Our approach to the RERS challenge 2020

Frédéric Lang, Wendelin Serwe
Inria & LIG – CONVECS team (Grenoble, France)

Franco Mazzanti
ISTI-CNR – FM&&T group (Pisa, Italy)
Rigorous Examination of Reactive Systems

• Every year since 2012

• Several categories:
  - Sequential systems (C programs): reachability, LTL
  - Parallel systems (Networks of automata): CTL, LTL

• Competition flow:
  - Organizers release problems of various complexities built by property-preserving transformations of smaller systems
  - Participants have 4-6 weeks to solve the problems and submit results
  - Two categories of medals (gold, silver, bronze) are awarded during the event: achievements and ranking
  - Participants also give a talk where they explain their approach

• In 2019, we won all gold medals in the parallel LTL and parallel CTL categories with 100% success using CADP
Our participation in RERS 2020

- Category: parallel CTL
- Main tool: CADP (cadp.inria.fr)
- Auxiliary tools:
  - PMC (convecs.inria.fr/software/pmc)  
    Partial model checking on top of CADP
  - KandISTI/FMC (fmt.isti.cnr.it/kandisti)  
    Cross-checking of CTL results
- We lacked time to tackle parallel LTL: not at the core of CADP, requires craftwork
The CADP toolbox
http://cadp.inria.fr

• Developed by Inria/CONVECS for more than 30 years
• Model & equivalence checking, rapid prototyping, test case generation, ... (> 80 tools and libraries)
• Enumerative techniques: LTS model
• Main languages and tools used in this work:
  • LNT system description language,
  • MCL property description language,
  • GENERATOR/DISTRIBUTOR state space generators,
  • EVALUATOR model checker,
  • BCG_MIN LTS minimizer,
  • SVL scripting language and compiler, ...
RERS parallel verification tasks

• System description $P_1 \parallel \ldots \parallel P_n$
  • 9 system descriptions 101 to 109
    from 5 to 16 parallel processes $P_1$, $P_2$, ...
    from 26 to 75 actions $L_0$, $L_1$, ...
  • Synchronization on the intersection of actions
  • We used the DOT representation provided by the organizers, automatically translated to LNT

• Property $\varphi$
  – 10 CTL properties for each system description
    from $10X#01$ to $10X#10$
Process reduction

Given a (mu-calculus, or CTL) formula $\varphi$

- extract from $\varphi$ the maximal set of actions that can be hidden in the system [MW14]...

- extract from $\varphi$ a set of so-called strong actions, defining a sharp bisimulation relation, stronger than branching and weaker than strong [LMM20]...

... so that hiding + reduction preserve $\varphi$

Applied whenever possible


1\textsuperscript{st} attempt: On-the-fly verification

- Using the Evaluator model checker of CADP
- Successful on \textbf{48/90} (53 \%) problems:
  - Problem \textbf{101}: \textit{10/10}
  - Problem \textbf{102}: \textit{8/10} (all but 102\#\{07,08\})
  - Problem \textbf{103}: \textit{9/10} (all but 103\#02)
  - Problem \textbf{104}: \textit{2/10} (104\#\{04,09\})
  - Problem \textbf{105}: \textit{3/10} (105\#\{02,03,04\})
  - Problem \textbf{106}: \textit{2/10} (106\#\{01,07\})
  - Problem \textbf{107}: \textit{3/10} (107\#\{01,09,10\})
  - Problem \textbf{108}: \textit{7/10} (all but 108\#\{02,07,08\})
  - Problem \textbf{109}: \textit{4/10} (109\#\{01,04,06,10\})
"problem.exp" = leaf sharp reduction hold L9 of hide all but L1, L9 in
   par
   L0, L1, L3, L4, L6, L7, L8, L25, L26 -> P1
   || L0, L1, L3, L4, L6, L7, L8, L11, L13 -> P2
   || L11, L13, L14, L15, L16, L20, L22, L23, L24 -> P3
   || L14, L15, L16, L17, L18, L23, L24, L25, L26 -> P4
   || L17, L18, L20, L22 -> P5
end par
end hide;
"problem.exp" \models AF (AW ([ "L1" ] false, < "L9" > true));
2\textsuperscript{nd} attempt: Partial model checking

