

Maple's Simplification Process as Constraint Contextual Rewriting

Alessandro Armando

MRG-Lab

DIST, University of Genova



CALCULEMUS Autumn School

Pisa, Sept 30 – Oct 1, 2002

The Problem of Simplification

Hard: Not decidable for many domains.

Problem: representation often does not provide sufficient information.

The Problem of Simplification

Hard: Not decidable for many domains.

Problem: representation often does not provide sufficient information.

$$\begin{array}{l} \frac{x^2 - 1}{x - 1} \rightsquigarrow x + 1 \\ \quad \quad \quad = \quad \quad \quad \text{over } \mathbb{Q}(x) \\ \quad \quad \quad \neq \quad \quad \quad \text{over } \mathbb{Q} \rightarrow \mathbb{Q} \\ \quad \quad \quad = \quad \quad \quad \text{for } x \neq 1 \end{array}$$

The Problem of Simplification

Hard: Not decidable for many domains.

Problem: representation often does not provide sufficient information.

$$\frac{x^2 - 1}{x - 1} \rightsquigarrow x + 1$$

$$=$$

over $\mathbb{Q}(x)$

$$\neq$$

over $\mathbb{Q} \rightarrow \mathbb{Q}$

$$=$$

for $x \neq 1$

$$\sqrt{x^2} \rightsquigarrow x$$

$$=$$

$x \geq 0$

$$\neq$$

$x < 0$

$$?$$

$x \in \mathbb{C}$, consider branch cuts!

Maple's Assume Facility

- Maintains a global context of user-provided assumptions.
- Simplification is performed only if conditions can be derived from the context.
- Examples:

$$\begin{aligned} \text{assume}(x > 1, y > 1) &\rightsquigarrow \text{is}(x + 2y - 3 > 0) = \text{true} \\ &\rightsquigarrow \sqrt{(x + 2y - 3)^2} = x + 2y - 3 \\ \text{assume}(x, \text{real}) &\rightsquigarrow \text{is}(e^x \geq 0) = \text{true} \\ &\rightsquigarrow \sqrt{(e^x)^2} = e^x \end{aligned}$$

- Is integrated to Maple's evaluator.

The Property Reasoner

- **Objects:** Maple's objects of computation (e.g.: π , \sqrt{x})
- **Properties:** sets of objects (e.g. $[0, +\infty)$, set of continuous functions)
- They are denoted by **object terms** and **property terms** respectively.
- If $t \in p$ then we say that t **has property p** and write $t : p$.

Property Terms

Property Terms are defined in the following way:

- (i) Atomic symbols: `real`, `positive`, `monotonic`, . . .
- (ii) Expressions of the form (l, r) , $[l, r)$, $(l, r]$, and $[l, r]$, where l and r are object terms denoting numbers.
- (iii) Are built out of simpler property terms using the constructors \neg , \wedge , and \vee .

Property Functions

- Knowledge about functions is encoded in **property functions**.
- Every function f has an associated property function \bar{f} .
- A property function maps properties into properties, e.g.:

$$[0, \infty) \bar{*} [0, \infty) = [0, \infty)$$

$$(-\infty, 0] \bar{*} (-\infty, 0] = [0, \infty)$$

$$[0, \infty) \bar{*} (-\infty, 0] = (-\infty, 0]$$

$$(-\infty, 0] \bar{*} [0, \infty) = (-\infty, 0]$$

$$\overline{\sin}([0, \pi]) = [0, 1]$$

$$\overline{\sin}([0, \frac{\pi}{3}]) = [0, \frac{1}{2}\sqrt{3}]$$

- Property functions satisfy

$$f(t_1, \dots, t_n) : \bar{f}(p_1, \dots, p_n) \quad \text{for} \quad t_1 : p_1, \dots, t_n : p_n.$$

Maple's Evaluation as CCR

The assume facility is a reasoning specialist.
Its integration to the evaluator can be seen as CCR.

First step: regard evaluation functions as rewrite rules.

