

# Finding Lexicographic Orders for Termination Proofs

Lukas Bulwahn

Fakultät für Informatik  
Technische Universität München

February 16th, 2007



## Structure of Termination Proofs

Shown with an example

General Structure

## Goal of the Termination Tactic

## Algorithm of the Termination Tactic

Create all basic measure functions

Prove local descents

Search for a global termination ordering

Instantiate the relation and prove the subgoals

## Mutual recursion

Necessary extension of the algorithm

Ordering the functions



# Structure of Termination Proofs shown with an example

## Ackermann Function

$\text{ack} :: \text{nat} \Rightarrow \text{nat} \Rightarrow \text{nat}$

$\text{ack } 0 \ m = \text{Suc } m$

$\text{ack } (\text{Suc } n) \ 0 = \text{ack } n \ 1$

$\text{ack } (\text{Suc } n) \ (\text{Suc } m) = \text{ack } n \ (\text{ack } (\text{Suc } n) \ m)$



# Structure of Termination Proofs shown with an example

## Ackermann Function

is translated in the following Termination Proof:

0.  $wf ?R$
1.  $\bigwedge n. ((n, 1), (Suc\ n, 0)) \in ?R$
2.  $\bigwedge n\ m. ((Suc\ n, m), (Suc\ n, Suc\ m)) \in ?R$
3.  $\bigwedge n\ m. ((n, ack\ (Suc\ n, m)), (Suc\ n, Suc\ m)) \in ?R$



# General Structure of Termination Proofs

## Structure of Termination Proofs

0. wf ?R
1.  $\bigwedge \text{var}_1, \dots, \text{var}_m. \Gamma_1 \implies (\text{lhs}_1, \text{rhs}_1) \in ?R$
2.  $\bigwedge \text{var}_1, \dots, \text{var}_m. \Gamma_2 \implies (\text{lhs}_2, \text{rhs}_2) \in ?R$
- ⋮
- n.  $\bigwedge \text{var}_1, \dots, \text{var}_m. \Gamma_n \implies (\text{lhs}_n, \text{rhs}_n) \in ?R$



## Goal of the Termination Tactic

Find a suitable relation  $?R$   
and show that  $?R$  is wellfounded  
and the parameters of all function calls are decreasing in  $?R$ .

The tactic searches for lexicographic combinations of size measures on the arguments.



## Structure of Termination Proofs

Shown with an example

General Structure

## Goal of the Termination Tactic

### Algorithm of the Termination Tactic

Create all basic measure functions

Prove local descents

Search for a global termination ordering

Instantiate the relation and prove the subgoals

### Mutual recursion

Necessary extension of the algorithm

Ordering the functions



# Algorithm of the Termination Tactic

## Algorithm of the Termination Tactic

The tactic works in four steps:

1. Create all basic measure functions for the analysis
2. Prove local descents of the measure functions for the recursive calls and gather information in a matrix
3. Search for a global termination order in the matrix
4. Instantiate the relation and prove termination using the proofs of local descent



# 1. step: Create all basic measure functions

Measure functions can be generated from the type  $\alpha$  of the function argument:

$M :: \alpha \Rightarrow (\alpha \Rightarrow \text{nat}) \text{ set}$

$M(\alpha \times \beta) = \{\lambda x.f(\text{fst } x) \mid f \in M(\alpha)\} \cup \{\lambda x.f(\text{snd } x) \mid f \in M(\beta)\}$

$M(\alpha) = \{|\cdot|_\alpha\}$  for every inductive type  $\alpha$

$M(\alpha) = \{\}$  for any other type



## 2. step: Prove local descents

### Definition of a call matrix

A call matrix is a matrix  $C$  with  $\{<, \leq, N, F\}$  entries for each call and each measure function.

The entry  $C_{ij}$  states that the  $j$ .th measure function for  $i$ .th call becomes less, lessequal, none is known about, or proving led to an contradiction.



## 2. step: Prove local descents

### Prove local descents

For every call

$$\bigwedge \text{var}_1, \dots, \text{var}_m. \Gamma_i \Longrightarrow (\text{lhs}_i, \text{rhs}_i) \in ?R$$

we construct new goals and prove them with auto if possible:

$$\bigwedge \text{var}_1, \dots, \text{var}_m. \Gamma_i \Longrightarrow \text{mf}_j(\text{lhs}_i) < \text{mf}_j(\text{rhs}_i)$$

If it fails, we retry to prove the goal with  $\leq$ .

We collect the proof and the information ( $<$ ,  $\leq$ , N oder F) in a matrix.



### 3. step: Search for a global termination ordering

Finding a global termination ordering can be restated as:

Reorder the columns in the matrix in such a way that each row starts with a (possibly empty) sequence of  $\leq$ , followed by a  $<$ .

We now only have to find a suitable permutation for the columns.



### 3. step: Search for a global termination ordering

Finding a global termination ordering can be restated as:

Reorder the columns in the matrix in such a way that each row starts with a (possibly empty) sequence of  $\leq$ , followed by a  $<$ .

We now only have to find a suitable permutation for the columns.

## Algorithm for the search

1. Find a column with at least one  $<$ -entry and else only  $\leq$ -entries.  
Choose this as first column. If there is no such columns the search fails.
2. Remove all rows, where the first column has  $<$ -entries and then remove the column itself. Continue the search on the remaining matrix.
3. The search ends successful when the matrix is empty.



