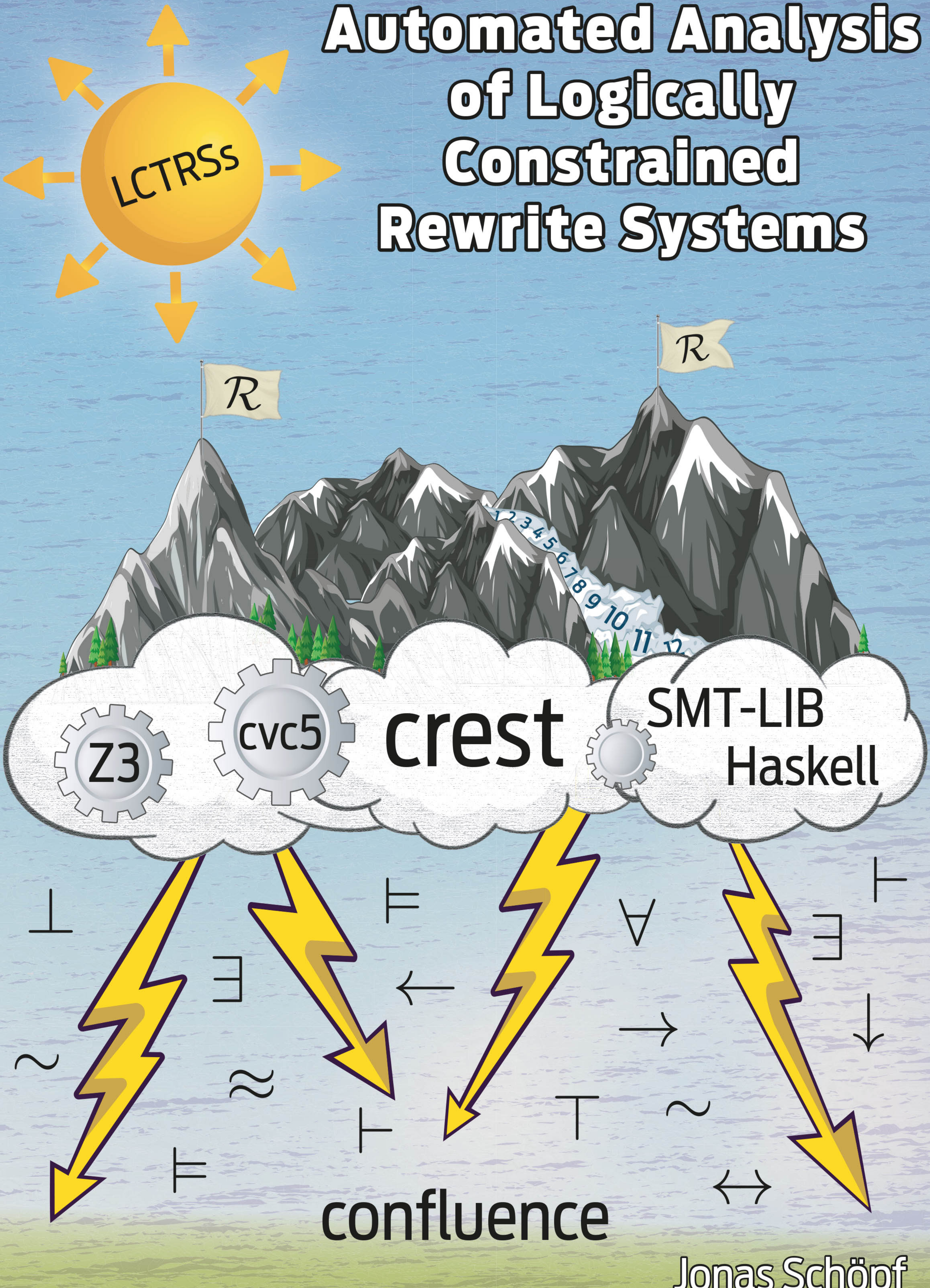
Automated Analysis of Logically Constrained Rewrite Systems



Jonas Schöpf

Automated Analysis of Logically Constrained Rewrite Systems

cumulative dissertation

by

Jonas Schöpf

submitted to the Faculty of Mathematics, Computer Science and Physics of the University of Innsbruck

in partial fulfillment of the requirements for the degree of Doctor of Philosophy (PhD)

advisor: Univ.-Prof. Dr. Aart Middeldorp

Innsbruck, 15 July 2025

universität
innsbruck

cumulative dissertation

Automated Analysis of Logically Constrained Rewrite Systems

Jonas Schöpf (01416055)

15 July 2025

advisor: Univ.-Prof. Dr. Aart Middeldorp

Abstract

Term rewriting serves as a simple but powerful model of computation that is Turing-complete. It offers a rich set of techniques to reason about properties such as termination or confluence, where the latter ensures that the final result is independent of the order in which rewrite rules are applied. Logically constrained term rewriting extends this model by equipping rewrite rules with logical constraints over some theory. This enables built-in support for data structures and thus eases the analysis of real-world code. While plain term rewriting has been extensively studied, logically constrained rewriting—especially confluence analysis—is still in its infancy. In this thesis, we lift several well-known confluence criteria from term rewrite systems to logically constrained rewriting. Additionally, we present the first known non-confluence criterion for logically constrained rewrite systems, along with modular methods that significantly improve the automation of confluence analysis. We also introduce formal semantics for logically constrained rewriting, that serve as the foundation for future research on rewriting induction. Finally, our tool crest presents an automated push-button confluence analysis that implements all of our obtained results.

Acknowledgments

First and foremost, I am deeply grateful to my supervisor, Aart Middeldorp, for always believing in me and my scientific abilities, as well as for providing such a supportive and inspiring environment throughout my PhD. I can not express my gratitude enough. Despite your limited time, you were always there for me whenever I needed help or guidance. Thank you for teaching me how to conduct solid research and for showing me the value of typesetting, teaching, attention to detail, and precision. I am also truly grateful that you shared your love for 日本 with me—it opened my eyes to its beauty and made me fall in love with its culture, too. It was a true pleasure to pursue my PhD under your supervision and I will always vividly remember the many CL group lunches and CL group events.

I would also like to thank Christian Sternagel for inviting me to join the Computational Logic research group in 2017 and for sparking my interest in theoretical computer science research. Honestly, I would not be where I am today without you and I am deeply grateful for that.

I want to thank all the people with whom I shared the office 3M03 during my PhD: Fabian Mitterwallner, Christina Kirk, Johannes Niederhauser, Teppei Saito, Alexander Lochmann and Michael Färber. Thank you all for being such great colleagues. Even on tough days, we always managed to bring some joy into the day. Special thanks go to Fabian Mitterwallner—not only for our collaboration, but also for the countless deep and enjoyable conversations we shared, whether about rewriting, Haskell, Arch Linux, XMonad, Vim, coffee, climbing, and so much more . . . and no, it's not my Arch setup's fault ;). Special thanks also to Johannes Niederhauser—I really appreciated your company at the conferences we attended, and I especially enjoyed the many delicious ラーメン we ate.

I also want to thank the entire ARI team. Special thanks to Nao Hirokawa for leading the Japanese side of the project. I am especially thankful to my collaborators, Takahito Aoto and Naoki Nishida, for warmly welcoming me during my research stays at Niigata University and Nagoya University, and for the many insightful and productive online research meetings. Moreover, I want to thank René Thiemann and Cezary Kaliszyk, who always had an open door for me and provided valuable advice and support whenever I needed it.

I also want to thank Georg Moser for welcoming me to the Theoretical Computer Science research group as a research assistant and for his guidance during those two years. I am also thankful to all the people in the office 2W04 during that time: Albert Nössig, Samuel Frontull, Elisabeth Hell, Monika Steidl and Valentina Golendukhina—for having such a pleasant and supportive office atmosphere. Special thanks go to Martin Avanzini for his helpful insights during that period.

I would like to thank all the people who crossed my path during the years of my PhD, as well as those during my (student) research assistant positions: Sarah Winkler, Manuel Eberl, Martina Ingenhaeff, Gernot Baumgartner, Daniel Ranalter, Liao Zhang, Vincent van Oostrom, Burak Ekici, Max Haslbeck, Yutaka Nagashima, Ralph Bottesch, Florian Meßner, Julian Parsert, and all the others that are not explicitly mentioned above.

Zu guter Letzt möchte ich mich von ganzem Herzen bei meiner Familie bedanken—insbesondere bei meinen Eltern Birgit und Markus, meiner Schwester Stephanie und meiner Freundin Elena. Eure bedingunslose Unterstützung, euer Verständnis und eure Liebe bedeuten mir mehr als ich mit Worte ausdrücken könnte. Ihr habt mich in Zeiten voller Stress, Zweifel und Gedankenversunkenheit erlebt, und mich immer daran erinnert, was wirklich zählt. Ohne euch hätte ich diesen Weg nicht gehen können. Danke, dass ich mich immer auf euch verlassen kann.

Contents

Chapter 1

Introduction

Term rewriting is a computational model that is Turing-complete, i.e., it can be used to simulate any Turing machine. Therefore term rewriting is capable of implementing any algorithm and can thus be used to analyze arbitrary computer programs. Research has been carried out in various directions, including termination analysis, confluence analysis, reachability analysis, rewriting induction, complexity analysis, and rewriting strategies. The results from this research have found numerous applications in areas such as programming language semantics, theorem proving, automated deduction, symbolic computation and compiler optimization. A particular interesting property is *confluence*, which ensures that the final result is independent of the order of the applied computations.

However, the simplicity of term rewriting poses significant challenges when representing standard data structures, such as integers. A recent extension called *logically constrained term rewrite systems* (LCTRSs) tries to overcome this issue by implicitly representing such data structures as background theories. This enables LCTRSs to focus syntactically in their computation rules only on those aspects that are not handled by the theory. As a result, LCTRSs offer a much simpler way to represent real-world programs. For instance, an LCTRS over the theory of integers implicitly supports integer and boolean arithmetic. All of this is achieved by adding logical constraints to the rewrite rules, which must be satisfied before a rule is applied. These constraints range over the background theory and thus, e.g., take care of arithmetic on integers.

Prior to this thesis, only the basic confluence criterion of (weak) orthogonality was known for LCTRSs. Hence the goal of this thesis is to extend the available methods using the rich literature on confluence of term rewriting. The main idea is to adapt well-known criteria from the latter into the setting of LCTRSs. Using this approach we obtain the following sufficient confluence criteria: (1) left-linear strongly closed LCTRSs are confluent, (2) left-linear (almost) parallel closed LCTRSs are confluent, (3) left-linear (almost) development closed LCTRSs are confluent, and (4) left-linear LCTRSs with parallel closed (parallel) critical pairs are confluent. In addition, we develop transformations based on splitting constrained critical pairs and merging constrained rewrite rules, and introduce the first known non-confluence criterion for LCTRSs. We also make a first attempt at defining formal semantics for LCTRSs. All the confluence results are efficiently implemented in an automated analysis, yielding a competitive push-button confluence checker in our tool crest. We are pleased to report that crest won the LCTRS category in the Confluence Competition 2024.

Motivation. My journey into theoretical computer science—particularly term rewriting—began with the Term Rewriting lecture 2017 during my bachelor's studies, taught by Aart Middeldorp and Christian Sternagel. After that course I joined the project "Certification Redux", where we developed a formally verified solver for homogeneous linear Diophantine equations in Isabelle/HOL [52]. For my bachelor's thesis, I extended various termination methods in the automatic termination prover $\mathsf{T_TT_2}$ to restrict the search space toward more specific termination proofs [70]. During my master's thesis, I implemented the weighted path order (WPO)—a powerful termination method for term rewrite systems—into $\mathsf{T_TT_2}$, resulting in the first tool capable of producing machine-checkable certificates for WPO [82]. Before starting my PhD, I briefly worked on adversarial machine learning [61], followed by research on the static complexity analysis of probabilistic programs. All of this work significantly contributed to the motivation to pursue a PhD, which ultimately led to the thesis at hand. The interplay between deep theoretical results and their efficient automated approximation in tools continues to fascinate me.

Related Work. We highlight a selection of different research directions on LCTRSs over the past decade. The first paper on LCTRSs was published by Kop and Nishida in [45]. This was followed by the development of a rewriting induction calculus for LCTRSs [23, 46] and the introduction of Ctrl [47], the first automated tool for analyzing LCTRSs. Basic termination methods were subsequently extended to more advanced techniques in [44], and approaches for non-termination were introduced in [58]. In addition, the LCTRS formalism has been extended to support completion procedures [88] and complexity analysis [89]. All of the aforementioned methods were implemented in Ctrl, except for the latter, which is available in a fork of the complexity analyzer TcT.[1] At CoCo 2024, the tool CRaris[2] made its first appearance and participated in the LCTRS category. It supports the confluence methods (weak) orthogonality and the Knuth-Bendix criterion. Recently, the LCTRS formalism has been extended to higher-order LCTRSs, with methods for termination analysis and program equivalence developed in [26–28]. Additionally, alternative formalisms that are not directly comparable to the original LCTRS framework have been proposed in [12, 13].

Outline. In Chapter 2, we introduce the essential concepts used throughout this thesis, including an overview of LCTRSs, confluence and termination, and SMT solving. Following that, Chapter 3 discusses automation aspects not necessarily covered in later chapters, as well as publications not explicitly included elsewhere in this thesis. The subsequent Chapters 4 to 7 present the peer-reviewed conference papers that form the core of this cumulative thesis. In Chapter 8, we summarize and give an outlook for future work. Appendices A to D contain missing proofs and lemmata that were omitted from the published versions of the papers in Chapters 4 to 7, respectively.

[1] https://github.com/bytekid/tct-lctrs
[2] https://www.trs.css.i.nagoya-u.ac.jp/craris/

Chapter 2

Preliminaries

The intention of this chapter is to give a state-of-the-art introduction to *logically constrained rewrite systems* (LCTRSs) and the most important concepts used in this thesis. However, we ignore concepts that are the same for LCTRSs and *term rewrite systems* (TRSs). To this end, we assume at least some basic knowledge on term rewriting and refer to these books [9, 59, 81] for additional information. The content of later chapters does not necessarily build on this chapter but rather on their respective preliminaries.

2.1 Logically Constrained Term Rewrite Systems

A set of sorts is denoted by $\mathcal{S}$ which can be split into theory ($\mathcal{S}_{\mathsf{th}}$) and term sorts ($\mathcal{S}_{\mathsf{te}}$). The many-sorted set of variables on $\mathcal{S}$ is given by $\mathcal{V}$. For a given sort $\iota \in \mathcal{S}$ we write x^ι if x is of sort ι and $\mathcal{V}^\iota = \{x^\iota \mid x^\iota \in \mathcal{V}\}$ for the set of variables of sort ι. The set of many-sorted function symbols $\mathcal{F}$, also called signature, can be partitioned into the set of term symbols $\mathcal{F}_{\mathsf{te}}$ and theory symbols $\mathcal{F}_{\mathsf{th}}$ such that $\mathcal{F} = \mathcal{F}_{\mathsf{te}} \uplus \mathcal{F}_{\mathsf{th}}$. In this thesis, function symbols that are not theory symbols, such as f, are highlighted in violet. To make the sort annotation associated with a function symbol $f \in \mathcal{F}$ explicit, we write $f\colon \iota_1 \times \cdots \times \iota_n \to \kappa$ for $\iota_1, \ldots, \iota_n, \kappa \in \mathcal{S}$. For a constant symbol v, having no arguments, we simply write $v\colon \kappa$. We further assume a *mapping* $\mathcal{I}$ which assigns the carrier set to every sort $\iota \in \mathcal{S}_{\mathsf{th}}$, and a special *interpretation* $\mathcal{J}$ that assigns a semantic interpretation $f_{\mathcal{J}}\colon \mathcal{I}(\iota_1) \times \cdots \times \mathcal{I}(\iota_n) \to \mathcal{I}(\kappa)$ to every function symbol $f \in \mathcal{F}_{\mathsf{th}}$ with its sort annotation $f\colon \iota_1 \times \cdots \times \iota_n \to \kappa$. This is the key point where the separation of syntax and semantics of LCTRSs is apparent.

Example 2.1. Consider the theory sort $\mathsf{Int} \in \mathcal{S}_{\mathsf{th}}$ and the many-sorted theory symbols

$$\mathsf{succ}\colon \mathsf{Int} \to \mathsf{Int} \qquad\qquad +\colon \mathsf{Int} \times \mathsf{Int} \to \mathsf{Int} \qquad\qquad \mathsf{id}\colon \mathsf{Int} \to \mathsf{Int}$$

for which we assume their usual semantics. Hence we define $\mathcal{I}(\mathsf{Int}) = \mathbb{Z}$ and the interpretation $\mathcal{J}$ as follows:

$$\mathsf{succ}_{\mathcal{J}} := \lambda x.\ x + 1 \qquad\qquad +_{\mathcal{J}} := \lambda x\ y.\ x + y \qquad\qquad \mathsf{id}_{\mathcal{J}} := \lambda x.\ x$$

It is obvious that succ models the successor function, $+$ addition and id the identity function on integers.

For every $\iota \in \mathcal{S}_{\mathsf{th}}$ we have a non-empty set of values $\mathcal{V}\mathsf{al}^{\iota} \subseteq \mathcal{F}_{\mathsf{th}}$, which are constants of sort ι. The set of values $\mathcal{V}\mathsf{al}$ is then defined as $\bigcup_{\iota \in \mathcal{S}_{\mathsf{th}}} \mathcal{V}\mathsf{al}^{\iota}$. Function symbols do not overlap on values, i.e., $\mathcal{F}_{\mathsf{te}} \cap \mathcal{F}_{\mathsf{th}} = \varnothing$, but values are theory symbols: $\mathcal{V}\mathsf{al} \subseteq \mathcal{F}_{\mathsf{th}}$. In this thesis, variables such as x, that are only instantiable by values, are highlighted in orange.

Remark 2.2. The original formulation in [45], and subsequent publications, values were allowed to appear in both $\mathcal{F}_{\mathsf{te}}$ and $\mathcal{F}_{\mathsf{th}}$. However, from our point of view there is no evidence in the literature that it is desirable to allow values in the term part of the signature.

The set $\mathcal{T}(\mathcal{F}, \mathcal{V})$ consists of all correctly sorted terms and elements of $\mathcal{T}(\mathcal{F}_{\mathsf{th}}, \mathcal{V})$ are called logical terms. If a logical term is of sort Bool then it is called a constraint. Here Bool is the boolean sort with its usual elements such that $\mathcal{I}(\mathsf{Bool}) = \mathbb{B} = \{\mathsf{false}, \mathsf{true}\}$.

Example 2.3. Consider the two theory sorts Bool and Int, a many-sorted signature $\mathcal{F}_{\mathsf{te}} = \{\mathsf{f}\colon \mathsf{Bool} \times \mathsf{Bool} \to \mathsf{Bool}, \mathsf{g}\colon \mathsf{Int} \to \mathsf{Bool}\}$, $\mathcal{F}_{\mathsf{th}} = \{\mathsf{true}\colon \mathsf{Bool}, \wedge\colon \mathsf{Bool} \times \mathsf{Bool} \to \mathsf{Bool}\}$, and a set of many-sorted variables $\mathcal{V} = \{x\colon \mathsf{Bool}, y\colon \mathsf{Bool}, z\colon \mathsf{Int}\}$. Using this we obtain

$$\underbrace{\mathsf{f}(x, y) \qquad \mathsf{f}(\mathsf{g}(z), y) \qquad \mathsf{true} \qquad y}_{\in\, \mathcal{T}(\mathcal{F},\mathcal{V})} \qquad\qquad \underbrace{\mathsf{g}(\mathsf{f}(x, y)) \qquad \mathsf{g}(x) \qquad \mathsf{true} \wedge z}_{\notin\, \mathcal{T}(\mathcal{F},\mathcal{V})}$$

where true and y are constraints.

In the following, we omit explicit sorts in signatures if they are irrelevant.

Remark 2.4. Constraints play a central role in LCTRSs, and in order to be able to construct them we need to demand $\mathcal{F}_{\mathsf{th}} \neq \varnothing$. The minimal requirement for this is a value true, a conjunction symbol $\wedge$, and the equality symbol $=$ including their usual semantics. The former is used to represent empty or trivial constraints, while the latter two combine constraints and assign variables within a constraint. Also sorts for these symbols are needed, therefore we further assume $\mathsf{Bool} \in \mathcal{S}_{\mathsf{th}}$. If these requirements are not met then no constraints and consequently no constrained rewrite rules can be constructed.

Logical terms that are ground, i.e., contain no variables, are mapped by the interpretation $\mathcal{J}$ to values by $[\![f(s_1, \ldots, s_n)]\!] = f_{\mathcal{J}}([\![s_1]\!], \ldots, [\![s_n]\!])$. Hence we assume a bijection between values and the elements of the co-domain of $\mathcal{J}$, e.g., in the case of the integers this yields $[\![\mathsf{n}]\!] = n \in \mathcal{I}(\mathsf{Int})$ where $\mathsf{n} \in \mathcal{V}\mathsf{al}^{\mathsf{Int}}$ for $n \in \mathbb{Z}$.

During rewriting we employ *constrained rewrite rules*, written as $\ell \to r\ [\varphi]$, consisting of two terms $\ell, r \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ of the same sort and a constraint φ such that $\ell = f(\ell_1, \ldots, \ell_n)$ for some $f \in \mathcal{F}_{\mathsf{te}}$.

Remark 2.5. Some of the literature [23, 40, 43, 44, 46], instead of having a term symbol as the root symbol on the left-hand side of rules, requires only that the left-hand side of a rule is not allowed to be a logical term.

Implicitly, LCTRSs are equipped with a set of calculation rules, which are rules of the form $f(x_1, \ldots, x_n) \to y\ [f(x_1, \ldots, x_n) = y]$ with a fresh variable y defined for all

$f \in \mathcal{F}_{\mathsf{th}} \setminus \mathcal{V}\mathsf{al}$. The calculation rules are not part of the rules of an LCTRS $\mathcal{R}$, hence we denote the set of all calculation rules by $\mathcal{R}_{\mathsf{ca}}$ and the combination of both sets by $\mathcal{R}_{\mathsf{rc}} = \mathcal{R} \cup \mathcal{R}_{\mathsf{ca}}$.

The *logical variables* of a constrained rewrite rule $\rho\colon \ell \to r\ [\varphi]$ are defined as $\mathcal{V}\mathsf{ar}(\varphi) \cup (\mathcal{V}\mathsf{ar}(r) \setminus \mathcal{V}\mathsf{ar}(\ell))$ and denoted by $\mathcal{LV}\mathsf{ar}(\rho)$. The set of *extra variables*, written as $\mathcal{EV}\mathsf{ar}(\rho)$, are given by $\mathcal{V}\mathsf{ar}(r) \setminus (\mathcal{V}\mathsf{ar}(\ell) \cup \mathcal{V}\mathsf{ar}(\varphi))$. A term s is *linear* if every variable in $\mathcal{V}\mathsf{ar}(s)$ occurs at most once in s. A constrained rewrite rule $\rho\colon \ell \to r\ [\varphi]$ is called left-linear if the variables $\mathcal{V}\mathsf{ar}(\ell) \setminus \mathcal{V}\mathsf{ar}(\varphi)$ occur at most once in ℓ, whereas ρ is linear if in addition the variables $\mathcal{V}\mathsf{ar}(r) \setminus \mathcal{LV}\mathsf{ar}(\rho)$ occur at most once in r. An LCTRS is a set of constrained rewrite rules, implicitly equipped with a set of sorts, a signature, a set of variables, a mapping, an interpretation function, and its respective calculation rules.

Remark 2.6. We want to remark that in Chapter 6 the notation of constrained rules and constrained terms is extended by a set of logical variables. This avoids the necessity to always work explicitly with the notions $\mathcal{LV}\mathsf{ar}$ and $\mathcal{EV}\mathsf{ar}$. However, since no other chapter follows this notation, we will not introduce it in this section.

Example 2.7. Consider the signature and set of variables of Example 2.3 including the theory symbols $=$ and $>$. We construct the following rules where α, β, γ are labels:

$$\alpha\colon \mathsf{f}(x,y) \to y\ [x = y] \qquad \beta\colon \mathsf{f}(x,y) \to \mathsf{g}(z)\ [x > y \wedge y > 1] \qquad \gamma\colon \mathsf{g}(z) \to \mathsf{true}\ [\mathsf{true}]$$

We obtain the following sets of logical and extra variables $\mathcal{LV}\mathsf{ar}(\alpha) = \{x,y\}$, $\mathcal{EV}\mathsf{ar}(\beta) = \{z\}$, $\mathcal{LV}\mathsf{ar}(\beta) = \{x,y,z\}$ and $\mathcal{EV}\mathsf{ar}(\alpha) = \mathcal{EV}\mathsf{ar}(\gamma) = \mathcal{LV}\mathsf{ar}(\gamma) = \varnothing$. The calculation rule induced by the theory symbol $\wedge$ is $\delta\colon x \wedge y \to z\ [x \wedge y = z]$ has $\mathcal{EV}\mathsf{ar}(\delta) = \varnothing$ and $\mathcal{LV}\mathsf{ar}(\delta) = \{x,y,z\}$.

The LCTRS in Example 2.7 is not computationally interesting per se, but its purpose is to illustrate certain idiosyncrasies. Rule α uses two distinct variables in the left-hand side that must be instantiated by the same values due to the constraint; rule β permits arbitrary input (or values) in the right-hand side; and rule γ can be applied completely independently of its constraint.

The following example shows that LCTRSs often allow for modeling computations in a more succinct way than is the case for TRSs. For this purpose consider the following well-known function, used in the Collatz conjecture which is named after the mathematician Lothar Collatz:

$$f(n) = \begin{cases} \frac{n}{2} & \text{if } n \equiv 0 \pmod 2 \\ 3n+1 & \text{if } n \equiv 1 \pmod 2 \end{cases}$$

Note that the input n is assumed to be a positive integer. Modeling such a simple function as a TRS already presents difficulties. The tedious part to model is the underlying integer arithmetic.

Example 2.8. We model the function f as a TRS, thus we represent the (positive) integers using unary notation, i.e., $0, 1, 2, 3, \ldots$ corresponds to $\mathsf{0}, \mathsf{s}(\mathsf{0}), \mathsf{s}(\mathsf{s}(\mathsf{0})), \mathsf{s}(\mathsf{s}(\mathsf{s}(\mathsf{0}))), \ldots$.

We start by defining basic arithmetic such as addition and multiplication,

$$\begin{aligned} \mathsf{add}(0, y) &\to y & \mathsf{mul}(0, y) &\to 0 \\ \mathsf{add}(\mathsf{s}(x), y) &\to \mathsf{s}(\mathsf{add}(x, y)) & \mathsf{mul}(\mathsf{s}(x), y) &\to \mathsf{add}(y, \mathsf{mul}(x, y)) \end{aligned}$$

cut-off subtraction and integer division:

$$\begin{aligned} \mathsf{csub}(0, y) &\to 0 & \mathsf{idiv}(x, y) &\to \mathsf{idiv2}(x, 0, y, \mathsf{geq}(x, y)) \\ \mathsf{csub}(x, 0) &\to x & \mathsf{idiv2}(r, q, d, \bot) &\to q \\ \mathsf{csub}(\mathsf{s}(x), \mathsf{s}(y)) &\to \mathsf{csub}(x, y) & \mathsf{idiv2}(r, q, d, \top) &\to \mathsf{idiv2}(\mathsf{csub}(r, d), \mathsf{s}(q), d, \\ & & &\qquad \mathsf{geq}(\mathsf{csub}(r, d), d)) \end{aligned}$$

Then we define the greater-equal comparison,

$$\begin{aligned} \mathsf{geq}(0, 0) &\to \top & \mathsf{geq}(\mathsf{s}(x), 0) &\to \top \\ \mathsf{geq}(\mathsf{s}(x), \mathsf{s}(y)) &\to \mathsf{geq}(x, y) & \mathsf{geq}(0, \mathsf{s}(y)) &\to \bot \end{aligned}$$

a predicate to test for even positive integers,

$$\begin{aligned} \mathsf{isEven}(0) &\to \top & \mathsf{isEven}(\mathsf{s}(0)) &\to \bot \\ \mathsf{isEven}(\mathsf{s}(\mathsf{s}(x))) &\to \mathsf{isEven}(x) \end{aligned}$$

and finally the main function using the function symbol f:

$$\begin{aligned} \mathsf{f}(\mathsf{s}(x)) &\to \mathsf{seq}(\mathsf{s}(x), \mathsf{isEven}(\mathsf{s}(x))) & \mathsf{seq}(n, \top) &\to \mathsf{idiv}(n, \mathsf{s}(\mathsf{s}(0))) \\ & & \mathsf{seq}(n, \bot) &\to \mathsf{s}(\mathsf{mul}(\mathsf{s}(\mathsf{s}(\mathsf{s}(0))), n)) \end{aligned}$$

Note that f is only defined on positive integers in the TRS. Now assume an LCTRS over the integer theory (this includes usual integer functions as specified in the SMT-LIB format[1]). This allows us to encode the same behavior as in the TRS above by using only two rules.

$$\begin{aligned} \mathsf{f}(n) &\to \mathsf{div}(n, 2) & &[\mathsf{mod}(n, 2) = 0 \land n > 0] \\ \mathsf{f}(n) &\to (3 * n) + 1 & &[\mathsf{mod}(n, 2) = 1 \land n > 0] \end{aligned}$$

The LCTRS is much more concise and also more readable. The reason is that boolean and integer arithmetic is transferred to the constraints and the semantic part of the LCTRS.

The previous example shows that certain computations in LCTRSs are handled by the theory. An important notion to ensure that the constraint is instantiated with the correct values during rewriting is called *respects*. A substitution σ respects a constraint φ, written as $\sigma \vDash \varphi$, if $\sigma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$ and the constraint is valid, i.e., $[\![\varphi]\!] = \mathsf{true}$. A substitution σ respects a constrained rewrite rule $\rho\colon \ell \to r\ [\varphi]$ if $\sigma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$ and $[\![\varphi\sigma]\!] = \mathsf{true}$. The former notion requires the domain of the substitution σ to satisfy $\mathcal{D}\mathsf{om}(\sigma) \subseteq \mathcal{V}\mathsf{ar}(\varphi)$, whereas the latter requires $\mathcal{D}\mathsf{om}(\sigma) \subseteq \mathcal{V}\mathsf{ar}(\ell) \cup \mathcal{V}\mathsf{ar}(r) \cup \mathcal{V}\mathsf{ar}(\varphi)$. We proceed with the rewrite relation for LCTRSs.

[1] https://smt-lib.org/theories-Ints.shtml

Remark 2.9. Note that this definition differs from the initial one in [45] where the rewrite relation is separated into a calculation part and a rule part. The motivation behind this change lies in the simplification of several proofs, which is discussed in Chapter 5.

Definition 2.10. A term s and a constrained rewrite rule $\rho\colon \ell \to r\ [\varphi]$ admit the rewrite step $s[\ell\sigma]_p \to s[r\sigma]_p$ if there exists a substitution σ and a position p such that $s|_p = \ell\sigma$ and $\sigma \vDash \rho$.

The ultimate goal for LCTRSs is to analyze their specific properties, however, for this purpose rewriting terms is not always convenient. If we want to show that, for the constrained rewrite rules β and γ from Example 2.7, after applying β subsequently also γ applies, then we must show this for all terms of the form $\mathsf{f}(x, y)$ where x and y are instantiated such that $x > y \wedge y > 1$ holds. During the analysis of LCTRSs similar questions, e.g., whether a set of multiple instances of a term rewrites to another set of instances, naturally arise. Clearly, rewriting infinitely many terms is not in our interest, especially with an automated analysis as ultimate goal. To tackle these issues, the concept of rewriting constrained terms was introduced in [45].

A *constrained term* is a tuple $s\ [\varphi]$ consisting of a term s and constraint φ. Intuitively, a constrained term can be seen as the set of all instances of this term that satisfy the constraint, i.e., for $s\ [\varphi]$ the set of instances is $\{s\sigma \mid \sigma \vDash \varphi\}$. Two constrained terms can represent the same set of instances modulo renaming of variables with a term sort, e.g., $\mathsf{f}(x, y)\ [x \geqslant 1]$ and $\mathsf{f}(x, z)\ [x > 0]$ represent the same sets of instances modulo renaming of y and z. We denote the *equivalence* of two constrained terms by $s\ [\varphi] \sim t\ [\psi]$ and define it as follows:

1. for all substitutions σ with $\sigma \vDash \varphi$ there exists a substitution γ with $\gamma \vDash \psi$ such that $s\sigma = t\gamma$, and

2. for all substitutions γ with $\gamma \vDash \psi$ there exists a substitution σ with $\sigma \vDash \varphi$ such that $t\gamma = s\sigma$.