This is adequate:

- Most evaluation functions perform local transformations on terms.
- Subterms are evaluated recursively.

Property Reasoner as Reasoning Specialist

Constraint store contains property judgements: $\{t_1 : p_1, \dots, t_n : p_n\}$ or $\vec{t} : \vec{p}$.

- $\text{cs-init}(C)$ holds if and only if C is the empty set of judgements.
- $\text{cs-unsat}(C)$ holds if and only if $(u : \perp) \in C$ for some term u .
- $P :: C \xrightarrow{\text{cs-simp}} C'$ is defined in terms of `assume` and `is`.

$$(assume) \frac{}{\{u : p\} :: (\vec{t} : \vec{p}) \xrightarrow{\text{cs-simp}} \{u : p\} \cup (\vec{t} : \vec{p})}$$

$$(is) \frac{(\vec{t} : \vec{p}) :: u \xrightarrow{\text{solve}} v[\vec{t}] \quad v[\vec{p}] \xrightarrow{\text{prop-eval}} p_0}{\{\neg(u : p)\} :: (\vec{t} : \vec{p}) \xrightarrow{\text{cs-simp}} \{u : \perp\} \cup (\vec{t} : \vec{p})} \text{ if } p_0 \preceq p$$

Property Functions and Augmentation

Augmentation not needed to reconstruct Maple's assume facility.

Weibel and Gonnet's view of property functions:

- Property functions are part of the Property Reasoner.

Proposed view:

- Property functions are lemmas.
- They state properties of functions that are uninterpreted for the reasoning specialist.

⇒ Weak form of augmentation.

Strengthening the Property Reasoner

Property functions as lemmas.

- They can only express relations between **properties** of the input arguments and the output.
- Relations that **cannot** be expressed:
 - Monotonicity of a function f

$$Y - X : [0, \infty) \implies f(Y) - f(X) : [0, \infty).$$

- “Squeeze” theorem $x \in [a, b] \implies g(x) \leq f(x) \leq h(x)$

$$X : [a, b] \implies f(X) - g(X) : [0, \infty)$$

$$X : [a, b] \implies h(X) - f(X) : [0, \infty)$$

- Augmentation does not suffer from this limitation!

Conclusions

Modular design of the simplifier:

- Evaluation as a set of **cooperating reasoning modules** with neatly **specified interfaces**.
- Paves the way to adding other (user-provided) reasoning specialists to Maple.
 \implies **more powerful** simplifier.
- Properties of user-defined functions should be **declared** rather than **programmed**.

Reference Paper

- A. Armando and C. Ballarin. [Maple's Evaluation Process as Constraint Contextual Rewriting](#). In the Proceedings of the Intl. Symposium on Symbolic and Algebraic Computation (ISSAC'2001), 2001.

Example: Constraint Contextual Rewriting

Consider a reasoning specialist for total orders and let R contain the rewrite rule

$$X \geq 0 \implies \sqrt{X^2} = X.$$

Then $\sqrt{\cos^2 a}$ rewrites to $\cos a$ in context

$$C = \{0 \leq \sin a, 0 \neq \sin a, \sin a \leq \cos a\}$$

The derivation is

$$C :: \sqrt{\cos^2 a} \xrightarrow[\text{ccr}]{C :: \cos a \geq 0} \text{true} \xrightarrow[\text{ccr}]{\{ \cos a \neq 0 \} :: C \xrightarrow[\text{cs-simp}]{} C'} \cos a$$

where $C' = C \cup \{0 \leq \cos a, \cos a = 0, 0 \neq \cos a\}$, and $\text{cs-unsat}(C')$ holds.

