Termination Tools in Automated Reasoning

Sarah Winkler*
Computational Logic Group, Institute for Computer Science, University of Innsbruck

Abstract. Automated reasoning systems often rely on a reduction ordering as input. This parameter is hard to choose appropriately in advance, but critical for success. The subject of this project is to investigate under which circumstances this fixed ordering can be replaced by the use of automatic termination tools, aiming at flexible and more powerful yet efficient implementations of reasoning systems.

What is automated reasoning?

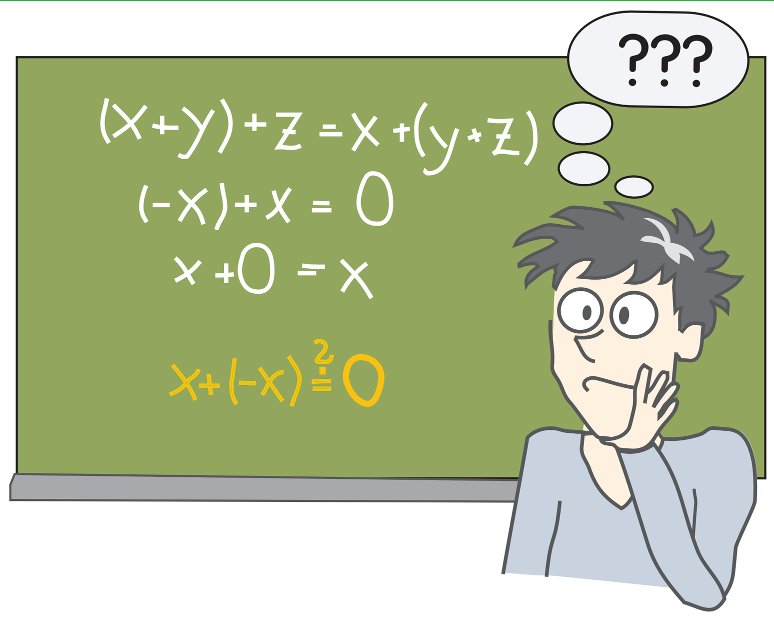
Reasoning refers to the ability to draw conclusions from certain facts. Automated reasoning aims to design computing systems that automate this process. This requires a precise algorithmic description of some formal deduction calculus such that reasoning in this context can be efficiently implemented.

... and what is it good for?

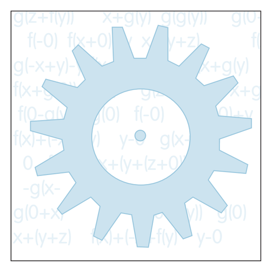
Automated reasoning gained considerable interest over the last two decades and has by now a variety of applications in computer science, such as formal specification of software and hardware systems or verification of cryptographic protocols. Lately, automated reasoning systems are also used in mathematical research, and mathematical software provides proof assistance using automated reasoning techniques.

Example

Knuth-Bendix completion can construct confluent and terminating rewrite systems for input equalities. These allow to determine the validity of arbitrary identities in the theory.



$$\begin{aligned} (x+y)+z &\approx x+(y+z)\\ x+(-x) &\approx 0\\ x+0 &\approx x\\ f(x)+f(y) &\approx f(x+y)\\ g(x)+g(y) &\approx g(x+y)\\ f(x)+g(y) &\approx g(y)+f(x) \end{aligned}$$



$$\begin{aligned} (x+y)+z &\to x+(y+z)\\ -(x+y) &\to ((-y)+(-x))\\ f(x)+(f(y)+z) &\to g(x+y)+z\\ f(x)+(f(y)+z) &\to g(x+y)+z\\ &\vdots\\ -x+(x+y) &\to y \end{aligned}$$

$x+(-x) \approx 0$ YES

$-(g(y)+f(x)) \quad -f(y)-g(x)$ NO

A modern automated reasoning tool

The underlying calculus

Automated reasoning systems are based on different calculi. Well known examples are

- Knuth-Bendix completion
- ordered completion
- paramodulation
- congruence closure
- inductive theorem proving

Example (Paramodulation)

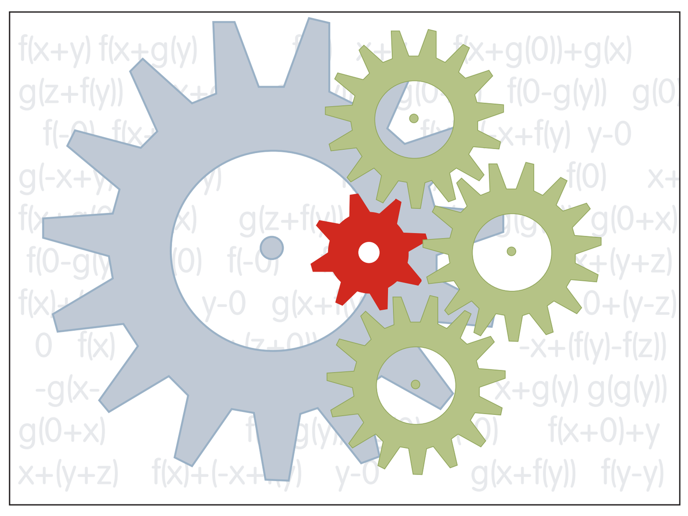
$$\frac{l \approx r \quad s \approx t}{(s[r]_p \approx t)\,\sigma}$$

where $\sigma = mgu(s|_p, l)$, $s|_p$ is not a variable and for some ground substitution θ it holds that $l\sigma\theta \succ r\sigma\theta$.

