to experiment with the synthesis process in a game-based approach to the synthesis of reactive modules [46].
- **New Types of Automata and Logics:** Previously in the 1st gen, two-way alternating automata were only used internally in PLTL2BA. Now the user can draw a two-way alternating automaton and convert it to an equivalent transition-based generalized Büchi automaton if it is very weak [15]. For new logics, GOAL supports the translation of an ACTL formula [38] to a

maximal model (represented as an automaton) of the formula [18,29]. Such model can be used in model checking or synthesis. The translation of $\omega$-regular expressions is also available in GOAL.

- **Classification of $\omega$-Regular Languages:** We have implemented an algorithm for testing whether an $\omega$-regular language or Büchi automaton is star-free [11]. If an $\omega$-regular language is star-free, it can be specified by a formula in PTL, which is less expressive than QPTL. We have also implemented the classification of $\omega$-regular languages into the temporal hierarchy of Manna and Pnueli [34]. Such classification can be used not only for educational purposes but also for helping model checking [7].
- **Script Interpreter:** GOAL provides an interpreter that can execute scripts in a customized language. In the 1st gen, GOAL commands can only be executed in shell scripts, which create a GOAL process per command. Now a batch of GOAL commands can be written as a script and executed by a single GOAL process to achieve better performance.

## 4 Remarks

With these enhancements and new features, GOAL now lives up to an alternative source of its acronym "**G**ames, **O**mega-**A**utomata, and **L**ogics". Even classic finite automata and regular expressions are also supported by GOAL. With the new architecture, we expect that GOAL will be extended by third-party plugins. We will also continue to extend GOAL with more algorithms, e.g., translation algorithms for Property Specification Language (PSL) [21] and game solving algorithms that produce better winning strategies.

**Acknowledgements.** This work was partially supported by the National Science Council, Taiwan, under grants NSC97-2221-E-002-074-MY3, NSC100-2221-E-002-116, and NSC101-2221-E-002-093. We thank Jinn-Shu Chang and Yi-Wen Chang for helping with the implementations of some algorithms.

## References

1. Abdulla, P.A., Chen, Y.-F., Clemente, L., Holík, L., Hong, C.-D., Mayr, R., Vojnar, T.: Simulation subsumption in Ramsey-based Büchi automata universality and inclusion testing. In: Touili, T., Cook, B., Jackson, P. (eds.) CAV 2010. LNCS, vol. 6174, pp. 132–147. Springer, Heidelberg (2010)
2. Althoff, C.S., Thomas, W., Wallmeier, N.: Observations on determinization of Büchi automata. TCS 363(2), 224–233 (2006)
3. Alur, R., Weiss, G.: RTComposer: a framework for real-time components with scheduling interfaces. In: EMSOFT, pp. 159–168. ACM (2008)
4. Boker, U., Kupferman, O.: Co-ing Büchi made tight and useful. In: LICS, pp. 245–254. IEEE Computer Society (2009)
5. Büchi, J.R.: On a decision method in restricted second order arithmetic. In: CLMPS, pp. 1–12. Stanford University Press (1962)

6. Carton, O., Maceiras, R.: Computing the Rabin index of a parity automaton. Informatique Théorique et Applications 33(6), 495–506 (1999)
7. Černá, I., Pelánek, R.: Relating hierarchy of temporal properties to model checking. In: Rovan, B., Vojtáš, P. (eds.) MFCS 2003. LNCS, vol. 2747, pp. 318–327. Springer, Heidelberg (2003)
8. Chatterjee, K., Henzinger, T.A., Jobstmann, B., Singh, R.: QUASY: Quantitative synthesis tool. In: Abdulla, P.A., Leino, K.R.M. (eds.) TACAS 2011. LNCS, vol. 6605, pp. 267–271. Springer, Heidelberg (2011)
9. Cichoń, J., Czubak, A., Jasiński, A.: Minimal Büchi automata for certain classes of LTL formulas. In: DepCos-RELCOMEX, pp. 17–24. IEEE (2009)
10. Couvreur, J.-M.: On-the-fly verification of linear temporal logic. In: Wing, J.M., Woodcock, J. (eds.) FM 1999. LNCS, vol. 1708, pp. 253–271. Springer, Heidelberg (1999)
11. Diekert, V., Gastin, P.: First-order definable languages. In: Logic and Automata. Texts in Logic and Games, vol. 2, pp. 261–306. Amsterdam Univ. Press (2008)
12. Etessami, K., Wilke, T., Schuller, R.A.: Fair simulation relations, parity games, and state space reduction for Büchi automata. SICOMP 34(5), 1159–1175 (2005)
13. Friedmann, O., Lange, M.: Solving parity games in practice. In: Liu, Z., Ravn, A.P. (eds.) ATVA 2009. LNCS, vol. 5799, pp. 182–196. Springer, Heidelberg (2009)
14. 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)
15. Gastin, P., Oddoux, D.: LTL with past and two-way very-weak alternating automata. In: Rovan, B., Vojtáš, P. (eds.) MFCS 2003. LNCS, vol. 2747, pp. 439–448. Springer, Heidelberg (2003)
16. Gerth, R., Peled, D., Vardi, M.Y., Wolper, P.: Simple on-the-fly automatic verification of linear temporal logic. In: PSTV, pp. 3–18. Chapman & Hall (1995)
17. 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)
18. Grumberg, O., Long, D.E.: Model checking and modular verification. TOPLAS 16(3), 843–871 (1994)
19. Gurumurthy, S., Bloem, R., Somenzi, F.: Fair simulation minimization. In: Brinksma, E., Larsen, K.G. (eds.) CAV 2002. LNCS, vol. 2404, pp. 610–624. Springer, Heidelberg (2002)
20. Holzmann, G.J.: The SPIN Model Checker: Primer and Reference Manual. Addison-Wesley (September 2003)
21. IEEE standard for property specification language (PSL). IEEE Std 1850-2010 (Revision of IEEE Std 1850-2005), 1–182 (2010)
22. Java Plugin Framework, `http://jpf.sourceforge.net`
23. Jurdziński, M.: Small progress measures for solving parity games. In: Reichel, H., Tison, S. (eds.) STACS 2000. LNCS, vol. 1770, pp. 290–301. Springer, Heidelberg (2000)
24. Jurdziński, M., Paterson, M., Zwick, U.: A deterministic subexponential algorithm for solving parity games. SICOMP 38(4), 1519–1532 (2008)
25. 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)