## 4. step: Instantiate the relation and prove the subgoals

### Definition of lexicographic orders

measures ::  $(\alpha \Rightarrow \text{nat}) \text{ list} \Rightarrow (\alpha \times \alpha) \text{ set}$

$(x, y) \in \text{measures } [] = \text{False}$

$(x, y) \in \text{measures } (f\#\text{fs}) = (f\ x < f\ y) \vee ((f\ x = f\ y) \wedge (x, y) \in \text{measures } \text{fs})$



## 4. step: Instantiate the relation and prove the subgoals

From an column order  $\sigma$  construct measures  $[mf_{\sigma_1} , mf_{\sigma_2} \dots mf_{\sigma_k}]$



## Rules for measures

$$\text{wf\_measures: } \frac{}{\text{wf (measures fs)}}$$

$$\text{measures\_less: } \frac{f\ x < f\ y}{(x, y) \in \text{measures (f\#\#fs)}}$$

$$\text{measures\_lesseq: } \frac{f\ x \leq f\ y \quad (x, y) \in \text{measures fs}}{(x, y) \in \text{measures (f\#\#fs)}}$$



## Proving the termination subgoals

The first subgoal (wellfoundedness of  $R$ ) can be shown with `wf_measures`.

For the other subgoals use the following strategy:

For each subgoal inspect the row in the ordered matrix:

For an  $<$ -entry use `measures_less` and the proof in the matrix.

For an  $\leq$ -entry use `measures_lesseq` use the proof in the matrix which results in a new subgoal to continue.



## Structure of Termination Proofs

Shown with an example

General Structure

## Goal of the Termination Tactic

## Algorithm of the Termination Tactic

Create all basic measure functions

Prove local descents

Search for a global termination ordering

Instantiate the relation and prove the subgoals

## Mutual recursion

Necessary extension of the algorithm

Ordering the functions



## Treatment of mutual recursion

The mutual recursive functions are converted to a single function with a sum type

### Example even and odd

even 0 = True

odd 0 = False

even (Suc n) = odd n

odd (Suc n) = even n

is transformed to the following termination goals:

0. wf ?R
1.  $\bigwedge n. (\text{Inr } (n), \text{Inl } (\text{Suc } n)) \in ?R$
2.  $\bigwedge n. (\text{Inl } (n), \text{Inr } (\text{Suc } n)) \in ?R$



## Necessary extension of the algorithm

In the first step also construct measure functions for sum types

$$M(\alpha + \beta) = \{ \text{sum\_case } m_1 \ m_2 \mid m_1 \in M(\alpha); m_2 \in M(\beta) \}$$

The further steps do not need any changes.



## Necessary extension of the algorithm

In the first step also construct measure functions for sum types

$$M(\alpha + \beta) = \{ \text{sum\_case } m_1 \ m_2 \mid m_1 \in M(\alpha); m_2 \in M(\beta) \}$$

The further steps do not need any changes.



## Problem: Too many measure functions

### Example

$f, g, h, i, j :: \text{nat} \Rightarrow \text{nat} \Rightarrow \text{nat}$

$f \ 0 \ m = 0$

$g \ 0 \ m = 0$

$f \ (\text{Suc } n) \ m = g \ m \ n$

$g \ (\text{Suc } n) \ m = f \ m \ n$

$h \ n \ m = 0$

$i \ n \ m = 0$

$j \ n \ m = 0$

- We construct 32 measure functions
- We only need 4 measure functions to look at.
- The necessary proofs are calculated 8 times.



## Ordering the functions

Problem: Local descent has to be shown for every call.

### Example

$$f\ 0 = 0$$

$$f\ (\text{Suc } n) = g\ n$$

$$g\ n = \text{Suc } (f\ n)$$

Solution: Define an order on the functions (here:  $g > f$ )



## Ordering the functions

Problem: Local descent has to be shown for every call.

### Example

$$f\ 0 = 0$$

$$f\ (\text{Suc } n) = g\ n$$

$$g\ n = \text{Suc } (f\ n)$$

Solution: Define an order on the functions (here:  $g > f$ )



## Ordering the functions

Create measure functions of the following form:

$$M(\alpha + \beta) = \{ \text{sum\_case } (\lambda x. 0) m \mid m \in M(\beta) \} \\ \cup \{ \text{sum\_case } m (\lambda x. 0) \mid m \in M(\alpha) \}$$

$$M(\alpha) = \{ \lambda x. 1 \} \quad \text{if } \alpha \text{ is not a sum type}$$

### Example

$$\text{sum\_case } (\lambda x. 0) (\lambda x. 1) \equiv f < g$$

$$\text{sum\_case } (\lambda x. 1) (\lambda x. 0) \equiv g < f$$

With lexicographic combinations every acyclic order on the functions can be captured



## Ordering the functions

Create measure functions of the following form:

$$M(\alpha + \beta) = \{ \text{sum\_case } (\lambda x. 0) m \mid m \in M(\beta) \} \\ \cup \{ \text{sum\_case } m (\lambda x. 0) \mid m \in M(\alpha) \}$$

$$M(\alpha) = \{ \lambda x. 1 \} \quad \text{if } \alpha \text{ is not a sum type}$$

### Example

$$\text{sum\_case } (\lambda x. 0) (\lambda x. 1) \equiv f < g$$

$$\text{sum\_case } (\lambda x. 1) (\lambda x. 0) \equiv g < f$$

With lexicographic combinations every acyclic order on the functions can be captured