Simulating multiple runs

Simulating multiple runs in parallel increases the success rate. To gain efficiency, many inference steps can be shared among the parallel processes using nodes:

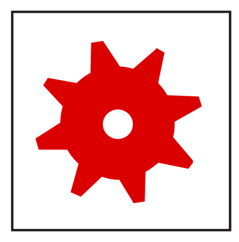
$\langle x+(y+z) : (x+y)+z, R_1, R_2, E, C_1, C_2 \rangle$



Termination tools

Termination tools implement a variety of techniques to check whether a rewrite system is terminating.

$(x+y)+z \to x+(y+z)$
$(-x)+x \to 0$ — YES



$x+y \to x+y+0$
$(-x)+x \to 0$ — NO

Automated reasoning systems often require a well-founded term ordering as input.

- critical for success
- hard to determine in advance

Idea: use termination tools instead of fixed orderings, gain flexibility and computational power.

Optimizations

To maintain large amount of data, use

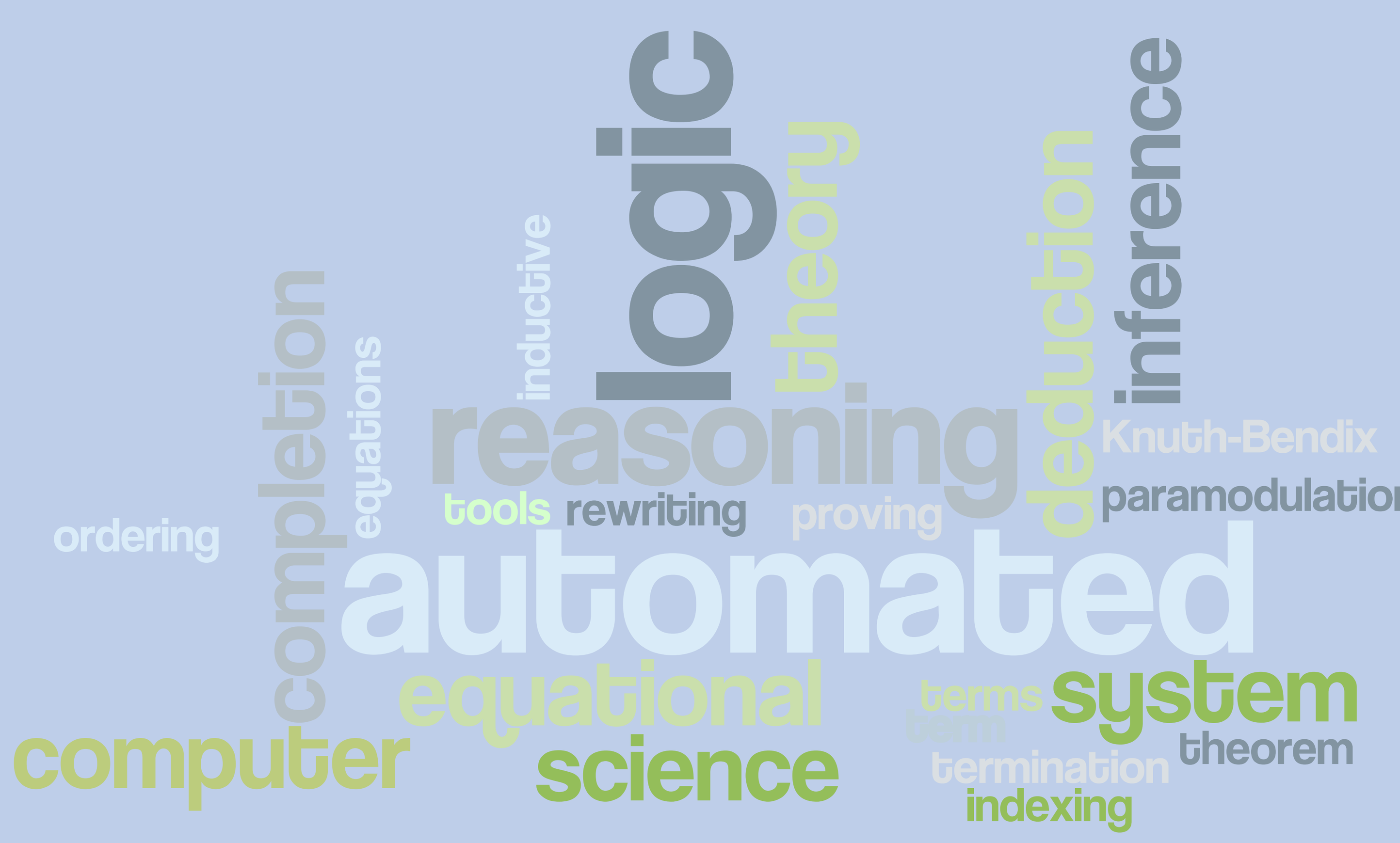
- term indexing techniques
- critical pair criteria
- checking for ismorphic processes
- careful selection strategies

Results

- The Knuth-Bendix completion tool **mkb$_{\mathsf{TT}}$** implements multi-completion with termination tools. It can e.g. complete the theories CGE_3 and CGE_4 of three and four commuting group endomorphisms, which were not completed automatically before. The tool is available via a web interface.
- Our approach extends to ordered completion and completion-based theorem proving. An outline of this method was submitted for publication, and a prototype implemented in the tool **omkb$_{\mathsf{TT}}$**.

Web

`http://cl-informatik.uibk.ac.at/users/swinkler`



*Recipient of a DOC-fFORTE fellowship of the Austrian Academy of Sciences at the Institute for Computer Science, University of Innsbruck