In other words, s and t are only allowed to differ at positions with values or variables of a theory sort, and at positions with variables of a term sort such that between the latter position a renaming exists.

Definition 2.11. For a constrained term $s\ [\varphi]$ and a constrained rewrite rule $\rho\colon \ell \to r\ [\psi] \in \mathcal{R}_{\mathsf{rc}}$ we obtain the rewrite step $s[\ell\sigma]_p\ [\varphi] \to s[r\sigma]_p\ [\varphi]$ for a substitution σ and a position p if

- $s|_p = \ell\sigma$,
- $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}\mathsf{ar}(\varphi)$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$,
- φ is satisfiable, and
- $\varphi \Rightarrow \psi\sigma$ is valid.

The final rewrite relation $\xrightarrow{\sim}$ is then defined as $\sim \cdot \to \cdot \sim$.

The third item ensures that there exists at least one instance for the constrained term while the fourth item ensures that the applied rule covers all instances induced by the constraint. Let us illustrate this with a concrete example.

Example 2.12. Consider the constrained rewrite rules from Example 2.7 and the constrained term $\mathsf{f}(x_1, x_2)\ [\varphi]$ with $\varphi := x_1 > x_2 \land x_2 > 1$. The following rewrite steps are possible:

$$\begin{aligned}\mathsf{f}(x_1, x_2)\ [\varphi] \sim \mathsf{f}(x_1, x_2)\ [\varphi \land z_1 = z_1] &\to \mathsf{g}(z_1)\ [\varphi \land z_1 = z_1] \\ &\to \mathsf{true}\ [\varphi \land z_1 = z_1] \sim \mathsf{true}\ [\mathsf{true}]\end{aligned}$$

We are now able to rewrite sets of terms, represented by constrained terms. In later chapters, it is apparent that rewriting constrained terms becomes very useful and is often crucial for performing an analysis. So far we have only seen single rewrite steps on (constrained) terms. In the following we introduce the parallel and multi-step rewrite relations on (constrained) terms. These are crucial for various confluence criteria presented in Chapters 4 and 5.

Definition 2.13. Let $\mathcal{R}$ be an LCTRS. The parallel rewrite relation $\mathrel{+\!\!\!\!\!\!\!\to}$ on terms is inductively defined as follows: (1) $x \mathrel{+\!\!\!\!\!\!\!\to} x$ for all variables x, (2) $f(s_1, \ldots, s_n) \mathrel{+\!\!\!\!\!\!\!\to} f(t_1, \ldots, t_n)$ if $s_i \mathrel{+\!\!\!\!\!\!\!\to} t_i$ with $1 \leqslant i \leqslant n$, (3) $\ell\sigma \mathrel{+\!\!\!\!\!\!\!\to} r\sigma$ if $\ell \to r\ [\varphi] \in \mathcal{R}_{\mathsf{rc}}$ and $\sigma \vDash \ell \to r\ [\varphi]$.

This allows us to contract redexes at parallel positions similar to parallel rewriting for TRSs. In the following definition we define parallel rewriting on constrained terms.

Definition 2.14. Let $\mathcal{R}$ be an LCTRS. The parallel rewrite relation $\mathrel{+\!\!\!\!\!\!\!\to}$ on constrained terms is inductively defined as follows:

1. $x\ [\varphi] \mathrel{+\!\!\!\!\!\!\!\to} x\ [\varphi]$ for all variables x,
2. $f(s_1, \ldots, s_n)\ [\varphi] \mathrel{+\!\!\!\!\!\!\!\to} f(t_1, \ldots, t_n)\ [\varphi]$ if $s_i\ [\varphi] \mathrel{+\!\!\!\!\!\!\!\to} t_i\ [\varphi]$ for $1 \leqslant i \leqslant n$,
3. $\ell\sigma\ [\varphi] \mathrel{+\!\!\!\!\!\!\!\to} r\sigma\ [\varphi]$ if $\rho\colon \ell \to r\ [\psi] \in \mathcal{R}_{\mathsf{rc}}$, $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}\mathsf{ar}(\varphi)$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$, φ is satisfiable, $\varphi \Rightarrow \psi\sigma$ is valid.

The relation $\overset{\sim}{\mathrel{+\!\!\!\!\!\!\!\to}}$ on constrained terms is defined as $\sim \cdot \mathrel{+\!\!\!\!\!\!\!\to} \cdot \sim$.

Note that within the definition of parallel rewriting we do not allow any equivalence step, i.e., only before or after the parallel step. This decision is inspired by the single-step rewrite relation on constrained terms. Before we introduce multi-step rewriting, let us see parallel rewriting in action.

Example 2.15. Consider the LCTRS $\mathcal{R}$ with the rules from Example 2.7 over the theory of integers. Starting from the constrained term $\mathsf{f}(2 + x, x + 2)\ [x \geqslant 0]$ we obtain the rewrite sequence

$$\begin{aligned}\mathsf{f}(2 + x, x + 2)\ [x \geqslant 0] &\overset{\sim}{\mathrel{+\!\!\!\!\!\!\!\to}} \mathsf{f}(z, z')\ [x \geqslant 0 \land z = 2 + x \land z' = x + 2] \\ &\overset{\sim}{\mathrel{+\!\!\!\!\!\!\!\to}} z'\ [x \geqslant 0 \land z' = x + 2]\end{aligned}$$

Definition 2.16. Let $\mathcal{R}$ be an LCTRS. The multi-step relation $\twoheadrightarrow\!\!\!\!\!\!\circ\;\,$ on terms is defined inductively as follows: (1) $x \twoheadrightarrow\!\!\!\!\!\!\circ\;\, x$ for all variables x, (2) $f(s_1, \dots, s_n) \twoheadrightarrow\!\!\!\!\!\!\circ\;\, f(t_1, \dots, t_n)$ if $s_i \twoheadrightarrow\!\!\!\!\!\!\circ\;\, t_i$ with $1 \leqslant i \leqslant n$, (3) $\ell\sigma \twoheadrightarrow\!\!\!\!\!\!\circ\;\, r\tau$ if $\ell \to r\ [\varphi] \in \mathcal{R}_{\mathsf{rc}}$, $\sigma \vDash \ell \to r\ [\varphi]$ and $\sigma \twoheadrightarrow\!\!\!\!\!\!\circ\;\, \tau$, where $\sigma \twoheadrightarrow\!\!\!\!\!\!\circ\;\, \tau$ denotes $\sigma(x) \twoheadrightarrow\!\!\!\!\!\!\circ\;\, \tau(x)$ for all variables $x \in \mathcal{D}\mathsf{om}(\sigma)$.

We naturally extend this to constrained terms as follows.

Definition 2.17. Let $\mathcal{R}$ be an LCTRS. The multi-step relation $\twoheadrightarrow\!\!\!\!\!\!\circ\;\,$ on constrained terms is defined inductively as follows:

1. $x\ [\varphi] \twoheadrightarrow\!\!\!\!\!\!\circ\;\, x\ [\varphi]$ for all variables x,
2. $f(s_1, \dots, s_n)\ [\varphi] \twoheadrightarrow\!\!\!\!\!\!\circ\;\, f(t_1, \dots, t_n)\ [\varphi]$ if $s_i\ [\varphi] \twoheadrightarrow\!\!\!\!\!\!\circ\;\, t_i\ [\varphi]$ for $1 \leqslant i \leqslant n$,
3. $\ell\sigma\ [\varphi] \twoheadrightarrow\!\!\!\!\!\!\circ\;\, r\tau\ [\varphi]$ if $\rho\colon \ell \to r\ [\psi] \in \mathcal{R}_{\mathsf{rc}}$, $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}\mathsf{ar}(\varphi)$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$, φ is satisfiable, $\varphi \Rightarrow \psi\sigma$ is valid, and $\sigma\ [\varphi] \twoheadrightarrow\!\!\!\!\!\!\circ\;\, \tau\ [\varphi]$.

Here $\sigma\ [\varphi] \twoheadrightarrow\!\!\!\!\!\!\circ\;\, \tau\ [\varphi]$ denotes $\sigma(x)\ [\varphi] \twoheadrightarrow\!\!\!\!\!\!\circ\;\, \tau(x)\ [\varphi]$ for all variables $x \in \mathcal{D}\mathsf{om}(\sigma)$. The relation $\tilde{\twoheadrightarrow\!\!\!\!\!\!\circ}\;\,$ on constrained terms is defined as $\sim \cdot \twoheadrightarrow\!\!\!\!\!\!\circ\;\, \cdot \sim$.

Example 2.18. Consider the LCTRS $\mathcal{R}$ and the rewrite step from Example 2.15. This step can be performed using a single multi-step:

$$\mathsf{f}(\mathsf{2} + x, x + \mathsf{2})\ [x \geqslant \mathsf{0}] \;\tilde{\twoheadrightarrow\!\!\!\!\!\!\circ}\;\; z'\ [x \geqslant \mathsf{0} \wedge z' = x + \mathsf{2}]$$

2.2 Confluence and Termination

We are now ready to define constrained (parallel) critical pairs which are the key ingredient for the confluence analysis in later chapters. We write $\mathcal{EC}_\rho$ for the constraint $\bigwedge\{x = x \mid x \in \mathcal{EV}\mathsf{ar}(\rho)\}$ for a given constrained rewrite rule ρ.

Definition 2.19. An *overlap* of an LCTRS $\mathcal{R}$ is a triple $\langle \rho_1, p, \rho_2 \rangle$ with rules $\rho_1\colon \ell_1 \to r_1\ [\varphi_1]$ and $\rho_2\colon \ell_2 \to r_2\ [\varphi_2]$, satisfying the following conditions:

1. ρ_1 and ρ_2 are variable-disjoint variants of rewrite rules in $\mathcal{R}_{\mathsf{rc}}$,
2. $p \in \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell_2)$,
3. ℓ_1 and $\ell_2|_p$ unify with mgu σ such that $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2)$,
4. $\varphi_1\sigma \wedge \varphi_2\sigma$ is satisfiable, and
5. if $p = \epsilon$ then ρ_1 and ρ_2 are not variants, or $\mathcal{V}\mathsf{ar}(r_1) \nsubseteq \mathcal{V}\mathsf{ar}(\ell_1)$.

We call $\ell_2\sigma[r_1\sigma]_p \approx r_2\sigma\ [\varphi_1\sigma \wedge \varphi_2\sigma \wedge \psi\sigma]$ a *constrained critical pair* (CCP) obtained from the overlap $\langle \rho_1, p, \rho_2 \rangle$ with $\psi = \mathcal{EC}_{\rho_1} \wedge \mathcal{EC}_{\rho_2}$. The peak $\ell_2\sigma[r_1\sigma]_p\ [\Phi] \leftarrow \ell_2\sigma\ [\Phi] \to_\epsilon r_2\sigma\ [\Phi]$ with $\Phi = (\varphi_1 \wedge \varphi_2 \wedge \psi)\sigma$, from which the constrained critical pair originates, is called a *constrained critical peak*. The set $\mathsf{CCP}(\mathcal{R})$ consists of all constrained critical pairs of $\mathcal{R}$.

In order to check that the left- and right-hand side of a CCP are indeed the same, it does not suffice to check syntactic equality. To this end we introduce the notion of triviality: A constrained critical pair $s \approx t\ [\varphi]$ is *trivial* if $s\sigma = t\sigma$ for every substitution σ with $\sigma \vDash \varphi$. In the following we define constrained parallel critical pairs.

Definition 2.20. Let $\mathcal{R}$ be an LCTRS, $\rho\colon \ell \to r\ [\varphi]$ a rule in $\mathcal{R}_{\mathsf{rc}}$, and $P \subseteq \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell)$ a non-empty set of parallel positions. For every $p \in P$ let $\rho_p\colon \ell_p \to r_p\ [\varphi_p]$ be a variant of a rule in $\mathcal{R}_{\mathsf{rc}}$. Let $\psi = \mathcal{EC}_\rho \wedge \bigwedge_{p \in P} \mathcal{EC}_{\rho_p}$ and $\Phi = \varphi\sigma \wedge \psi\sigma \wedge \bigwedge_{p \in P} \varphi_p\sigma$. The peak $\ell\sigma[r_p\sigma]_{p \in P}\ [\Phi] \mathrel{\leftarrow\!\!\!\Vdash} \ell\sigma\ [\Phi] \to_{\epsilon,\mathcal{R}} r\sigma\ [\Phi]$ forms a *constrained parallel critical pair* (CPCP) $\ell\sigma[r_p\sigma]_{p \in P} \approx r\sigma\ [\Phi]$ if the following conditions are satisfied:

1. $\mathcal{V}\mathsf{ar}(\rho_1) \cap \mathcal{V}\mathsf{ar}(\rho_2) = \varnothing$ for different rules ρ_1 and ρ_2 in $\{\rho\} \cup \{\rho_p \mid p \in P\}$,
2. σ is an mgu of $\{\ell_p = \ell|_p \mid p \in P\}$ such that $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho) \cup \bigcup_{p \in P} \mathcal{LV}\mathsf{ar}(\rho_p)$,
3. $\varphi\sigma \wedge \bigwedge_{p \in P} \varphi_p\sigma$ is satisfiable,
4. if $P = \{\epsilon\}$ then ρ_ϵ is not a variant of ρ or $\mathcal{V}\mathsf{ar}(r) \nsubseteq \mathcal{V}\mathsf{ar}(\ell)$.

A constrained peak forming a constrained parallel critical pair is called a *constrained parallel critical peak*. The set $\mathsf{CPCP}(\mathcal{R})$ consists of all constrained parallel critical pairs of $\mathcal{R}$.

The following example demonstrates the computation of constrained (parallel) critical pairs.

Example 2.21. The rules from Example 2.7 have no overlaps and hence no constrained critical pairs. Therefore we consider the LCTRS $\mathcal{R}$ over the theory of integers with the rules

$$\alpha\colon \mathsf{f}(x + y, \mathsf{g}(x)) \to z\ [x > y \wedge y > z] \qquad\qquad \beta\colon \mathsf{g}(x) \to x\ [\mathsf{true}]$$

There exist the three overlaps $\langle \alpha, \epsilon, \alpha\rangle$, $\langle \beta, 2, \alpha\rangle$, $\langle \gamma, 1, \alpha\rangle$, where $\gamma\colon x + y \to z\ [z = x + y]$. The set $\mathsf{CCP}(\mathcal{R})$ consists of the three constrained critical pairs:

$$\begin{aligned} z' &\approx z\ [x > y \wedge y > z \wedge x > y \wedge y > z' \wedge z = z \wedge z' = z'] \\ \mathsf{f}(x + y, x) &\approx z\ [x > y \wedge y > z \wedge \mathsf{true} \wedge z = z] \\ \mathsf{f}(z', \mathsf{g}(x)) &\approx z\ [x > y \wedge y > z \wedge z' = x + y \wedge z = z \wedge z' = z'] \end{aligned}$$

The set $\mathsf{CPCP}(\mathcal{R})$ contains all of $\mathsf{CCP}(\mathcal{R})$ and additionally the constrained parallel critical pair $\mathsf{f}(z', x) \approx z\ [x > y \wedge y > z \wedge z' = x + y \wedge \mathsf{true} \wedge z = z \wedge z' = z']$ using the rules β and γ on the parallel positions 2 and 1 to obtain the left-hand side, respectively.

To conclude this section, we recall the properties *termination* and *confluence* in connection with LCTRSs. In general, checking whether one of these properties holds is undecidable for both. Similar to TRSs, an LCTRS is terminating whenever it does not admit an infinite rewrite sequence. To prove termination, it suffices to check that

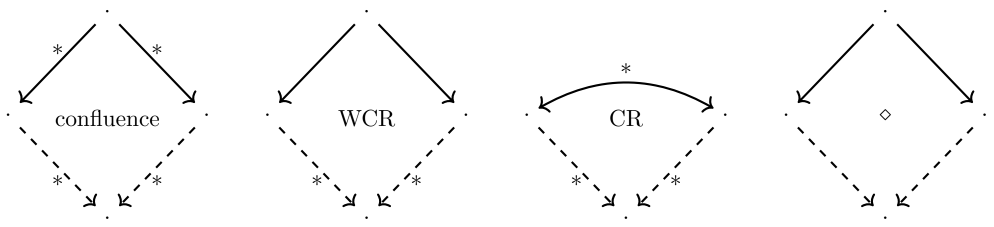


Figure 2.1: Properties related to confluence.

each constrained rewrite rule of an LCTRS is strictly decreasing from left to right by a well-founded order. There exist several methods to prove termination of LCTRSs, e.g., the recursive path ordering [45, Section 6.2] or the value criterion [44, Section 5].

Confluence denotes that for all terms s, t and u we have $t \downarrow u$ whenever they originate from the same peak $t \mathrel{{}^*{\leftarrow}} s \to^* u$. In Figure 2.1 we see the diagrams of the confluence property, the weak Church-Rosser property (WCR) also known as local confluence, the Church-Rosser property (CR) and the diamond property ($\diamond$). Note that the confluence property coincides with the Church-Rosser property, whereas the diamond property implies strong confluence ($\leftarrow \cdot \to \; \subseteq \; \to^* \cdot {}^{=}{\leftarrow}$) which further implies confluence. Local confluence plays an important role for the Knuth-Bendix criterion, which states that local confluence implies confluence for terminating (LC)TRSs. While this is a decidable property for TRSs, we discuss in Chapter 5 its undecidability for LCTRSs. In later chapters of this thesis several sufficient (parallel) critical pair closing criteria for confluence of LCTRSs are presented.

It is important to note that not all concepts from plain TRSs are readily transferable to the constrained setting, especially in the presence of constrained terms. Notably, this includes joinability between two constrained terms and the normal form property of a constrained term. The former is discussed in Chapter 4 and the latter in Chapter 7.

2.3 SMT Solving

At this point it is obvious that the key difference between LCTRSs and plain TRSs lies in the use of logical constraints. In general, LCTRSs do support arbitrary theories for these constraints. Therefore LCTRSs do not necessarily have tools supporting them, e.g., for checking validity of constraints. The ultimate goal, however, is to enable automated analysis for theories where satisfiability and validity of constraints can be checked using existing tools.

A natural choice that exactly provides this is satisfiability modulo theories (SMT) solving. Its key idea is to check whether specific first-order logic formulas are satisfiable. This can be seen as an extension of the boolean satisfiability problem (SAT) to more complex formulas involving data structures such as integers or bit vectors. SAT is well-known to be NP-complete, while solving an SMT problem is for most theories undecidable and in general NP-hard. SMT solvers are tools that given an SMT formula check whether it is satisfiable and, if possible, provide an assignment as witness. In the last decade, a lot of research in this direction was conducted and from this several powerful SMT solvers emerged, e.g., Z3 [17] and cvc5 [10]. Many of these SMT solvers support a common input format, called the SMT-LIB[2] format. This format supports common theories such as Ints, Reals or FixedSizeBitVectors. Let us demonstrate this with an example.

Example 2.22. To compute the first constrained critical pair in Example 2.21, we need to check that $x > y \wedge y > z \wedge x > y \wedge y > z'$ is satisfiable. We can encode this using the following SMT-LIB expression:

```
(declare-const x Int)
(declare-const y Int)
(declare-const z Int)
(declare-const z1 Int)
(assert (and (> x y) (> y z)))
(assert (and (> x y) (> y z1)))
```

Appending the expression `(check-sat)` instructs an SMT solver to check for satisfiability. The SMT-solver Z3 returns in this case `sat` as this formula is satisfiable. In order to obtain a witness, adding the expression `(get-model)` returns the following satisfying assignment:

```
(
  (define-fun x () Int
    1)
  (define-fun y () Int
    0)
  (define-fun z () Int
    (- 1))
  (define-fun z1 () Int
    (- 1))
)
```

It is easy to see that this assignment satisfies the constraint. To check validity, we check that the negation of the constraint is unsatisfiable.

For this reason tools that automatically analyze LCTRSs usually only support theories that have an off-the-shelf checker for satisfiability available, and support the SMT-LIB format. In the next chapter we see that the ARI format heavily relies on the latter.

[2] https://smt-lib.org/

Chapter 3

Further Contributions

In this chapter, we discuss contributions that were not included as standalone chapters in this thesis. This includes the development of the ARI format and the ARI database, automation aspects of crest, and other (informal) publications. Before the first section, we briefly revisit the main research questions that laid the foundation for this thesis:

1. **Which confluence criteria for LCTRSs beyond orthogonality can be adapted to the constrained setting?** This led to the adaptation of several confluence criteria to LCTRSs, including strong closedness [64], (almost) parallel closedness [64], (almost) development closedness [68], and parallel closedness of constrained parallel critical pairs [68]. Moreover, we have recently introduced modular methods for confluence analysis [66,67] and the first known non-confluence criterion for LCTRSs [66].

2. **How to obtain decidable approximations of the confluence criteria and implement them to enable an automated analysis?** All methods mentioned in the previous item, except the reduction method, are implemented in our tool crest [66]. We developed suitable approximations to automate the results, which ultimately enabled our participation in the LCTRS category of CoCo 2024 where we took first place.

3. **How can existing research on the rewriting induction calculus for terminating LCTRSs be extended to non-terminating systems?** This is the only research question that was not directly solved. During our first research stay in Japan in 2023, we discussed with Takahito Aoto—an expert in rewriting induction for TRSs—the possibility of developing rewriting induction methods for non-terminating LCTRSs. In these discussions, we also recognized a more fundamental issue: there is a lack of semantics for LCTRSs. These insights led to the work presented in [4], where we investigate the semantic side of LCTRSs.

4. **What input format is most appropriate to represent LCTRSs? How can a database of interesting confluence problems be created? Can this form the basis for establishing a new LCTRS competition category in the CoCo?** As discussed in the following section, we developed a new format for rewrite systems [3], which includes a format for LCTRSs. The current LCTRSs in the ARI database were taken from the literature over the past decade. Together, these developments enabled the establishment of an LCTRS category for CoCo.

3.1 Automation

This section gives a brief overview on automation aspects. For this reason we briefly discuss the newly developed ARI (input) format, the creation of the ARI database and we also comment on the main reason of the development of crest.

The main idea behind LCTRSs is to ease the analysis of real-world code. As mentioned in the previous chapter, standard data structures found in modern programming languages—such as integers or floating point numbers—are tedious to represent in plain TRSs, as they must be encoded syntactically. Since such data structures are omnipresent, the idea behind LCTRSs is to include them directly in the underlying theory. Moreover, the theory incorporates default properties of these structures, such as in the case of integers the commutativity of addition.

Kop and Nishida started the development of the Constrained Term Rewriting tooL (for short Ctrl) [46, 47] more than a decade ago. It was the first tool that was capable of performing an automated analysis on LCTRSs. It supports basic confluence analysis [45], termination analysis [44], program equivalence via rewriting induction [23], completion [88] and non-termination analysis [58]. Initially we planned to extend Ctrl with our results. However, due to the lack of maintenance, limited multicore support in OCaml at that time, and differing design preferences, we decided to start the development of our own tool.

ARI Format and ARI Database

The Austria-Japan joint project ARI[1] (Automation of Rewriting Infrastructure) developed a new format for rewrite systems [3], intended as the successor of the old COPS format.[2] This format includes all relevant categories of the Confluence Competition (CoCo) such as TRSs, conditional TRSs (CTRSs), context-sensitive TRSs (CSTRSs), context-sensitive conditional (CSCTRSs) and many-sorted TRSs (MSTRSs). Notably, it introduces the first official format for LCTRSs. Moreover, it incorporates a compact format of multiple TRSs used for commutation problems, as well as a format for infeasibility problems. Currently, higher-order systems are not supported. The LCTRS format[3] is divided into two parts: (1) the general format and (2) a restriction for the ARI database. In the following we highlight interesting aspects of the LCTRS format.

In (1) we define the general format that covers LCTRSs over an arbitrary theory, following the syntax of the SMT-LIB format. The annotation `:smtlib` specifies the used SMT-LIB standard. However, it is also possible to employ a different theory having a different syntax, because `formula` does not need to be an ARI term. If theory symbols are used in left- or right-hand sides of rules, they must conform to the term syntax. Sorts are divided into term and theory sorts, since `identifier` does not permit whitespace characters. This would fail for sorts of bit vectors, e.g., `(_ BitVec 32)`, which do contain whitespace characters. To avoid repetition of constraints, we introduce the `smt-def`

[1] https://ari-informatik.uibk.ac.at/
[2] https://project-coco.uibk.ac.at/problems/index.php
[3] https://project-coco.uibk.ac.at/ARI/lctrs.php

```
(format LCTRS :smtlib 2.6)
(theory Ints)
(define-fun isEven ((x Int)) Bool (= (mod x 2) 0))
(fun f (-> Int Int))
(rule (f n) (div n 2) :guard (and (isEven n) (> n 0)))
(rule (f n) (+ (* 3 n) 1) :guard (and (not (isEven n)) (> n 0)))
```

Figure 3.1: LCTRS of Example 2.8 in ARI format.

construct, which allows us to define custom predicates. An example predicate, `isEven`, which checks whether a given integer is even, is shown in Figure 3.1. Additionally, we require that the sorts of all variables in rules are inferable from their context. It is important to note that a fully sorted LCTRS is always required for the analysis, as SMT solvers must know the sort of each variable.

In (2) we further restrict the format to ease its use within the ARI database. The ARI database[4] is the successor of the now outdated COPS database.[5] Both these databases are a collection of challenging and interesting confluence problems across the various categories of CoCo. All non-duplicate problems of the COPS database are included in the ARI database. The current LCTRSs in the latter are taken from the literature and benchmark set of Ctrl. When a new problem is submitted to the ARI database, it is first checked for syntactic correctness with respect to the ARI format and for duplication with existing LCTRSs in the database. If both checks pass, it is accepted into the database. Specifically for LCTRSs, an additional semantics checker is used to verify that the criteria in (2) are satisfied. Currently, the restrictions in (2) require that LCTRSs follow the SMT-LIB 2.6 format, that `formula` expressions follow a particular structure and that a fixed set of variables includes sort annotations. In (1) it was only mentioned that variables need to be inferable, however, the latter simplifies type inference for tool authors significantly. Additionally, we limit the supported theories to integers, reals, bit vectors, and a combination of integers and reals. Hence only theory symbols and theory sorts specified in those theories are supported.

Previously, most tools relied on their own ad-hoc formats which we wanted to avoid. However, future releases of the SMT-LIB format may require further adaptations to maintain compatibility. Those two components—the ARI format and the ARI database—laid the foundation for establishing the first LCTRS category in CoCo 2024. We are pleased to report that crest won this category in 2024!

Constrained REwriting Software Tool

In January 2023, we began with a prototype implementation, which was later extended to crest. This prototype used its own format and implemented the results from [64]. However, crest significantly extended this prototype by adopting the ARI format—in particular the ARI database format for LCTRSs—and incorporating all the results

[4]https://ari-cops.uibk.ac.at/ARI/
[5]https://ari-cops.uibk.ac.at/COPS/

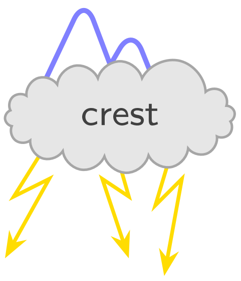


Figure 3.2: Logo of crest.

presented in Chapters 4, 5 and 7. For the following cases, the input format of crest is less strict than the ARI database format of LCTRSs: the implemented type inference algorithm allows us to avoid most sort annotations for variables, negative numbers can be written as `-1` instead of `(- 1)`, and for quantifier-free bit vector arithmetic it supports abbreviations[6] that are not part of the theory, e.g., `bv_sub` or `bv_div`. It is implemented using Haskell, the code is open source and can be downloaded from its website:

http://cl-informatik.uibk.ac.at/software/crest/

The logo of crest, shown in Figure 3.2, was designed by Aart Middeldorp. It visually represents the main three components of our automated analysis: constrained critical peaks which give rise to CCPs, crest and its confluence analysis and lightning bolts symbolizing the closing rewrite sequences to establish confluence.

The main functionality of crest, along with implementation details and extensive experiments is presented in Chapter 7. We conclude this section with an explanation of the simplified overview shown in Figure 3.3: (1) It parses the given LCTRS input file and the given property. During this parsing phase already light checks with respect to the format are performed. It also extracts the given theories and loads appropriate theory symbols/sorts into the LCTRS. (2) Then it performs type inference to determine any missing sorts and obtain a fully sorted LCTRS, before verifying that all conditions for a valid LCTRS are satisfied. At this stage, we have a fully sorted LCTRS, on which transformations are applied—such as moving values from left-hand sides of rules into the constraint and replacing them by fresh variables. (3) Finally, the main analysis begins. In this phase, the appropriate methods—depending on the desired property—are executed concurrently. Each method runs its own SMT solver instance. This phase involves a coordinated interplay between the SMT component, interfacing the SMT solver; the REW component, which provides a rewrite engine for single-step, parallel step, and multi-step rewriting; and the CCP component which computes the constrained (parallel) critical pairs. (4) If the analysis succeeds, crest outputs the result together with a proof supporting its correctness; otherwise, no result is provided.

[6]https://smt-lib.org/logics-all.shtml#QF_BV

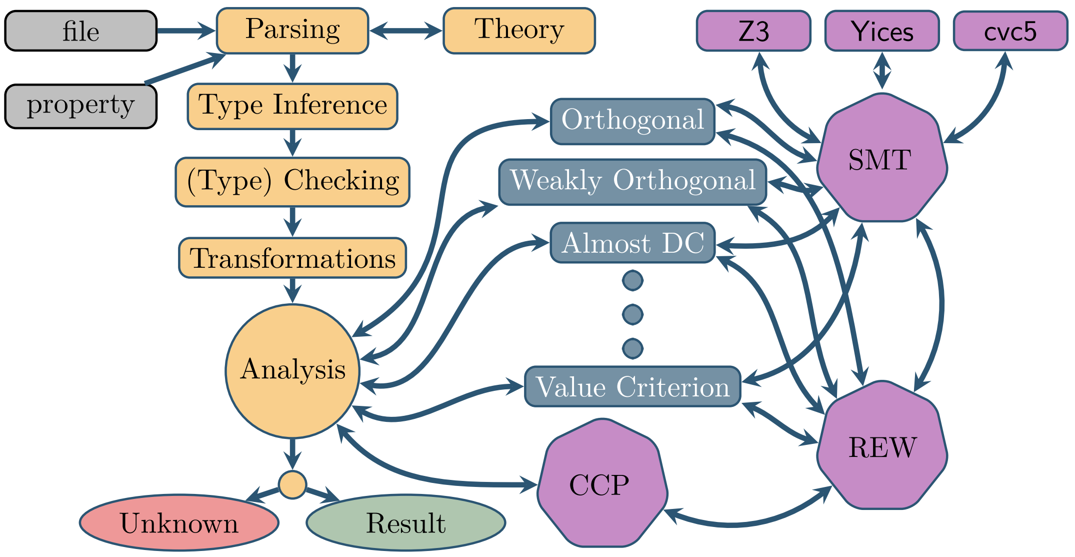


Figure 3.3: Simplified overview of crest.

3.2 Other Publications

Throughout this PhD study, several (informal) publications were produced, some of which form the basis for later chapters, while others are not included as standalone chapters. In the following we want to briefly comment on those.

[53] Fabian Mitterwallner, Jonas Schöpf, and Aart Middeldorp. Reducing confluence of LCTRSs to confluence of TRSs. In *Proceedings of the 12th International Workshop on Confluence (IWC)*, pages 3–8, 2023. Available at http://cl-informatik.uibk.ac.at/iwc/iwc2023.pdf

This workshop paper contains the initial idea for transforming LCTRSs into (possibly infinite) TRSs with the same rewrite relation. This transformation enabled us to lift the development closedness criterion into the constrained setting more easily without the need of replaying its original proof for TRSs. It served as basis for [68].

[3] Takahito Aoto, Nao Hirokawa, Dohan Kim, Misaki Kojima, Aart Middeldorp, Fabian Mitterwallner, Naoki Nishida, Teppei Saito, Jonas Schöpf, Kiraku Shintani, René Thiemann, and Akihisa Yamada. A new format for rewrite systems. In *Proceedings of the 12th International Workshop on Confluence (IWC)*, pages 32–37, 2023. Available at http://cl-informatik.uibk.ac.at/iwc/iwc2023.pdf

In this workshop paper we propose a new format for rewrite systems. The format follows S-expression syntax and tries to overcome problems inherent in the old COPS format. In particular, it contains the first official input format to represent LCTRSs, which made it possible for the annual Confluence Competition (CoCo) to host the first LCTRS category in its 2024 edition. Since 2024, CoCo uses this format for all competition categories.

