These benchmarks show the progress in LTL translation of various versions of Spot. Other tools are shown for comparison. In all experiments, smaller numbers are better.

Last updated on 2012-06-14 by Alexandre Duret-Lutz <adl@lrde.epita.fr>.

## Translation of 188 formulas from the literature

For references, see `bench/ltl2tgba/README` in the Spot distribution.

Cumulated sizes of automata for 188 formulas from the literature

Count of nondeterministic states and automata

Products with a random state-space of 200 states

| | | $\Sigma\|A_{\neg\varphi}\|$ | | non det. | | $\Sigma\|A_M \otimes A_{\neg\varphi}\|$ | |
|---|---|---|---|---|---|---|---|
| | | st. | tr. | st. | aut. | st. | tr. |
| BA | Spin 6.1.0 (☠×7) | 1 635 | 7 825 | 1 402 | 176 | 314 218 | 21 549 478 |
| | ltl2ba 1.1 | 1 080 | 3 646 | 871 | 177 | 215 717 | 12 766 425 |
| | LTL→NBA | 989 | 3 214 | 784 | 178 | 197 568 | 12 063 463 |
| | Modella 1.5.9 | 1 391 | 4 562 | 679 | 125 | 274 281 | 10 907 038 |
| | ltl3ba 1.0.1 | 924 | 2 815 | 722 | 177 | 184 589 | 10 710 928 |
| | ltl3ba 1.0.1 -M | 909 | 2 734 | 375 | 126 | 180 616 | 6 868 411 |
| | ltl3ba 1.0.1 -S | 846 | 2 332 | 641 | 176 | 168 991 | 9 462 074 |
| | ltl3ba 1.0.1 -M -S | 884 | 2 538 | 349 | 126 | 175 626 | 6 562 700 |
| | Spot-0.6 | 834 | 2 623 | 520 | 157 | 166 579 | 9 090 819 |
| | Spot-0.6 det. | 834 | 2 623 | 292 | 115 | 165 677 | 6 258 743 |
| | Spot-0.7.1 | 834 | 2 419 | 520 | 157 | 166 579 | 8 749 162 |
| | Spot-0.7.1 det. | 834 | 2 419 | 292 | 115 | 165 677 | 6 258 605 |
| | Spot-0.7.1 WDBA | 773 | 2 166 | 159 | 52 | 153 535 | 5 657 125 |
| | Spot-0.8.3 | 834 | 2 419 | 520 | 157 | 166 579 | 8 749 162 |
| | Spot-0.8.3 det. | 834 | 2 419 | 292 | 115 | 165 677 | 6 258 605 |
| | Spot-0.8.3 WDBA | 770 | 2 159 | 159 | 52 | 152 935 | 5 633 081 |
| | Spot-0.9 | 831 | 2 422 | 521 | 157 | 165 971 | 8 723 693 |
| | Spot-0.9 det. | 831 | 2 422 | 295 | 115 | 165 033 | 6 254 283 |
| | Spot-0.9 Sim | 789 | 2 205 | 245 | 105 | 157 003 | 5 802 744 |
| | Spot-0.9 WDBA | 772 | 2 173 | 164 | 52 | 153 344 | 5 668 811 |
| | Spot-0.9 WDBA+Sim | 746 | 2 032 | 146 | 51 | 148 514 | 5 436 445 |
| | Spot-0.9.1 | 822 | 2 387 | 512 | 157 | 164 179 | 8 673 053 |
| | Spot-0.9.1 det. | 822 | 2 387 | 284 | 115 | 163 277 | 6 193 160 |
| | Spot-0.9.1 Sim | 780 | 2 158 | 231 | 105 | 155 251 | 5 720 190 |
| | Spot-0.9.1 WDBA | 768 | 2 159 | 157 | 52 | 152 535 | 5 628 444 |
| | Spot-0.9.1 WDBA+Sim | 742 | 2 018 | 139 | 51 | 147 709 | 5 396 354 |
| TGBA | Spot-0.6 | 757 | 2 089 | 451 | 157 | 151 185 | 7 573 811 |
| | Spot-0.6 det. | 757 | 2 089 | 265 | 115 | 150 445 | 5 696 034 |
| | Spot-0.7.1 | 757 | 2 089 | 451 | 157 | 151 185 | 7 573 811 |
| | Spot-0.7.1 det. | 757 | 2 089 | 265 | 115 | 150 445 | 5 696 034 |
| | Spot-0.7.1 WDBA | 705 | 1 886 | 137 | 52 | 140 100 | 5 156 767 |
| | Spot-0.8.3 | 757 | 2 085 | 451 | 157 | 151 185 | 7 570 733 |
| | Spot-0.8.3 det. | 757 | 2 085 | 265 | 115 | 150 445 | 5 696 034 |
| | Spot-0.8.3 WDBA | 704 | 1 879 | 137 | 52 | 139 900 | 5 148 732 |
| | Spot-0.9.1 | 745 | 2 049 | 443 | 157 | 148 785 | 7 480 832 |
| | Spot-0.9.1 det. | 745 | 2 049 | 257 | 115 | 148 045 | 5 630 589 |
| | Spot-0.9.1 Sim | 706 | 1 890 | 211 | 105 | 140 621 | 5 209 858 |
| | Spot-0.9.1 WDBA | 702 | 1 875 | 135 | 52 | 139 500 | 5 144 095 |
| | Spot-0.9.1 WDBA+Sim | 676 | 1 761 | 119 | 51 | 134 676 | 4 915 159 |

