infChecker **at CoCo 2022**

Raúl Gutiérrez[1] Salvador Lucas[2] Miguel Vítores[2]

HAIFA, AUGUST 1TH, 2022

[1]Universidad Politécnica de Madrid
Spain
[2]DSIC & VRAIN
Universitat Politècnica de València
Spain

Description

- infChecker is a tool for checking **(in)feasibility of goals** $\mathcal{G} = \{F_i\}_{i=1}^{m}$, where $F_i = (s_{ij} \bowtie_{ij} t_{ij})_{i=1}^{n_i}$.
- $\bowtie_{ij}$ represents **predicates** on terms defined by provability of goals $s \bowtie_{ij} t$ with respect to a *first-order theories* $\mathrm{Th}_{\bowtie_{ij}}$.
- $\bowtie_{ij}$ can be one of the following predicates:
  - One (CS-)rewriting step (`->`, `\->`).
  - Zero or more (CS-)rewriting steps (`->*`, `\->*`).
  - One or more (CS-)rewriting steps (`->+`, `\->+`).
  - Subterm (`|>=`) and strict subterm (`|>`).
  - (CS-)Joinability (`->*<-`, `\->*<-/`).
  - One (CS-)convertibility step (`<-->`, `<-/\->`).
  - Zero or more (CS-)convertibility steps (`<-->*`, `<-/\->*`).

Implementation

- The tool is available here:
  `http://zenon.dsic.upv.es/infChecker/`.
- It is written in Haskell and provides a first implementation of the **Feasibility Framework**, where four **processors** have been implemented:
  - P^{Sat} integrates a satisfiability approach to **prove infeasibility using model generators** as AGES and Mace4 to find a proof.
  - P^{UR} **simplifies** problems by removing non-usable rules.
  - P^{Prov} integrates a logic-based approach to program analysis to **prove feasibility by theorem proving**. In infChecker, we use the theorem prover Prover9.
  - P^{NC} adapt the processor that **narrow conditions** in the 2D DP framework for proving operational termination of CTRs to be used with feasibility sequences.

Strategy and Results

- Our **proof strategy** is:
  1. we apply P^{UR} whenever it is sound and complete;
  2. we try to prove feasibility using $\mathsf{P}^{\mathsf{Prov}}$;
  3. if $\mathsf{P}^{\mathsf{Prov}}$ fails, we apply $\mathsf{P}^{\mathsf{Sat}}$;
  4. if $\mathsf{P}^{\mathsf{Sat}}$ fails, we apply P^{NC};
  5. if P^{NC} succeeds and modifies the feasibility sequence, we repeat the strategy, otherwise we return `MAYBE`.
- Bibliography:

GL20 R. Gutiérez and S. Lucas. Automatically Proving and Disproving Feasibility Conditions. In Proc. of IJCAR'2020, LNCS 12167:416–435. Springer, 2020.

Luc19 S. Lucas. Proving semantic properties as first-order satisfiability. Artificial Intelligence 277, paper 103174, 24 pages, 2019.

LG18 S. Lucas and R. Gutiérrez. Use of Logical Models for Proving Infeasibility in Term Rewriting. Information Processing Letters, 136:90-95, 2018.