[78] Kanta Takahata, Jonas Schöpf, Naoki Nishida, and Takahito Aoto. Most general constrained rewriting in logically constrained term rewriting systems. In Soichiro Hidaka and Takeshi Tsukada, editors, *Proceedings of the 27th JSSST Workshop on Programming and Programming Languages (PPL)*, 2025. In Japanese. Gamagori, Japan, March 5–7, 2025

During a meeting in Innsbruck with Takahito Aoto and Naoki Nishida, we discussed current challenges concerning the equivalence relation on constrained terms. In particular, the lack of a feasible method to compute equivalences between constrained terms, and the fact that such an equivalence not necessarily imply logical equivalence of the constraints. Our proposed solutions to these issues were formalized and further extended by Kanta Takahata. This was submitted to a national Japanese workshop.

[67] Jonas Schöpf and Aart Middeldorp. Improving confluence analysis for LCTRSs. In *Proceedings of the 14th International Workshop on Confluence (IWC)*, 2025. Available at https://iwc2025.github.io/

The current approaches for confluence analysis of LCTRSs rely on applying specific criteria and lack support for more modular methods, such as the redundant rules method and the reduction method. These two methods have proven highly effective for automation in the context of TRSs. In this workshop paper, we adapt both techniques to LCTRSs.

[76] Kanta Takahata, Jonas Schöpf, Naoki Nishida, and Takahito Aoto. Characterizing equivalence of logically constrained terms via existentially constrained terms. In Santiago Escobar and Laura Titolo, editors, *Proceedings of the 35th International Symposium on Logic-Based Program Synthesis and Transformation (LOPSTR)*, Lecture Notes in Computer Science, 2025. to appear

This paper extends the work presented in [78], where many of the underlying concepts and proofs have been simplified, improved and corrected. We introduce the concept of existentially constrained terms, which explicitly carry a set of logical variables and in the constraint a set of existentially quantified variables. We also present several characterizations of the equivalence relations between such terms. This allows us to eliminate the need for the notion of $\mathcal{LV}\mathsf{ar}$. The full version is available on arXiv [77].

[79] Kanta Takahata, Jonas Schöpf, Naoki Nishida, and Takahito Aoto. Recovering commutation of logically constrained rewriting and equivalence transformations. In Małgorzata Biernacka and Carlos Olarte, editors, *Proceedings of the 27th International Symposium on Principles and Practice of Declarative Programming (PPDP)*. Association of the Computing Machinery, 2025. to appear

The content of this paper builds on the results of [76, 78]. We define a class of constrained rewrite rules that are both left-linear and left-value-free. For these special rules, we show that on constrained terms the rewrite relation commutes with the equivalence relation. This result allows equivalence transformations during rewriting to be deferred until the end of the rewrite sequence. This is an important property for automation as most tools do not compute any equivalence steps in between rewrite steps. The full version is available on arXiv [80].

Chapter 4

Confluence Criteria for Logically Constrained Rewrite Systems

Publication Details

The content of this chapter is taken from the camera-ready version of the following publication:

[64] Jonas Schöpf and Aart Middeldorp. Confluence criteria for logically constrained rewrite systems. In Brigitte Pientka and Cesare Tinelli, editors, *Proceedings of the 29th International Conference on Automated Deduction (CADE)*, volume 14132 of *Lecture Notes in Artificial Intelligence*, pages 474–490, 2023. doi: 10.1007/978-3-031-38499-8_27

Adjustments were made to the presentation of the content to align with the format of this thesis. Additionally to the correction of obvious typos, we did change the set "$\mathcal{F}_{\mathsf{th}} \setminus \mathcal{F}_{\mathsf{te}}$" [64, Definition 3] of the root symbols of the calculation rules to "$\mathcal{F}_{\mathsf{th}} \setminus \mathcal{V}\mathsf{al}$" in Definition 4.7. Ignoring this change can lead to non-terminating sequences on constrained terms, i.e.,

$$s[v]\ [\mathsf{true}] \rightarrow_{\mathsf{ca}} s[x]\ [\mathsf{true} \wedge x = v] \sim s[v]\ [\mathsf{true}]$$

for a value $v \in \mathcal{F}_{\mathsf{th}} \setminus \mathcal{F}_{\mathsf{te}}$, a term s such that $s|_p = v$, and a position p. The same behavior occurs for the calculation relations on terms and constrained terms in [45]. Appendix A contains the missing lemmata and proofs, and a full version of this paper is available via arXiv [65].

The content of this publication, including the prototype implementation of crest, was developed primarily by me in collaboration with my supervisor Aart Middeldorp.

Abstract

Numerous confluence criteria for plain term rewrite systems are known. For logically constrained rewrite system, an attractive extension of term rewriting in which rules are equipped with logical constraints, much less is known. In this paper we extend the strongly-closed and (almost) parallel-closed critical pair criteria of Huet and Toyama to the logically constrained setting. We discuss the challenges for automation and present crest, a new tool for logically constrained rewriting in which the confluence criteria are implemented, together with experimental data.

4.1 Introduction

Logically constrained rewrite systems constitute a general rewrite formalism with native support for constraints that are handled by SMT solvers. They are useful for program analysis, as illustrated in numerous papers [13, 23, 40, 89]. Several results from term rewriting have been lifted to constrained rewriting. We mention termination analysis [42, 44, 88], rewriting induction [23], completion [88] as well as runtime complexity analysis [89].

In this paper we are concerned with confluence analysis of logically constrained rewrite systems (LCTRSs for short). Only two sufficient conditions for confluence of LCTRSs are known. Kop and Nishida considered (weak) orthogonality in [45]. Orthogonality is the combination of left-linearity and the absence of critical pairs, in a weakly orthogonal system trivial critical pairs are allowed. Completion of LCTRSs is the topic of [88] and the underlying confluence condition of completion is the combination of termination and joinability of critical pairs. In this paper we add two further confluence criteria. Both of these extend known conditions for standard term rewriting to the constrained setting. The first is the combination of linearity and strong closedness of critical pairs, introduced by Huet [32]. The second, also due to [32], is the combination of left-linearity and parallel closedness of critical pairs. We also consider an extension of the latter, due to Toyama [84].

Overview The remainder of this paper is organized as follows. In the next section we summarize the relevant background. Section 4.3 recalls the existing confluence criteria for LCTRSs and some of the underlying results. The new confluence criteria for LCTRSs are reported in Section 4.4. In Section 4.5 the automation challenges we faced are described and we present our prototype implementation crest. Experimental results are reported in Section 4.6, before we conclude in Section 4.7.

4.2 Preliminaries

We assume familiarity with the basic notions of term rewrite systems (TRSs) [9], but shortly recapitulate terminology and notation that we use in the remainder. In particular, we recall the notion of logically constrained rewriting as defined in [23, 45].

We assume a many-sorted signature $\mathcal{F}$ and a set $\mathcal{V}$ of (many-sorted) variables disjoint from $\mathcal{F}$. The signature $\mathcal{F}$ is split into term symbols from $\mathcal{F}_{\mathsf{te}}$ and theory symbols from $\mathcal{F}_{\mathsf{th}}$. The set $\mathcal{T}(\mathcal{F},\mathcal{V})$ contains the well-sorted terms over this signature and $\mathcal{T}(\mathcal{F}_{\mathsf{th}})$ denotes the set of well-sorted ground terms that consist entirely of theory symbols. We assume a mapping $\mathcal{I}$ which assigns to every sort ι occurring in $\mathcal{F}_{\mathsf{th}}$ a carrier set $\mathcal{I}(\iota)$, and an interpretation $\mathcal{J}$ that assigns to every symbol $f \in \mathcal{F}_{\mathsf{th}}$ with sort declaration $\iota_1 \times \cdots \times \iota_n \to \kappa$ a function $f_{\mathcal{J}} \colon \mathcal{I}(\iota_1) \times \cdots \times \mathcal{I}(\iota_n) \to \mathcal{I}(\kappa)$. Moreover, for every sort ι occurring in $\mathcal{F}_{\mathsf{th}}$ we assume a set $\mathcal{V}\mathsf{al}_\iota \subseteq \mathcal{F}_{\mathsf{th}}$ of value symbols, such that all $c \in \mathcal{V}\mathsf{al}_\iota$ are constants of sort ι and $\mathcal{J}$ constitutes a bijective mapping between $\mathcal{V}\mathsf{al}_\iota$ and $\mathcal{I}(\iota)$. Thus there exists a constant symbol in $\mathcal{F}_{\mathsf{th}}$ for every value in the carrier set. The interpretation $\mathcal{J}$ naturally extends to a mapping $[\![\cdot]\!]$ from ground terms in $\mathcal{T}(\mathcal{F}_{\mathsf{th}})$ to values in $\mathcal{V}\mathsf{al} = \bigcup_{\iota \in \mathcal{D}\mathsf{om}(\mathcal{I})} \mathcal{V}\mathsf{al}_\iota$: $[\![f(t_1,\dots,t_n)]\!] = f_{\mathcal{J}}([\![t_1]\!],\dots,[\![t_n]\!])$ for all $f(t_1,\dots,t_n) \in \mathcal{T}(\mathcal{F}_{\mathsf{th}})$. So every ground term in $\mathcal{T}(\mathcal{F}_{\mathsf{th}})$ has a unique value. We demand that theory symbols and term symbols overlap only on values, i.e., $\mathcal{F}_{\mathsf{te}} \cap \mathcal{F}_{\mathsf{th}} \subseteq \mathcal{V}\mathsf{al}$. A term in $\mathcal{T}(\mathcal{F}_{\mathsf{th}}, \mathcal{V})$ is called a *logical* term.

Positions are strings of positive natural numbers used to address subterms. The empty string is denoted by ϵ. We write $q \leqslant p$ and say that p is below q if $qq' = p$ for some position q', in which case $p \backslash q$ is defined to be q'. Furthermore, $q < p$ if $q \leqslant p$ and $q \neq p$. Finally, positions q and p are parallel, written as $q \parallel p$, if neither $q \leqslant p$ nor $p < q$. The set of positions of a term t is defined as $\mathcal{P}\mathsf{os}(t) = \{\epsilon\}$ if t is a variable or a constant, and as $\mathcal{P}\mathsf{os}(t) = \{\epsilon\} \cup \{iq \mid 1 \leqslant i \leqslant n \text{ and } q \in \mathcal{P}\mathsf{os}(t_i)\}$ if $t = f(t_1,\dots,t_n)$ with $n \geqslant 1$. The subterm of t at position $p \in \mathcal{P}\mathsf{os}(t)$ is defined as $t|_p = t$ if $p = \epsilon$ and as $t|_p = t_i|_q$ if $p = iq$ and $t = f(t_1,\dots,t_n)$. We write $s[t]_p$ for the result of replacing the subterm at position p of s with t. We write $\mathcal{P}\mathsf{os}_{\mathcal{V}}(t)$ for $\{p \in \mathcal{P}\mathsf{os}(t) \mid t|_p \in \mathcal{V}\}$ and $\mathcal{P}\mathsf{os}_{\mathcal{F}}(t)$ for $\mathcal{P}\mathsf{os}(t) \setminus \mathcal{P}\mathsf{os}_{\mathcal{V}}(t)$. The set of variables occurring in the term t is denoted by $\mathcal{V}\mathsf{ar}(t)$. A term t is linear if every variable occurs at most once in it. A substitution is a mapping σ from $\mathcal{V}$ to $\mathcal{T}(\mathcal{F}, \mathcal{V})$ such that its domain $\{x \in \mathcal{V} \mid \sigma(x) \neq x\}$ is finite. We write $t\sigma$ for the result of applying σ to the term t.

We assume the existence of a sort bool such that $\mathcal{I}(\mathsf{bool}) = \mathbb{B} = \{\top, \bot\}$, $\mathcal{V}\mathsf{al}_{\mathsf{bool}} = \{\mathsf{true}, \mathsf{false}\}$, $[\![\mathsf{true}]\!] = \top$, and $[\![\mathsf{false}]\!] = \bot$ hold. Logical terms of sort bool are called *constraints*. A constraint φ is *valid* if $[\![\varphi\gamma]\!] = \top$ for all substitutions γ such that $\gamma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$.

A *constrained rewrite rule* is a triple $\ell \to r\ [\varphi]$ where $\ell, r \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ are terms of the same sort such that $\mathsf{root}(\ell) \in \mathcal{F}_{\mathsf{te}} \setminus \mathcal{F}_{\mathsf{th}}$ and φ is a logical term of sort bool. If $\varphi = \mathsf{true}$ then the constraint is often omitted, and the rule is denoted as $\ell \to r$. We denote the set $\mathcal{V}\mathsf{ar}(\varphi) \cup (\mathcal{V}\mathsf{ar}(r) \setminus \mathcal{V}\mathsf{ar}(\ell))$ of *logical* variables in $\rho\colon \ell \to r\ [\varphi]$ by $\mathcal{LV}\mathsf{ar}(\rho)$. We write $\mathcal{EV}\mathsf{ar}(\rho)$ for the set $\mathcal{V}\mathsf{ar}(r) \setminus (\mathcal{V}\mathsf{ar}(\ell) \cup \mathcal{V}\mathsf{ar}(\varphi))$ of variables that appear only in the right-hand side of ρ. Note that extra variables in right-hand sides are allowed, but they may only be instantiated by values. This is useful to model user input or random choice [23]. A set of constrained rewrite rules is called a *logically constrained rewrite system* (LCTRS for short).

The LCTRS $\mathcal{R}$ introduced in the example below computes the maximum of two integers.

Example 4.1. Before giving the rules, we need to define the term and theory symbols, the carrier sets and interpretation functions:

$$\begin{aligned}
\mathcal{F}_{\mathsf{te}} &= \{\mathsf{max}\colon \mathsf{int} \times \mathsf{int} \Rightarrow \mathsf{int}\} \cup \{\mathsf{0}, \mathsf{1}, \dots : \mathsf{int}\} \qquad \mathcal{I}_{\mathsf{bool}} = \mathbb{B} \qquad \mathcal{I}_{\mathsf{int}} = \mathbb{Z}\\
\mathcal{F}_{\mathsf{th}} &= \{\mathsf{0}, \mathsf{1}, \dots : \mathsf{int}\} \cup \{\mathsf{true}, \mathsf{false}\colon \mathsf{bool}\} \cup \{\neg\colon \mathsf{bool} \Rightarrow \mathsf{bool}\}\\
&\cup \{-\colon \mathsf{int} \Rightarrow \mathsf{int}\} \cup \{\wedge\colon \mathsf{bool} \times \mathsf{bool} \Rightarrow \mathsf{bool}\}\\
&\cup \{+, -\colon \mathsf{int} \times \mathsf{int} \Rightarrow \mathsf{int}\} \cup \{\leqslant, \geqslant, <, >, = : \mathsf{int} \times \mathsf{int} \Rightarrow \mathsf{bool}\}
\end{aligned}$$

The interpretations for theory symbols follow the usual semantics given in the SMT-LIB theory Ints[1] used by the SMT-LIB logic $\mathsf{QF\_LIA}$. The LCTRS $\mathcal{R}$ consists of the following

[1]http://smtlib.cs.uiowa.edu/Theories/Ints.smt2

constrained rewrite rules

$$\mathsf{max}(x, y) \to x\ [x \geqslant y] \qquad \mathsf{max}(x, y) \to y\ [y \geqslant x] \qquad \mathsf{max}(x, y) \to \mathsf{max}(y, x)$$

In later examples we refrain from spelling out the signature and interpretations of the theory Ints. We now define rewriting using constrained rewrite rules. LCTRSs admit two kinds of rewrite steps. Rewrite rules give rise to *rule* steps, provided the constraint of the rule is satisfied. In addition, theory calls of the form $f(v_1, \dots, v_n)$ with $f \in \mathcal{F}_{\mathsf{th}} \setminus \mathcal{V}\mathsf{al}$ and values $v_1, \dots, v_n$ can be evaluated in a *calculation* step. In the definition below, a substitution σ is said to *respect* a rule $\rho\colon \ell \to r\ [\varphi]$, denoted by $\sigma \vDash \rho$, if $\mathcal{D}\mathsf{om}(\sigma) = \mathcal{V}\mathsf{ar}(\ell) \cup \mathcal{V}\mathsf{ar}(r) \cup \mathcal{V}\mathsf{ar}(\varphi)$, $\sigma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$, and $\varphi\sigma$ is valid. Moreover, a constraint φ is respected by σ, denoted by $\sigma \vDash \varphi$, if $\sigma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$ and $\varphi\sigma$ is valid.

Definition 4.2. Let $\mathcal{R}$ be an LCTRS. A *rule step* $s \to_{\mathsf{ru}} t$ satisfies $s|_p = \ell\sigma$ and $t = s[r\sigma]_p$ for some position p and constrained rewrite rule $\ell \to r\ [\varphi]$ that is respected by the substitution σ. A *calculation step* $s \to_{\mathsf{ca}} t$ satisfies $s|_p = f(v_1, \dots, v_n)$ and $t = s[v]_p$ for some $f \in \mathcal{F}_{\mathsf{th}} \setminus \mathcal{V}\mathsf{al}$, $v_1, \dots, v_n \in \mathcal{V}\mathsf{al}$ with $v = [\![f(v_1, \dots, v_n)]\!]$. In this case $f(x_1, \dots, x_n) \to y\ [y = f(x_1, \dots, x_n)]$ with a fresh variable y is a *calculation rule*. The set of all calculation rules is denoted by $\mathcal{R}_{\mathsf{ca}}$. The relation $\to_{\mathcal{R}}$ associated with $\mathcal{R}$ is the union of $\to_{\mathsf{ru}} \cup \to_{\mathsf{ca}}$.

We sometimes write $\to_{p|\rho|\sigma}$ to indicate that the rewrite step takes place at position p, using the constrained rewrite rule ρ with substitution σ.

Example 4.3. We have $\mathsf{max}(1 + 2, 4) \to_{\mathcal{R}} \mathsf{max}(3, 4) \to_{\mathcal{R}} \mathsf{max}(4, 3) \to_{\mathcal{R}} 4$ in the LCTRS of Example 4.1. The first step is a calculation step. In the third step we apply the rule $\mathsf{max}(x, y) \to x\ [x \geqslant y]$ with substitution $\sigma = \{x \mapsto 4, y \mapsto 3\}$.

4.3 Confluence

In this paper we are concerned with the confluence of LCTRSs. An LCTRS $\mathcal{R}$ is *confluent* if $t \to_{\mathcal{R}}^* \cdot {}_{\mathcal{R}}^*\!\leftarrow u$ for all terms s, t and u such that $t\ {}_{\mathcal{R}}^*\!\leftarrow s \to_{\mathcal{R}}^* u$. Confluence criteria for TRSs are based on critical pairs. Critical pairs for LCTRS were introduced in [45]. The difference with the definition below is that we add dummy constraints for *extra* variables in right-hand sides of rewrite rules.

Definition 4.4. An *overlap* of an LCTRS $\mathcal{R}$ is a triple $\langle \rho_1, p, \rho_2 \rangle$ with rules $\rho_1\colon \ell_1 \to r_1\ [\varphi_1]$ and $\rho_2\colon \ell_2 \to r_2\ [\varphi_2]$, satisfying the following conditions:

1. ρ_1 and ρ_2 are variable-disjoint variants of rewrite rules in $\mathcal{R} \cup \mathcal{R}_{\mathsf{ca}}$,
2. $p \in \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell_2)$,
3. ℓ_1 and $\ell_2|_p$ are unifiable with a mgu σ such that $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2)$,

4. $\varphi_1\sigma \wedge \varphi_2\sigma$ is satisfiable, and
5. if $p = \epsilon$ then ρ_1 and ρ_2 are not variants, or $\mathcal{V}\mathsf{ar}(r_1) \nsubseteq \mathcal{V}\mathsf{ar}(\ell_1)$.

In this case we call $\ell_2\sigma[r_1\sigma]_p \approx r_2\sigma\ [\varphi_1\sigma \wedge \varphi_2\sigma \wedge \psi\sigma]$ a *constrained critical pair* obtained from the overlap $\langle \rho_1, p, \rho_2 \rangle$. Here

$$\psi = \bigwedge \{x = x \mid x \in \mathcal{E}\mathcal{V}\mathsf{ar}(\rho_1) \cup \mathcal{E}\mathcal{V}\mathsf{ar}(\rho_2)\}$$

The set of all constrained critical pairs of $\mathcal{R}$ is denoted by $\mathsf{CCP}(\mathcal{R})$.

In the following we drop "constrained" and speak of critical pairs. The condition $\mathcal{V}\mathsf{ar}(r_1) \nsubseteq \mathcal{V}\mathsf{ar}(\ell_1)$ in the fifth condition is essential to correctly deal with extra variables in rewrite rules. The equations (ψ) added to the constraint of a critical pair save the information which variables in a critical pair were introduced by variables only occurring in the right-hand side of a rewrite rule and therefore should *only* be instantiated by values. Critical pairs as defined in [45, 88] lack this information. The proof of Theorem 4.19 in the next section makes clear why those trivial equations are essential for our confluence criteria, see also Example 4.17.

Example 4.5. Consider the LCTRS consisting of the rule

$$\rho\colon\ \mathsf{f}(x) \to z\ [x = z\,\hat{}\,2]$$

The variable z does not occur in the left-hand side and the condition $\mathcal{V}\mathsf{ar}(r_1) \nsubseteq \mathcal{V}\mathsf{ar}(\ell_1)$ ensures that ρ overlaps with (a variant of) itself at the root position. Note that $\mathcal{R}$ is not confluent due to the non-joinable local peak $-4 \leftarrow \mathsf{f}(16) \to 4$.

Example 4.6. The LCTRS $\mathcal{R}$ of Example 4.1 admits the following critical pairs:

$x \approx y\ [x \geqslant y \wedge y \geqslant x]$	$\langle 1, \epsilon, 2 \rangle$
$x \approx \mathsf{max}(y, x)\ [x \geqslant y]$	$\langle 1, \epsilon, 3 \rangle$
$y \approx \mathsf{max}(y, x)\ [y \geqslant x]$	$\langle 2, \epsilon, 3 \rangle$

The originating overlap is given on the right, where we number the rewrite rules from left to right in Example 4.1.

Actually, there are three more overlaps since the position of overlap (ϵ) is the root position. Such overlaps are called *overlays* and always come in pairs. For instance, $\mathsf{max}(y, x) \approx x\ [x \geqslant y]$ is the critical pair originating from $\langle 3, \epsilon, 1 \rangle$. For confluence criteria based on symmetric joinability conditions of critical pairs (like weak orthogonality and joinability of critical pairs for terminating systems) we need to consider just one critical pair, but this is not true for the criteria presented in the next section.

Logically constrained rewriting aims to rewrite (unconstrained) terms with constrained rules. However, for the sake of analysis, rewriting *constrained terms* is useful. In particular, since critical pairs in LCTRSs come with a constraint, confluence criteria need to consider constrained terms. The relevant notions defined below originate from [23, 45].

Definition 4.7. A *constrained term* is a pair $s\ [\varphi]$ of a term s and a constraint φ. Two constrained terms $s\ [\varphi]$ and $t\ [\psi]$ are *equivalent*, denoted by $s\ [\varphi] \sim t\ [\psi]$, if for every substitution γ respecting φ there is some substitution δ that respects ψ such that $s\gamma = t\delta$, and vice versa. Let $\mathcal{R}$ be an LCTRS and $s\ [\varphi]$ a constrained term. If $s|_p = \ell\sigma$ for some constrained rewrite rule $\rho\colon \ell \to r\ [\psi]$, position p, and substitution σ such that $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}\mathsf{ar}(\varphi)$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$, φ is satisfiable and $\varphi \Rightarrow \psi\sigma$ is valid then

$$s\ [\varphi] \to_{\mathsf{ru}} s[r\sigma]_p\ [\varphi]$$

is a *rule step*. If $s|_p = f(s_1, \dots, s_n)$ with $f \in \mathcal{F}_{\mathsf{th}} \setminus \mathcal{V}\mathsf{al}$ and $s_1, \dots, s_n \in \mathcal{V}\mathsf{al} \cup \mathcal{V}\mathsf{ar}(\varphi)$ then

$$s\ [\varphi] \to_{\mathsf{ca}} s[x]_p\ [\varphi \wedge x = f(s_1, \dots, s_n)]$$

is a *calculation step*. Here x is a fresh variable. We write $\to_{\mathcal{R}}$ for $\to_{\mathsf{ru}} \cup \to_{\mathsf{ca}}$ and the rewrite relation $\overset{\sim}{\to}_{\mathcal{R}}$ on constrained terms is defined as $\sim \cdot \to_{\mathcal{R}} \cdot \sim$.

Positions in connection with $\overset{\sim}{\to}_{\mathcal{R}}$ steps always refer to the underlying steps in $\to_{\mathcal{R}}$. We give an example of constrained rewriting.

Example 4.8. Consider again the LCTRS $\mathcal{R}$ of Example 4.1. We have

$$\begin{aligned} \mathsf{max}(x + y, 6)\ [x \geqslant 2 \wedge y \geqslant 4] &\to_{\mathcal{R}} \mathsf{max}(z, 6)\ [x \geqslant 2 \wedge y \geqslant 4 \wedge z = x + y] \\ &\to_{\mathcal{R}} z\ [x \geqslant 2 \wedge y \geqslant 4 \wedge z = x + y] \end{aligned}$$

The first step is a calculation step. The second step is a rule step using the rule $\mathsf{max}(x, y) \to x\ [x \geqslant y]$ with the substitution $\sigma = \{x \mapsto z, y \mapsto 6\}$. Note that the constraint $(x \geqslant 2 \wedge y \geqslant 4 \wedge z = x + y) \Rightarrow z \geqslant 6$ is valid.

Definition 4.9. A critical pair $s \approx t\ [\varphi]$ is *trivial* if $s\sigma = t\sigma$ for every substitution σ with $\sigma \vDash \varphi$.[2] A left-linear LCTRS having only trivial critical pairs is called *weakly orthogonal*. A left-linear TRS without critical pairs is called *orthogonal*.

The following result is from [45].

Theorem 4.10. *Weakly orthogonal LCTRS are confluent.* □

Example 4.11. The following left-linear LCTRS computes the Ackermann function using term symbols from $\mathcal{F}_{\mathsf{te}} = \{\mathsf{ack} : \mathsf{int} \times \mathsf{int} \Rightarrow \mathsf{int}\} \cup \{0, 1, \cdots : \mathsf{int}\}$ and the same theory symbols, carrier sets and interpretations as in Example 4.1:

$$\begin{aligned} \mathsf{ack}(0, n) &\to n + 1\ [n \geqslant 0] \\ \mathsf{ack}(m, 0) &\to \mathsf{ack}(m - 1, 1)\ [m > 0] \\ \mathsf{ack}(m, n) &\to \mathsf{ack}(m - 1, \mathsf{ack}(m, n - 1))\ [m > 0 \wedge n > 0] \\ \mathsf{ack}(m, n) &\to 0\ [m < 0 \vee n < 0] \end{aligned}$$

Since the conjunction of any two constraints is unsatisfiable, $\mathcal{R}$ lacks critical pairs. Hence $\mathcal{R}$ is confluent by Theorem 4.10.

[2] The triviality condition in [45] is wrong. Here we use the corrected version in an update of [45] announced on Cynthia Kop's website (accessible at https://www.cs.ru.nl/~cynthiakop/frocos13.pdf).

The following result is proved in [88] and forms the basis of completion of LCTRSs.

Lemma 4.12. *Let $\mathcal{R}$ be an LCTRS. If $t \;{}_{\mathcal{R}}{\leftarrow}\; s \to_{\mathcal{R}} u$ then $t \downarrow_{\mathcal{R}} u$ or $t \xleftrightarrow[\mathsf{CCP}(\mathcal{R})]{} u$.* $\square$

In combination with Newman's Lemma, the following confluence criterion is obtained.

Corollary 4.13. *A terminating LCTRS is confluent if all critical pairs are joinable.*

This is less obvious than it seems. Joinability of a critical pair $s \approx t\ [\varphi]$ cannot simply be defined as $s\ [\varphi] \xrightarrow{\sim}{}^{*}_{\mathcal{R}} \cdot {}^{*}_{\mathcal{R}}{\xleftarrow{\sim}}\ t\ [\varphi]$, as the following example shows.

Example 4.14. Consider the terminating LCTRS $\mathcal{R}$ consisting of the rewrite rules

$$\mathsf{f}(x,y) \to \mathsf{g}(x, 1+1) \qquad\qquad \mathsf{h}(\mathsf{f}(x,y)) \to \mathsf{h}(\mathsf{g}(y, 1+1))$$

The single critical pair $\mathsf{h}(\mathsf{g}(x,1+1)) \approx \mathsf{h}(\mathsf{g}(y,1+1))$ should not be joinable because $\mathcal{R}$ is not confluent, but we do have

$$\begin{aligned}
&\mathsf{h}(\mathsf{g}(x,1+1)) \to_{\mathsf{ca}} \mathsf{h}(\mathsf{g}(x,z))\ [z = 1+1] \sim \mathsf{h}(\mathsf{g}(y,v))\ [v = 1+1]\\
&\mathsf{h}(\mathsf{g}(y,1+1)) \to_{\mathsf{ca}} \mathsf{h}(\mathsf{g}(y,v))\ [v = 1+1]
\end{aligned}$$

due to the equivalence relation $\sim$ on constrained terms; since x and y do not appear in the constraints, there is no demand that they must be instantiated with values.

The solution is not to treat the two sides of a critical pair in isolation but define joinability based on rewriting constrained term pairs. So we view the symbol $\approx$ in a constrained equation $s \approx t\ [\varphi]$ as a binary constructor symbol such that the constrained equation can be viewed as a constrained term. Steps in s take place at positions $\geqslant 1$ whereas steps in t use positions $\geqslant 2$. The same is done in completion of LCTRSs [88].

Definition 4.15. We call a constrained equation $s \approx t\ [\varphi]$ *trivial* if $s\sigma = t\sigma$ for any substitution σ with $\sigma \vDash \varphi$. A critical pair $s \approx t\ [\varphi]$ is *joinable* if $s \approx t\ [\varphi] \xrightarrow{\sim}{}^{*}_{\mathcal{R}} u \approx v\ [\psi]$ and $u \approx v\ [\psi]$ is trivial.

We revisit Example 4.14.

Example 4.16. For the critical pair in Example 4.14 we obtain

$$\begin{aligned}
&\mathsf{h}(\mathsf{g}(x,1+1)) \approx \mathsf{h}(\mathsf{g}(y,1+1))\\
&\qquad \to_{\mathsf{ca}} \mathsf{h}(\mathsf{g}(x,v)) \approx \mathsf{h}(\mathsf{g}(y,1+1))\ [v = 1+1]\\
&\qquad \to_{\mathsf{ca}} \mathsf{h}(\mathsf{g}(x,v)) \approx \mathsf{h}(\mathsf{g}(y,z))\ [v = 1+1 \wedge z = 1+1]
\end{aligned}$$

The substitution $\sigma = \{v \mapsto 2, z \mapsto 2\}$ respects the constraint $v = 1+1 \wedge z = 1+1$ but does not equate $\mathsf{h}(\mathsf{g}(x,v))$ and $\mathsf{h}(\mathsf{g}(y,z))$.

The converse of Corollary 4.13 also holds, but note that in contrast to TRSs, joinability of critical pairs is not a decidable criterion for terminating LCTRSs, due to the undecidable triviality condition. Moreover, for the converse to hold, it is essental that critical pairs contain the trivial equations ψ in Definition 4.4.

Example 4.17. Consider the LCTRS $\mathcal{R}$ consisting of the rules

$$\mathsf{f}(x) \to \mathsf{g}(y) \qquad\qquad \mathsf{g}(y) \to \mathsf{a}\ [y = y]$$

which admits the critical pair $\mathsf{g}(y) \approx \mathsf{g}(y')\ [y = y \wedge y' = y']$ originating from the overlap $\langle \mathsf{f}(x) \to \mathsf{g}(y), \epsilon, \mathsf{f}(x') \to \mathsf{g}(y')\rangle$. This critical pair is joinable as y and y' are restricted to values and thus both sides rewrite to a using the second rule. As $\mathcal{R}$ is also terminating, it is confluent by Corollary 4.13. If we were to drop ψ in Definition 4.4, we would obtain the non-joinable critical pair $\mathsf{g}(y) \approx \mathsf{g}(y')$ instead and wrongly conclude non-confluence.

