infChecker at the 2019 Confluence Competition*

Raúl Gutiérrez[1]† and Salvador Lucas[1]

ELP group, DSIC, Universitat Politècnica de València
Camino de Vera s/n, E-46022 Valencia, Spain
{rgutierrez,slucas}@dsic.upv.es

1 Overview

infChecker 1.0 is a tool for checking *(in)feasibility* of sequences $(s_i \to^* t_i)_{i=1}^n$ [3] over *Conditional Term Rewriting Systems* (CTRSs) based on a *Feasibility Framework* similar to the *Dependency Pair Framework* used in termination:

http://zenon.dsic.upv.es/infChecker/

infChecker is written in Haskell. Three *processors* have been implemented for the aforementioned feasibility framework:

- $\mathsf{P}^{\mathsf{Sat}}$ integrates the satisfiability approach described in [3] to prove infeasibility. In infChecker, we use the model generators AGES [1] and Mace4 [5] to find a proof.
- $\mathsf{P}^{\mathsf{Prov}}$ integrates the logic-based approach to program analysis described in [2] to prove feasibility by theorem proving. In infChecker, we use the theorem prover Prover9 [5].
- P^{NC} adapt the processor that narrow conditions in the 2D DP framework [4] to be used with feasibility sequences.

Since we have three processors, our proof strategy is very simple: (1) first, we try to prove feasibility using $\mathsf{P}^{\mathsf{Prov}}$; (2) if $\mathsf{P}^{\mathsf{Prov}}$ fails, we apply $\mathsf{P}^{\mathsf{Sat}}$; (3) if $\mathsf{P}^{\mathsf{Sat}}$ fails, we apply P^{NC}; (4) if P^{NC} succeeds and modifies the feasibility sequence, we go to (2), otherwise we return MAYBE.

We succesfully participated in the 2019 Confluence Competition in the INF (infeasibility) category, being the most powerful tool for proving and disproving infeasibility.

References

[1] R. Gutiérrez and S. Lucas. Automatic Generation of Logical Models with AGES (System Description). In *Proc. of CADE 2019*, LNCS to appear. Springer, 2019.

[2] S. Lucas. Proving Program Properties as First-Order Satisfiability. In Selected and Revised papers from *LOPSTR 2018* LNCS 11408:3–21. Springer, 2019.

[3] S. Lucas and R. Gutiérrez. Use of Logical Models for Proving Infeasibility in Term Rewriting. *Information Processing Letters*, 136:90–95, 2018.

[4] S. Lucas, J. Meseguer, and R. Gutiérrez. The 2D Dependency Pair Framework for conditional rewrite systems. Part I: Definition and basic processors. *Journal of Computer and System Sciences*, 96:74–106, 2018.

[5] W. McCune. Prover9 and Mace4. [online]. Available at https://www.cs.unm.edu/~mccune/mace4/.

*Partially supported by the EU (FEDER), and projects TIN2015-69175-C4-1-R, PROMETEO/2019/098, and SP20180225.

†Raúl Gutiérrez was also supported by INCIBE program "Ayudas para la excelencia de los equipos de investigación avanzada en ciberseguridad".