BDD variables allocated during LTL simplifications in Spot-0.9 caused the degeneralization to use a less efficient order.

The TGBAs output by Spot-0.9 and 0.9.1 are identical.

☠ = 10min timeout

Modella is run with all available optimizations: `-o1 -g -e -r12`.

Keys for Spot:

| | |
|---|---|
| det. | More deterministic output (`-x`) |
| WDBA | Weak Deterministic Büchi Automaton minimization (`-x -Rm`) |
| Sim | Reduction based on direct simulation on TGBA (`-x -RDS`) |
| WDBA+Sim | All the above (`-x -Rm -RDS`) |

All formula reductions are always applied (`-r7`), and SCC simplifications are always performed (`-R3`).

## 5 classes of formulas

This benchmark consists in 5 parameterized classes of formulas studied by Cichoń et al. (DEPCOS'09). Each class is translated with parameter $n$ ranging from 1 to 20, so that makes a total of 100 formulas. Each of the tools below produces the theoretically smallest Büchi Automaton, so we only measure the total time it takes to translate these 100 formulas. See `bench/ltlclasses/README` to reproduce.

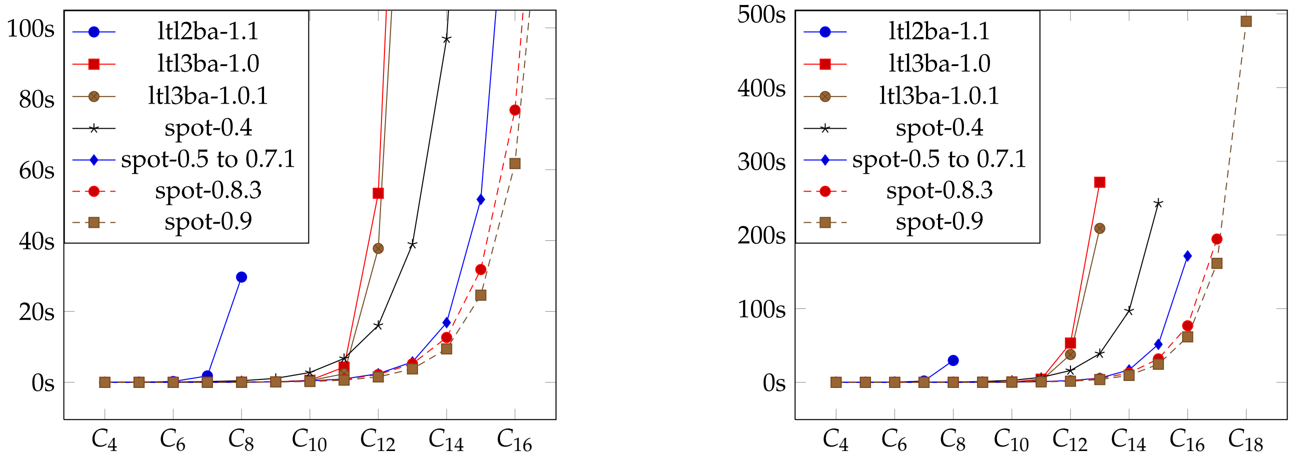
| | |
|---|---|
| Spot 0.8.3 | 562 seconds |
| Spot 0.9 | 315 seconds |
| Spot 0.9.1 | 198 seconds |
| ltl3ba 1.0.1 | 77 seconds |