4.4 Main Results

We start with extending a confluence result of Huet [32] for linear TRSs. Below we write $\to_{\geqslant p}$ to indicate that the position of the contracted redex in the step is below position p.

Definition 4.18. A critical pair $s \approx t\ [\varphi]$ is *strongly closed* if

1. $s \approx t\ [\varphi] \overset{\sim}{\to}^{*}_{\geqslant 1} \cdot \overset{\sim}{\to}^{=}_{\geqslant 2} u \approx v\ [\psi]$ for some trivial $u \approx v\ [\psi]$, and
2. $s \approx t\ [\varphi] \overset{\sim}{\to}^{*}_{\geqslant 2} \cdot \overset{\sim}{\to}^{=}_{\geqslant 1} u \approx v\ [\psi]$ for some trivial $u \approx v\ [\psi]$.

A binary relation $\to$ on terms is *strongly confluent* if $t \to^{*} \cdot {}^{=}\!\!\leftarrow u$ for all terms s, t and u with $t \leftarrow s \to u$. (By symmetry, also $t \to^{=} \cdot {}^{*}\!\!\leftarrow u$ is required.) Strong confluence is a well-known sufficient condition for confluence. Huet [32] proved that linear TRSs are strongly confluent if all critical pairs are strongly closed. Below we extend this result to LCTRSs, using the above definition of strongly closed constrained critical pairs.

Theorem 4.19. *A linear LCTRS is strongly confluent if all its critical pairs are strongly closed.*

We give full proof details in order to illustrate the complications caused by constrained rewrite rules. The following result from [88] plays an important role.

Lemma 4.20. *Suppose $s \approx t\ [\varphi] \overset{\sim}{\to}_{p} u \approx v\ [\psi]$ and $\gamma \vDash \varphi$. If $p \geqslant 1$ then $s\gamma \to u\delta$ and $t\gamma = v\delta$ for some substitution δ with $\delta \vDash \psi$. If $p \geqslant 2$ then $s\gamma = u\delta$ and $t\gamma \to v\delta$ for some substitution δ with $\delta \vDash \psi$.* ☐

Proof of Theorem 4.19. Consider an arbitrary local peak

$$t \leftarrow_{p_1 | \rho_1 | \sigma_1} s \to_{p_2 | \rho_2 | \sigma_2} u$$

with rewrite rules $\rho_1 \colon \ell_1 \to r_1\ [\varphi_1]$ and $\rho_2 \colon \ell_2 \to r_2\ [\varphi_2]$ from $\mathcal{R} \cup \mathcal{R}_{\mathsf{ca}}$. We may assume that ρ_1 and ρ_2 have no variables in common, and consequently $\mathcal{D}\mathsf{om}(\sigma_1) \cap \mathcal{D}\mathsf{om}(\sigma_2) = \varnothing$. We have $s|_{p_1} = \ell_1\sigma_1$, $t = s[r_1\sigma_1]_{p_1}$ and $\sigma_1 \vDash \varphi_1$. Likewise, $s|_{p_2} = \ell_2\sigma_2$, $u = s[r_2\sigma_2]_{p_2}$ and $\sigma_2 \vDash \varphi_2$. If $p_1 \parallel p_2$ then

$$t \to_{p_2 | \rho_2 | \sigma_2} t[r_2\sigma_2]_{p_2} = u[r_1\sigma_1]_{p_1} \leftarrow_{p_1 | \rho_1 | \sigma_1} u$$

Hence both $t \to^{*} \cdot {}^{=}\!\!\leftarrow u$ and $t \to^{=} \cdot {}^{*}\!\!\leftarrow u$. If p_1 and p_2 are not parallel then $p_1 \leqslant p_2$ or $p_2 < p_1$. Without loss of generality, we consider $p_1 \leqslant p_2$. Let $q = p_2 \backslash p_1$. We do a case analysis on whether or not $q \in \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell_1)$.

- First suppose $q \notin \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell_1)$. Let $q = q_1 q_2$ such that $q_1 \in \mathcal{P}\mathsf{os}_{\mathcal{V}}(\ell_1)$ and let x be the variable in ℓ_1 at position q_1. We have $\ell_2\sigma_2 = x\sigma_1|_{q_2}$ and thus $\sigma_1(x) \notin \mathcal{V}\mathsf{al}$. Define the substitution σ_1' as follows:

$$\sigma_1'(y) = \begin{cases} x\sigma_1[r_2\sigma_2]_{q_2} & \text{if } y = x \\ \sigma_1(y) & \text{otherwise} \end{cases}$$

  We show $t \to^{=} s[r_1\sigma_1']_{p_1} \leftarrow u$, which yields $t \to^{*} \cdot {}^{=}\!\!\leftarrow u$ and $t \to^{=} \cdot {}^{*}\!\!\leftarrow u$. Since $\mathcal{R}$ is left-linear, $\ell_1\sigma_1' = \ell_1\sigma_1[x\sigma_1']_{q_1} = \ell_1\sigma_1[x\sigma_1[r_2\sigma_2]_{q_2}]_{q_1} = \ell_1\sigma_1[r_2\sigma_2]_q$ and thus $u = s[r_2\sigma_2]_{p_2} = s[\ell_1\sigma_1[r_2\sigma_2]_q]_{p_1} = s[\ell_1\sigma_1']_{p_1}$. If we can show $\sigma_1' \vDash \rho_1$ then $u \to s[r_1\sigma_1']_{p_1}$. Consider an arbitrary variable $y \in \mathcal{LV}\mathsf{ar}(\rho_1)$. If $y \neq x$ then $\sigma_1'(y) = \sigma_1(y) \in \mathcal{V}\mathsf{al}$ since $\sigma_1 \vDash \rho_1$. If $y = x$ then $x \in \mathcal{V}\mathsf{ar}(\varphi)$ since $x \in \mathcal{V}\mathsf{ar}(\ell_1)$. However, this contradicts $\sigma_1 \vDash \rho_1$ as $\sigma_1(x) \notin \mathcal{V}\mathsf{al}$. So $\sigma_1'(y) = \sigma_1(y)$ for all $y \in \mathcal{LV}\mathsf{ar}(\rho_1)$ and thus $\sigma_1' \vDash \rho_1$ is an immediate consequence of $\sigma_1 \vDash \rho_1$. It remains to show $t \to^{=} s[r_1\sigma_1']_{p_1}$. If $x \notin \mathcal{V}\mathsf{ar}(r_1)$ then $r_1\sigma_1' = r_1\sigma_1$ and thus $t = s[r_1\sigma_1']_{p_1}$. If $x \in \mathcal{V}\mathsf{ar}(r_1)$ then there exists a unique position $q' \in \mathcal{P}\mathsf{os}_{\mathcal{V}}(r_1)$ such that $r_1|_{q'} = x$, due to the right-linearity of $\mathcal{R}$. Hence $r_1\sigma_1' = r_1\sigma_1[x\sigma_1[r_2\sigma_2]_{q_2}]_{q'} = r_1\sigma_1[r_2\sigma_2]_{q'q_2}$. Since $r_1\sigma_1|_{q'q_2} = \ell_2\sigma_2$ we obtain $t = s[r_1\sigma_1]_{p_1} \to_{p_1 q' q_2|\rho_2|\sigma_2} s[r_1\sigma_1']_{p_1}$ as desired.

- Next suppose $q \in \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell_1)$. The substitution $\sigma' = \sigma_1 \cup \sigma_2$ satisfies $\ell_1|_q\sigma' = \ell_1|_q\sigma_1 = \ell_2\sigma_2 = \ell_2\sigma'$ and thus is a unifier of $\ell_1|_q$ and ℓ_2. Since $\sigma_1 \vDash \rho_1$ and $\sigma_2 \vDash \rho_2$, $\sigma'(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2)$. Let σ be an mgu of $\ell_1|_q$ and ℓ_2. Since σ is at least as general as σ', $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2)$. Since $\varphi_1\sigma' = \varphi_1\sigma_1$ and $\varphi_2\sigma' = \varphi_2\sigma_2$ are valid, $\varphi_1\sigma \wedge \varphi_2\sigma$ is satisfiable. Hence conditions 1, 2, 3 and 4 in Definition 4.4 hold for the triple $\langle \rho_2, q, \rho_1 \rangle$. If condition 5 is *not* fulfilled then $q = \epsilon$ (and thus $p_1 = p_2$), ρ_2 and ρ_1 are variants, and $\mathcal{V}\mathsf{ar}(r_2) \subseteq \mathcal{V}\mathsf{ar}(\ell_2)$ (and thus also $\mathcal{V}\mathsf{ar}(r_1) \subseteq \mathcal{V}\mathsf{ar}(\ell_1)$). Hence $\ell_1\sigma_1 = \ell_2\sigma_2$ and $r_1\sigma_1 = r_2\sigma_2$, and thus $t = u$. In the remaining case condition 5 holds and hence $\langle \rho_2, q, \rho_1 \rangle$ is an overlap. By definition, $\ell_1\sigma[r_2\sigma]_q \approx r_1\sigma\ [\varphi_2\sigma \wedge \varphi_1\sigma \wedge \psi\sigma]$ with

$$\psi = \bigwedge \{x = x \mid x \in \mathcal{EV}\mathsf{ar}(\rho_1) \cup \mathcal{EV}\mathsf{ar}(\rho_2)\}$$

  is a critical pair. To simplify the notation, we abbreviate $\ell_1\sigma[r_2\sigma]_q$ to s', $r_1\sigma$ to t', and $\varphi_2\sigma \wedge \varphi_1\sigma \wedge \psi\sigma$ to φ'. Critical pairs are strongly closed by assumption, and thus both

  1. $s' \approx t'\ [\varphi']\ \overset{\sim}{\to}^{*}_{\geqslant 1} \cdot \overset{\sim}{\to}^{=}_{\geqslant 2}\ u \approx v\ [\psi']$ for some trivial $u \approx v\ [\psi']$, and
  2. $s' \approx t'\ [\varphi']\ \overset{\sim}{\to}^{*}_{\geqslant 2} \cdot \overset{\sim}{\to}^{=}_{\geqslant 1}\ u \approx v\ [\psi']$ for some trivial $u \approx v\ [\psi']$.

  Let γ be the substitution such that $\sigma\gamma = \sigma'$. We claim that γ respects φ'. So let $x \in \mathcal{V}\mathsf{ar}(\varphi') = \mathcal{V}\mathsf{ar}(\varphi_2\sigma \wedge \varphi_1\sigma \wedge \psi\sigma)$. We have

$$\mathcal{LV}\mathsf{ar}(\rho_1) = \mathcal{V}\mathsf{ar}(\varphi_1) \cup \mathcal{EV}\mathsf{ar}(\rho_1) \qquad \mathcal{LV}\mathsf{ar}(\rho_2) = \mathcal{V}\mathsf{ar}(\varphi_2) \cup \mathcal{EV}\mathsf{ar}(\rho_2)$$

  Together with $\mathcal{V}\mathsf{ar}(\psi) = \mathcal{EV}\mathsf{ar}(\rho_1) \cup \mathcal{EV}\mathsf{ar}(\rho_2)$ we obtain

$$\mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2) = \mathcal{V}\mathsf{ar}(\varphi_1) \cup \mathcal{V}\mathsf{ar}(\varphi_2) \cup \mathcal{V}\mathsf{ar}(\psi)$$

Since $\sigma'(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2)$, we obtain $\gamma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi')$ and thus $\gamma \vDash \varphi'$. At this point repeated applications of Lemma 4.20 to the constrained rewrite sequence in item 1 yields a substitution δ respecting ψ' such that $s'\gamma \to^* u\delta$ and $t'\gamma = v\delta$. Since $u \approx v\ [\psi']$ is trivial, $u\delta = v\delta$ and hence $s'\gamma \to^* \cdot\, {}^{=}\!\!\leftarrow t'\gamma$. Likewise, $s'\gamma \to^= \cdot\, {}^{*}\!\!\leftarrow t'\gamma$ is obtained from item 2. We have

$$s'\gamma = (\ell_1\sigma[r_2\sigma]_q)\gamma = \ell_1\sigma'[r_2\sigma']_q = \ell_1\sigma_1[r_2\sigma_2]_q \qquad t'\gamma = r_1\sigma' = r_1\sigma_1$$

Moreover, $t = s[r_1\sigma_1]_{p_1} = s[t'\gamma]_{p_1}$ and $u = s[\ell_1\sigma_1[r_2\sigma_2]_q]_{p_1} = s[s'\gamma]_{p_1}$. Since rewriting is closed under contexts, we obtain $u \to^* \cdot\, {}^{=}\!\!\leftarrow t$ and $u \to^= \cdot\, {}^{*}\!\!\leftarrow t$. This completes the proof.

□

Example 4.21. Consider the LCTRS $\mathcal{R}$ of Example 4.1 and its critical pairs in Example 4.6. The critical pair

$$x \approx \mathsf{max}(y, x)\ [x \geqslant y]$$

is not trivial, so Theorem 4.10 is not applicable and the rule $\mathsf{max}(x, y) \to \mathsf{max}(y, x)$ precludes the use of Corollary 4.13 to infer confluence. We do have

$$x \approx \mathsf{max}(y, x)\ [x \geqslant y] \xrightarrow{\geqslant 2} x \approx x\ [x \geqslant y]$$

by applying the rule $\mathsf{max}(x, y) \to y\ [y \geqslant x]$ and the resulting constrained equation $x \approx x\ [x \geqslant y]$ is obviously trivial. The same reasoning applies to the critical pair $y \approx \mathsf{max}(y, x)\ [y \geqslant x]$. The first critical pair $x \approx y\ [x \geqslant y \land y \geqslant x]$ in Example 4.6 is trivial since any (value) substitution satisfying its constraint $x \geqslant y \land y \geqslant x$ equates x and y. By symmetry, all critical pairs of $\mathcal{R}$ are strongly closed. Since $\mathcal{R}$ is linear, confluence follows from Theorem 4.19.

The second main result is the extension of Huet's parallel closedness condition on critical pairs in left-linear TRSs [32] to LCTRSs. To this end, we first define parallel rewriting for LCTRSs.

Definition 4.22. Let $\mathcal{R}$ be an LCTRS. The relation $\mathrel{+\!\!\!\!\!\rightarrow}_{\mathcal{R}}$ is defined on terms inductively as follows:

1. $x \mathrel{+\!\!\!\!\!\rightarrow}_{\mathcal{R}} x$ for all variables x,
2. $f(s_1, \ldots, s_n) \mathrel{+\!\!\!\!\!\rightarrow}_{\mathcal{R}} f(t_1, \ldots, t_n)$ if $s_i \mathrel{+\!\!\!\!\!\rightarrow}_{\mathcal{R}} t_i$ for all $1 \leqslant i \leqslant n$,
3. $\ell\sigma \mathrel{+\!\!\!\!\!\rightarrow}_{\mathcal{R}} r\sigma$ with $\ell \to r\ [\varphi] \in \mathcal{R}$ and $\sigma \vDash \ell \to r\ [\varphi]$,
4. $f(v_1, \ldots, v_n) \mathrel{+\!\!\!\!\!\rightarrow} v$ with $f \in \mathcal{F}_{\mathsf{th}} \setminus \mathcal{V}\mathsf{al}$, $v_1, \ldots, v_n \in \mathcal{V}\mathsf{al}$ and $v = [\![f(v_1, \ldots, v_n)]\!]$.

We write $\mathrel{+\!\!\!\!\!\rightarrow}_{\geqslant p}$ to indicate that all positions of contracted redexes in the parallel step are below p. In the next definition we add constraints to parallel rewriting.

Definition 4.23. Let $\mathcal{R}$ be an LCTRS. The relation $\nrightarrow\!\!\!\!\!\rightarrow_{\mathcal{R}}$ is defined on constrained terms inductively as follows:

1. $x\ [\varphi] \nrightarrow\!\!\!\!\!\rightarrow_{\mathcal{R}} x\ [\varphi]$ for all variables x,
2. $f(s_1, \ldots, s_n)\ [\varphi] \nrightarrow\!\!\!\!\!\rightarrow_{\mathcal{R}} f(t_1, \ldots, t_n)\ [\varphi \wedge \psi]$ if $s_i\ [\varphi] \nrightarrow\!\!\!\!\!\rightarrow_{\mathcal{R}} t_i\ [\varphi \wedge \psi_i]$ for all $1 \leqslant i \leqslant n$ and $\psi = \psi_1 \wedge \cdots \wedge \psi_n$,
3. $\ell\sigma\ [\varphi] \nrightarrow\!\!\!\!\!\rightarrow_{\mathcal{R}} r\sigma\ [\varphi]$ with $\rho\colon \ell \to r\ [\omega] \in \mathcal{R}$, $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}\mathsf{ar}(\varphi)$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$, φ is satisfiable and $\varphi \Rightarrow \omega\sigma$ is valid,
4. $f(v_1, \ldots, v_n)\ [\varphi] \nrightarrow\!\!\!\!\!\rightarrow v\ [\varphi \wedge v = f(v_1, \ldots, v_n)]$ with $v_1, \ldots, v_n \in \mathcal{V}\mathsf{al} \cup \mathcal{V}\mathsf{ar}(\varphi)$, $f \in \mathcal{F}_{\mathsf{th}} \setminus \mathcal{V}\mathsf{al}$ and v is a fresh variable.

Here we assume that different applications to case 4 result in different fresh variables. The constraint ψ in case 2 collects the assignments introduced in earlier applications of case 4. (If there are none, $\psi = \mathsf{true}$ is omitted.) The same holds for $\psi_1, \ldots, \psi_n$. We write $\overset{\sim}{\nrightarrow\!\!\!\!\!\rightarrow}$ for the relation $\sim \cdot \nrightarrow\!\!\!\!\!\rightarrow_{\mathcal{R}} \cdot \sim$.

In light of the earlier developments, the following definition is the obvious adaptation of parallel closedness for LCTRSs.

Definition 4.24. A critical pair $s \approx t\ [\varphi]$ is *parallel closed* if

$$s \approx t\ [\varphi] \overset{\sim}{\nrightarrow\!\!\!\!\!\rightarrow}_{\geqslant 1} u \approx v\ [\psi]$$

for some trivial $u \approx v\ [\psi]$.

Note that the right-hand side t of the constrained equation $s \approx t\ [\varphi]$ may change due to the equivalence relation $\sim$, cf. the statement of Lemma 4.20.

Theorem 4.25. *A left-linear LCTRS is confluent if its critical pairs are parallel closed.*

To prove this result, we adapted the formalized proof presented in [57] to the constrained setting. The required changes are very similar to the ones in the proof of Theorem 4.19.

Example 4.26. Consider the LCTRS $\mathcal{R}$ with rules

$$\begin{aligned} \mathsf{f}(x, y) &\to \mathsf{g}(\mathsf{a}, y + y)\ [y \geqslant x \wedge y = 1] & \mathsf{a} &\to \mathsf{b} \\ \mathsf{h}(\mathsf{f}(x, y)) &\to \mathsf{h}(\mathsf{g}(\mathsf{b}, 2))\ [x \geqslant y] & \mathsf{g}(x, y) &\to \mathsf{g}(y, x) \end{aligned}$$

The single critical pair $\mathsf{h}(\mathsf{g}(\mathsf{a}, y + y)) \approx \mathsf{h}(\mathsf{g}(\mathsf{b}, 2))\ [y \geqslant x \wedge y = 1 \wedge x \geqslant y]$ is parallel closed:

$$\begin{aligned} &\mathsf{h}(\mathsf{g}(\mathsf{a}, y + y)) \approx \mathsf{h}(\mathsf{g}(\mathsf{b}, 2))\ [y \geqslant x \wedge y = 1 \wedge x \geqslant y] \\ &\qquad \overset{\sim}{\nrightarrow\!\!\!\!\!\rightarrow}_{\geqslant 1} \mathsf{h}(\mathsf{g}(\mathsf{b}, z)) \approx \mathsf{h}(\mathsf{g}(\mathsf{b}, 2))\ [y \geqslant x \wedge y = 1 \wedge x \geqslant y \wedge z = y + y] \end{aligned}$$

and the obtained equation is trivial. Hence $\mathcal{R}$ is confluent by Theorem 4.25. Note that the earlier confluence criteria do no apply.

We also consider the extension of Huet's result by Toyama [84], which has a less restricted joinability condition on critical pairs stemming from overlapping rules at the root position. Such critical pairs are called *overlays* whereas critical pairs originating from overlaps $\langle \rho_1, p, \rho_2 \rangle$ with $p > \epsilon$ are called *inner* critical pairs.

Definition 4.27. An LCTRS $\mathcal{R}$ is almost parallel-closed if every inner critical pair is parallel closed and every overlay $s \approx t\ [\varphi]$ satisfies

$$s \approx t\ [\varphi]\ \overset{\sim}{\mathrel{+\!\!\!\!\twoheadrightarrow}}_{\geqslant 1} \cdot \overset{\sim}{\rightarrow}{}^{*}_{\geqslant 2}\ u \approx v\ [\psi]$$

for some trivial $u \approx v\ [\psi]$.

Theorem 4.28. *Left-linear almost parallel-closed LCTRSs are confluent.*

Again, the formalized proof of the corresponding result for plain TRSs in [57] can be adapted to the constrained setting.

Example 4.29. Consider the following variation of the LCTRS $\mathcal{R}$ in Example 4.26:

$$\mathsf{f}(x, y) \to \mathsf{g}(\mathsf{a}, y + y)\ [y \geqslant x \wedge y = 1] \qquad \mathsf{a} \to \mathsf{b}$$
$$\mathsf{f}(x, y) \to \mathsf{g}(\mathsf{b}, 2)\ [x \geqslant y] \qquad \mathsf{g}(x, y) \to \mathsf{g}(y, x)$$

The overlay $\mathsf{g}(\mathsf{b}, 2) \approx \mathsf{g}(\mathsf{a}, y + y)\ [x \geqslant y \wedge y \geqslant x \wedge y = 1]$ is not parallel closed but one readily confirms that the condition in Definition 4.27 applies.

4.5 Automation

As it is very inconvenient and tedious to test by hand if an LCTRS satisfies one of the confluence criteria presented in the preceding sections, we provide an implementation. The natural choice would be to extend the existing tool Ctrl [47] because it is currently the only tool capable of analyzing confluence of LCTRSs. However, Ctrl is not actively maintained and not very well documented, so we decided to develop a new tool for the analysis of LCTRSs. Our tool is called crest (constrained rewriting software tool). It is written in Haskell, based on the Haskell `term-rewriting`[3] library and allows the logics `QF_LIA`, `QF_NIA`, `QF_LRA`.

The input format of crest is described on its website.[4] After parsing the input, crest checks that the resulting LCTRS is well-typed. Missing sort information is inferred. Next it is checked concurrently whether one of the implemented confluence criteria applies. crest supports (weak) orthogonality, strong closedness and (almost) parallel closedness. The tool outputs the computed critical pairs and a "proof" describing how these are closed, based on the first criterion that reports a YES result. Below we describe some of the challenges that one faces when automating the confluence criteria presented in the preceding sections.

[3] https://hackage.haskell.org/package/term-rewriting-0.4.0.2
[4] http://cl-informatik.uibk.ac.at/software/crest/

First of all, how can we determine whether a constrained critical pair or more generally a constrained equation $s \approx t\ [\varphi]$ is trivial? The following result explains how this can be solved by an SMT solver.

Definition 4.30. Given a constrained equation $s \approx t\ [\varphi]$, the formula $T(s,t,\varphi)$ is inductively defined as follows:

$$T(s,t,\varphi) = \begin{cases} \mathsf{true} & \text{if } s = t \\ s = t & \text{if } s,t \in \mathcal{V}\mathsf{al} \cup \mathcal{V}\mathsf{ar}(\varphi) \\ \displaystyle\bigwedge_{i=1}^{n} T(s_i,t_i,\varphi) & \text{if } s = f(s_1,\ldots,s_n) \text{ and } t = f(t_1,\ldots,t_n) \\ \mathsf{false} & \text{otherwise} \end{cases}$$

Lemma 4.31. *A constrained equation $s \approx t\ [\varphi]$ is trivial if and only if the formula $\varphi \implies T(s,t,\varphi)$ is valid.*

Proof. First suppose $\varphi \implies T(s,t,\varphi)$ is valid. Let σ be a substitution with $\sigma \vDash \varphi$. Since $\sigma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$, we can apply σ to the formula $\varphi \implies T(s,t,\varphi)$. We obtain $[\![\varphi\sigma]\!] = \top$ from $\sigma \vDash \varphi$. Hence also $[\![T(s,t,\varphi)\sigma]\!] = \top$. Since $T(s,t,\varphi)$ is a conjunction, the final case in the definition of $T(s,t,\varphi)$ is not used. Hence $\mathcal{P}\mathsf{os}(s) = \mathcal{P}\mathsf{os}(t)$, $s(p) = t(p)$ for all internal positions p in s and t, and $s|_p\sigma = t|_p\sigma$ for all leaf positions p in s and t. Consequently, $s\sigma = t\sigma$. This concludes the triviality proof of $s \approx t\ [\varphi]$.

For the only if direction, suppose $s \approx t\ [\varphi]$ is trivial. Note that the variables appearing in the formula $\varphi \implies T(s,t,\varphi)$ are those of φ. Let σ be an arbitrary assignment such that $[\![\varphi\sigma]\!] = \top$. We need to show $[\![T(s,t,\varphi)\sigma]\!] = \top$. We can view σ as a substitution with $\sigma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$. We have $\sigma \vDash \varphi$ and thus $s\sigma = t\sigma$ by the triviality of $s \approx t\ [\varphi]$. Hence $T(s,t,\varphi)$ is a conjunction of equations between values and variables in φ, which are turned into identities by σ. Hence $[\![T(s,t,\varphi)\sigma]\!] = \top$ as desired. □

The second challenge is how to implement rewriting on constrained equations in particular, how to deal with the equivalence relation $\sim$ defined in Definition 4.7.

Example 4.32. The LCTRS $\mathcal{R}$

$$\mathsf{f}(x) \to z\ [z = 3] \qquad \mathsf{g}(\mathsf{f}(x)) \to \mathsf{a} \qquad \mathsf{g}(3) \to \mathsf{a}$$

over the integers admits two critical pairs:

$$z \approx z'\ [z = 3 \wedge z' = 3] \qquad \mathsf{g}(z) \approx \mathsf{a}\ [z = 3]$$

The first one is trivial, but to join the second one, an initial equivalence step is required:

$$\mathsf{g}(z) \approx \mathsf{a}\ [z = 3] \sim \mathsf{g}(3) \approx \mathsf{a}\ [z = 3] \to \mathsf{a} \approx \mathsf{a}\ [z = 3]$$

The transformation introduced below avoids having to look for an initial equivalence step before a rule becomes applicable.

Definition 4.33. Let $\mathcal{R}$ be an LCTRS. Given a term $t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$, we replace values in t by fresh variables and return the modified term together with the constraint that collects the bindings:

$$\mathsf{tf}(t) = \begin{cases} (t, \mathsf{true}) & \text{if } t \in \mathcal{V} \\ (z, z = t) & \text{if } t \in \mathcal{V}\mathsf{al} \text{ and } z \text{ is a fresh variable} \\ (f(s_1, \ldots, s_n), \varphi_1 \wedge \cdots \wedge \varphi_n) & \text{if } t = f(t_1, \ldots, t_n) \text{ and } \mathsf{tf}(t_i) = (s_i, \varphi_i) \end{cases}$$

Applying the transformation tf to the left-hand sides of the rules in $\mathcal{R}$ produces

$$\mathsf{tf}(\mathcal{R}) = \{\ell' \to r\ [\varphi \wedge \psi] \mid \ell \to r\ [\varphi] \in \mathcal{R} \text{ and } \mathsf{tf}(\ell) = (\ell', \psi)\}$$

Example 4.34. Applying the transformation tf to the LCTRS $\mathcal{R}$ of Example 4.32 produces the rules

$$\mathsf{f}(x) \to z\ [z = 3] \qquad \mathsf{g}(\mathsf{f}(x)) \to \mathsf{a} \qquad \mathsf{g}(z) \to \mathsf{a}\ [z = 3]$$

The critical pair $\mathsf{g}(z) \approx \mathsf{a}\ [z = 3]$ can now be joined by an application of the modified third rule. Note that the modified rule does not overlap with the second rule because z may not be instantiated with $\mathsf{f}(x)$. Hence the modified LCTRS $\mathsf{tf}(\mathcal{R})$ is strongly closed and, because it is linear, also confluent.

In the following we show the correctness of the transformation. In particular we prove that the initial rewrite relation is preserved.

Lemma 4.35. *The relations $\to_{\mathcal{R}}$ and $\to_{\mathsf{tf}(\mathcal{R})}$ coincide on unconstrained terms.*

Proof. Consider $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$. Since the transformation tf does not affect calculation steps, it suffices to consider rule steps. First assume $s = C[\ell\sigma] \to_{\mathsf{ru}} C[r\sigma] = t$ by applying the rule $\ell \to r\ [\varphi] \in \mathcal{R}$ and let $\ell' \to r\ [\varphi'] \in \mathsf{tf}(\mathcal{R})$ be its transformation. So $\mathsf{tf}(\ell) = (\ell', \psi)$ and $\varphi' = \varphi \wedge \psi$. Define the substitution

$$\sigma' = \{\ell'|_p \mapsto \ell|_p \mid (\ell', \psi) = \mathsf{tf}(\ell),\ p \in \mathcal{P}\mathsf{os}(\ell) \text{ and } \ell|_p \in \mathcal{V}\mathsf{al}\}$$

and let $\tau = \sigma \cup \sigma'$. Since $\mathcal{D}\mathsf{om}(\sigma) \cap \mathcal{D}\mathsf{om}(\sigma') = \varnothing$ by construction, τ is well-defined. From $\sigma \vDash \ell \to r\ [\varphi]$ and $\sigma' \vDash \psi$ we immediately obtain $\tau \vDash \ell' \to r\ [\varphi']$, which yields $s = C[\ell'\tau] \to_{\mathsf{ru}} C[r\tau] = t$ in $\mathsf{tf}(\mathcal{R})$.

For the other direction consider $s = C[\ell'\sigma] \to_{\mathsf{ru}} C[r'\sigma] = t$ by applying the rule $\ell' \to r'\ [\varphi'] \in \mathsf{tf}(\mathcal{R})$. The difference between ℓ' and its originating left-hand side ℓ in $\mathcal{R}$ is that value positions in ℓ are occupied by fresh variables in ℓ'. Because σ' respects $\varphi' = \varphi \wedge \psi$, σ' substitutes the required values at these positions in ℓ. As $\sigma \vDash \ell' \to r'\ [\varphi']$, there exists a rule $\ell \to r\ [\varphi]$ which is respected by σ and thus $s = C[\ell\sigma] \to_{\mathsf{ru}} C[r\sigma] = t$ in $\mathcal{R}$. $\square$

As the transformation is used in the implementation and rewriting on constrained terms plays a key role, the following result is needed. The proof is similar to the first half of the proof of Lemma 4.35 and omitted.

Lemma 4.36. *The inclusion $\to_{\mathcal{R}} \subseteq \to_{\mathsf{tf}(\mathcal{R})}$ holds on constrained terms.*

Table 4.1: Specific experimental results.

	result	method	time (in ms)
[88, Example 23]	Timeout	—	10017.70
[88, Example 23] corrected	YES	strongly closed	103.71
Example 4.11	YES	orthogonal	34.35
[45, Example 3]	YES	weakly orthogonal	50.87
Example 4.1	YES	strongly closed	115.33
[57, Example 1]	YES	strongly closed	3806.84
Example 4.26	YES	parallel closed	38.42
Example 4.29	YES	almost parallel closed	130.36

4.6 Experimental Results

In order to evaluate our tool we performed some experiments. As there is no official database of interesting confluence problems for LCTRSs, we collected several LCTRSs from the literature and the repository of Ctrl. The problem files in the latter that contain an equivalence problem of two functions for rewriting induction were split into two separate files. The experiments were performed on an AMD Ryzen 7 PRO 4750U CPU with a base clock speed of 1.7 GHz, 8 cores and 32 GB of RAM. The full set of benchmarks consists of 127 problems of which crest can prove 90 confluent, 11 result in MAYBE and 26 in a timeout. With a timeout of 5 seconds crest needs 141.09 seconds to analyze the set of benchmarks. We have tested the implementation with 3 well-known SMT solvers: Z3, Yices and cvc5. Among those Z3 gives the best performance regarding time and the handling of non-linear arithmetic. Hence we use Z3 as the default SMT solver in our implementation. In Table 4.1 we list some interesting systems from this paper and the relevant literature. Full details are available from the website of crest. We choose 5 as the maximum number of steps in the $\to^*$ parts of the strongly closed and almost parallel closed criteria.

