Higher-Order Termination

Status Report

Julian Nagele

Seminar 3 @ Computational Logic

May 16, 2012

Higher-Order Rewriting

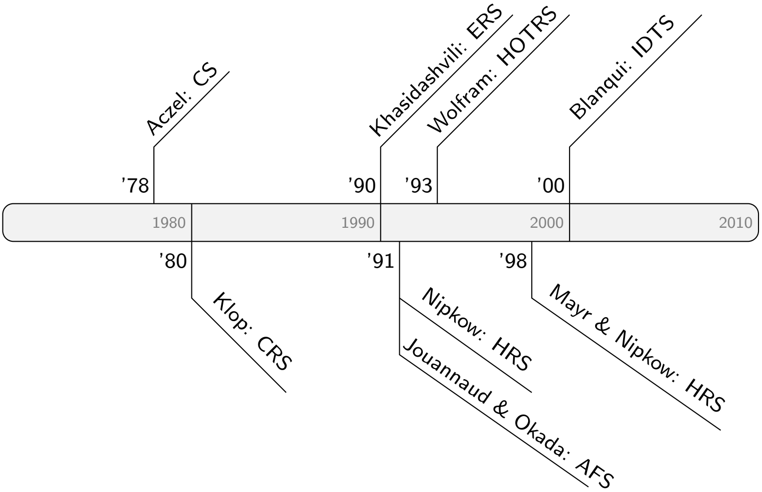
- transformation of functions
- rewriting with bound variables
- combine Term Rewriting and λ-calculus

Example

```
map f [] = []
map f (h : t) = f h : map f t
```

$$\mathrm{map}(F, \mathrm{nil}) \to \mathrm{nil}$$
$$\mathrm{map}(F, \mathrm{cons}(h, t)) \to \mathrm{cons}(F \cdot h, \mathrm{map}(F, t))$$

Historical Overview: Some Formalisms



Simply typed λ-calculus ($\lambda^{\rightarrow}$)

- types, built from base types (sorts) and $\rightarrow$

$$\mathbb{T} ::= \mathbb{S} \mid (\mathbb{T} \rightarrow \mathbb{T})$$

- set $\mathcal{V}$ of typed variables $(x : \sigma)$

Terms

s is a $\lambda^{\rightarrow}$-term if $s : \sigma$ can be inferred using

(var)	$x : \sigma$	if $x : \sigma \in \mathcal{V}$
(app)	$u \cdot t : \sigma$	if $u : \tau \rightarrow \sigma$ and $t : \tau$
(abs)	$\lambda x.t : \tau \rightarrow \rho$	if $x : \tau \in \mathcal{V}$ and $t : \rho$

β-reduction and η-expansion

Definitions

- context C is a term with a single occurrence of some $\Box_\sigma$
- substitutions are type-preserving and do not capture free variables
- $C[(\lambda x.s) \cdot t] \rightarrow_\beta C[s\{x \mapsto t\}]$
- $C[s] \rightarrow_\eta C[\lambda x.s \cdot x]$
    - $x : \sigma$ is a fresh variable
    - s is of functional type
    - no β-redex is created
- $s\updownarrow_\beta^\eta$ denotes unique η-long β-normal form of s

Algebraic Functional Systems vs Higher-Order Rewrite Systems

Algebraic Functional Systems (AFS)

- combine $\lambda^{\rightarrow}$ with algebraic terms
- uses first-order pattern matching
- every system has β-reduction as explicit "built-in" rule

Higher-Order Rewrite Systems (HRS)

- extend $\lambda^{\rightarrow}$ by typed constants
- uses higher-order pattern matching modulo $\beta\eta$

Terms

AFS

- signature $\mathcal{F}$: functions symbols with unique type declarations $(\sigma_1 \times \cdots \times \sigma_n) \to \sigma$
- Terms are built using (var) (abs) (app) and

$$\text{(fun)}\ f(s_1, \ldots, s_n) : \sigma \qquad \text{if } f : (\sigma_1 \times \cdots \times \sigma_n) \to \sigma \in \mathcal{F}$$
$$\text{and } s_1 : \sigma_1, \ldots, s_n : \sigma_n$$

HRS

- signature $\mathcal{F}$: constant symbols of unique type $\mathbb{T}$.
- Pre-terms are built using (var) (abs) (app) and

$$\text{(fun)}\ f : \sigma \qquad \text{if } f : \sigma \in \mathcal{F}$$

- $s \updownarrow_{\beta}^{\eta}$ is a term for a pre-term s

Example (map as Algebraic Functional System)

$$\mathcal{F}: 0 : \mathsf{nat} \quad \mathsf{s} : \mathsf{nat} \to \mathsf{nat}$$
$$\mathsf{nil} : \mathsf{natlist} \quad \mathsf{cons} : (\mathsf{nat} \times \mathsf{natlist}) \to \mathsf{natlist}$$
$$\mathsf{map} : ((\mathsf{nat} \to \mathsf{nat}) \times \mathsf{natlist}) \to \mathsf{natlist}$$

$$\mathcal{R}: \quad \mathsf{map}(F, \mathsf{nil}) \to \mathsf{nil}$$
$$\mathsf{map}(F, \mathsf{cons}(h, t)) \to \mathsf{cons}(F \cdot h, \mathsf{map}(F, t))$$

$$\begin{aligned} & \mathsf{map}(\lambda x.x, \mathsf{cons}(\mathsf{s}(0), \mathsf{nil})) \\ \to_{\mathcal{R}}\ & \mathsf{cons}((\lambda x.x) \cdot \mathsf{s}(0), \mathsf{map}(\lambda x.x, \mathsf{nil})) \\ \to_{\mathcal{R}}\ & \mathsf{cons}(\mathsf{s}(0), \mathsf{map}(\lambda x.x, \mathsf{nil})) \\ \to_{\mathcal{R}}\ & \mathsf{cons}(\mathsf{s}(0), \mathsf{nil}) \end{aligned}$$