26. Kesten, Y., Manna, Z., McGuire, H., Pnueli, A.: A decision algorithm for full propositional temporal logic. In: Courcoubetis, C. (ed.) CAV 1993. LNCS, vol. 697, pp. 97–109. Springer, Heidelberg (1993)
27. Kesten, Y., Pnueli, A.: Verification by augmented finitary abstraction. Information and Computation 163(1), 203–243 (2000)
28. Kesten, Y., Pnueli, A.: Complete proof system for QPTL. Journal of Logic and Computation 12(5), 701–745 (2002)
29. Klotz, T., Seßler, N., Straube, B., Fordran, E.: Compositional verification of material handling systems. In: ETFA, pp. 1–8. IEEE (2012)
30. Kupferman, O., Vardi, M.Y.: Weak alternating automata are not that weak. TOCL 2(3), 408–429 (2001)
31. Kurshan, R.P.: Complementing deterministic Büchi automata in polynomial time. JCSS 35(1), 59–71 (1987)
32. Küsters, R.: Memoryless determinacy of parity games. In: Grädel, E., Thomas, W., Wilke, T. (eds.) Automata, Logics, and Infinite Games. LNCS, vol. 2500, pp. 95–106. Springer, Heidelberg (2002)
33. Landweber, L.H.: Decision problems for omega-automata. Mathematical Systems Theory 3(4), 376–384 (1969)
34. Manna, Z., Pnueli, A.: A hierarchy of temporal properties. In: PODC, pp. 377–410 (1990)
35. Mazala, R.: Infinite games. In: Grädel, E., Thomas, W., Wilke, T. (eds.) Automata, Logics, and Infinite Games. LNCS, vol. 2500, pp. 23–38. Springer, Heidelberg (2002)
36. McNaughton, R.: Infinite games played on finite graphs. Annals of Pure and Applied Logic 65(2), 149–184 (1993)
37. Muller, D.E., Schupp, P.E.: Simulating alternating tree automata by nondeterministic automata: New results and new proofs of the theorems of Rabin, McNaughton and Safra. TCS 141(1&2), 69–107 (1995)
38. Peng, H., Mokhtari, Y., Tahar, S.: Environment synthesis for compositional model checking. In: ICCD, pp. 70–75. IEEE Computer Society (2002)
39. Piterman, N.: From nondeterministic Büchi and Streett automata to deterministic parity automata. LMCS 3(3), paper 5 (2007)
40. Rodger, S., Finley, T.: JFLAP, `http://www.jflap.org/`
41. Safra, S.: On the complexity of $\omega$-automata. In: FOCS, pp. 319–327. IEEE (1988)
42. Schewe, S.: Solving parity games in big steps. In: Arvind, V., Prasad, S. (eds.) FSTTCS 2007. LNCS, vol. 4855, pp. 449–460. Springer, Heidelberg (2007)
43. Schewe, S.: Büchi complementation made tight. In: STACS. LIPIcs, vol. 3, pp. 661–672 (2009)
44. Sebastiani, R., Tonetta, S.: "More" deterministic vs. "smaller" Büchi automata for efficient LTL model checking. In: Geist, D., Tronci, E. (eds.) CHARME 2003. LNCS, vol. 2860, pp. 126–140. Springer, Heidelberg (2003)
45. Sistla, A.P., Vardi, M.Y., Wolper, P.: The complementation problem for Büchi automata with appplications to temporal logic. TCS 49, 217–237 (1987)
46. Sohail, S., Somenzi, F.: Safety first: A two-stage algorithm for LTL games. In: FMCAD, pp. 77–84. IEEE (2009)
47. 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)
48. Thomas, W.: Complementation of Büchi automata revisited. In: Jewels are Forever, pp. 109–120. Springer (1999)

49. Tsai, M.-H., Chan, W.-C., Tsay, Y.-K., Luo, C.-J.: Incremental translation of full PTL formulae to Büchi automata (manuscript 2013)
50. Tsai, M.-H., Fogarty, S., Vardi, M.Y., Tsay, Y.-K.: State of Büchi complementation. In: Domaratzki, M., Salomaa, K. (eds.) CIAA 2010. LNCS, vol. 6482, pp. 261–271. Springer, Heidelberg (2011)
51. 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)
52. Tsay, Y.-K., Chen, Y.-F., Tsai, M.-H., Wu, K.-N., Chan, W.-C.: GOAL: A graphical tool for manipulating Büchi automata and temporal formulae. In: Grumberg, O., Huth, M. (eds.) TACAS 2007. LNCS, vol. 4424, pp. 466–471. Springer, Heidelberg (2007)
53. Tsay, Y.-K., Tsai, M.-H., Chang, J.-S., Chang, Y.-W., Liu, C.-S.: Büchi Store: An open repository of $\omega$-automata. STTT 15(2), 109–123 (2013)
54. Zielonka, W.: Infinite games on finitely coloured graphs with applications to automata on infinite trees. TCS 200(1-2), 135–183 (1998)