For the above translations Spot is configured with basic LTL rewritings enabled (-r1).
`ltl3ba` is run with its default options, except for the family of formulas of the form $f_n = \mathsf{F}(p_1 \wedge \mathsf{F}(p_2 \wedge \ldots \mathsf{F}(p_n))) \wedge \mathsf{F}(1_1 \wedge \mathsf{F}(q_2 \wedge \ldots \mathsf{F}(q_n)))$ where LTL simplifications have been disabled.[1]

Spot 0.9 and 0.9.1 actually spend all their time translating the class of formulas representing weak fairness constraints: $g_n = \bigwedge_{i=1}^{n} \mathsf{G}\,\mathsf{F}\,p_i$. The other four classes are translated instantaneously, or nearly so: the worst formula outside the $g_n$ class is $f_{20}$ and it takes 2.5s to translate. Spot 0.9 needs 215s to translate $g_{20}$. Thanks to an improved translation of the G operator, Spot 0.9.1 needs only 109s to translate $g_{20}$ (45% of this time is spent in the degeneralization procedure which is really inefficient). Comparatively, `ltl3ba`, which has a specific handling of subformulas that have the form of $g_n$, will translate $g_{20}$ in only 42s!

## Rozier's LTL Counters

This parameterized family of LTL formulas, $C_n$, describes circular automata with $n2^n$ states and as much transitions. These formulas are quite *heavy*; e.g. $C_3 = ((a \wedge (\mathsf{G}(a \rightarrow (\mathsf{X}(\neg a \wedge \mathsf{X}(\neg a \wedge \mathsf{X}\,a)))))) \wedge ((\neg b) \wedge \mathsf{X}(\neg b \wedge \mathsf{X}\,\neg b)) \wedge (\mathsf{G}((a \wedge \neg b) \rightarrow (\mathsf{X}((\mathsf{X}\,\mathsf{X}\,b) \wedge (((\neg a) \wedge (b \rightarrow \mathsf{X}\,\mathsf{X}\,\mathsf{X}\,b) \wedge ((\neg b) \rightarrow (\mathsf{X}\,\mathsf{X}\,\mathsf{X}\,\neg b))) \,\mathsf{U}\, a))))) \wedge (\mathsf{G}((a \wedge b) \rightarrow (\mathsf{X}((\mathsf{X}\,\mathsf{X}\,\neg b) \wedge ((b \wedge (\neg a) \wedge \mathsf{X}\,\mathsf{X}\,\mathsf{X}\,\neg b) \,\mathsf{U}(a \vee ((\neg a) \wedge (\neg b) \wedge (\mathsf{X}((\mathsf{X}\,\mathsf{X}\,b) \wedge (((\neg a) \wedge (b \rightarrow \mathsf{X}\,\mathsf{X}\,\mathsf{X}\,b) \wedge ((\neg b) \rightarrow \mathsf{X}\,\mathsf{X}\,\mathsf{X}\,\neg b)) \,\mathsf{U}\, a)))))))))))$. See `bench/ltlcounter/README` for references. The following plots (two zoom levels for the same data) show the time it takes to generate a TGBA for increasing $n$. A timeout was set at 10 minutes, so values above that are not shown.



The values for spot-0.6 and spot-0.7.1, not shown, are the same as those for spot-0.5. Spot-0.9.1, not shown, is only 2% faster than Spot-0.9: this difference is too small to be seen on such a plot. Spot is run without any pre- or post-processings: they are not needed to translate these formulas. `ltl2ba` is run with options `-p -l -c -U` disabling pre- and post- processings that would unfairly increase the runtime; similarly `ltl3ba` is run with options `-p -l -c -C -U`. Both tools have been patched to add the option `-U` so they exit immediately after the TGBA has been constructed. This way we measure the actual translation from LTL to TGBA without any extra cost of translating it to a Büchi automaton.

All experiments were ran under GNU/Linux on an Intel Core2 Q9550 running at 2.83GHz with 8GB of RAM.

[1] These simplifications have no effect on this class. The problem is actually twofold. Spot handles $\wedge$ as a commutative operator so it is unable to distinguish between $\mathsf{F}(p_1 \wedge \mathsf{F}(p_2 \wedge \mathsf{F}(p_3)))$ and $\mathsf{F}(\mathsf{F}(\mathsf{F}(p_3) \wedge p_2) \wedge p_1)$ which have exactly the same internal representation. Therefore when Spot constructs these benchmarking formulas, it may output one of these two forms. `ltl3ba` on the other hand, distinguishes these two formulas and its LTL simplifications are much slower (1h versus 1s for $f_{17}$) when the latter order is used.