Layer Systems for Confluence — Formalized

Bertram Felgenhauer, Franziska Rapp

University of Innsbruck, Allgemeines Rechenzentrum Innsbruck

ICTAC, Stellenbosch 2018-10-16

1supported by FWF project P27528
• automated confluence of first-order term rewrite systems
• certified automated confluence of first-order term rewrite systems
Motivation

Big Picture

- certified automated confluence of first-order term rewrite systems
- here: formalizing layer systems
Term Rewriting

\[ + (x, 0) \rightarrow x \quad + (x, S(y)) \rightarrow S (+ (x, y)) \]

\[ + (0, + (0, S(S(0)))) \]
Term Rewriting

\[ (x, 0) \rightarrow x \]
\[ (x, S(y)) \rightarrow S((x, y)) \]

\[ (0, (0, S(S(0)))) \]
\[ \downarrow \]
\[ (0, S((0, S(0)))) \]
Term Rewriting

\[+(x, 0) \rightarrow x\]

\[+(x, S(y)) \rightarrow S(+(x, y))\]

\[+(0, +(0, S(S(0))))\]

\[\downarrow\]

\[+(0, S(+(0, S(0))))\]

\[\rightarrow\]

\[+(0, S(+(0, 0)))\]

\[\downarrow\]

\[S(+(0, +(0, S(0))))\]
Motivation

Term Rewriting

\[ + (x, 0) \rightarrow x \]

\[ + (x, S(y)) \rightarrow S(+ (x, y)) \]

\[ + (0, + (0, S(S(0)))) \]

\[ \downarrow \]

\[ + (0, S(+ (0, S(0)))) \]

\[ \downarrow \]

\[ S(+ (0, + (0, S(0)))) \]

\[ \downarrow \]

\[ S(S(+ (0, + (0, S(0))))) \]

\[ \downarrow \]

\[ S(S(S(0))) \]
Confluence

Definition

\[ s \leftarrow t \rightarrow u \leftarrow v \]
Confluence

Definition

\[ * \leftarrow \cdot \rightarrow * \subseteq \rightarrow * \cdot * \leftarrow \]

Criteria for TRSs

- orthogonality: left-linear, no critical pairs
- Knuth-Bendix: terminating, joinable critical pairs
- ...
Example

\[
\begin{align*}
\text{@(@(@K, x), y)} & \rightarrow x \\
\text{@(@(@S, x), y), z} & \rightarrow \text{@(@x, z), @y, z)} \\
\text{e(x, x)} & \rightarrow \top
\end{align*}
\]

Orthogonal?

• not left-linear 😞
• no critical pairs 😊
Example

\[ \text{@(@(@K, x), y) } \rightarrow x \quad \text{@(@(@S, x), y), z} \rightarrow \text{@(@x, z), @y, z)} \]
\[ e(x, x) \rightarrow T \]

Orthogonal?

- not left-linear ☹
- no critical pairs 😊

Knuth-Bendix?

- non-terminating 😞

\[ \text{@(@(@S, I), I), @(@S, I))} \rightarrow^+ \text{@(@(@S, I), I), @(@S, I))} \]

where \( I = @(@S, K), K \)
- joinable critical pairs 😊
Motivation

Modularity

Theorem

Let $R_1$, $R_2$ be TRSs over disjoint signatures. Then

$$\text{CR}(R_1 \cup R_2) \iff \text{CR}(R_1) \land \text{CR}(R_2)$$

Example

\[
\begin{align*}
@(@(@K, x), y) & \rightarrow x @(@(@S, x), y), z) \\
& \rightarrow @(@x, z), @y, z)
\end{align*}
\]

- first two rules are orthogonal,
- last rule is terminating, and has no critical pairs,
- disjoint signatures $\Rightarrow$ confluent by modularity,

Bertram Felgenhauer (UIBK)
Layer Systems — Formalized
ICTAC 2018 6/22
Modularity

Theorem

Let $\mathcal{R}_1$, $\mathcal{R}_2$ be TRSs over disjoint signatures. Then

$$\text{CR}(\mathcal{R}_1 \cup \mathcal{R}_2) \iff \text{CR}(\mathcal{R}_1) \land \text{CR}(\mathcal{R}_2)$$

Example

$$\begin{align*}
@(@(@K, x), y) & \rightarrow x \\
@(@(@S, x), y), z) & \rightarrow @(@(@x, z), @y, z) \\
e(x, x) & \rightarrow \top
\end{align*}$$

- The first two rules are orthogonal,
- The last rule is terminating, and has no critical pairs,
- Disjoint signatures $\Rightarrow$ confluent by modularity.
Theorem