From Table 4.2 the relative power of each implemented confluence criterion on our benchmark can be inferred, i.e., it depicts how many of the 127 problems both methods can prove confluent. This illustrates that the relative applicability in theory (e.g., weakly orthogonal LCTRSs are parallel closed), is preserved in our implementation. We conclude this section with an interesting observation discovered by crest when testing [88, Example 23].

We also tested the applicability of Corollary 4.13, using the tool Ctrl as a black box for proving termination. Of the 127 problems, Ctrl claims 102 to be terminating and 67 of those can be shown locally confluent by crest, where we limit the number of steps in the joining sequence to 100. It is interesting to note that all of these problems are orthogonal, and so proving termination and finding a joining sequence is not necessary

Table 4.2: Comparison between confluence criteria implemented in crest.

	O	W	S	P	A
orthogonality (O)	74	74	11	74	74
weak orthogonality (W)		78	13	78	78
strongly closed (S)			20	16	20
parallel closed (P)				83	83
almost parallel closed (A)					89

to conclude confluence, on the current set of problems. Of the remaining 35 problems, crest can show confluence of 5 of these by almost parallel closedness.

Example 4.37. The LCTRS $\mathcal{R}$ is obtained by completing a system consisting of four constrained equations:

1. $\mathsf{f}(x, y) \to \mathsf{f}(z, y) + 1\ [x \geqslant 1 \wedge z = x - 1]$
2. $\mathsf{f}(x, 0) \to \mathsf{g}(1, x)\ [x \leqslant 1]$
3. $\mathsf{g}(0, y) \to y\ [x \leqslant 0]$
4. $\mathsf{g}(1, 1) \to \mathsf{g}(1, 0) + 1$
5. $\mathsf{h}(x) \to \mathsf{g}(1, x) + 1\ [x \leqslant 1]$
6. $\mathsf{h}(x) \to \mathsf{f}(x - 1, 0) + 2\ [x \geqslant 1]$

Calling crest on $\mathcal{R}$ results in a timeout. As a matter of fact, the LCTRS is not confluent because the critical pair

$$\mathsf{g}(1, x) + 1 \approx \mathsf{f}(x - 1, 0) + 2\ [x \leqslant 1 \wedge x \geqslant 1]$$

between rules 5 and 6 is not joinable. Inspecting the steps in [88, Example 23] reveals some incorrect applications of the inference rules of constrained completion, which causes rule 6 to be wrong. Replacing it with the correct rule

$$6'.\quad \mathsf{h}(x) \to (\mathsf{f}(z, 0) + 1) + 1\ [x > 1 \wedge z = x - 1]$$

causes crest to report confluence by strong closedness.

4.7 Concluding Remarks

In this paper we presented new confluence criteria for LCTRSs as well as a new tool in which these criteria have been implemented. We clarified the subtleties that arise when analyzing joinability of critical pairs in LCTRSs and reported experimental results.

For plain rewrite systems many more confluence criteria are known and implemented in powerful tools that compete in the yearly Confluence Competition (CoCo).[5] In the

[5]http://project-coco.uibk.ac.at/

near future we will investigate which of these can be lifted to LCTRSs. We will also advance the creation of a competition category on confluence of LCTRSs in CoCo.

Our tool crest has currently no support for termination. Implementing termination techniques in crest is of clear interest. The starting point here are the methods reported in [42, 44, 88]. Many LCTRSs coming from applications are actually non-confluent.[6] So developing more powerful techniques for LCTRSs is on our agenda as well.

Acknowledgments.

We thank Fabian Mitterwallner for valuable discussions on the presented topics and our Haskell implementation. The detailed comments by the reviewers improved the presentation. Cynthia Kop and Deivid Vale kindly provided us with instructions and a working implementation of Ctrl.

[6]Naoki Nishida, personal communication (February 2023).

Chapter 5

Confluence of Logically Constrained Rewrite Systems Revisited

Publication Details

The content of this chapter is taken from the camera-ready version of the following publication:

[68] Jonas Schöpf, Fabian Mitterwallner, and Aart Middeldorp. Confluence of logically constrained rewrite systems revisited. In Christoph Benzmüller, Marijn J.H. Heule, and Renate A. Schmidt, editors, *Proceedings of the 12th International Joint Conference on Automated Reasoning (IJCAR)*, volume 14740 of *Lecture Notes in Artificial Intelligence*, pages 298–316, 2024. doi: 10.1007/978-3-031-63501-4_16

Adjustments were made to the presentation of the content to align with the format of this thesis and obvious typos were corrected. Preliminary research on this subject appeared in a workshop paper with the same co-authors [53]. Appendix B contains the missing lemmata and proofs, and a full version of this paper is available via arXiv [69].

The initial idea for this publication emerged from a discussion with my co-author, Fabian Mitterwallner, with whom I also collaborated in developing the concepts presented in Section 5.4. Regarding the remaining sections, Fabian conceived the content of Section 5.3, while I primarily focused on Section 5.5. However, the content in Section 5.6 was developed solely by me. All of the work was carried out in collaboration and under the supervision of Aart Middeldorp.

Abstract

We show that (local) confluence of terminating logically constrained rewrite systems is undecidable, even when the underlying theory is decidable. Several confluence criteria for logically constrained rewrite systems are known. These were obtained by replaying existing proofs for plain term rewrite systems in a constrained setting, involving a non-trivial effort. We present a simple transformation from logically constrained rewrite systems to term rewrite systems such that critical pairs of the latter correspond to constrained critical pairs of the former. The usefulness of the transformation is illustrated by lifting the advanced confluence results based on (almost) development closed critical pairs as well as on parallel critical pairs to the constrained setting.

5.1 Introduction

Logically constrained rewrite systems (LCTRSs) [45] are a natural extension of plain term rewrite systems (TRSs) with native support for constraints that are handled by SMT solvers. The latter makes LCTRSs suitable for program analysis [12, 13, 23, 88]. In this paper we are concerned with confluence techniques for LCTRSs. Numerous techniques exist to (dis)prove confluence of TRSs. For LCTRSs much less is known. Kop and Nishida [45] established (weak) orthogonality as sufficient confluence criteria for LCTRSs. Joinability of critical pairs for terminating systems is implicit in [88]. Very recently, strong closedness for linear LCTRSs and (almost) parallel closedness for left-linear LCTRSs were established [64]. The proofs of these results were obtained by *replaying* existing proofs for TRSs in a constrained setting, involving a non-trivial effort. For more advanced confluence criteria, this is not feasible.

In particular, the conclusion in [45] that LCTRSs "are *flexible*: common analysis techniques for term rewriting extend to LCTRSs without much effort" is not accurate. On the contrary, in Section 5.3 we show that (local) confluence of terminating LCTRSs is undecidable, even for a decidable fragment of the theory of integers.

In Section 5.4 we present a simple transformation from LCTRSs to TRSs which allows us to relate results for the latter to the former. We use the transformation to extend two advanced confluence criteria based on (parallel) critical pairs from TRSs to LCTRSs: In Section 5.5 we prove that (almost) development closed left-linear LCTRSs are confluent by *reusing* the corresponding result for TRSs obtained by van Oostrom [87] and in Section 5.6 we lift the result of Toyama [83] based on parallel critical pairs from TRSs to LCTRSs. Both results are employed in state-of-the-art confluence provers for TRSs (ACP [6], CSI [55], Hakusan [73]) and have only recently been formally verified in the Isabelle proof assistant [29, 38, 39].

For the LCTRS extension of the result of Toyama [83] we observed a subtle problem in the definition of the equivalence relation on constrained terms, which goes back to [45] and has been used in subsequent work on LCTRSs [23, 64, 88]. We briefly discuss the issue at the end of the next section, after recalling basic notions for LCTRSs. For space reasons some of the more technical proofs are only available in an extended version of this paper [69]. The results in Section 5.4 and Section 5.5 were first announced in [53].

5.2 Preliminaries

We assume familiarity with the basic notions of term rewriting. In this section we recall a few key notions for LCTRSs. For more background information we refer to [45, 64, 88]. We assume a many-sorted signature $\mathcal{F} = \mathcal{F}_{\mathsf{te}} \cup \mathcal{F}_{\mathsf{th}}$ with a term and theory part. For every sort ι in $\mathcal{F}_{\mathsf{th}}$ we have a non-empty set $\mathcal{V}\mathsf{al}_\iota \subseteq \mathcal{F}_{\mathsf{th}}$ of value symbols, such that all $c \in \mathcal{V}\mathsf{al}_\iota$ are constants of sort ι. We demand $\mathcal{F}_{\mathsf{te}} \cap \mathcal{F}_{\mathsf{th}} \subseteq \mathcal{V}\mathsf{al}$ where $\mathcal{V}\mathsf{al} = \bigcup_\iota \mathcal{V}\mathsf{al}_\iota$. In the case of integers this results in an infinite signature with $\mathbb{Z} \subseteq \mathcal{V}\mathsf{al} \subseteq \mathcal{F}_{\mathsf{th}}$. A term in $\mathcal{T}(\mathcal{F}_{\mathsf{th}}, \mathcal{V})$ is called a *logical* term. Ground logical terms are mapped to values by an interpretation $\mathcal{J}$: $[\![f(t_1, \ldots, t_n)]\!] = f_{\mathcal{J}}([\![t_1]\!], \ldots, [\![t_n]\!])$. We assume a bijection between value symbols and elements in the domain of $\mathcal{J}$, e.g., for integers: $[\![0]\!] = 0$, $[\![-1]\!] = -1$, $[\![1]\!] = 1$ and so on. Logical terms of sort bool are called *constraints*. A constraint φ is *valid* if $[\![\varphi\gamma]\!] = \top$ for all substitutions γ such that $\gamma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$. A *constrained rewrite rule* is a triple $\ell \to r\ [\varphi]$ where $\ell, r \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ are terms of the same sort such that $\mathsf{root}(\ell) \in \mathcal{F}_{\mathsf{te}} \setminus \mathcal{F}_{\mathsf{th}}$ and φ is a constraint. We denote the set $\mathcal{V}\mathsf{ar}(\varphi) \cup (\mathcal{V}\mathsf{ar}(r) \setminus \mathcal{V}\mathsf{ar}(\ell))$ of *logical* variables in $\ell \to r\ [\varphi]$ by $\mathcal{LV}\mathsf{ar}(\ell \to r\ [\varphi])$. A constrained rewrite rule is left-linear (right-linear) if non-logical variables in the left-hand side (right-hand side) occur at most once. If a rule is left-linear and right-linear then it is called linear. An LCTRS is a set of constrained rewrite rules.

A substitution σ is said to *respect* a rule $\ell \to r\ [\varphi]$, denoted by $\sigma \vDash \ell \to r\ [\varphi]$, if $\mathcal{D}\mathsf{om}(\sigma) \subseteq \mathcal{V}\mathsf{ar}(\ell) \cup \mathcal{V}\mathsf{ar}(r) \cup \mathcal{V}\mathsf{ar}(\varphi)$, $\sigma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{LV}\mathsf{ar}(\ell \to r\ [\varphi])$, and $[\![\varphi\sigma]\!] = \top$. Moreover, a constraint φ is respected by σ, denoted by $\sigma \vDash \varphi$, if $\sigma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$ and $[\![\varphi\sigma]\!] = \top$. We call $f(x_1, \ldots, x_n) \to y\ [y = f(x_1, \ldots, x_n)]$ with a fresh variable y and $f \in \mathcal{F}_{\mathsf{th}} \setminus \mathcal{V}\mathsf{al}$ a *calculation rule*. Calculation rules are not part of the rules of an LCTRS $\mathcal{R}$. The set of all calculation rules induced by the signature $\mathcal{F}_{\mathsf{th}}$ of an LCTRS $\mathcal{R}$ is denoted by $\mathcal{R}_{\mathsf{ca}}$ and we abbreviate $\mathcal{R} \cup \mathcal{R}_{\mathsf{ca}}$ to $\mathcal{R}_{\mathsf{rc}}$. An LCTRS is called linear (left-linear, right-linear) if all its rules in $\mathcal{R}$ are linear (left-linear, right-linear). A rewrite step $s \to_{\mathcal{R}} t$ satisfies $s|_p = \ell\sigma$ and $t = s[r\sigma]_p$ for some position p, constrained rewrite rule $\ell \to r\ [\varphi]$ in $\mathcal{R}_{\mathsf{rc}}$, and substitution σ such that $\sigma \vDash \ell \to r\ [\varphi]$. We drop the subscript $\mathcal{R}$ from $\to_{\mathcal{R}}$ when no confusion arises. An LCTRS $\mathcal{R}$ is confluent if there exists a term v with $t \to^* v \;{}^*\!\leftarrow u$ whenever $t \;{}^*\!\leftarrow s \to^* u$, for all terms s, t and u. For confluence analysis we need to rewrite constrained terms.

A *constrained term* is a pair $s\ [\varphi]$ consisting of a term s and a constraint φ. Two constrained terms $s\ [\varphi]$ and $t\ [\psi]$ are *equivalent*, denoted by $s\ [\varphi] \sim t\ [\psi]$, if for every substitution $\gamma \vDash \varphi$ with $\mathcal{D}\mathsf{om}(\gamma) = \mathcal{V}\mathsf{ar}(\varphi)$ there is some substitution $\delta \vDash \psi$ with $\mathcal{D}\mathsf{om}(\delta) = \mathcal{V}\mathsf{ar}(\psi)$ such that $s\gamma = t\delta$, and vice versa. Let $s\ [\varphi]$ be a constrained term. If $s|_p = \ell\sigma$ for some constrained rewrite rule $\rho\colon \ell \to r\ [\psi] \in \mathcal{R}_{\mathsf{rc}}$, position p, and substitution σ such that $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}\mathsf{ar}(\varphi)$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$, φ is satisfiable and $\varphi \Rightarrow \psi\sigma$ is valid then $s\ [\varphi] \to_{\mathcal{R}} s[r\sigma]_p\ [\varphi]$. The rewrite relation $\xrightarrow{\sim}_{\mathcal{R}}$ on constrained terms is defined as $\sim \cdot \to_{\mathcal{R}} \cdot \sim$ and $s\ [\varphi] \xrightarrow{\sim}_p t\ [\psi]$ indicates that the rewrite step in $\xrightarrow{\sim}_{\mathcal{R}}$ takes place at position p. Similarly, we write $s\ [\varphi] \xrightarrow{\sim}_{\geqslant p} t\ [\psi]$ if the position in the rewrite step is below position p. Note that in our definition of $\to_{\mathcal{R}}$ the constraint is not modified. This equals [23, Definition 2.15], but is different from [45, 64] where calculation steps $s[f(v_1, \ldots, v_n)]_p\ [\varphi] \to s[v]_p\ [\varphi \land v = f(v_1, \ldots, v_n)]$ modify the constraint. However, the relation $\xrightarrow{\sim}$ can simulate the relation $\to_{\mathcal{R}}$ from [45, 64] as exemplified below.

Example 5.1. Consider the constrained term $x + 1\ [x > 3]$. Calculation steps as defined in [45,64] permit $x+1\ [x > 3] \to z\ [z = x+1 \land x > 3]$. In our setting, an initial equivalence step is required to introduce the fresh variable z and the corresponding assignment needed to perform a calculation: $x+1\ [x > 3] \sim x+1\ [z = x+1 \land x > 3] \to z\ [z = x+1 \land x > 3]$.

Our treatment allows for a much simpler definition of parallel and multi-step rewriting since we do not have to merge different constraints.

Equivalence on Constrained Terms

The equivalence on constrained terms $\sim$ used in this paper also differs from the equivalence relation used in [45, 64], which we will denote by $\sim'$. In $\sim'$ the domain of substitutions is not restricted, i.e., $s\ [\varphi] \sim' t\ [\psi]$ if and only if for all substitutions $\gamma \vDash \varphi$ there exists a substitution δ where $\delta \vDash \psi$ and $s\gamma = t\delta$. Intuitively, constrained terms are equivalent with respect to $\sim'$ if their sets of "allowed" instances are equivalent, while for $\sim$ we only instantiate variables appearing in the constraints and therefore representing some value. We have $\sim\, \subsetneq\, \sim'$. This can be seen as follows. First of all, any substitution γ with $\gamma \vDash \varphi$ can be split into γ_1 and γ_2 such that $\gamma = \gamma_1 \cup \gamma_2 = \gamma_1\gamma_2$ with $\mathcal{D}\mathsf{om}(\gamma_1) = \mathcal{V}\mathsf{ar}(\varphi)$ and $\gamma_1 \vDash \varphi$. From $s\ [\varphi] \sim t\ [\psi]$ we obtain a substitution δ_1 where $\mathcal{D}\mathsf{om}(\delta_1) = \mathcal{V}\mathsf{ar}(\psi)$, $\delta_1 \vDash \psi$ and $s\gamma_1 = t\delta_1$. Hence also $s\gamma = s\gamma_1\gamma_2 = t\delta_1\gamma_2 = t\delta$ for $\delta = \delta_1\gamma_2$, which implies $s\ [\varphi] \sim' t\ [\psi]$. However, $\sim'\, \subseteq\, \sim$ does not hold since $x\ [\mathsf{true}] \sim' y\ [\mathsf{true}]$ and $x\ [\mathsf{true}] \not\sim y\ [\mathsf{true}]$.

The change is necessary, since we have to differentiate (non-logical) variables in constrained terms from one another, to keep track of them through rewrite sequences. Take the (LC)TRS $\mathcal{R}$ consisting of the rule $\mathsf{f}(x, y) \to x$. When rewriting unconstrained terms we have $\mathsf{f}(x, y) \to_{\mathcal{R}} x$ and $\mathsf{f}(x, y) \not\to_{\mathcal{R}} y$. When rewriting on constrained terms with respect to $\sim'$, however, we have $\mathsf{f}(x, y)\ [\mathsf{true}] \sim' \cdot \to \cdot \sim' x\ [\mathsf{true}]$ and $\mathsf{f}(x, y)\ [\mathsf{true}] \sim' \cdot \to \cdot \sim' y\ [\mathsf{true}]$, losing any information connecting the resulting variable to the initial term. This is especially problematic in our analysis of parallel critical pairs in Section 5.6, where keeping track of variables through rewrite sequences is essential. Note that $\mathsf{f}(x, y)\ [\mathsf{true}] \xrightarrow{\sim} x\ [\mathsf{true}]$ but not $\mathsf{f}(x, y)\ [\mathsf{true}] \xrightarrow{\sim} y\ [\mathsf{true}]$.

5.3 Undecidability

Confluence is a decidable property of finite terminating TRSs, a celebrated result of Knuth and Bendix [37] which forms the basis of completion. For LCTRSs matters are more complicated.

Theorem 5.2. *Local confluence is undecidable for terminating* LCTRS*s.*

Proof. We use a reduction from PCP [62]. Let $P = \{(\alpha_1, \beta_1), \dots, (\alpha_N, \beta_N)\}$ with $\alpha_1, \dots, \alpha_N, \beta_1, \dots, \beta_N \in \{0, 1\}^+$ be an instance of PCP, where we assume that $\alpha_i \neq \beta_i$ for at least one $i \in \{1, \dots, N\}$. This entails no loss of generality, since instances that violate this assumption are trivially solvable. We encode candidate strings over $\{1, \dots, N\}$ as natural numbers where the empty string ϵ is represented by $[\epsilon] = 0$, and a non-empty string $i_0 i_1 \cdots i_k$ is represented by $[i_0 i_1 \cdots i_k] = N \cdot [i_1 \cdots i_k] + i_0$. So $[i_0 i_1 \cdots i_k] = i_0 + i_1 \cdot N + \cdots + i_k \cdot N^k$. For instance, assuming $N = 3$, the number 102 encodes the candidate string $\underline{3313}$ since $102 = 3 \cdot 33 + \underline{3}$, $33 = 3 \cdot 10 + \underline{3}$, $10 = 3 \cdot 3 + \underline{1}$ and $3 = 3 \cdot 0 + \underline{3}$. Conversely, the candidate string $\underline{112}$ is mapped to $22 = \underline{1} + \underline{1} \cdot 3^1 + \underline{2} \cdot 3^2$. It is not difficult to see that this results in a bijection between $\mathbb{N}$ and candidate strings, for each $N > 0$.

The LCTRS $\mathcal{R}_P$ that we construct is defined over the theory Ints, with theory symbols $\mathcal{F}_{\mathsf{th}} = \{>, +, \cdot, =, \wedge\} \cup \mathcal{V}\mathsf{al}$ and values $\mathcal{V}\mathsf{al} = \mathbb{B} \cup \mathbb{Z}$, with the additional sorts PCP and

String and the following term signature:

$$\begin{aligned} \mathsf{e} &: \mathsf{String} & \mathsf{0}, \mathsf{1} &: \mathsf{String} \to \mathsf{String} \\ \mathsf{start}, \top, \bot &: \mathsf{PCP} & \mathsf{test} &: \mathsf{String} \times \mathsf{String} \to \mathsf{PCP} \\ \mathsf{alpha}, \mathsf{beta} &: \mathsf{Int} \to \mathsf{String} \end{aligned}$$

The LCTRS $\mathcal{R}_P$ consists of the following rules:

$$\begin{aligned} \mathsf{start} &\to \mathsf{test}(\mathsf{alpha}(n), \mathsf{beta}(n)) \quad [n > 0] \\ \mathsf{test}(\mathsf{e}, \mathsf{e}) &\to \top \\ \mathsf{test}(\mathsf{0}(x), \mathsf{0}(y)) &\to \mathsf{test}(x, y) & \mathsf{test}(\mathsf{0}(x), \mathsf{1}(y)) &\to \bot \\ \mathsf{test}(\mathsf{1}(x), \mathsf{1}(y)) &\to \mathsf{test}(x, y) & \mathsf{test}(\mathsf{1}(x), \mathsf{0}(y)) &\to \bot \\ \mathsf{test}(\mathsf{0}(x), \mathsf{e}) &\to \bot & \mathsf{test}(\mathsf{e}, \mathsf{0}(y)) &\to \bot \\ \mathsf{test}(\mathsf{1}(x), \mathsf{e}) &\to \bot & \mathsf{test}(\mathsf{e}, \mathsf{1}(y)) &\to \bot \\ \mathsf{alpha}(0) &\to \mathsf{e} & \mathsf{beta}(0) &\to \mathsf{e} \end{aligned}$$

and, for all $i \in \{1, \ldots, N\}$,

$$\begin{aligned} \mathsf{alpha}(n) &\to \alpha_i(\mathsf{alpha}(m)) \quad [N \cdot m + i = n \wedge n > 0] \\ \mathsf{beta}(n) &\to \beta_i(\mathsf{beta}(m)) \quad [N \cdot m + i = n \wedge n > 0] \end{aligned}$$

Here, for a string $\gamma \in \{0, 1\}^*$ and a term $t : \mathsf{String}$, $\gamma(t) : \mathsf{String}$ is defined as

$$\gamma(t) = \begin{cases} t & \text{if } \gamma = \epsilon \\ \mathsf{0}(\gamma'(t)) & \text{if } \gamma = 0\gamma' \\ \mathsf{1}(\gamma'(t)) & \text{if } \gamma = 1\gamma' \end{cases}$$

Note that in the constraints n and m are variables, while N and i are values. Hence all constraints are in the decidable fragment of linear integer arithmetic and the rewrite relation $\to_{\mathcal{R}_P}$ is computable.

We claim that $\mathcal{R}_P$ is locally confluent if and only if P has no solution. The LCTRS $\mathcal{R}_P$ admits the constrained critical pair

$$\mathsf{test}(\mathsf{alpha}(n), \mathsf{beta}(n)) \approx \mathsf{test}(\mathsf{alpha}(m), \mathsf{beta}(m)) \quad [n > 0 \wedge m > 0]$$

with $n \neq m$. The rules with left-hand sides $\mathsf{alpha}(n)$ and $\mathsf{beta}(n)$ give rise to further constrained critical pairs but these are harmless since for all $n, N > 0$ there are unique numbers i and m satisfying the constraint $[N \cdot m + i = n \wedge n > 0]$.

By construction of the rules for test, $\mathsf{test}(\mathsf{alpha}(n), \mathsf{beta}(n)) \to^* \top$ if n represents a solution of P and $\mathsf{test}(\mathsf{alpha}(n), \mathsf{beta}(n)) \to^* \bot$ if n does not represent a solution of P. Since we assume that P is non-trivial, the latter happens for some $n > 0$. Hence all instances of the constrained critical pairs can only be joined if $\mathsf{test}(\mathsf{alpha}(n), \mathsf{beta}(n)) \to^* \bot$ for all $n > 0$. Hence $\mathcal{R}_P$ is locally confluent if and only if P has no solution.

The LCTRS $\mathcal{R}_P$ is terminating by the recursive path order [45] with the precedence $\mathsf{start} > \mathsf{test} > \mathsf{alpha} > \mathsf{beta} > \mathsf{1} > \mathsf{0} > \mathsf{e} > \top > \bot$ and the well-founded order $\sqsupset_{\mathsf{Int}}$ on integers where $x \sqsupset_{\mathsf{Int}} y$ if and only if $x > y$ and $x \geqslant 0$. The key observation is that the constraint $[N \cdot m + i = n \wedge n > 0]$ in the recursive rules for alpha and beta ensure $n > m$ since $N > 0$ and $i \geqslant 1$. $\square$

A key difference between TRSs and LCTRSs leading to this undecidability result can be seen in the first rule: $\mathsf{start} \to \mathsf{test}(\mathsf{alpha}(n), \mathsf{beta}(n))\ [n > 0]$. Plain TRSs usually do not allow variables appearing only in the right-hand side of a rule, as is the case for n here, because then termination never holds. However, in LCTRSs such variables are useful, since they can be used to model computations on arbitrary values which are often used to represent user input in program analysis. For $\mathcal{R}_P$ this leads to infinitely many possible steps starting from the term start and in turn to infinitely many critical pairs, breaking decidability.

5.4 Transformation

In this section we present a simple transformation from LCTRSs to possibly infinite TRSs, which exactly corresponds to the intuition behind LCTRSs. This allows us to lift results on TRSs more easily to LCTRSs than previously possible.

Definition 5.3. Given an LCTRS $\mathcal{R}$, the TRS $\overline{\mathcal{R}}$ consists of the following rules: $\ell\tau \to r\tau$ for all $\rho\colon \ell \to r\ [\varphi] \in \mathcal{R}_{\mathsf{rc}}$ with $\tau \vDash \rho$ and $\mathcal{D}\mathsf{om}(\tau) = \mathcal{LV}\mathsf{ar}(\rho)$.

Note that $\overline{\mathcal{R}}$ typically consists of infinitely many rules.

Lemma 5.4. *The rewrite relations of $\mathcal{R}$ and $\overline{\mathcal{R}}$ are the same. Moreover $\to_{p,\mathcal{R}} = \to_{p,\overline{\mathcal{R}}}$ for all positions p.*

Proof. We first show $\to_{p,\mathcal{R}} \subseteq \to_{p,\overline{\mathcal{R}}}$. Assume $s \to_{p,\mathcal{R}} t$. We have $s = s[\ell\sigma]_p \to s[r\sigma]_p = t$ for some $\rho\colon \ell \to r\ [\varphi] \in \mathcal{R}_{\mathsf{rc}}$ and $\sigma \vDash \rho$. We split σ into two substitutions $\tau = \{x \mapsto \sigma(x) \mid x \in \mathcal{LV}\mathsf{ar}(\rho)\}$ and $\delta = \{x \mapsto \sigma(x) \mid x \in \mathcal{V}\mathsf{ar}(\ell) \setminus \mathcal{LV}\mathsf{ar}(\rho)\}$. From $\sigma \vDash \rho$ we infer $\tau \vDash \rho$ and thus $\tau(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$. Hence $\sigma = \tau \cup \delta = \tau\delta$. We have $\ell\tau \to r\tau \in \overline{\mathcal{R}}$. Hence $s = s[\ell\tau\delta]_p \to_{p,\overline{\mathcal{R}}} s[r\tau\delta]_p = t$ as desired. To show the reverse inclusion $\to_{p,\overline{\mathcal{R}}} \subseteq \to_{p,\mathcal{R}}$ we assume $s \to_{p,\overline{\mathcal{R}}} t$. Otherwise $s = s[\ell\mu\nu]_p \to_{p,\overline{\mathcal{R}}} s[r\mu\nu]_p$ for some rule $\rho\colon \ell \to r\ [\varphi] \in \mathcal{R}$ with $\mu \vDash \rho$. Let $\sigma = \mu\nu$. Since $\mu(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$, we have $x\sigma = x\mu$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$. Hence $\sigma \vDash \rho$ and thus $s = s[\ell\sigma]_p \to_{p,\mathcal{R}} s[r\sigma]_p = t$. $\square$

Since $\to_{\mathcal{R}}$ and $\to_{\overline{\mathcal{R}}}$ coincide, we drop the subscript in the sequel. We write $\mathcal{EV}\mathsf{ar}(\ell \to r\ [\varphi])$ for the set $\mathcal{V}\mathsf{ar}(r) \setminus (\mathcal{V}\mathsf{ar}(\ell) \cup \mathcal{V}\mathsf{ar}(\varphi))$ of extra variables of a rule. In the computation of constrained critical pairs these variables of the overlapping rules would lose the property of being a logical variable without adding trivial constraints. Given a constrained rewrite rule ρ, we write $\mathcal{EC}_\rho$ for $\bigwedge\{x = x \mid x \in \mathcal{EV}\mathsf{ar}(\rho)\}$. The set of positions in a term s is denoted by $\mathcal{P}\mathsf{os}(s)$. We write ϵ for the root position and $\mathcal{P}\mathsf{os}_{\mathcal{F}}(s)$ for the set of positions of function symbols in s.

Definition 5.5. An *overlap* of an LCTRS $\mathcal{R}$ is a triple $\langle \rho_1, p, \rho_2 \rangle$ with rules $\rho_1\colon \ell_1 \to r_1\ [\varphi_1]$ and $\rho_2\colon \ell_2 \to r_2\ [\varphi_2]$, satisfying the following conditions: (1) ρ_1 and ρ_2 are variable-disjoint variants of rewrite rules in $\mathcal{R}_{\mathsf{rc}}$, (2) $p \in \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell_2)$, (3) ℓ_1 and $\ell_2|_p$ unify with mgu σ such that $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2)$, (4) $\varphi_1\sigma \wedge \varphi_2\sigma$ is satisfiable, and (5) if $p = \epsilon$ then ρ_1 and ρ_2 are not variants, or $\mathcal{V}\mathsf{ar}(r_1) \nsubseteq \mathcal{V}\mathsf{ar}(\ell_1)$. In this case we call $\ell_2\sigma[r_1\sigma]_p \approx r_2\sigma\ [\varphi_1\sigma \wedge \varphi_2\sigma \wedge \psi\sigma]$ a *constrained critical pair* obtained from the overlap $\langle \rho_1, p, \rho_2 \rangle$. Here $\psi = \mathcal{EC}_{\rho_1} \wedge \mathcal{EC}_{\rho_2}$. The peak $\ell_2\sigma[r_1\sigma]_p\ [\Phi] \leftarrow \ell_2\sigma\ [\Phi] \to_\epsilon r_2\sigma\ [\Phi]$ with $\Phi = (\varphi_1 \wedge \varphi_2 \wedge \psi)\sigma$, from which the constrained critical pair originates, is called a *constrained critical peak*. The set of all constrained critical pairs of $\mathcal{R}$ is denoted by $\mathsf{CCP}(\mathcal{R})$. A constrained critical pair $s \approx t\ [\varphi]$ is *trivial* if $s\sigma = t\sigma$ for every substitution σ with $\sigma \vDash \varphi$.

A key ingredient of our approach is to relate critical pairs of the transformed TRS to constrained critical pairs of the original LCTRS.