Example: Maple's Evaluator as CCR

$$\{\neg(x + 2y - 3 : [0, \infty))\} ::$$

$$\{\}$$

$$\downarrow \text{ (assume) } x - 1 : [0, \infty)$$

$$\{x - 1 : [0, \infty)\}$$

$$\downarrow \text{ (assume) } y - 1 : [0, \infty)$$

$$\{x - 1 : [0, \infty), y - 1 : [0, \infty)\}$$

$$\downarrow \text{ (is) contradiction } \begin{array}{ccc} x + 2y - 3 & \xrightarrow{\text{solve}} & (x - 1) + 2(y - 1) \\ [0, \infty) + 2[0, \infty) & \xrightarrow{\text{prop-eval}} & [0, \infty) \preceq [0, \infty) \end{array}$$

$$\{x - 1 : [0, \infty), y - 1 : [0, \infty), x + 2y - 3 : \perp\}$$

Example: Augmentation

Let R contain the facts:

$$X : [0, \infty) \wedge Y - X : [0, \infty) \implies \sqrt{Y} - \sqrt{X} : [0, \infty)$$

$$X : [-\frac{\Pi}{2}, \frac{\Pi}{2}] \wedge Y : [-\frac{\Pi}{2}, \frac{\Pi}{2}] \wedge Y - X : [0, \infty) \\ \implies \sin Y - \sin X : [0, \infty)$$

and consider the problem of determining whether $\sqrt{\sin y} - \sqrt{\sin x} : [0, \infty)$ in context

$$C = \{x : [0, \frac{\Pi}{2}], y : [0, \frac{\Pi}{2}], y - x : [0, \infty)\}$$

By (*cxt-entails*) we need to find a C' with

$$\{\neg(\sqrt{\sin y} - \sqrt{\sin x} : [0, \infty))\} :: C \xrightarrow{\text{cs-simp}} C'$$

and $\text{cs-unsat}(C')$.

Example: Augmentation (2)

$$\{\neg(\sqrt{\sin y} - \sqrt{\sin x}) : [0, \infty)\} ::$$

$$\{x : [0, \frac{\pi}{2}], y : [0, \frac{\pi}{2}], y - x : [0, \infty)\}$$

$$\begin{array}{l} \downarrow \\ \text{(augment) with } x : [-\frac{\pi}{2}, \frac{\pi}{2}] \wedge y : [-\frac{\pi}{2}, \frac{\pi}{2}] \wedge y - x : [0, \infty) \\ \implies \sin y - \sin x : [0, \infty) \end{array}$$

$$\{\dots, \sin y - \sin x : [0, \infty)\}$$

$$\begin{array}{l} \downarrow \\ \text{(augment) with } \sin x : [0, \infty) \wedge \sin y - \sin x : [0, \infty) \\ \implies \sqrt{\sin y} - \sqrt{\sin x} : [0, \infty) \end{array}$$

$$\{\dots, \sqrt{\sin y} - \sqrt{\sin x} : [0, \infty)\}$$

$$\begin{array}{l} \downarrow \\ \text{(is) contradiction } \sqrt{\sin y} - \sqrt{\sin x} \xrightarrow{\text{solve}} \sqrt{\sin y} - \sqrt{\sin x} \\ [0, \infty) \xrightarrow{\text{prop-eval}} [0, \infty) \not\leq [0, \infty) \end{array}$$

$$\{\dots, \sqrt{\sin y} - \sqrt{\sin x} : \perp\}$$

Example: Augmentation (3)

The second application of (*augment*) requires a further invocation of the Property Reasoner.

$$\begin{array}{c}
 \{x : [0, \frac{\pi}{2}], \dots\} :: \sin x : [0, \infty) \xrightarrow{\text{ccr}} \mathbf{true} \\
 \hline
 \{\neg(\sin x : [0, \infty))\} :: \{x : [0, \frac{\pi}{2}], \dots\} \xrightarrow{\text{cs-simp}} \{x : [0, \frac{\pi}{2}], \dots, \sin x : \perp\} \\
 \hline
 \{x : [0, \frac{\pi}{2}], \dots\} :: \sin x \xrightarrow{\text{solve}} \sin x \qquad \overline{\sin}[0, \frac{\pi}{2}] \xrightarrow{\text{prop-eval}} [0, 1] \preceq [0, \infty)
 \end{array}$$