Example (map as Higher-Order Rewrite System)

$$\mathcal{F}: 0 : \mathsf{nat} \quad \mathsf{s} : \mathsf{nat} \to \mathsf{nat}$$
$$\mathsf{nil} : \mathsf{natlist} \quad \mathsf{cons} : \mathsf{nat} \to \mathsf{natlist} \to \mathsf{natlist}$$
$$\mathsf{map} : (\mathsf{nat} \to \mathsf{nat}) \to \mathsf{natlist} \to \mathsf{natlist}$$

$$\mathcal{R}: \quad \mathsf{map} \cdot (\lambda x.F \cdot x) \cdot \mathsf{nil} \to \mathsf{nil}$$
$$\mathsf{map} \cdot (\lambda x.F \cdot x) \cdot (\mathsf{cons} \cdot h \cdot t) \to \mathsf{cons} \cdot (F \cdot h) \cdot (\mathsf{map} \cdot (\lambda x.F \cdot x) \cdot t)$$

$$\mathsf{map} \cdot (\lambda y.(\lambda x.x) \cdot y) \cdot (\mathsf{cons} \cdot (\mathsf{s} \cdot 0) \cdot \mathsf{nil})$$
$$\to_{\mathcal{R}} (\mathsf{cons} \cdot ((\lambda x.x) \cdot (\mathsf{s} \cdot 0)) \cdot (\mathsf{map} \cdot (\lambda y.(\lambda x.x) \cdot y) \cdot \mathsf{nil})) \updownarrow_{\beta}^{\eta}$$
$$\to_{\mathcal{R}} \mathsf{cons} \cdot (\mathsf{s} \cdot 0) \cdot (\mathsf{nil} \updownarrow_{\beta}^{\eta})$$

Master Project

[. . .] The aim of this master project is to summarise the existing literature on higher-order termination and to develop an implementation.

AFS vs HRS

- similar results in literature
- similar termination techniques
- sound transformations in both directions exist

AFS

- termination competition
- more recent results
- adaption of techniques originally for HRSs

HRS

- no tool support yet
- not in TPDB
- tightly connected to Isabelle (HO Termfun?)

Termfun: use termination tools to proof totality of Isabelle functions

Termination Techniques

Technique	AFS	HRS
Path Orderings		
HORPO	✓	✓
MHOSPO	✓	
HOIPO	✓	
CPO	✓	
HORPOLO	✓	
Monotone Algebras	✓	✓
POLO	✓	
Dependency Pairs		
Static DPs	✓	✓
Dynamic DPs	✓	✓
Dependency Graph	✓	✓
Subterm Criterion	✓	✓
Usable Rules	✓	✓
Argument Filters	✓	✓
Formative Rules	✓	

Wanda
THOR
My Impl.

HORPO

Example

$$\text{map}(F, \text{cons}(h, t)) \rightarrow \text{cons}(F \cdot h, \text{map}(F, t))$$

- $\text{map}(F, \text{cons}(h, t)) \succ \text{cons}(F \cdot h, \text{map}(F, t))$
- $\text{map}(F, \text{cons}(h, t)) \succ F \cdot h$
- $\text{map}(F, \text{cons}(h, t)) \succ \text{map}(F, t)$
- $\{\{F, \text{cons}(h, t)\}\} \succ_{Mul} \{\{F, t\}\}$

precedence: $\text{map} >_{\mathcal{F}} \text{cons}$

status: $map \in \mathcal{F}_{Mul}$

Dependency Pairs

Example

$$\mathrm{map}(F, \mathrm{cons}(h, t)) \to \mathrm{cons}(F \cdot h, \mathrm{map}(F, t))$$

- static DPs:

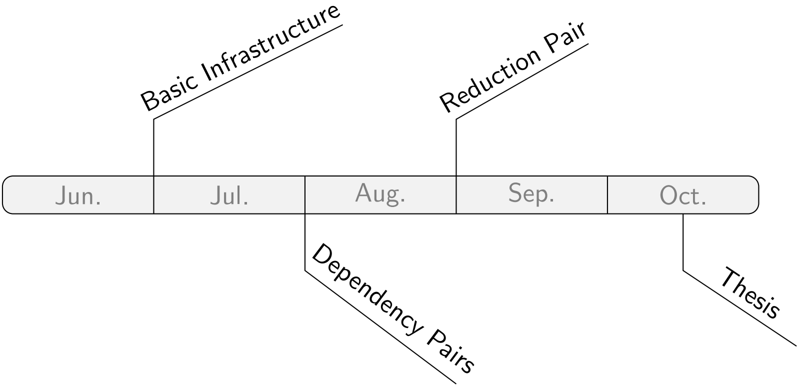
$$\mathrm{map}^{\#}(F, \mathrm{cons}(h, t)) \rightsquigarrow \mathrm{map}^{\#}(F, t)$$

- dynamic DPs:

$$\mathrm{map}^{\#}(F, \mathrm{cons}(h, t)) \rightsquigarrow F \cdot h$$
$$\mathrm{map}^{\#}(F, \mathrm{cons}(h, t)) \rightsquigarrow \mathrm{map}^{\#}(F, t)$$

Schedule

- 27.5 ECTS $\widehat{=}$ 687.5 hours $\widehat{=}$ 17 weeks at 40 hours/week



Summary

- higher-order rewriting combines term rewriting and simply typed lambda-calculus
- many different formalisms exist
- which one should a tool support? AFS vs HRS
- implementation: static DPs & friends and Reduction Pair