Theorem 5.6. *For every critical pair $s \approx t$ of $\overline{\mathcal{R}}$ there exists a constrained critical pair $s' \approx t'\ [\varphi']$ of $\mathcal{R}$ and a substitution γ such that $s = s'\gamma$, $t = t'\gamma$ and $\gamma \vDash \varphi'$.*

Proof. Let $s \approx t$ be a critical pair of $\overline{\mathcal{R}}$, originating from the critical peak $\ell_2\mu\sigma[r_1\nu\sigma]_p \leftarrow \ell_2\mu\sigma = \ell_2\mu\sigma[\ell_1\nu\sigma]_p \to r_2\mu\sigma$ with variants $\rho_1\colon \ell_1 \to r_1\ [\varphi_1]$ and $\rho_2\colon \ell_2 \to r_2\ [\varphi_2]$ of rules in $\mathcal{R}_{\mathsf{rc}}$ without shared variables. Let $\psi_i = \mathcal{EC}_{\rho_i}$ for $i \in \{1,2\}$. Furthermore we have $\mathcal{D}\mathsf{om}(\nu) = \mathcal{LV}\mathsf{ar}(\rho_1)$, $\mathcal{D}\mathsf{om}(\mu) = \mathcal{LV}\mathsf{ar}(\rho_2)$, $\nu \vDash \varphi_1 \wedge \psi_1$, $\mu \vDash \varphi_2 \wedge \psi_2$, $p \in \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell_2\mu)$, and σ is an mgu of $\ell_2\mu|_p$ and $\ell_1\nu$. Moreover, if $p = \epsilon$ then $\ell_1\nu \to r_1\nu$ and $\ell_2\mu \to r_2\mu$ are not variants. Define $\tau = \nu \uplus \mu$. We have $\mathcal{D}\mathsf{om}(\tau) = \mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2)$. Let $\varphi = \varphi_1 \wedge \varphi_2 \wedge \psi_1 \wedge \psi_2$. Clearly, $\ell_1\tau = \ell_1\nu$, $r_1\tau = r_1\nu$, $\ell_2\tau = \ell_2\mu$, $r_2\tau = r_2\mu$ and $\tau \vDash \varphi$. Hence the given peak can be written as $\ell_2\tau\sigma[r_1\tau\sigma]_p \leftarrow \ell_2\tau\sigma = \ell_2\tau\sigma[\ell_1\tau\sigma]_p \to r_2\tau\sigma$ and $\tau \vDash \varphi$. Since $\ell_2|_p\tau\sigma = \ell_1\tau\sigma$ there exists an mgu δ of $\ell_2|_p$ and ℓ_1, and a substitution γ such that $\delta\gamma = \tau\sigma$. Let $s' = \ell_2\delta[r_1\delta]_p$ and $t' = r_2\delta$. We claim that $\langle \rho_1, p, \rho_2 \rangle$ is an overlap of $\mathcal{R}$, resulting in the constrained critical pair $s' \approx t'\ [\varphi\delta]$. Condition (1) of Definition 5.5 is trivially satisfied. For condition (2) we need to show $p \in \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell_2)$. This follows from $p \in \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell_2\mu)$, $\mu(x) \in \mathcal{V}\mathsf{al}$ for every $x \in \mathcal{D}\mathsf{om}(\mu)$, and $\mathsf{root}(\ell_2\mu|_p) = \mathsf{root}(\ell_1\nu) \in \mathcal{F} \setminus \mathcal{V}\mathsf{al}$. For condition (3) it remains to show that $\delta(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2)$. Suppose to the contrary that $\mathsf{root}(\delta(x)) \in \mathcal{F} \setminus \mathcal{V}\mathsf{al}$ for some $x \in \mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2)$. Then $\mathsf{root}(\delta(x)) = \mathsf{root}(\gamma(\delta(x))) = \mathsf{root}(\sigma(\tau(x))) \in \mathcal{F} \setminus \mathcal{V}\mathsf{al}$, which contradicts $\tau \vDash \varphi$. Condition (4) follows from the identity $\delta\gamma = \tau\sigma$ together with $\tau \vDash \varphi$ which imply $\delta\gamma \vDash \varphi$ and thus $\varphi\delta$ is satisfiable. Hence also $\varphi_1\delta \wedge \varphi_2\delta$ is satisfiable. It remains to show condition (5), so let $p = \epsilon$ and further assume that ρ_1 and ρ_2 are variants. So there exists a variable renaming π such that $\rho_1\pi = \rho_2$. In particular, $\ell_1\pi = \ell_2$ and $r_1\pi = r_2$. Let $x \in \mathcal{V}\mathsf{ar}(\ell_1)$. If $x \in \mathcal{LV}\mathsf{ar}(\rho_1) = \mathcal{D}\mathsf{om}(\nu)$ then $\tau(x) = \nu(x) \in \mathcal{V}\mathsf{al}$. Moreover, $\pi(x) \in \mathcal{LV}\mathsf{ar}(\rho_2) = \mathcal{D}\mathsf{om}(\mu)$ and thus $\tau(\pi(x)) = \mu(\pi(x)) \in \mathcal{V}\mathsf{al}$. Since $\ell_1\tau$ and $\ell_2\tau$ are unifiable, $\pi(\tau(x)) = \tau(x) = \tau(\pi(x))$. If $x \notin \mathcal{LV}\mathsf{ar}(\rho_1)$ then $\tau(x) = x$, $\pi(x) \notin \mathcal{LV}\mathsf{ar}(\rho_2)$ and similarly $\tau(\pi(x)) = \pi(x) = \pi(\tau(x))$. All in all, $\ell_1\tau\pi = \ell_1\pi\tau = \ell_2\tau$. Now, if $\mathcal{V}\mathsf{ar}(r_1) \subseteq \mathcal{V}\mathsf{ar}(\ell_1)$ then we obtain $r_1\tau\pi = r_1\pi\tau = r_2\tau$, contradicting the fact that $\ell_1\nu \to r_1\nu$ and $\ell_2\mu \to r_2\mu$ are not variants. We conclude that $s' \approx t'\ [\varphi\delta]$ is a constrained critical pair of $\mathcal{R}$. So we can take $\varphi' = \varphi\delta$. Clearly, $s = s'\gamma$ and $t = t'\gamma$. Moreover, $\gamma \vDash \varphi'$ since $\varphi'\gamma = \varphi\tau\sigma = \varphi\tau$ and $\tau \vDash \varphi$. □

The converse does not hold in general.

Example 5.7. Consider the LCTRS $\mathcal{R}$ consisting of the single rule $\mathsf{a} \to x\ [x = 0]$ where the variable x ranges over the integers. Since x appears on the right-hand side but not the left, we obtain a constrained critical pair $x \approx x'\ [x = 0 \wedge x' = 0]$. Since the constraint uniquely determines the values of x and x', the TRS $\overline{\mathcal{R}}$ consists of the single rule $\mathsf{a} \to 0$. Obviously $\overline{\mathcal{R}}$ has no critical pairs.

The above example also shows that orthogonality of $\overline{\mathcal{R}}$ does not imply orthogonality of $\mathcal{R}$. However, the counterexample relies somewhat on a technicality in condition (5) of Definition 5.5. It only occurs when the two rules $\ell_1 \to r_1\ [\varphi_1]$ and $\ell_2 \to r_2\ [\varphi_2]$ involved in the critical pair overlap at the root and have instances $\ell_1\tau_1 \to r_1\tau_1$ and $\ell_2\tau_2 \to r_2\tau_2$ in $\overline{\mathcal{R}}$ which are variants of each other. By dealing with such cases separately we can prove the following theorem.

Theorem 5.8. *For every constrained critical pair $s \approx t\ [\varphi]$ of $\mathcal{R}$ and every substitution σ with $\sigma \vDash \varphi$, (1) $s\sigma = t\sigma$ or (2) there exist a critical pair $u \approx v$ of $\overline{\mathcal{R}}$ and a substitution δ such that $s\sigma = u\delta$ and $t\sigma = v\delta$.*

Proof. Let $s \approx t\ [\varphi]$ be a constrained critical pair of $\mathcal{R}$ originating from the critical peak $s = \ell_2\theta[r_1\theta]_p \leftarrow \ell_2\theta[\ell_1\theta]_p \to r_2\theta = t$ with variants $\rho_1\colon \ell_1 \to r_1\ [\varphi_1]$ and $\rho_2\colon \ell_2 \to r_2\ [\varphi_2]$ of rules in $\mathcal{R}_{\mathsf{rc}}$, and an mgu θ of $\ell_2|_p$ and ℓ_1 where $p \in \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell_2)$. Moreover $\theta(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2)$, and $\varphi = \varphi_1\theta \wedge \varphi_2\theta \wedge \psi\theta$ with $\psi = \mathcal{EC}_{\rho_1} \wedge \mathcal{EC}_{\rho_2}$. Let σ be a substitution with $\sigma \vDash \varphi$. Hence $\theta\sigma \vDash \varphi_1 \wedge \varphi_2 \wedge \psi$ and further $\sigma(\theta(x)) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2)$. We split $\theta\sigma$ into substitutions τ_1, τ_2 and π as follows: $\tau_i(x) = x\theta\sigma$ if $x \in \mathcal{LV}\mathsf{ar}(\rho_i)$ and $\tau_i(x) = x$ otherwise, for $i \in \{1,2\}$, and $\pi(x) = x\theta\sigma$ if $x \in \mathcal{D}\mathsf{om}(\theta\sigma) \setminus (\mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2))$ and $\pi(x) = x$ otherwise. From $\theta\sigma \vDash \varphi_1 \wedge \varphi_2 \wedge \psi$ and $\mathcal{V}\mathsf{ar}(\varphi_i) \subseteq \mathcal{LV}\mathsf{ar}(\rho_i)$ we infer $\tau_i \vDash \varphi_i$ for $i \in \{1,2\}$. Since $\mathcal{D}\mathsf{om}(\tau_i) = \mathcal{LV}\mathsf{ar}(\rho_i)$, $\ell_i\tau_i \to r_i\tau_i \in \overline{\mathcal{R}}$ for $i \in \{1,2\}$. Furthermore, $\tau_i\pi = \tau_i \cup \pi$ for $i \in \{1,2\}$. Hence $\ell_2|_p\tau_2\pi = \ell_2|_p\theta\sigma = \ell_1\theta\sigma = \ell_1\tau_1\pi$, implying that $\ell_2|_p\tau_2$ and $\ell_1\tau_1$ are unifiable. Let γ be an mgu of these two terms. There exists a substitution δ such that $\gamma\delta = \pi$. Clearly $p \in \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell_2\tau_2)$. If $p \neq \epsilon$ or $\ell_1\tau_1 \to r_1\tau_1$ and $\ell_2\tau_2 \to r_2\tau_2$ are not variants, then $u \approx v$ with $u = \ell_2\tau_2\gamma[r_1\tau_1\gamma]_p$ and $v = r_2\tau_2\gamma$ is a critical pair of $\overline{\mathcal{R}}$. Moreover $t\sigma = r_2\theta\sigma = r_2\tau_2\pi = r_2\tau_2\gamma\delta = v\delta$, and similarly $s\sigma = u\delta$. Thus option (2) is satisfied. If $p = \epsilon$ and $\ell_1\tau_1 \to r_1\tau_1$ and $\ell_2\tau_2 \to r_2\tau_2$ are variants then $s\sigma = r_1\tau_1\gamma\delta = r_2\tau_2\gamma\delta = t\sigma$, fulfilling (1). ☐

A TRS (LCTRS) is weakly orthogonal if it is left-linear and all its (constrained) critical pairs are trivial. Since $\overline{\mathcal{R}}$ is left-linear if and only if $\mathcal{R}$ is left-linear, a direct consequence of Theorem 5.8 is that weak orthogonality of $\overline{\mathcal{R}}$ implies weak orthogonality of $\mathcal{R}$.

Our transformation is not only useful for confluence analysis.

Example 5.9. For the LCTRS $\mathcal{R}_P$ in the proof of Theorem 5.2 the TRS $\overline{\mathcal{R}}_P$ consists of all unconstrained rules of $\mathcal{R}_P$ together with $f(v_1, \dots, v_n) \to [\![f(v_1, \dots, v_n)]\!]$ for all $f \in \mathcal{F}_{\mathsf{th}} \setminus \mathcal{V}\mathsf{al}$ and $v_1, \dots, v_n \in \mathcal{V}\mathsf{al}$, $\mathsf{start} \to \mathsf{test}(\mathsf{alpha}(n), \mathsf{beta}(n))$ for all $n > 0$, $\mathsf{alpha}(n) \to \alpha_i(\mathsf{alpha}(m))$ and $\mathsf{beta}(n) \to \beta_i(\mathsf{beta}(m))$ for all $i \in \{1, \dots, N\}$, $n > 0$ and $m \geqslant 0$ such that $N \cdot m + i = n$. Termination of the infinite TRS $\overline{\mathcal{R}}_P$ is easily shown by LPO or dependency pairs.

5.5 Development Closed Critical Pairs

Using Theorem 5.6 we can easily transfer confluence criteria for TRSs to LCTRSs. Rather than reproving the confluence results reported in [45, 64, 88], in this section we illustrate this by extending the result of van Oostrom [87] concerning (almost) development closed critical pairs from TRSs to LCTRSs. This result subsumes most critical-pair based confluence criteria, as can be seen in Figure 5.2 in the concluding section.

Definition 5.10. Let $\mathcal{R}$ be an LCTRS. The multi-step relation $\twoheadrightarrow\!\!\!\!\!\circ\,$ on terms is defined inductively as follows: (1) $x \;\circ\!\!\!\twoheadrightarrow\; x$ for all variables x, (2) $f(s_1, \ldots, s_n) \;\circ\!\!\!\twoheadrightarrow\; f(t_1, \ldots, t_n)$ if $s_i \;\circ\!\!\!\twoheadrightarrow\; t_i$ with $1 \leqslant i \leqslant n$, (3) $\ell\sigma \;\circ\!\!\!\twoheadrightarrow\; r\tau$ if $\ell \to r\ [\varphi] \in \mathcal{R}_{\mathsf{rc}}$, $\sigma \vDash \ell \to r\ [\varphi]$ and $\sigma \;\circ\!\!\!\twoheadrightarrow\; \tau$, where $\sigma \;\circ\!\!\!\twoheadrightarrow\; \tau$ denotes $\sigma(x) \;\circ\!\!\!\twoheadrightarrow\; \tau(x)$ for all variables $x \in \mathcal{D}\mathsf{om}(\sigma)$.

Definition 5.11. A critical pair $s \approx t$ is *development closed* if $s \;\circ\!\!\!\twoheadrightarrow\; t$. It is *almost development closed* if it is not an overlay and development closed, or it is an overlay and $s \;\circ\!\!\!\twoheadrightarrow\; \cdot\ {}^*\!\!\leftarrow t$. A TRS is called (almost) development closed if all its critical pairs are (almost) development closed.

The following result from [87] has recently been formalized in Isabelle [38, 39].

Theorem 5.12. *Left-linear almost development closed* TRSs *are confluent.* ☐

We define multi-step rewriting on constrained terms.

Definition 5.13. Let $\mathcal{R}$ be an LCTRS. The multi-step relation $\circ\!\!\!\twoheadrightarrow$ on constrained terms is defined inductively as follows:

1. $x\ [\varphi] \;\circ\!\!\!\twoheadrightarrow\; x\ [\varphi]$ for all variables x,
2. $f(s_1, \ldots, s_n)\ [\varphi] \;\circ\!\!\!\twoheadrightarrow\; f(t_1, \ldots, t_n)\ [\varphi]$ if $s_i\ [\varphi] \;\circ\!\!\!\twoheadrightarrow\; t_i\ [\varphi]$ for $1 \leqslant i \leqslant n$,
3. $\ell\sigma\ [\varphi] \;\circ\!\!\!\twoheadrightarrow\; r\tau\ [\varphi]$ if $\rho\colon \ell \to r\ [\psi] \in \mathcal{R}_{\mathsf{rc}}$, $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}\mathsf{ar}(\varphi)$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$, φ is satisfiable, $\varphi \Rightarrow \psi\sigma$ is valid, and $\sigma\ [\varphi] \;\circ\!\!\!\twoheadrightarrow\; \tau\ [\varphi]$.

Here $\sigma\ [\varphi] \;\circ\!\!\!\twoheadrightarrow\; \tau\ [\varphi]$ denotes $\sigma(x)\ [\varphi] \;\circ\!\!\!\twoheadrightarrow\; \tau(x)\ [\varphi]$ for all variables $x \in \mathcal{D}\mathsf{om}(\sigma)$. The relation $\tilde{\circ\!\!\!\twoheadrightarrow}$ on constrained terms is defined as $\sim \cdot \circ\!\!\!\twoheadrightarrow \cdot \sim$.

Example 5.14. Consider the following LCTRS $\mathcal{R}$ over the theory Ints with the rules:

$$\mathsf{max}(x, y) \to x\ [x \geqslant y] \qquad\qquad \mathsf{max}(x, y) \to y\ [y \geqslant x]$$

Rewriting the term $\mathsf{max}(1 + 2, 3 + 2)$ to its normal form 5 requires three single steps. These steps can be combined into a single multi-step $\mathsf{max}(1 + 2, 3 + 2) \;\circ\!\!\!\twoheadrightarrow\; 5$.

The constrained term $\mathsf{max}(1 + x, 3 + y)\ [x > 3 \wedge y = 1]$ rewrites in a single multi-step to its normal form $z\ [z = 1 + x \wedge x > 3]$. This involves the following parts of Definition 5.13.

$$\begin{tikzcd}
s \approx t \in \mathsf{CP}(\overline{\mathcal{R}}) \arrow[rr, Rightarrow, shift left, "\text{Theorem 5.6}"] \arrow[d, bend right, "\circ"'] \arrow[d, bend left, "*"] & & s' \approx t' ~[\varphi] \in \mathsf{CCP}(\mathcal{R}) \arrow[ll, Rightarrow, shift left, "\text{Lemma 5.18}"] \arrow[d, bend right, "\circ"'] \arrow[d, bend left, "*"] \\
\bullet & & \bullet
\end{tikzcd}$$

Figure 5.1: Proof idea for Theorem 5.19.

Let φ be $x > 3 \land y = 1 \land z = 1 + x \land z' = 3 + y$. Case (3) gives $1 + x ~[\varphi] \mathrel{\circ\!\!\!\rightarrow} z ~[\varphi]$ and $3 + y ~[\varphi] \mathrel{\circ\!\!\!\rightarrow} z' ~[\varphi]$. Using this we obtain $\mathsf{max}(1 + x, 3 + y) ~[\varphi] \mathrel{\circ\!\!\!\rightarrow} \mathsf{max}(z, z') ~[\varphi]$ by case (2). A final application of case (3) yields $\mathsf{max}(z, z') ~[\varphi] \mathrel{\circ\!\!\!\rightarrow} z ~[\varphi]$. Together with the equivalences

$$\begin{aligned}
\mathsf{max}(1 + x, 3 + y) ~[x > 3 \land y = 1] &\sim \mathsf{max}(1 + x, 3 + y) ~[\varphi] \\
z ~[\varphi] &\sim z ~[z = 1 + x \land x > 3]
\end{aligned}$$

we obtain $\mathsf{max}(1 + x, 3 + y) ~[x > 3 \land y = 1] \mathrel{\widetilde{\circ\!\!\!\rightarrow}} z ~[z = 1 + x \land x > 3]$.

Definition 5.11 is extended to LCTRSs as follows.

Definition 5.15. A constrained critical pair $s \approx t ~[\varphi]$ is *development closed* if $s \approx t ~[\varphi] \mathrel{\widetilde{\circ\!\!\!\rightarrow}}_{\geqslant 1} u \approx v ~[\psi]$ for some trivial $u \approx v ~[\psi]$. A constrained critical pair is *almost development closed* if it is not an overlay and development closed, or it is an overlay and $s \approx t ~[\varphi] \mathrel{\widetilde{\circ\!\!\!\rightarrow}}_{\geqslant 1} \cdot \tilde{\rightarrow}^*_{\geqslant 2} u \approx v ~[\psi]$ for some trivial $u \approx v ~[\psi]$. An LCTRS is called (almost) development closed if all its constrained critical pairs are (almost) development closed.

Similar to [64, 88], the symbol $\approx$ is treated as a fresh binary function symbol, resulting in constrained equations whose positions are addressed in the usual way. Therefore positions below 1 in $s \approx t ~[\varphi]$ refer to subterms of s.

Figure 5.1 conveys the idea how the main result (Theorem 5.19) in this section is obtained. For every critical pair in the transformed TRS $\overline{\mathcal{R}}$ there exists a corresponding constrained critical pair in the original LCTRS $\mathcal{R}$ (Theorem 5.6). Almost development closure of the constrained critical pair implies almost development closure of the critical pair (Lemma 5.18). Since the rewrite relations of $\mathcal{R}$ and $\overline{\mathcal{R}}$ coincide (Lemma 5.4), we obtain the confluence of almost development closed left-linear LCTRSs from the corresponding result in [87].

We now present a few technical results that relate rewrite sequences and multi-steps on (constrained) terms. These prepare for the use of Theorem 5.6 to obtain the confluence of (almost) development closed LCTRSs. The proofs of the following two lemmata can be found in [69].

Lemma 5.16. *Suppose $s \approx t ~[\varphi] \tilde{\rightarrow}^*_{\geqslant p} u \approx v ~[\psi]$ with $\gamma \vDash \varphi$ and position p. If $p = 1q$ for a position q then $s\gamma \rightarrow^*_{\geqslant q} u\delta$ and $t\gamma = v\delta$ for some substitution δ with $\delta \vDash \psi$. If $p = 2q$ for a position q then $s\gamma = u\delta$ and $t\gamma \tilde{\rightarrow}^*_{\geqslant q} v\delta$ for some substitution δ with $\delta \vDash \psi$.* ☐

Lemma 5.17. *If $s \approx t ~[\varphi] \mathrel{\widetilde{\circ\!\!\!\rightarrow}}_{\geqslant 1} u \approx v ~[\psi]$ then for all substitutions $\sigma \vDash \varphi$ there exists $\delta \vDash \psi$ such that $s\sigma \mathrel{\circ\!\!\!\rightarrow} u\delta$ and $t\sigma = v\delta$.* ☐

Lemma 5.18. *If a constrained critical pair $s \approx t\ [\varphi]$ is almost development closed then for all substitutions σ with $\sigma \vDash \varphi$ we have $s\sigma \mathrel{\circ\!\!\!\!\!\twoheadrightarrow} \cdot {}^{*}\!\leftarrow t\sigma$.*

Proof. Let $s \approx t\ [\varphi]$ be an almost development closed constrained critical pair, and $\sigma \vDash \varphi$ some substitution. From Definition 5.15 we obtain

$$s \approx t\ [\varphi] \mathrel{\overset{\sim}{\circ\!\!\!\!\!\twoheadrightarrow}}_{\geqslant 1} u' \approx v'\ [\psi'] \mathrel{\overset{\sim}{\rightarrow}}{}^{*}_{\geqslant 2}\, u \approx v\ [\psi] \tag{5.1}$$

where $u\tau = v\tau$ for all $\tau \vDash \psi$ for some constrained term $u' \approx v'\ [\psi']$. We apply Lemma 5.17 to the first step in (5.1). This yields a substitution δ where $s\sigma \mathrel{\circ\!\!\!\!\!\twoheadrightarrow} u'\delta$, $t\sigma = v'\delta$ and $\delta \vDash \psi'$. For the second part of (5.1) we use Lemma 5.16 and obtain $v'\delta \to^{*} v\gamma$, $u'\delta = u\gamma$ for some $\gamma \vDash \psi$. Moreover we have $u\gamma = v\gamma$. Hence $s\sigma \mathrel{\circ\!\!\!\!\!\twoheadrightarrow} u'\delta = u\gamma = v\gamma\ {}^{*}\!\leftarrow v'\delta = t\sigma$. ☐

Theorem 5.19. *If an* LCTRS $\mathcal{R}$ *is almost development closed then so is* $\overline{\mathcal{R}}$.

Proof. Take any critical pair $s \approx t$ from $\overline{\mathcal{R}}$. From Theorem 5.6 we know that there exists a constrained critical pair $s' \approx t'\ [\varphi]$ in $\mathcal{R}$ where $s'\sigma = s$ and $t'\sigma = t$ for some $\sigma \vDash \varphi$. Since the constrained critical pair must be almost development closed, Lemma 5.18 yields $s = s'\sigma \mathrel{\circ\!\!\!\!\!\twoheadrightarrow} \cdot {}^{*}\!\leftarrow t'\sigma = t$ if it is an overlay and $s = s'\sigma \mathrel{\circ\!\!\!\!\!\twoheadrightarrow} t'\sigma = t$ otherwise. This proves that $\overline{\mathcal{R}}$ is almost development closed. ☐

Interestingly, the converse does not hold, as seen in the following example.

Example 5.20. Consider the LCTRS $\mathcal{R}$ over the theory Ints with the rules

$$\begin{array}{ll} \mathsf{f}(x) \to \mathsf{g}(x) & \mathsf{g}(x) \to \mathsf{h}(2)\ [x = 2z] \\ \mathsf{f}(x) \to \mathsf{h}(x)\ [1 \leqslant x \leqslant 2] & \mathsf{g}(x) \to \mathsf{h}(1)\ [x = 2z + 1] \end{array}$$

The TRS $\overline{\mathcal{R}}$ consists of the rules

$$\begin{array}{llll} \mathsf{f}(x) \to \mathsf{g}(x) & \mathsf{f}(1) \to \mathsf{h}(1) & \mathsf{g}(n) \to \mathsf{h}(1) & \text{for all odd } n \in \mathbb{Z} \\ & \mathsf{f}(2) \to \mathsf{h}(2) & \mathsf{g}(n) \to \mathsf{h}(2) & \text{for all even } n \in \mathbb{Z} \end{array}$$

and has two (modulo symmetry) critical pairs $\mathsf{g}(1) \approx \mathsf{h}(1)$ and $\mathsf{g}(2) \approx \mathsf{h}(2)$. Since $\mathsf{g}(1) \mathrel{\circ\!\!\!\!\!\twoheadrightarrow} \mathsf{h}(1)$ and $\mathsf{g}(2) \mathrel{\circ\!\!\!\!\!\twoheadrightarrow} \mathsf{h}(2)$, $\overline{\mathcal{R}}$ is almost development closed. The constrained critical pair $\mathsf{g}(x) \approx \mathsf{h}(x)\ [1 \leqslant x \leqslant 2]$ is not almost development closed, since it is a normal form with respect to the rewrite relation on constrained terms.

This also makes intuitive sense, since a rewrite step $s \approx t\ [\varphi] \mathrel{\overset{\sim}{\rightarrow}} u \approx v\ [\psi]$ implies that the same step can be taken on all instances $s\sigma \approx t\sigma$ where $\sigma \vDash \varphi$. However it may be the case, like in the above example, that different instances of the constrained critical pair require different steps to obtain a closing sequence, which cannot directly be modeled using rewriting on constrained terms.

Since left-linearity of $\overline{\mathcal{R}}$ is preserved, the following corollary is obtained from Theorems 5.12 and 5.19. In fact $\mathcal{R}$ only has to be linear in the variables $x \notin \mathcal{LV}\mathsf{ar}$, since that is sufficient for $\overline{\mathcal{R}}$ to be linear.

Corollary 5.21. *Left-linear almost development closed* LCTRSs *are confluent.* ☐

Example 5.22. The LCTRS $\mathcal{R}$ over the theory Ints with the rules

$$\begin{array}{lll} \mathsf{f}(x,y) \to \mathsf{h}(\mathsf{g}(y, 2 \cdot 2))\ [x \leqslant y \wedge y = 2] & \mathsf{g}(x,y) \to \mathsf{g}(y,x) & \mathsf{h}(x) \to x \\ \mathsf{f}(x,y) \to \mathsf{c}(4,x)\ [y \leqslant x] & \mathsf{c}(x,y) \to \mathsf{g}(4,2)\ [x \neq y] & \end{array}$$

admits the two constrained critical pairs (with simplified constraints)

$$\mathsf{h}(\mathsf{g}(y, 2 \cdot 2)) \approx \mathsf{c}(4,x)\ [\varphi] \qquad\qquad \mathsf{c}(4,x) \approx \mathsf{h}(\mathsf{g}(y, 2 \cdot 2))\ [\varphi]$$

Both are almost development closed:

$$\begin{array}{ll} \mathsf{h}(\mathsf{g}(y, 2 \cdot 2)) \approx \mathsf{c}(4,x)\ [\varphi] & \mathsf{c}(4,x) \approx \mathsf{h}(\mathsf{g}(y, 2 \cdot 2))\ [\varphi] \\ \overset{\sim}{\circ\!\!\!\rightarrow}_{\geqslant 1} \mathsf{g}(4,2) \approx \mathsf{c}(4,x)\ [x = 2] & \overset{\sim}{\circ\!\!\!\rightarrow}_{\geqslant 1} \mathsf{g}(4,2) \approx \mathsf{h}(\mathsf{g}(y, 2 \cdot 2))\ [y = 2] \\ \overset{\sim}{\rightarrow}_{\geqslant 2} \mathsf{g}(4,2) \approx \mathsf{g}(4,2)\ [\mathsf{true}] & \overset{\sim}{\rightarrow}{}^{*}_{\geqslant 2} \mathsf{g}(4,2) \approx \mathsf{g}(4,2)\ [\mathsf{true}] \end{array}$$

Here φ is the constraint $x = y \wedge y = 2$. Hence $\mathcal{R}$ is almost development closed. Since $\mathcal{R}$ is left-linear, confluence follows by Corollary 5.21.

5.6 Parallel Critical Pairs

In this section we extend the confluence result by Toyama [83] based on parallel critical pairs to LCTRSs. Recently there is a renewed interest in this result; Shintani and Hirokawa proved in [73] that it subsumes Toyama's later confluence result in [84]. The latter was already lifted to LCTRSs in [64] and is also subsumed by Corollary 5.21. The result of Toyama [83] is a proper extension of the confluence criterion on parallel critical pairs by Gramlich [25]. In the sequel we mainly follow the notions from [73].

Definition 5.23. Let $\mathcal{R}$ be an LCTRS. The parallel rewrite relation $\mathrel{\parallel\!\!\!\rightarrow}$ on terms is defined inductively as follows:

1. $x \mathrel{\parallel\!\!\!\rightarrow} x$ for all variables x,
2. $f(s_1, \ldots, s_n) \mathrel{\parallel\!\!\!\rightarrow} f(t_1, \ldots, t_n)$ if $s_i \mathrel{\parallel\!\!\!\rightarrow} t_i$ for $1 \leqslant i \leqslant n$,
3. $\ell\sigma \mathrel{\parallel\!\!\!\rightarrow} r\sigma$ if $\ell \to r\ [\varphi] \in \mathcal{R}_{\mathsf{rc}}$ and $\sigma \vDash \ell \to r\ [\varphi]$

We extend $\mathrel{\parallel\!\!\!\rightarrow}$ to constrained terms inductively as follows:

1. $x\ [\varphi] \mathrel{\parallel\!\!\!\rightarrow} x\ [\varphi]$ for all variables x,
2. $f(s_1, \ldots, s_n)\ [\varphi] \mathrel{\parallel\!\!\!\rightarrow} f(t_1, \ldots, t_n)\ [\varphi]$ if $s_i\ [\varphi] \mathrel{\parallel\!\!\!\rightarrow} t_i\ [\varphi]$ for $1 \leqslant i \leqslant n$,
3. $\ell\sigma\ [\varphi] \mathrel{\parallel\!\!\!\rightarrow} r\sigma\ [\varphi]$ if $\rho\colon \ell \to r\ [\psi] \in \mathcal{R}_{\mathsf{rc}}$, $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}\mathsf{ar}(\varphi)$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$, φ is satisfiable and $\varphi \Rightarrow \psi\sigma$ is valid.

The parallel rewrite relation $\overset{\sim}{\mathrel{\parallel\!\!\!\rightarrow}}$ on constrained terms is defined as $\sim \cdot \mathrel{\parallel\!\!\!\rightarrow} \cdot \sim$.

Let s be a term and $P \subseteq \mathcal{P}\mathsf{os}(s)$ be a set of parallel positions. Given terms t_p for $p \in P$, we denote by $s[t_p]_{p \in P}$ the simultaneous replacement of the terms at position $p \in P$ in s by t_p. We recall the definition of parallel critical pairs for TRSs.

Definition 5.24. Let $\mathcal{R}$ be a TRS, $\rho\colon \ell \to r$ a rule in $\mathcal{R}$, and $P \subseteq \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell)$ a non-empty set of parallel positions. For every $p \in P$ let $\rho_p\colon \ell_p \to r_p$ be a variant of a rule in $\mathcal{R}$. The peak $\ell\sigma[r_p\sigma]_{p \in P} \mathrel{+\!\!\!\!\leftarrow\!\!\!\!\!+} \ell\sigma \to_{\epsilon,\mathcal{R}} r\sigma$ forms a *parallel critical pair* $\ell\sigma[r_p\sigma]_{p \in P} \approx r\sigma$ if the following conditions are satisfied:

1. $\mathcal{V}\mathsf{ar}(\rho_1) \cap \mathcal{V}\mathsf{ar}(\rho_2) = \varnothing$ for different rules ρ_1 and ρ_2 in $\{\rho\} \cup \{\rho_p \mid p \in P\}$,
2. σ is an mgu of $\{\ell_p \approx \ell|_p \mid p \in P\}$,
3. if $P = \{\epsilon\}$ then ρ_ϵ is not a variant of ρ.