• Following [Anderson-95]: Iteratively turn the check $P_1 || ... || P_n \models \varphi$ into $P_2 || ... || P_n \models \varphi \ // P_1$ (quotienting) until quotient gets a true/false value

• Successful on 30/42 (71%) remaining problems
  • Problem 102: 2/2
  • Problem 103: 1/1
  • Problem 104: 8/8
  • Problem 105: 7/7
  • Problem 106: 4/8 (all but 106#{03,04,06,10})
  • Problem 107: 5/7 (all but 107#{05,06})
  • Problem 108: 1/3 (108#{08})
  • Problem 109: 2/6 (109#{02,09})
SVL script example: 102#07

Property \( AF ((< L9 \triangleright true) \land AG ([ L6 ] false)) \) Visible: L6, L9 Strong: L9

"problem.exp" = \texttt{leaf sharp reduction hold L9 of}
\texttt{hide all but L6, L9 in}
\begin{verbatim}
par
  L0, L2, L3, L6, L7, L8, L9, L30, L31 -> P1
  || L0, L2, L3, L6, L7, L8, L9, L10, L11 -> P2
  || L10, L11, L12, L13, L19, L20, L21, L22 -> P3
  || L12, L13, L15, L17, L18, L21, L22, L23, L24, L25 -> P4
  || L15, L17, L18, L19, L20 -> P5
  || L23, L24, L25, L26, L27 -> P6
  || L26, L27, L30, L31 -> P7
end par
end hide;
\end{verbatim}

\% pmc -lefrighth -orelim "problem.exp" "problem102_07.mcl"

-orelim to save time and memory
(otherwise check does not succeed)
Remarks

• Some properties were simplified taking into account other results
  Example:
  • 106#05: « AG ([ L0 ] false) » is TRUE
  • Thus, 5 properties of problem 106 using L0 can be simplified
    e.g., 106#02: A ((< L1 > true) ⇒ A ([ L8 ] false U < L0 > true) W < L7 > true)
    becomes A ([ L1 ] false W < L7 > true)
  • This allows more labels to be hidden / less labels defined as strong
    e.g., 106#02: L8 hidden, L1 not strong ⇒ greater reduction
  • For properties of the form AG ([ Lx ] false), we minimized the
    processes wrt. safety equivalence (weaker relation than sharp)
  • Invariance properties of the form AG (φ₁ ∧ φ₂) were split into
    independent invariance properties: AG (φ₁) and AG (φ₂)
Other attempts

• 106#06 solved by reasoning and searching a counterexample using interactive simulation

• We also tried distributed state space generation using CADP on the Grid’5000 platform

• But state spaces remained too big, even using compositional reduction (billions of states)
Comparison with RERS 2019

• In 2019:
  • All examples could be verified fastly using compositional sharp reduction (smart heuristic)
  • On-the-fly verification was less successful than in 2020
  • Partial model checking was not even tried because of the success of compositional verification

• In 2020, the problems evolved:
  • Processes are larger
  • There is less concurrency (up to 16 // processes instead of 70)
  • The branching factor and nondeterminism are higher
  • This probably limits the « confluence » of concurrent transitions eliminable by sharp bisimulation
Conclusions

• Gold medals (achievement and ranking), no bad answer
• We found RERS 2020 much harder than RERS 2019
  • 0/180 problems remained unsolved in 2019
  • 11/90 (12%) problems remained unsolved in 2020
  (106#{03,04,10}, 107#{05,06}, 108#{02,07}, 109#{03,05,07,08})
• As ever, techniques are complementary: what works on some problems may not work on others
• Sharp bisimulation reduction remains very useful
• But we did not yet implement full minimization... Don’t know if results would have been better