Let $\mathcal{R}_1$, $\mathcal{R}_2$ be TRSs over disjoint signatures. Then

$$
\text{CR}(\mathcal{R}_1 \cup \mathcal{R}_2) \iff \text{CR}(\mathcal{R}_1) \land \text{CR}(\mathcal{R}_2)
$$

Example

$$\begin{align*}
@(@(@K, x), y) & \rightarrow x \\
@(@(@(@S, x), y), z) & \rightarrow @(@x, z), @y, z)) \\
e(x, x) & \rightarrow T
\end{align*}$$

• first two rules are orthogonal 😊
Modularity

Theorem

Let $\mathcal{R}_1$, $\mathcal{R}_2$ be TRSs over disjoint signatures. Then

$$\text{CR}(\mathcal{R}_1 \cup \mathcal{R}_2) \iff \text{CR}(\mathcal{R}_1) \land \text{CR}(\mathcal{R}_2)$$

Example

\[
\begin{align*}
@(@(@K, x), y) & \rightarrow x \\
@(@(@S, x), y), z) & \rightarrow @(@x, z), @y, z) \\
e(x, x) & \rightarrow \top
\end{align*}
\]

- first two rules are orthogonal 😊
- last rule is terminating, and has no critical pairs 😊
Modularity

Theorem

Let $\mathcal{R}_1$, $\mathcal{R}_2$ be TRSs over disjoint signatures. Then

$$\text{CR}(\mathcal{R}_1 \cup \mathcal{R}_2) \iff \text{CR}(\mathcal{R}_1) \land \text{CR}(\mathcal{R}_2)$$

Example

$$\begin{align*}
\text{@ (@(K, x), y)} & \rightarrow x \\
\text{@ (@ (@(S, x), y), z)} & \rightarrow \text{@ (@(x, z), @ (y, z))} \\
\text{e}(x, x) & \rightarrow \top
\end{align*}$$

- first two rules are orthogonal 😊
- last rule is terminating, and has no critical pairs 😊
- disjoint signatures $\implies$ confluent by modularity 😊
# Table of Contents

- Motivation
- Layer Systems
- Formalization
- Implementation
Proving Modularity

History

- Toyama 1987
- Klop et al. 1994
- van Oostrom 2008
- ...
Proving Modularity

History

- Toyama 1987
- Klop et al. 1994
- van Oostrom 2008
- ...

Proof idea

- $\implies$ is easy (homogeneous terms are closed under rewriting)
Proving Modularity

History

- Toyama 1987
- Klop et al. 1994
- van Oostrom 2008
- ...

Proof idea

- $\implies$ is easy (homogeneous terms are closed under rewriting)
- decompose terms into maximal top and aliens recursively
Example

\[ \@(@(@K, x), y) \rightarrow x \]
\[ \@(@(@S, x), y), z \rightarrow @(@x, z), @y, z) \]
\[ e(x, x) \rightarrow T \]

- \[ e(@x, e(S, x)), K) \]
Example

\[ @(@(@K, x), y) \rightarrow x \quad @(@(@S, x), y), z) \rightarrow @(@x, z), @y, z) \]
\[ e(x, x) \rightarrow T \]

- \[ e(@x, e(S, x)), K) \]

\[ e(x, x) \rightarrow T \]
Example

\[ \text{@(@(@K, x), y) } \rightarrow x \]
\[ \text{@(@(@S, x), y), z) } \rightarrow @(@x, z), @y, z) \]
\[ \text{e}(x, x) \rightarrow \top \]

- \[ \text{e}(@x, eSx), K) \]

- \text{max-top e(□, □), aliens @}(x, eSx) \text{ and K, rank 4} \]
Proving Modularity

History
- Toyama 1987
- Klop et al. 1994
- van Oostrom 2008
- ...

Proof idea
- $\Rightarrow$ is easy (homogeneous terms are closed under rewriting)
- decompose terms into maximal top and aliens recursively
- use induction on rank
Layer Systems

Proving Modularity

History

- Toyama 1987
- Klop et al. 1994
- van Oostrom 2008
- ...

Proof idea

- \(\Rightarrow\) is easy (homogeneous terms are closed under rewriting)
- decompose terms into maximal top and aliens recursively
- use induction on rank
- ... details are complicated
Related Results

Results

- persistence (Aoto and Toyama 1997)
- layer preservation (Ohlebusch 1994)
- currying (Kahrs 1995)
- ...

Proof idea

- similar to modularity
- different decomposition into max-top and aliens
Related Results

Results

• persistence (Aoto and Toyama 1997)
• layer preservation (Ohlebusch 1994)
• currying (Kahrs 1995)
• ...