The set of all constrained parallel critical pairs of $\mathcal{R}$ is denoted by $\mathsf{PCP}(\mathcal{R})$.

We lift this notion to the constrained setting and define it for LCTRSs.

Definition 5.25. Let $\mathcal{R}$ be an LCTRS, $\rho\colon \ell \to r\ [\varphi]$ a rule in $\mathcal{R}_{\mathsf{rc}}$, and $P \subseteq \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell)$ a non-empty set of parallel positions. For every $p \in P$ let $\rho_p\colon \ell_p \to r_p\ [\varphi_p]$ be a variant of a rule in $\mathcal{R}_{\mathsf{rc}}$. Let $\psi = \mathcal{EC}_\rho \wedge \bigwedge_{p \in P} \mathcal{EC}_{\rho_p}$ and $\Phi = \varphi\sigma \wedge \psi\sigma \wedge \bigwedge_{p \in P} \varphi_p\sigma$. The peak $\ell\sigma[r_p\sigma]_{p \in P}\ [\Phi] \mathrel{+\!\!\!\!\leftarrow\!\!\!\!\!+} \ell\sigma\ [\Phi] \to_{\epsilon,\mathcal{R}} r\sigma\ [\Phi]$ forms a *constrained parallel critical pair* $\ell\sigma[r_p\sigma]_{p \in P} \approx r\sigma\ [\Phi]$ if the following conditions are satisfied:

1. $\mathcal{V}\mathsf{ar}(\rho_1) \cap \mathcal{V}\mathsf{ar}(\rho_2) = \varnothing$ for different rules ρ_1 and ρ_2 in $\{\rho\} \cup \{\rho_p \mid p \in P\}$,
2. σ is an mgu of $\{\ell_p = \ell|_p \mid p \in P\}$ such that $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho) \cup \bigcup_{p \in P} \mathcal{LV}\mathsf{ar}(\rho_p)$,
3. $\varphi\sigma \wedge \bigwedge_{p \in P} \varphi_p\sigma$ is satisfiable,
4. if $P = \{\epsilon\}$ then ρ_ϵ is not a variant of ρ or $\mathcal{V}\mathsf{ar}(r) \nsubseteq \mathcal{V}\mathsf{ar}(\ell)$.

A constrained peak forming a constrained parallel critical pair is called a *constrained parallel critical peak*. The set of all constrained parallel critical pairs of $\mathcal{R}$ is denoted by $\mathsf{CPCP}(\mathcal{R})$.

For a term t and a set of parallel positions P in t, we write $\mathcal{V}\mathsf{ar}(t, P)$ to denote $\bigcup_{p \in P} \mathcal{V}\mathsf{ar}(t|_p)$. For a set of parallel positions P we denote by $\mathrel{+\!\!\!\!\!\!\to}^{P}$ that each rewrite step obtained in case (3) of Definition 5.23 is performed at a position $p \in P$ and no two steps share a position. Moreover, for a set of parallel positions P and a position q we denote by $\mathrel{+\!\!\!\!\!\!\to}^{P}_{\geqslant q}$ that $p \geqslant q$ for all $p \in P$.

Definition 5.26. A critical pair $s \approx t$ is *1-parallel closed* if $s \mathrel{+\!\!\!\!\!\!\to} \cdot \,{}^{*}\!\!\leftarrow t$. A TRS is 1-parallel closed if all its critical pairs are 1-parallel closed. A parallel critical pair $\ell\sigma[r_p\sigma]_{p \in P} \approx r\sigma$ originating from the peak $\ell\sigma[r_p\sigma]_{p \in P} \mathrel{+\!\!\!\!\leftarrow\!\!\!\!\!+} \ell\sigma \to_\epsilon r\sigma$ is *2-parallel closed* if there exists a term v and a set of parallel positions Q such that $\ell\sigma[r_p\sigma]_{p \in P} \to^* v \ {}^{Q}\!\!\mathrel{+\!\!\!\!\leftarrow\!\!\!\!\!+} r\sigma$ with $\mathcal{V}\mathsf{ar}(v, Q) \subseteq \mathcal{V}\mathsf{ar}(\ell\sigma, P)$. A TRS is 2-parallel closed if all its parallel critical pairs are 2-parallel closed. A TRS is parallel closed if it is 1-parallel closed and 2-parallel closed.

The following result from [83] has recently been formalized in Isabelle [29].

Theorem 5.27. *Left-linear parallel closed* TRS*s are confluent.* $\square$

In the remainder of this section we extend this result to LCTRSs. To this end we introduce the notion $\mathcal{TV}\mathsf{ar}(t, \varphi) = \mathcal{V}\mathsf{ar}(t) \setminus \mathcal{V}\mathsf{ar}(\varphi)$ denoting the set of non-logical variables in term t with respect to the logical constraint φ. We restrict this to non-logical variables in subterms below a set of parallel positions P in t: $\mathcal{TV}\mathsf{ar}(t, \varphi, P) = \bigcup_{p \in P} \mathcal{TV}\mathsf{ar}(t|_p, \varphi)$.

Definition 5.28. A constrained critical pair $s \approx t\ [\varphi]$ is *1-parallel closed* if $s \approx t\ [\varphi] \overset{\sim}{\mathrel{+\!\!\!\!\!\rightarrow}}_{\geqslant 1} \cdot \overset{\sim}{\rightarrow}^*_{\geqslant 2} u \approx v\ [\psi]$ for some trivial $u \approx v\ [\psi]$. An LCTRS is 1-parallel closed if all its constrained critical pairs are 1-parallel closed. A constrained parallel critical pair $\ell\sigma[r_p\sigma]_{p \in P} \approx r\sigma\ [\varphi]$ is *2-parallel closed* if there exists a set of parallel positions Q such that

$$\ell\sigma[r_p\sigma]_{p \in P} \approx r\sigma\ [\varphi] \overset{\sim}{\mathrel{+\!\!\!\!\!\rightarrow}}{}^{Q}_{\geqslant 2} \cdot \overset{\sim}{\rightarrow}^*_{\geqslant 1} u \approx v\ [\psi]$$

for some trivial $u \approx v\ [\psi]$ and $\mathcal{TV}\mathsf{ar}(v, \psi, Q) \subseteq \mathcal{TV}\mathsf{ar}(\ell\sigma, \varphi, P)$. An LCTRS is 2-parallel closed if all its constrained parallel critical pairs are 2-parallel closed. An LCTRS is parallel closed if it is 1-parallel closed and 2-parallel closed.

Recall from Section 5.2 that our definition of $\sim$ differs from the equivalence relation $\sim'$ defined in [45, 64]. The change is necessary for the variable condition of 2-parallel closedness to make sense, as illustrated in the following example.

Example 5.29. Consider the (LC)TRS consisting of the rules

$$\mathsf{f}(\mathsf{g}(x), y) \to \mathsf{f}(\mathsf{b}, y) \qquad \mathsf{g}(x) \to \mathsf{a} \qquad \mathsf{f}(\mathsf{a}, x) \to x \qquad \mathsf{f}(\mathsf{b}, x) \to x$$

The peak $\mathsf{f}(\mathsf{a}, y)\ [\mathsf{true}]\ {}^{\{1\}}\!\!\mathrel{\leftarrow\!\!\!\!\!+}\ \mathsf{f}(\mathsf{g}(x), y)\ [\mathsf{true}] \to \mathsf{f}(\mathsf{b}, y)\ [\mathsf{true}]$ gives rise to the (constrained) parallel critical pair $\mathsf{f}(\mathsf{a}, y) \approx \mathsf{f}(\mathsf{b}, y)\ [\mathsf{true}]$. Using $\sim'$ we have

$$\mathsf{f}(\mathsf{a}, y) \approx \mathsf{f}(\mathsf{b}, y)\ [\mathsf{true}] \mathrel{+\!\!\!\!\!\rightarrow}^{\{\epsilon\}}_{\geqslant 2} \cdot \rightarrow^*_{\geqslant 1} y \approx y\ [\mathsf{true}] \sim' x \approx x\ [\mathsf{true}]$$

and the variable condition $\mathcal{TV}\mathsf{ar}(x, \mathsf{true}, \{\epsilon\}) \subseteq \mathcal{TV}\mathsf{ar}(\mathsf{f}(\mathsf{g}(x), y), \mathsf{true}, \{1\})$ holds. Since the system has no logical constraints it can also be analyzed in the TRS setting. Following Definition 5.26 we would have to check the variable condition $\mathcal{V}\mathsf{ar}(y, \{\epsilon\}) \subseteq \mathcal{V}\mathsf{ar}(\mathsf{f}(\mathsf{g}(x), y), \{1\})$, which does not hold. Using $\sim$ resolves this difference, since $y \approx y\ [\mathsf{true}] \not\sim x \approx x\ [\mathsf{true}]$. So the conditions in Definition 5.28 reduce to the ones in Definition 5.26 for TRSs.

In Theorem 5.6 in Section 5.4 we related critical pairs of the transformed TRS to constrained critical pairs of the originating LCTRS. The following theorem does the same for parallel critical pairs.

Theorem 5.30. *For every parallel critical pair $s \approx t$ of $\overline{\mathcal{R}}$ there exists a constrained parallel critical pair $s' \approx t'\ [\varphi']$ of $\mathcal{R}$ and a substitution γ such that $s = s'\gamma$, $t = t'\gamma$ and $\gamma \vDash \varphi'$.*

Proof. Let $s \approx t$ be a parallel critical pair of $\overline{\mathcal{R}}$, originating from the parallel critical peak $\ell\mu\sigma[r_p\nu_p\sigma]_{p\in P} \nleftrightarrow \ell\mu\sigma = \ell\mu\sigma[\ell_p\nu_p\sigma]_{p\in P} \rightarrow_\epsilon r\mu\sigma$ with variants $\rho\colon \ell \rightarrow r\ [\varphi]$ and $\rho_p\colon \ell_p \rightarrow r_p\ [\varphi_p]$ for $p \in P$ of rules in $\mathcal{R}_{\mathsf{rc}}$ without shared variables, $\psi = \mathcal{EC}_\rho$ and $\psi_p = \mathcal{EC}_{\rho_p}$ for $p \in P$. Furthermore, $\mathcal{D}\mathsf{om}(\nu_p) = \mathcal{LV}\mathsf{ar}(\rho_p)$ for $p \in P$, $\mathcal{D}\mathsf{om}(\mu) = \mathcal{LV}\mathsf{ar}(\rho)$, $\nu_p \vDash \varphi_p \wedge \psi_p$ for $p \in P$, $\mu \vDash \varphi \wedge \psi$, $p \in \mathcal{P}\mathsf{os}_\mathcal{F}(\ell\mu)$, and σ is an mgu of $\{\ell\mu|_p \approx \ell_p\nu_p \mid p \in P\}$. Moreover, if $P = \{\epsilon\}$ then $\ell_\epsilon\nu_\epsilon \rightarrow r_\epsilon\nu_\epsilon\ [\varphi_\epsilon\nu_\epsilon]$ and $\ell\mu \rightarrow r\mu\ [\varphi\mu]$ are not variants. Define the substitution τ as $\bigcup\{\nu_p \mid p \in P\} \uplus \mu$. Clearly, $\ell_p\tau = \ell_p\nu_p$ and $r_p\tau = r_p\nu_p$ for $p \in P$, $\ell\tau = \ell\mu$, $r\tau = r\mu$, $\tau \vDash \varphi \wedge \psi$ and $\tau \vDash \varphi_p \wedge \psi_p$ for all $p \in P$. Hence the given peak can be written as $\ell\tau\sigma[r_p\tau\sigma]_{p\in P} \nleftrightarrow \ell\tau\sigma = \ell\tau\sigma[\ell_p\tau\sigma]_{p\in P} \rightarrow_\epsilon r\tau\sigma$ with $\tau \vDash \varphi''$ where

$$\varphi'' = \varphi \wedge \mathcal{EC}_\rho \wedge \bigwedge_{p\in P} (\varphi_p \wedge \mathcal{EC}_{\rho_p})$$

Since $\ell|_p\tau\sigma = \ell_p\tau\sigma$ for all $p \in P$ there exists an mgu δ of $\{\ell|_p = \ell_p \mid p \in P\}$ and a substitution γ such that $\delta\gamma = \tau\sigma$. Let $s' = \ell\delta[r_p\delta]_{p\in P}$ and $t' = r\delta$. We claim that this results in the constrained parallel critical pair $s' \approx t'\ [\varphi''\delta]$. Condition (1) of Definition 5.25 is trivially satisfied. We obtain $P \subseteq \mathcal{P}\mathsf{os}_\mathcal{F}(\ell)$ because $P \subseteq \mathcal{P}\mathsf{os}_\mathcal{F}(\ell\mu)$, $\mu(x) \in \mathcal{V}\mathsf{al}$ for every $x \in \mathcal{D}\mathsf{om}(\mu)$, and $\mathsf{root}(\ell\mu|_p) = \mathsf{root}(\ell_p\nu) \in \mathcal{F} \setminus \mathcal{V}\mathsf{al}$ for all $p \in P$. For condition (2) it remains to show that $\delta(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho) \cup \bigcup_{p\in P} \mathcal{LV}\mathsf{ar}(\rho_p)$. Suppose to the contrary that $\mathsf{root}(\delta(x)) \in \mathcal{F} \setminus \mathcal{V}\mathsf{al}$ for some $x \in \mathcal{LV}\mathsf{ar}(\rho) \cup \bigcup_{p\in P} \mathcal{LV}\mathsf{ar}(\rho_p)$. Then $\mathsf{root}(\delta(x)) = \mathsf{root}(\gamma(\delta(x))) = \mathsf{root}(\sigma(\tau(x))) \in \mathcal{F} \setminus \mathcal{V}\mathsf{al}$, which contradicts $\tau \vDash \varphi''$. Condition (3) follows from the identity $\delta\gamma = \tau\sigma$ together with $\tau \vDash \varphi''$ which imply $\delta\gamma \vDash \varphi''$ and thus $\varphi''\delta$ is satisfiable. Hence also $\varphi\delta \wedge \bigwedge_{p\in P} \varphi_p\delta$ is satisfiable. It remains to show condition (4), so let $P = \{\epsilon\}$ and further assume that ρ_ϵ and ρ are variants. So there exists a variable renaming π such that $\rho_\epsilon\pi = \rho$. In particular, $\ell_\epsilon\pi = \ell$ and $r_\epsilon\pi = r$. We show $\tau(\pi(x)) = \pi(\tau(x))$ for all $x \in \mathcal{V}\mathsf{ar}(\ell_\epsilon)$. Let $x \in \mathcal{V}\mathsf{ar}(\ell_\epsilon)$. If $x \in \mathcal{LV}\mathsf{ar}(\rho_\epsilon) = \mathcal{D}\mathsf{om}(\nu)$ then $\tau(x) = \nu(x) \in \mathcal{V}\mathsf{al}$. Moreover, $\pi(x) \in \mathcal{LV}\mathsf{ar}(\rho) = \mathcal{D}\mathsf{om}(\mu)$ and thus $\tau(\pi(x)) = \mu(\pi(x)) \in \mathcal{V}\mathsf{al}$. Since $\ell_\epsilon\tau$ and $\ell\tau$ are unifiable, $\pi(\tau(x)) = \tau(x) = \tau(\pi(x))$. If $x \notin \mathcal{LV}\mathsf{ar}(\rho_\epsilon)$ then $\tau(x) = x$, $\pi(x) \notin \mathcal{LV}\mathsf{ar}(\rho)$ and similarly $\tau(\pi(x)) = \pi(x) = \pi(\tau(x))$. All in all, $\ell_\epsilon\tau\pi = \ell_\epsilon\pi\tau = \ell\tau$. Now, if $\mathcal{V}\mathsf{ar}(r_\epsilon) \subseteq \mathcal{V}\mathsf{ar}(\ell_\epsilon)$ then we obtain $r_\epsilon\tau\pi = r_\epsilon\pi\tau = r\tau$, contradicting the fact that $\ell_\epsilon\nu \rightarrow r_\epsilon\nu$ and $\ell\mu \rightarrow r\mu$ are not variants. We conclude that $s' \approx t'\ [\varphi''\delta]$ is a constrained parallel critical pair of $\mathcal{R}$. So we can take $\varphi' = \varphi''\delta$. Clearly, $s = s'\gamma$ and $t = t'\gamma$. Moreover, $\gamma \vDash \varphi'$ since $\varphi'\gamma = \varphi''\tau\sigma = \varphi''\tau$ and $\tau \vDash \varphi''$. ☐

The proofs of the following lemmata are given in [69].

Lemma 5.31. *If $s \approx t\ [\varphi] \overset{\top}{\twoheadrightarrow}{}^{P}_{\geqslant 1} u \approx v\ [\psi]$ then for all substitutions $\sigma \vDash \varphi$ there exists a substitution δ such that $\delta \vDash \psi$, $s\sigma \twoheadrightarrow^P u\delta$ and $t\sigma = v\delta$.* ☐

Lemma 5.32. *If a constrained critical pair $s \approx t\ [\varphi]$ is 1-parallel closed then $s\sigma \twoheadrightarrow \cdot {}^*\!\leftarrow t\sigma$ for all substitutions σ with $\sigma \vDash \varphi$.* ☐

Lemma 5.33. *If a constrained parallel critical pair $s = \ell\sigma'[r_p\sigma']_{p\in P} \approx r\sigma' = t\ [\varphi]$ is 2-parallel closed then there exist a term v and a set Q of parallel positions such that $s\sigma \rightarrow^* v \ {}^{Q}\!\!\twoheadleftarrow t\sigma$ and $\mathcal{V}\mathsf{ar}(v, Q) \subseteq \mathcal{V}\mathsf{ar}(\ell\sigma'\sigma, P)$ for all substitutions σ with $\sigma \vDash \varphi$.* ☐

Theorem 5.34. *If an* LCTRS $\mathcal{R}$ *is parallel closed then* $\overline{\mathcal{R}}$ *is parallel closed.*

Proof. Let $\mathcal{R}$ be a parallel closed LCTRS. First consider an arbitrary critical pair $s \approx t \in \mathsf{CP}(\overline{\mathcal{R}})$. From Theorem 5.6 we know that there exist a constrained critical pair $s' \approx t'\ [\varphi] \in \mathsf{CCP}(\mathcal{R})$ and a substitution σ such that $s'\sigma = s$, $t'\sigma = t$ and $\sigma \vDash \varphi$. Since the constrained critical pair is 1-parallel closed, Lemma 5.32 yields $s \mathrel{+\!\!\!\!\rightarrow} \cdot {}^{*}\!\!\leftarrow t$. Hence $\overline{\mathcal{R}}$ is 1-parallel closed.

Next consider an arbitrary parallel critical pair $s \approx t \in \mathsf{PCP}(\overline{\mathcal{R}})$. Theorem 5.30 yields a constrained parallel critical pair $s' = \ell\sigma'[r_p\sigma']_{p \in P} \approx r\sigma' = t'\ [\varphi]$ in $\mathsf{CPCP}(\mathcal{R})$ and a substitution σ such that $s'\sigma = s$, $t'\sigma = t$ and $\sigma \vDash \varphi$. Since the constrained parallel critical pair is 2-parallel closed, by Lemma 5.33 there exist a term v and a set of parallel positions Q such that $s \rightarrow^{*} v \;{}^{Q}\!\!\leftarrow\!\!\!\!+\; t$ and $\mathcal{V}\mathsf{ar}(v, Q) \subseteq \mathcal{V}\mathsf{ar}(\ell\sigma'\sigma, P)$. Hence $\overline{\mathcal{R}}$ is 2-parallel closed. $\square$

Since left-linearity of $\mathcal{R}$ is preserved in $\overline{\mathcal{R}}$ and left-linear, parallel closed TRSs are confluent by Theorem 5.27, we obtain the following corollary via Theorems 5.30 and 5.34. Again, $\mathcal{R}$ only has to be left-linear in the variables $x \notin \mathcal{LV}\mathsf{ar}$, since that is sufficient for $\overline{\mathcal{R}}$ to be left-linear.

Corollary 5.35. *Every left-linear parallel closed* LCTRS *is confluent.* $\square$

We illustrate the corollary on a concrete example.

Example 5.36. Consider the LCTRS $\mathcal{R}$ over the theory Ints with the rules

$$\mathsf{f}(\mathsf{a}) \rightarrow \mathsf{g}(4,4) \qquad \mathsf{a} \rightarrow \mathsf{g}(1+1, 3+1) \qquad \mathsf{g}(x, y) \rightarrow \mathsf{f}(\mathsf{g}(z, y))\ [z = x - 2]$$

The constrained (parallel) critical pair $\mathsf{f}(\mathsf{g}(1+1, 3+1)) \approx \mathsf{g}(4,4)\ [\mathsf{true}]$ originating from the peak $\mathsf{f}(\mathsf{g}(1+1,3+1))\ [\mathsf{true}]\ {}^{\{1\}}\!\!\leftarrow\!\!\!\!+\ \mathsf{f}(\mathsf{a})\ [\mathsf{true}] \rightarrow_{\epsilon} \mathsf{g}(4,4)\ [\mathsf{true}]$ is 2-parallel closed:

$$\begin{aligned} \mathsf{f}(\mathsf{g}(1+1,3+1)) \approx \mathsf{g}(4,4)\ [\mathsf{true}] &\ \tilde{\mathrel{+\!\!\!\!\rightarrow}}_{\geqslant 1}\ \mathsf{f}(\mathsf{g}(2,4)) \approx \mathsf{g}(4,4)\ [\mathsf{true}] \\ &\ \tilde{\mathrel{+\!\!\!\!\rightarrow}}^{\{2\}}_{\geqslant 2}\ \mathsf{f}(\mathsf{g}(2,4)) \approx \mathsf{f}(\mathsf{g}(2,4))\ [\mathsf{true}] \end{aligned}$$

Note that the condition $\mathcal{TV}\mathsf{ar}(\mathsf{f}(\mathsf{g}(2,4)), \mathsf{true}, \{2\}) \subseteq \mathcal{TV}\mathsf{ar}(\mathsf{f}(\mathsf{a}), \mathsf{true}, \{1\})$ is trivially satisfied. One easily checks that the corresponding constrained critical pair is 1-parallel closed. Since the only other remaining constrained critical pair is trivial, we conclude confluence by Corollary 5.35.

5.7 Conclusion

We presented a left-linearity preserving transformation from LCTRSs to TRSs such that (parallel) critical pairs in the latter correspond to constrained (parallel) critical pairs in the former. As a consequence, confluence results for TRSs based on restricted joinability conditions easily carry over to LCTRSs. This was illustrated by generalizing the advanced confluence results of van Oostrom [87] and Toyama [83] from TRSs to LCTRSs. We also proved that (local) confluence of terminating LCTRSs over a decidable theory is undecidable in general.

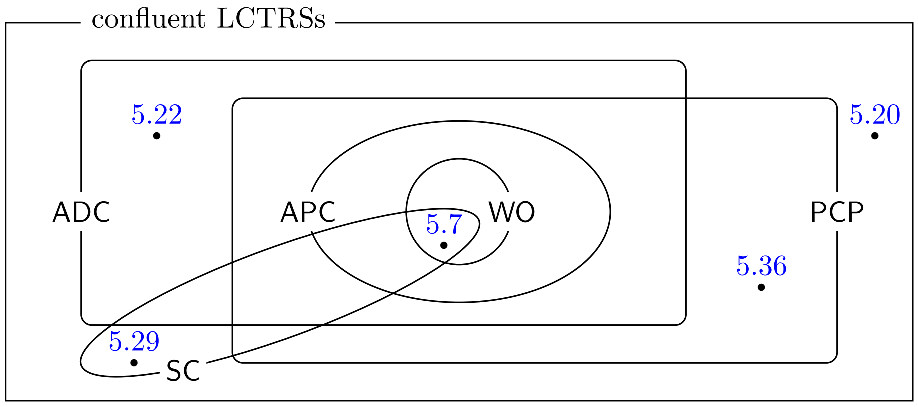


Figure 5.2: Relating confluence criteria for LCTRSs.

Figure 5.2 relates the confluence criteria in this paper to the earlier ones from [45, 64]. The acronyms stand for weak orthogonality (WO, [45, Theorem 4]), strong closedness (SC, [64, Theorem 2]), almost parallel closedness (APC, [64, Theorem 4]), almost development closedness (ADC, Corollary 5.21), and parallel closedness of (parallel) critical pairs (PCP, Corollary 5.35). All areas are inhabited and the numbers refer to examples in this paper.

The confluence results of [45, 64] have been implemented in crest.[1] The tool is currently under heavy development, not only to incorporate the results in this paper but also termination and completion techniques. Confluence of LCTRSs is a new category in the upcoming edition of the Confluence Competition[2] and we expect to present experimental results obtained with crest at the conference.

For TRSs numerous other confluence techniques, not based on restricted joinability conditions of critical pairs, as well as sufficient conditions for non-confluence are known [2, 31, 73, 92]. We plan to investigate which techniques generalize to LCTRSs with our transformation. The transformation also makes the formal verification of confluence criteria for LCTRSs in a proof assistant a more realistic goal.

Acknowledgments.

The detailed feedback of the reviewers improved the presentation.

Disclosure of Interests.

The authors have no competing interests to declare that are relevant to the content of this article.

[1]http://cl-informatik.uibk.ac.at/software/crest/
[2]https://project-coco.uibk.ac.at/2024/

Chapter 6

Equational Theories and Validity for Logically Constrained Term Rewriting

Publication Details

The content of this chapter is taken from the camera-ready version of the following publication:

[4] Takahito Aoto, Naoki Nishida, and Jonas Schöpf. Equational theories and validity for logically constrained term rewriting. In Jakob Rehof, editor, *Proceedings of the 9th International Conference on Formal Structures for Computation and Deduction (FSCD)*, volume 299 of *Leibniz International Proceedings in Informatics*, pages 31:1–31:21, 2024. doi: 10.4230/LIPIcs.FSCD.2024.31

Adjustments were made to the presentation of the content to align with the format of this thesis and obvious typos were corrected. In particular, the appearance of the inference rules in Figure 6.2 was improved. Appendix C contains the missing lemmata and proofs, and a full version of this paper is available via arXiv [5].

The idea for this publication was developed during the midterm meeting of the ARI[1] project in Innsbruck. The effort behind this publication was equally shared among all co-authors, and I do not claim sole authorship of any part.

Abstract

Logically constrained term rewriting is a relatively new formalism where rules are equipped with constraints over some arbitrary theory. Although there are many recent advances with respect to rewriting induction, completion, complexity analysis and confluence analysis for logically constrained term rewriting, these works solely focus on the syntactic side of the formalism lacking detailed investigations on semantics. In this paper, we investigate a semantic side of logically constrained term rewriting. To this end, we first define constrained equations, constrained equational theories and validity of the former based on the latter. After presenting the relationship of validity and conversion of rewriting, we then construct a sound inference system to prove validity of constrained equations in constrained equational theories. Finally, we give an algebraic semantics,

[1]https://ari-informatik.uibk.ac.at/

which enables one to establish invalidity of constrained equations in constrained equational theories. This algebraic semantics derives a new notion of consistency for constrained equational theories.

6.1 Introduction

Logically constrained term rewriting is a relatively new formalism building upon many-sorted term rewriting and built-in theories. The rules of a logically constrained term rewrite system (LCTRS, for short) are equipped with constraints over some arbitrary theory, which have to be fulfilled in order to apply rules in rewrite steps. This formalism intends to live up with data structures which are often difficult to represent in basic rewriting, such as integers and bit-vectors, with the help of external provers and their built-in theories.

Logical syntax and semantics are often conceived as two sides of the same coin. This is not exceptional, especially for equational logic in which term rewriting lies. On the other hand, although there are many recent advances in rewriting induction [23], completion [88], complexity analysis [89], confluence analysis [45, 58, 64] and (all-path) reachability [13, 40, 41] for LCTRSs, these works solely focus on the syntactic side of the formalism, lacking detailed investigations on semantics.

In this paper, we investigate a semantic side of the LCTRS formalism. To this end, we first define *constrained equations* (CEs, for short) and *constrained equational theories* (CE-theories, for short). In (first-order) term rewriting, the equational version of rewrite rules is obtained by removing the orientation of the rules. However, in the case of LCTRSs, if we consider a constrained rule $\ell \to r\ [\varphi]$ and relate this naively to a CE $\ell \approx r\ [\varphi]$, which does not distinguish between left- and right-hand sides, we lose information about the restriction on the possible instantiation of variables. This motivates us to add an explicit set X to each CE $\ell \approx r\ [\varphi]$ as $\Pi X.\, \ell \approx r\ [\varphi]$[2]—we name variables in X as *logical variables* with respect to the equation. A CE-theory is then defined as a set of CEs. Similar to the rewrite steps of LCTRSs, we define validity by convertibility if all logical variables are instantiated by values—we denote this notion of validity as *CE-validity* for clarity.

After establishing fundamental properties of the CE-validity, we present its relation to the conversion of rewriting. However, the conversion of rewriting is useful in general to establish the validity of arbitrary CEs. This motivates us to introduce $\mathbf{CEC}_0$, an inference calculus for deriving valid CEs. After demonstrating the usefulness of $\mathbf{CEC}_0$ via some derivations, we present a soundness theorem for the calculus. We also show a partial completeness result, followed by a discussion why our system seems incomplete. Afterwards we consider the opposite question, namely how to prove that a CE is not valid for a particular CE-theory. To this end, we introduce an algebraic semantics that captures CE-validity. We give a natural notion of models for CE-theory, which we call

[2]In the literature, some other approaches exist. The computation of critical pairs is also prone of losing information [64]. They solved it by adding dummy constraints $x = x$ to the critical pair. Another approach was proposed in [88] where $\mathcal{LV}\mathsf{ar}(\ell \approx r\ [\varphi])$ was simply defined as $\mathcal{V}\mathsf{ar}(\varphi)$.

$$\begin{tikzcd}
& \boxed{\text{CE-Validity}} & & \boxed{\text{Algebraic Semantics}} \\
& E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi] \arrow[rr, bend left, "\text{Theorem 6.41}"] \arrow[d, dashed, bend left, "\text{Theorem 6.31}"] & & E \models \Pi X.\, s \approx t\ [\varphi] \arrow[ll, bend left, "\text{Theorem 6.52}"] \\
\boxed{\text{Provability in } \mathbf{CEC}_0} & E \vdash_{\mathbf{CEC}_0} \Pi X.\, s \approx t\ [\varphi] \arrow[u, bend left, "\text{Theorem 6.27}"] \arrow[rru, bend right, "\text{Corollary 6.42}"'] & &
\end{tikzcd}$$

Figure 6.1: An overview of the main results of this paper.

CE-algebras. We establish soundness and completeness with respect to CE-validity for this.

Figure 6.1 presents the relationships between the introduced notions and results of this paper. The following concrete contributions are covered in this paper:

1. We propose a formulation of CEs and CE-theories.
2. On top of that we devise a notion of validity of a CE for a CE-theory E, which we call CE-validity.
3. We give a proof system $\mathbf{CEC}_0$, and show soundness (Theorem 6.27) and a partial completeness result (Theorem 6.31) with respect to CE-validity.
4. We give a notion of CE-algebras and based on it we define algebraic semantics, which is sound (Theorem 6.41) and complete (Theorem 6.52) with respect to CE-validity for *consistent* CE-theories.

We want to discuss some highlights of the last item for readers who are familiar with algebraic semantics of equational logic. First of all, our definition of CE-algebras admits extended underlying models, contrast to those that precisely contain the same underlying models; we will demonstrate why this generalization is required to obtain the completeness result. To reflect this definition, it was necessary to modify the definition of congruence relation to a non-standard one. Also, the notion of consistency with respect to values arises to guarantee this modified notion of congruence in the term algebras. Moreover, it also turns out that value-consistency is equivalent to a more intuitive notion of consistency.

The remainder of the paper is organized as follows. In the next section, we briefly explain the LCTRS formalism, and present some basic lemmas that are necessary for our proofs. Section 6.3 introduces the notion of CEs, CE-theories and CE-validity, and presents basic properties on CE-validity and its relation to the conversion of rewriting. Section 6.4 is devoted to our inference system $\mathbf{CEC}_0$, including its soundness and partial completeness with respect to CE-validity. In Section 6.5, we present algebraic semantics, and soundness and completeness results with respect to CE-validity. Before concluding this paper in Section 6.7, we briefly describe related work in Section 6.6. We provide only brief proof sketches of selected results in this paper. However, all detailed proofs are given in the full version of this paper [5].