Proof idea

• similar to modularity
• different decomposition into max-top and aliens
Layer Systems in a Nutshell

Idea

- layer system $\mathcal{L}$: set of admissible tops
Layer Systems in a Nutshell

Idea

- layer system \( \mathcal{L} \): set of admissible tops
- theorem: if \( \mathcal{R} \) is confluent on \( \mathcal{L} \) then \( \mathcal{R} \) is confluent
Layer Systems in a Nutshell

Idea

- layer system $\mathcal{L}$: set of admissible tops
- theorem: if $\mathcal{R}$ is confluent on $\mathcal{L}$ then $\mathcal{R}$ is confluent
- adapt modularity proof
Layer Systems in a Nutshell

Idea

- layer system $\mathcal{L}$: set of admissible tops
- theorem: if $\mathcal{R}$ is confluent on $\mathcal{L}$ then $\mathcal{R}$ is confluent
- adapt modularity proof

Complications

- max-tops must be unique
Layer Systems in a Nutshell

Idea

• layer system $\mathcal{L}$: set of admissible tops
• theorem: if $\mathcal{R}$ is confluent on $\mathcal{L}$ then $\mathcal{R}$ is confluent
• adapt modularity proof

Complications

• max-tops must be unique
• rewriting must not increase rank
Layer Systems in a Nutshell

Idea

• layer system $\mathcal{L}$: set of admissible tops
• theorem: if $\mathcal{R}$ is confluent on $\mathcal{L}$ then $\mathcal{R}$ is confluent
• adapt modularity proof

Complications

• max-tops must be unique
• rewriting must not increase rank
• several restrictions on fusion
Layer Systems

Layer Systems Definition

Definition

Under the following conditions, the TRS $\mathcal{R}$ is layered wrt $\mathcal{L} \subseteq C(\mathcal{F}, \mathcal{V})$:

$L_1$ Every term in $\mathcal{T}(\mathcal{F}, \mathcal{V})$ has a non-empty top

$L_2$ If $x \in \mathcal{V}$ and $C \in \mathcal{C}$, then $C[x]_p \in \mathcal{L}$ if and only if $C[\square]_p \in \mathcal{L}$

$L_3$ If $L, N \in \mathcal{L}$, $p \in \mathcal{Pos}_\mathcal{F}(L)$, and $L|_p \sqcup N$ is defined then $L[L|_p \sqcup N]_p \in \mathcal{L}$

$W$ If $M$ is the max-top of $s$, $p \in \mathcal{Pos}_\mathcal{F}(M)$, and $s \rightarrow_{p, \ell \rightarrow r} t$ with $\ell \rightarrow r \in \mathcal{R}$, then $M \rightarrow_{p, \ell \rightarrow r} L$ for some $L \in \mathcal{L}$

$C_1$ In $(W)$, either $L$ is the max-top of $t$ or $L = \square$

$C_2$ If $L, N \in \mathcal{L}$ and $L \sqsubseteq N$, then $L[N]_p \in \mathcal{L}$ for any $p \in \mathcal{Pos}_\square(L)$
Layer Systems Definition

Under the following conditions, the TRS $\mathcal{R}$ is layered wrt $\mathcal{L} \subseteq \mathcal{C}(\mathcal{F}, \mathcal{V})$:

1. Every term in $\mathcal{T}(\mathcal{F}, \mathcal{V})$ has a non-empty top.
2. If $x \in \mathcal{V}$ and $C \in \mathcal{C}$, then $C[x] \in \mathcal{L}$ if and only if $C[\Box] \in \mathcal{L}$.
3. If $L, N \in \mathcal{L}$, $p \in \text{Pos} \mathcal{F}(L)$, and $L \mid p \sqcup N$ is defined, then $L[\overline{L} \mid p \sqcup N] \in \mathcal{L}$.
4. If $M$ is the max-top of $s$, $p \in \text{Pos} \mathcal{F}(M)$, and $s \rightarrow p, \ell \rightarrow r$ with $\ell \rightarrow r \in \mathcal{R}$, then $M \rightarrow p, \ell \rightarrow r$ for some $L \in \mathcal{L}$.

DON’T PANIC

it’s formalized
Main result(s)

- $\mathcal{R}$ layered by $\mathcal{L}$ and rank 1 terms confluent $\implies \mathcal{R}$ confluent
- ...
Main result(s)

- $\mathcal{R}$ layered by $\mathcal{L}$ and rank 1 terms confluent $\implies \mathcal{R}$ confluent
- ...

Applications

- modularity: $\mathcal{R}_1 \cup \mathcal{R}_2$ is layered by $\mathcal{C}(\mathcal{F}_1, \mathcal{V}) \cup \mathcal{C}(\mathcal{F}_2, \mathcal{V})$.
- persistence: $\mathcal{R}$ is layered by well-sorted contexts
- currying: $\text{PP}(\mathcal{R})$ is layered by...
Table of Contents

- Motivation
- Layer Systems
- Formalization
- Implementation
Challenges

Software engineering

• interface between confluence result and applications (separation of concerns)
• structuring the big induction proof
Challenges

Software engineering

- interface between confluence result and applications (separation of concerns)
- structuring the big induction proof

Miscellanea

- obvious
- express properties algebraically
- open: nice abstraction for multi-hole contexts
Using Locales

Locales

- bundle assumptions and conclusions (the interface)
- can be instantiated (for applications)
Using Locales

Locales

• bundle assumptions and conclusions (the interface)
• can be instantiated (for applications)
• main result is in \textit{layered} locale
Using Locales

Locales
- bundle assumptions and conclusions (the interface)
- can be instantiated (for applications)
- main result is in *layered* locale
- we also use locales for the induction on rank
Results and Effort

• definitions, basic results about layers 3.2k
• if $R$ is layered by $L$, and $R$ is confluent on $L$, then $R$ is confluent 2.0k
• for disjoint $R_1$ and $R_2$, $R_1 \cup R_2$ is layered by homogeneous terms $\Rightarrow$ modularity 0.8k
• for many-sorted $R$, $R$ is layered by well-typed terms $\Rightarrow$ persistence 1.5k
• for any $R$, $Cu(R)$ is layered by a layer system $\Rightarrow$ preservation of confluence by currying 3.8k
• executable persistent decomposition check for CeTA 0.6k

$\sum 12k$ lines of Isar
Results and Effort

- definitions, basic results about layers 3.2k (20×)
- if $\mathcal{R}$ is layered by $\mathcal{L}$, and $\mathcal{R}$ is confluent on $\mathcal{L}$, then $\mathcal{R}$ is confluent 2.0k (13×)
- for disjoint $\mathcal{R}_1$ and $\mathcal{R}_2$, $\mathcal{R}_1 \cup \mathcal{R}_2$ is layered by homogeneous terms $\implies$ modularity 0.8k (30×)
- for many-sorted $\mathcal{R}$, $\mathcal{R}$ is layered by well-typed terms $\implies$ persistence 1.5k (50×)
- for any $\mathcal{R}$, $\text{Cu}(\mathcal{R})$ is layered by a layer system $\implies$ preservation of confluence by currying 3.8k (40×)
- executable persistent decomposition check for CeTA 0.6k

\[ \sum 12k \text{ lines of Isar} \]
Table of Contents

- Motivation
- Layer Systems
- Formalization
- Implementation
CeTA

- extend CPF format
- formalize persistent decomposition
- define executable code
Implementation

CeTA
- extend CPF format
- formalize persistent decomposition
- define executable code

CSI
- order-sorted persistence was there
- add many-sorted persistence
- add proof output
## Experiments

<table>
<thead>
<tr>
<th></th>
<th>CSI ✓</th>
<th>+pd ✓</th>
<th>CSI</th>
</tr>
</thead>
<tbody>
<tr>
<td>yes</td>
<td>148</td>
<td>154</td>
<td>244</td>
</tr>
<tr>
<td>no</td>
<td>162</td>
<td>162</td>
<td>162</td>
</tr>
<tr>
<td>maybe</td>
<td>127</td>
<td>121</td>
<td>31</td>
</tr>
<tr>
<td>total</td>
<td>437</td>
<td>437</td>
<td>437</td>
</tr>
</tbody>
</table>

http://cl-informatik.uibk.ac.at/software/lisa/ictac2018/
Conclusions

- formalization of layer systems in Isabelle/HOL
- first formalization of Toyama’s theorem
- persistence, currying
- certification for persistence-based decomposition
Contributions

- formalization of layer systems in Isabelle/HOL
- first formalization of Toyama’s theorem
- persistence, currying
- certification for persistence-based decomposition

Future work

- order-sorted persistence
- further applications
- currying is foundation for efficient ground TRS confluence check
Conclusion

Contributions

- formalization of layer systems in Isabelle/HOL
- first formalization of Toyama’s theorem
- persistence, currying
- certification for persistence-based decomposition

Future work

- order-sorted persistence
- further applications
- currying is foundation for efficient ground TRS confluence check

Thanks!
Non-Modularity of Termination

\[
F(0, 1, x) \rightarrow F(x, x, x) \\
h(x, y) \rightarrow x \\
h(x, y) \rightarrow y
\]

\[
F(0, 1, h(0, 1)) \rightarrow F(h(0, 1), h(0, 1), h(0, 1)) \rightarrow F(0, 1, h(0, 1)) \rightarrow \ldots
\]
Related Work — First-Order TRSs

Formalization and certification

- CiME3 – generates Coq scripts (Knuth-Bendix criterion)
- trs – PVS (Knuth-Bendix criterion, orthogonality)
- Ruiz-Reina et al., 2002 – ACL2 (Knuth-Bendix criterion)

Tools

- ACP
- CoLL-Saigawa