6.2 Preliminaries

In this section, we briefly recall LCTRSs [23, 45, 64]. Familiarity with the basic notions on mathematical logic [20, 86] and term rewriting [9, 59] is assumed.

The (sorted) signature of an LCTRS is given by the set S of sorts and the set $\mathcal{F}$ of S-sorted function symbols. Each $f \in \mathcal{F}$ is equipped with a sort declaration $f\colon \tau_1 \times \cdots \times \tau_n \to \tau_0$ with $\tau_0, \ldots, \tau_n \in S$; $\tau_1 \times \cdots \times \tau_n \to \tau_0$ is said to be the sort of f, and we denote by $\mathcal{F}^{\tau_1 \times \cdots \times \tau_n \to \tau_0}$ the set of function symbols of sort $\tau_1 \times \cdots \times \tau_n \to \tau_0$. For constants of sort $\to \tau$ we drop $\to$ and write τ instead of $\to \tau$. The set of S-sorted variables is denoted by $\mathcal{V}$ and the set of S-sorted terms over $\mathcal{F}, \mathcal{V}$ is $\mathcal{T}(\mathcal{F}, \mathcal{V})$. For each $\tau \in S$, we denote by $\mathcal{V}^\tau$ the set of variables of sort τ and by $\mathcal{T}(\mathcal{F}, \mathcal{V})^\tau$ the set of terms of sort τ; we also write t^τ for a term t such that $t \in \mathcal{T}(\mathcal{F}, \mathcal{V})^\tau$. The set of variables occurring in a term $t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ is denoted by $\mathcal{V}\mathsf{ar}(t)$ and can be restricted by a set of sorts T with $\mathcal{V}\mathsf{ar}^T(t) = \{x^\tau \in \mathcal{V}\mathsf{ar}(t) \mid \tau \in T\}$. A substitution σ is a mapping $\mathcal{V} \to \mathcal{T}(\mathcal{F}, \mathcal{V})$ such that $\mathcal{D}\mathsf{om}(\sigma) = \{x \in \mathcal{V} \mid x \neq \sigma(x)\}$ is finite and $\sigma(x^\tau) \in \mathcal{T}(\mathcal{F}, \mathcal{V})^\tau$ is satisfied for all $x \in \mathcal{D}\mathsf{om}(\sigma)$.

In the LCTRS formalism, sorts are divided into two categories, that is, each sort $\tau \in S$ is either a *theory sort* or a *term sort*, where we denote by S_{th} the set of theory sorts and by S_{te} the set of term sorts, i.e. $S = S_{\mathsf{th}} \uplus S_{\mathsf{te}}$. Accordingly, the set of variables is partitioned as $\mathcal{V} = \mathcal{V}_{\mathsf{th}} \uplus \mathcal{V}_{\mathsf{te}}$ by letting $\mathcal{V}_{\mathsf{th}}$ for the set of variables of sort $\tau \in S_{\mathsf{th}}$ and $\mathcal{V}_{\mathsf{te}}$ for the set of variables of sort $\tau \in S_{\mathsf{te}}$. Furthermore, we assume each function symbol $f \in \mathcal{F}$ is either a theory symbol or a term symbol, where all former symbols $f\colon \tau_1 \times \cdots \times \tau_n \to \tau_0$ need to satisfy $\tau_i \in S_{\mathsf{th}}$ for all $0 \leqslant i \leqslant n$. The sets of theory and term symbols are denoted by $\mathcal{F}_{\mathsf{th}}$ and $\mathcal{F}_{\mathsf{te}}$, respectively: $\mathcal{F} = \mathcal{F}_{\mathsf{th}} \uplus \mathcal{F}_{\mathsf{te}}$. Throughout the paper, we consider signatures consisting of four components $\langle S_{\mathsf{th}}, S_{\mathsf{te}}, \mathcal{F}_{\mathsf{th}}, \mathcal{F}_{\mathsf{te}} \rangle$. In some cases term/theory signature stands for the two respective term/theory components of such a signature.

An LCTRS is also equipped with a model over the sorts S_{th} and the symbols $\mathcal{F}_{\mathsf{th}}$, which is given by $M = \langle \mathcal{I}, \mathcal{J} \rangle$, where $\mathcal{I}$ assigns each $\tau \in S_{\mathsf{th}}$ a *non-empty* set $\mathcal{I}(\tau)$, specifying its domain, and $\mathcal{J}$ assigns each $f\colon \tau_1 \times \cdots \times \tau_n \to \tau_0 \in \mathcal{F}_{\mathsf{th}}$ an interpretation function $\mathcal{J}(f)\colon \mathcal{I}(\tau_1) \times \cdots \times \mathcal{I}(\tau_n) \to \mathcal{I}(\tau_0)$. In particular, $\mathcal{J}(c) \in \mathcal{I}(\tau)$ for any constant $c \in \mathcal{F}^\tau_{\mathsf{th}}$. We suppose for each $\tau \in S_{\mathsf{th}}$, there exists a subset $\mathcal{V}\mathsf{al}_\tau \subseteq \mathcal{F}^\tau_{\mathsf{th}}$ of constants of sort τ such that (the restriction of) $\mathcal{J}$ to $\mathcal{V}\mathsf{al}_\tau$ forms a bijection $\mathcal{V}\mathsf{al}_\tau \cong \mathcal{I}(\tau)$. We let $\mathcal{V}\mathsf{al} = \bigcup_{\tau \in S_{\mathsf{th}}} \mathcal{V}\mathsf{al}_\tau$, whose elements are called *values*. For simplicity, we do not distinguish between $c \in \mathcal{V}\mathsf{al}$ and $\mathcal{J}(c)$. Note that, in [23, 45], an arbitrary overlap between term and theory symbols is allowed provided it is covered by values. For simplicity, we assume $\mathcal{F}_{\mathsf{th}} \cap \mathcal{F}_{\mathsf{te}} = \varnothing$.

A *valuation* over a model $M = \langle \mathcal{I}, \mathcal{J} \rangle$ is a family $\rho = (\rho_\tau)_{\tau \in S_{\mathsf{th}}}$ of mappings $\rho_\tau\colon \mathcal{V}^\tau \to \mathcal{I}(\tau)$. The *interpretation* $[\![t]\!]_{M,\rho} \in \mathcal{I}(\tau)$ of a term $t^\tau \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, \mathcal{V})$ in the model M with respect to the valuation $\rho = (\rho_\tau)_{\tau \in S_{\mathsf{th}}}$ is inductively defined as follows: $[\![x^\tau]\!]_{M,\rho} = \rho^\tau(x)$

and $[\![f(t_1,\dots,t_n)]\!]_{M,\rho} = \mathcal{J}(f)([\![t_1]\!]_{M,\rho},\dots,[\![t_n]\!]_{M,\rho})$. We abbreviate $[\![t]\!]_{M,\rho}$ as $[\![t]\!]_\rho$ if M is known from the context. Furthermore, for any ground term $t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}})$, the valuation ρ has no impact on the interpretation $[\![t]\!]_\rho$ which can be safely ignored and written as $[\![t]\!]$.

We suppose a special sort $\mathsf{Bool} \in S_{\mathsf{th}}$ such that $\mathcal{I}(\mathsf{Bool}) = \mathbb{B} = \{\mathsf{true}, \mathsf{false}\}$, and usual logical connectives $\neg, \wedge, \vee, \ldots \in \mathcal{F}_{\mathsf{th}}$ with their default sorts. We assume that there exists for each $\tau \in S_{\mathsf{th}}$ an equality symbol $=_\tau$ of sort $\tau \times \tau \to \mathsf{Bool}$ in $\mathcal{F}_{\mathsf{th}}$. For brevity we will omit $_\tau$ from $=_\tau$. We assume, for all of these theory symbols, that their interpretation functions model their default semantics. The terms in $\mathcal{T}(\mathcal{F}_{\mathsf{th}},\mathcal{V})^{\mathsf{Bool}}$ are called *logical constraints*.[3] Note that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{V}_{\mathsf{th}}$ for any logical constraint φ, thus in this case $\mathcal{T}(\mathcal{F}_{\mathsf{th}},\mathcal{V})^{\mathsf{Bool}} = \mathcal{T}(\mathcal{F}_{\mathsf{th}},\mathcal{V}_{\mathsf{th}})^{\mathsf{Bool}}$. We say that a logical constraint φ is over a set $X \subseteq \mathcal{V}_{\mathsf{th}}$ of theory variables if $\mathcal{V}\mathsf{ar}(\varphi) \subseteq X$. A logical constraint φ is said to be *valid* in a model M, written as $\models_M \varphi$ (or $\models \varphi$ when the model M is known from the context), if $[\![\varphi]\!]_{M,\rho} = \mathsf{true}$ for any valuation ρ over the model M. Considering the bijection $\mathcal{V}\mathsf{al}_\tau \cong \mathcal{I}(\tau)$, an arbitrary substitution σ is equivalent to a valuation ρ. Suppose that $\mathcal{VD}\mathsf{om}(\sigma) = \{x \in \mathcal{D}\mathsf{om}(\sigma) \mid \sigma(x) \in \mathcal{V}\mathsf{al}\}$ and $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$. Then the substitution σ can be seen as a valuation over φ, and $\models_M \varphi\sigma$ coincides with $[\![\varphi]\!]_{M,\sigma} = \mathsf{true}$. More generally, we have the following.

Lemma 6.1. *Let $t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}},\mathcal{V}_{\mathsf{th}})$, ρ a valuation, and σ a substitution.*

1. *Suppose $\sigma(x) \in \mathcal{T}(\mathcal{F}_{\mathsf{th}},\mathcal{V}_{\mathsf{th}})$ for all $x \in \mathcal{V}_{\mathsf{th}}$. Let $[\![\sigma]\!]_{M,\rho}$ be a valuation defined as $[\![\sigma]\!]_{M,\rho}(x) = [\![\sigma(x)]\!]_{M,\rho}$. Then, $[\![t]\!]_{M,[\![\sigma]\!]_{M,\rho}} = [\![t\sigma]\!]_{M,\rho}$.*
2. *Suppose that $\mathcal{V}\mathsf{ar}(t) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$. Then, $[\![t]\!]_{M,\hat\sigma} = [\![t\sigma]\!]_M$, where the valuation $\hat\sigma$ is defined by $\hat\sigma(x^\tau) = \xi(\sigma(x)) \in \mathcal{I}(\tau)$ for $x \in \mathcal{VD}\mathsf{om}(\sigma)$, where ξ is a bijection $\mathcal{V}\mathsf{al}^\tau \cong \mathcal{I}(\tau)$.*

Proof (Sketch). 1. Use structural induction on $t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}},\mathcal{V}_{\mathsf{th}})$. *2.* Similar to *1.*, using the assumption $\mathcal{V}\mathsf{ar}(t) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$. $\square$

From Lemma 6.1 the following characterizations, which are used later on, are obtained. Note that $\models \varphi = \mathsf{true}$ ($\models \varphi = \mathsf{false}$) if and only if $\models \varphi$ ($\models \neg\varphi$), for a logical constraint φ.

Lemma 6.2. *Let φ be a logical constraint.*

1. *$\models_M \varphi$ if and only if $\models_M \varphi\sigma$ for all substitutions σ such that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$.*
2. *If $\models_M \varphi$, then $\models_M \varphi\sigma$ for all substitutions σ such that $\sigma(x) \in \mathcal{T}(\mathcal{F}_{\mathsf{th}},\mathcal{V}_{\mathsf{th}})$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi) \cap \mathcal{D}\mathsf{om}(\sigma)$.*
3. *The following statements are equivalent: (1) $\models_M \neg\varphi$, (2) $\not\models_M \varphi\sigma$ for all substitutions σ such that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$, and (3) $\sigma \models_M \varphi$ for no substitution σ.*

[3]Logical constraints are quantifier-free, which is not restrictive: Consider, for example, a formula $\forall x.\ \varphi$ with n free variables $x_1,\dots,x_n$ and a quantifier-free formula φ. By introducing an n-ary predicate symbol p defined as $[\![p(x_1,\dots,x_n)]\!]_{M,\rho} = [\![\forall x.\ \varphi]\!]_{M,\rho}$, we can replace the formula by the quantifier-free formula $p(x_1,\dots,x_n)$. Clearly, this applies to arbitrary first-order formulas. Another approach can be seen in [23, Section 2.2].

Here, $\sigma \models_M \varphi$ denotes that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$ and $\models_M \varphi\sigma$ hold.

Proof (Sketch). 1. ($\Rightarrow$) Let σ be a substitution such that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$, and $\hat{\sigma}$ be defined as in Lemma 6.1. Then, $[\![\varphi]\!]_{M,\hat{\sigma}} = \mathsf{true}$, and hence $[\![\varphi\sigma]\!]_M = \mathsf{true}$ by Lemma 6.1. Therefore, $\models_M \varphi\sigma$. ($\Leftarrow$) Let ρ be a valuation over a model $M = \langle \mathcal{I}, \mathcal{J} \rangle$. Then, in the view of $\mathcal{V}\mathsf{al}_\tau \cong \mathcal{I}(\tau)$, we can take a substitution $\check{\rho}$ given by $\check{\rho}(x) = \rho(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$. Then, use Lemma 6.1 to obtain $[\![\varphi]\!]_{M,\rho} = [\![\varphi]\!]_{M,\hat{\check{\rho}}} = \mathsf{true}$, from which $\models_M \varphi$ follows. *2.* Take a substitution σ' such that $\sigma'(x) = \sigma(x)$ for $x \in \mathcal{V}\mathsf{ar}(\varphi)$ and $\sigma'(x) = x$ otherwise. Then, using Lemma 6.1, we have $[\![\varphi]\!]_{[\![\sigma]\!]_\rho} = [\![\varphi]\!]_{[\![\sigma']\!]_\rho} = [\![\varphi\sigma']\!]_\rho = [\![\varphi\sigma]\!]_\rho$. Thus, $[\![\varphi\sigma]\!]_\rho = \mathsf{true}$ for any ρ. Therefore, $\models_M \varphi\sigma$. *3.* Use *1.* □

LCTRSs admit special rewrite steps over $\mathcal{T}(\mathcal{F}, \mathcal{V})$ specified by the underlying model $M = \langle \mathcal{I}, \mathcal{J} \rangle$. Such rewrite steps are called *calculation steps* and denoted by $s \to_{\mathsf{ca}} t$, which is defined as follows: $s \to_{\mathsf{ca}} t$ if $s = C[f(c_1, \dots, c_n)]$ and $t = C[c_0]$ for $f \in \mathcal{F}_{\mathsf{th}} \setminus \mathcal{V}\mathsf{al}$ and $c_0, \dots, c_n \in \mathcal{V}\mathsf{al}$ with $c_0 = \mathcal{J}(f)(c_1, \dots, c_n)$ and a context C. The following lemma connects calculation steps and interpretations over ground theory terms $\mathcal{T}(\mathcal{F}_{\mathsf{th}})$. In the following $s \to^! t$ is used for $s \to^* t$ with t being a normal form with respect to $\to$.

Lemma 6.3. *Let $s, t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}})$. Then, all of the following holds: 1. $[\![t]\!] \in \mathcal{V}\mathsf{al}$, 2. $t \to^!_{\mathsf{ca}} [\![t]\!]$, 3. $s \to^*_{\mathsf{ca}} t$ implies $[\![s]\!] = [\![t]\!]$, and 4. $s \leftrightarrow^*_{\mathsf{calc}} t$ if and only if $[\![s]\!] = [\![t]\!]$.*

Proof (Sketch). 1. This claim follows as $[\![t^\tau]\!] \in \mathcal{I}(\tau) \cong \mathcal{V}\mathsf{al}^\tau$. *2.* Show $t \to^*_{\mathsf{ca}} [\![t]\!]$ by structural induction on t. Then, the claim follows, since values are normal forms with respect to calculation steps. *3.* We use the fact that the set of calculation rules forms a confluent LCTRS [45]. Since $s \to^!_{\mathsf{ca}} [\![s]\!]$ and $t \to^!_{\mathsf{ca}} [\![t]\!]$ from *2*, $s \to^*_{\mathsf{ca}} t$ implies $[\![s]\!] = [\![t]\!]$ by confluence. *4.* The *only-if* part follows from *3*, and the *if* part follows from *1*. □

The other type of rewrite steps in LCTRSs are rule steps specified by rewrite rules. Let us fix a signature $\langle \mathcal{S}_{\mathsf{th}}, \mathcal{S}_{\mathsf{te}}, \mathcal{F}_{\mathsf{th}}, \mathcal{F}_{\mathsf{te}} \rangle$. A constrained rule of an LCTRS is a triple $\ell \to r\ [\varphi]$ of terms ℓ, r with the same sort satisfying $\mathit{root}(\ell) \in \mathcal{F}_{\mathsf{te}}$ and a logical constraint φ. We define $\mathcal{LV}\mathsf{ar}(\ell \to r\ [\varphi]) = (\mathcal{V}\mathsf{ar}(r) \setminus \mathcal{V}\mathsf{ar}(\ell)) \cup \mathcal{V}\mathsf{ar}(\varphi)$, whose members are called *logical variables* of the rule. The intention is that the logical variables of rules in LCTRSs are required to be instantiated only by values. Let us also fix a model M. Then, a substitution γ is said to *respect a rewrite rule* $\ell \to r\ [\varphi]$ if $\mathcal{LV}\mathsf{ar}(\ell \to r\ [\varphi]) \subseteq \mathcal{VD}\mathsf{om}(\gamma)$ and $\models_M \varphi\gamma$. Using this notation, a rule step $s \to_{\mathsf{ru}} t$ over the model M by the rewrite rule $\ell \to r\ [\varphi]$ is given as follows: $s \to_{\mathsf{ru}} t$ if and only if $s = C[\ell\gamma]$ and $t = C[r\gamma]$ for some context C and some substitution γ that respects the rewrite rule $\ell \to r\ [\varphi]$.

Finally, a *logically constrained term rewrite system* (LCTRS, for short) consists of a signature $\Sigma = \langle \mathcal{S}_{\mathsf{th}}, \mathcal{S}_{\mathsf{te}}, \mathcal{F}_{\mathsf{th}}, \mathcal{F}_{\mathsf{te}} \rangle$, a model M over $\Sigma_{\mathsf{th}} = \langle \mathcal{S}_{\mathsf{th}}, \mathcal{F}_{\mathsf{th}} \rangle$ (which induces the set $\mathcal{V}\mathsf{al} \subseteq \mathcal{F}_{\mathsf{th}}$ of values) and a set $\mathcal{R}$ of constrained rules over the signature Σ. All this together defines rewrite steps consisting of calculation steps and rule steps. In a practical setting, often some predefined (semi-)decidable theories are assumed and used as model M and theory signature $\langle \mathcal{S}_{\mathsf{th}}, \mathcal{F}_{\mathsf{th}} \rangle$. An example of such a theory is *linear integer arithmetic*, whose model consists of standard boolean functions and the set of integers including standard predefined functions on them. From this point of view, we call the triple $\mathfrak{U} = \langle \mathcal{S}_{\mathsf{th}}, \mathcal{F}_{\mathsf{th}}, \mathcal{M} \rangle$ of the theory signature and its respective model the *underlying model* or *background theory* of the LCTRS. We also denote an LCTRS as $\langle \mathcal{M}, \mathcal{R} \rangle$ with an implicit signature or $\langle \mathcal{M}, \mathcal{R} \rangle$ over the signature $\Sigma = \langle \mathcal{S}_{\mathsf{th}}, \mathcal{S}_{\mathsf{te}}, \mathcal{F}_{\mathsf{th}}, \mathcal{F}_{\mathsf{te}} \rangle$ for an explicit signature.

6.3 Validity of Constrained Equational Theories

In this section, we introduce validity of constrained equational theories (CE-validity), which is a key concept used throughout the paper. Subsequently, we present fundamental properties of CE-validity, and show their relation to the conversion of rewriting.

Constrained Equational Theory and Its Validity

In this subsection, after introducing the notion of CEs, we define equational systems, which are sets of CEs, and rewriting with respect to such systems. This gives an equational version of the rewrite step in LCTRSs. Furthermore, based on these notions, we define the validity of CEs.

Recall that logical variables of a constrained rule are those which are only allowed to be instantiated by values. As we have seen in the previous section, rewrite steps of LCTRSs depend on the correct instantiation of the logical variables of the applied rule. However, the sets of logical variables $\mathcal{LV}\mathsf{ar}(\ell \to r\ [\varphi])$ and $\mathcal{LV}\mathsf{ar}(r \to \ell\ [\varphi])$ are not necessarily equivalent, and the CE $\ell \approx r\ [\varphi]$ alone does not suffice to specify the correct logical variables. This motivates us to add an explicit set X to the CE $\ell \approx r\ [\varphi]$ as $\Pi X.\, \ell \approx r\ [\varphi]$ which specifies its logical variables.

Definition 6.4 (constrained equation)**.** Let $\Sigma_{\mathsf{te}} = \langle \mathcal{S}_{\mathsf{te}}, \mathcal{F}_{\mathsf{te}} \rangle$ be a term signature over the underlying model $\mathfrak{U} = \langle \mathcal{S}_{\mathsf{th}}, \mathcal{F}_{\mathsf{th}}, \mathcal{M} \rangle$. A *constrained equation* (CE, for short) over $\mathfrak{U}$ and Σ_{te} is a quadruple $\Pi X.\, s \approx t\ [\varphi]$ where s, t are terms with the same sort, φ is a logical constraint, and $X \subseteq \mathcal{V}_{\mathsf{th}}$ is a set of theory variables satisfying $\mathcal{V}\mathsf{ar}(\varphi) \subseteq X$. A *logically constrained equational system* (LCES, for short) is a set of CEs. We abbreviate $\Pi X.\, s \approx t\ [\varphi]$ to $s \approx t\ [\varphi]$ if $\mathcal{V}\mathsf{ar}(\varphi) = X$. A CE $\Pi X.\, s \approx t\ [\mathsf{true}]$ is abbreviated to $\Pi X.\, s \approx t$.

We remark that a constrained rewrite rule $\ell \to r\ [\varphi]$ is naturally encoded as a CE $\Pi X.\, \ell \approx r\ [\varphi]$ by taking $X = \mathcal{LV}\mathsf{ar}(\ell \to r\ [\varphi])$. Furthermore, let us illustrate the aforementioned issues, without an explicit set of logical variables, by an example.

Example 6.5. Consider the LCTRS $\mathcal{R}$ over the theory of integer arithmetic and its (labeled) rules

$$\alpha\colon \mathsf{f}(x, y) \to \mathsf{g}(z)\ [x = 1] \qquad\qquad \beta\colon \mathsf{g}(z) \to \mathsf{f}(x, y)\ [x = 1]$$

with their sets of logical variables $\mathcal{LV}\mathsf{ar}(\alpha) = \{x, z\}$ and $\mathcal{LV}\mathsf{ar}(\beta) = \{x, y\}$. Transforming them naively into the CE $\mathsf{f}(x, y) \approx \mathsf{g}(z)\ [x = 1]$ and $\mathsf{g}(z) \approx \mathsf{f}(x, y)\ [x = 1]$ would give the set of logical variables $\{x\}$ for both. We use the notion of logical variables in Winkler and Middeldorp [88], where the set of logical variables of a CE consists of the variables appearing in the constraint. Obviously, we lose concrete information about the logical variables of the original rules. Clearly, in our notion this information remains intact: $\Pi\{x,z\}.\,\mathsf{f}(x,y) \approx \mathsf{g}(z)\ [x = 1]$ and $\Pi\{x,y\}.\,\mathsf{g}(z) \approx \mathsf{f}(x,y)\ [x = 1]$. Note that variables appearing solely in the set of logical variables and not in the CE have no effect but are allowed. For example, in the CE $\Pi\{x,z,z'\}.\,\mathsf{f}(x,y) \approx \mathsf{g}(z)\ [x = 1]$ the logical variable z' has no effect and could be dropped.

In the following we extend the notion of rewrite steps by using CEs instead of rewrite rules.

Definition 6.6 ($\leftrightarrow_E$)**.** Let E be an LCES over the underlying model $\mathfrak{U} = \langle \mathcal{S}_{\mathsf{th}}, \mathcal{F}_{\mathsf{th}}, M\rangle$ and the term signature $\Sigma_{\mathsf{te}} = \langle \mathcal{S}_{\mathsf{te}}, \mathcal{F}_{\mathsf{te}}\rangle$. For terms $s, t \in \mathcal{T}(\mathcal{F},\mathcal{V})$, we define a *rule* step $s \leftrightarrow_{\mathsf{rule},E} t$ if $s = C[\ell\sigma]$ and $t = C[r\sigma]$ (or vice versa) for some CE $\Pi X.\,\ell \approx r\ [\varphi] \in E$ and some X-valued substitution σ such that $\models_M \varphi\sigma$. Here, a substitution is said to be *X-valued* if $X \subseteq \mathcal{VD}\mathsf{om}(\sigma)$. We let $\leftrightarrow_E = \leftrightarrow_{\mathsf{calc}} \cup \leftrightarrow_{\mathsf{rule},E}$, where $\leftrightarrow_{\mathsf{calc}}$ is the symmetric closure of the calculation steps $\rightarrow_{\mathsf{ca}}$ specified by M.

We give examples on rewriting with CEs.

Example 6.7. Consider integer arithmetic as underlying model M. We consider the term sorts $\mathcal{S}_{\mathsf{te}} = \{\mathsf{Unit}\}$ and the term signature $\mathcal{F}_{\mathsf{te}} = \{\mathsf{cong}\colon \mathsf{Int} \to \mathsf{Unit}\}$ where Int is the respective sort of the integers. The set E of CEs consists of $\{\mathsf{cong}(x) \approx \mathsf{cong}(y)\ [\mathsf{mod}(x,12) = \mathsf{mod}(y,12)]\}$. Arithmetic values in intermediate steps of rewrite sequences wrapped in cong have the property that they are *congruent modulo 12* and thus E simulates modular arithmetic with modulus 12. Consider the following sequence:

$$\mathsf{cong}(7+31) \leftrightarrow_{\mathsf{calc}} \mathsf{cong}(38) \leftrightarrow_{\mathsf{rule},E} \mathsf{cong}(14)$$

From this we conclude that $7+31$, which gives 38, and 14 are congruent modulo 12. Note that the rule step $\leftrightarrow_{\mathsf{rule},E}$ does not allow to directly convert $\mathsf{cong}(7+31)$ and $\mathsf{cong}(14)$.

Example 6.8. Consider integer arithmetic as the underlying model M. We take a term signature $\mathcal{S}_{\mathsf{te}} = \{\mathsf{G}\}$ and $\mathcal{F}_{\mathsf{te}} = \{\mathsf{e}\colon \mathsf{G}, \mathsf{inv}\colon \mathsf{G} \to \mathsf{G}, *\colon \mathsf{G} \times \mathsf{G} \to \mathsf{G}, \mathsf{exp}\colon \mathsf{G} \times \mathsf{Int} \to \mathsf{G}\}$. Let the set E of CEs consist of:

$$\begin{aligned}
(x * y) * z &\approx x * (y * z) & \mathsf{e} * x &\approx x\\
\mathsf{inv}(x) * x &\approx \mathsf{e} & \mathsf{exp}(x,0) &\approx \mathsf{e}\\
\mathsf{exp}(x,1) &\approx x & \Pi\{n,m\}.\,\mathsf{exp}(x,n) * \mathsf{exp}(x,m) &\approx \mathsf{exp}(x,m+n)
\end{aligned}$$

As in first-order equational reasoning, one can show $x * \mathsf{e} \overset{*}{\leftrightarrow}_E x$. Thus, $\mathsf{exp}(x,-1) \leftrightarrow_E \mathsf{e} * \mathsf{exp}(x,-1) \leftrightarrow_E (\mathsf{inv}(x) * x) * \mathsf{exp}(x,-1) \leftrightarrow_E \mathsf{inv}(x) * (x * \mathsf{exp}(x,-1)) \leftrightarrow_E \mathsf{inv}(x) * (\mathsf{exp}(x,1) * \mathsf{exp}(x,-1)) \leftrightarrow_E \mathsf{inv}(x) * \mathsf{exp}(x,1+(-1)) \leftrightarrow_E \mathsf{inv}(x) * \mathsf{exp}(x,0) \leftrightarrow_E \mathsf{inv}(x) * \mathsf{e} \overset{*}{\leftrightarrow}_E \mathsf{inv}(x)$ as expected. This encodes a system of groups with an explicit exponentiation operator exp.

Example 6.9. Consider integer arithmetic as the underlying model M. We take a term signature $S_{\mathsf{te}} = \{\mathsf{Elem}, \mathsf{List}, \mathsf{ElemOp}\}$ and $\mathcal{F}_{\mathsf{te}} = \{\mathsf{nil}\colon \mathsf{List}, \mathsf{cons}\colon \mathsf{Elem} \times \mathsf{List} \to \mathsf{List}, \mathsf{none}\colon \mathsf{ElemOp}, \mathsf{some}\colon \mathsf{Elem} \to \mathsf{ElemOp}, \mathsf{length}\colon \mathsf{List} \to \mathsf{Int}, \mathsf{nth}\colon \mathsf{List} \times \mathsf{Int} \to \mathsf{ElemOp}\}$. Let the set E of CEs consist of

$$\begin{array}{rcl@{\qquad}rcll}
\mathsf{length}(\mathsf{nil}) & \approx & 0 & \mathsf{length}(\mathsf{cons}(x, xs)) & \approx & \mathsf{length}(xs) + 1 & \\
\Pi\{n\}.\, \mathsf{nth}(\mathsf{nil}, n) & \approx & \mathsf{none} & \mathsf{nth}(xs, n) & \approx & \mathsf{none} & [n < 0] \\
\mathsf{nth}(\mathsf{cons}(x, xs), 0) & \approx & \mathsf{some}(x) & \mathsf{nth}(\mathsf{cons}(x, xs), n) & \approx & \mathsf{nth}(xs, n - 1) & [n > 0]
\end{array}$$

This LCES encodes common list functions that use integers. For program verification purposes, one may deal with the validity problem of a formula such as $\mathsf{nth}(xs, n) \not\approx \mathsf{none} \Leftrightarrow 0 \leqslant n \wedge n < \mathsf{length}(xs)$.

We continue by giving some immediate facts which are used later on.

Lemma 6.10. *Let E be an LCES over the underlying model $\mathfrak{U} = \langle S_{\mathsf{th}}, \mathcal{F}_{\mathsf{th}}, M\rangle$ and the term signature $\Sigma_{\mathsf{te}} = \langle S_{\mathsf{te}}, \mathcal{F}_{\mathsf{te}}\rangle$. Then, all of the following hold: 1. $\leftrightarrow_E$ is symmetric, 2. $\leftrightarrow_E$ is closed under contexts i.e. $s \leftrightarrow_E t$ implies $C[s] \leftrightarrow_E C[t]$ for any context C, and 3. $\leftrightarrow_E$ is closed under substitutions, i.e. $s \leftrightarrow_E t$ implies $s\sigma \leftrightarrow_E t\sigma$ for any substitution σ.*

Proof (Sketch). *1.* and *2.* are trivial. For *3.*, the case $s \leftrightarrow_{\mathsf{calc}} t$ is clear. Suppose $s \leftrightarrow_{\mathsf{rule},E} t$. Then $s = C[\ell\rho]$ and $t = C[r\rho]$ (or vice versa) for some CE $\Pi X.\, \ell \approx r\ [\varphi] \in E$ and an X-valued substitution ρ such that $\models_M \varphi\rho$. Then $\varphi\rho = (\varphi\rho)\sigma = \varphi(\sigma \circ \rho)$ and hence $\models_M \varphi(\sigma \circ \rho)$. Then, the claim follows, as $s\sigma = C[\ell\rho]\sigma = C\sigma[\ell(\sigma \circ \rho)]$ and $t\sigma = C[r\rho]\sigma = C\sigma[r(\sigma \circ \rho)]$. □

We proceed by defining constrained equational theories (CE-theories) and validity of CEs (CE-validity) with respect to a CE-theory.

Definition 6.11 (constrained equational theory)**.** A *constrained equational theory* is specified by a triple $\mathfrak{T} = \langle \mathfrak{U}, \Sigma_{\mathsf{te}}, E\rangle$, where $\mathfrak{U} = \langle S_{\mathsf{th}}, \mathcal{F}_{\mathsf{th}}, M\rangle$ is an underlying model, Σ_{te} is a term signature over $\mathfrak{U}$ (as given in the LCTRS formalism), and E is an LCES over $\mathfrak{U}, \Sigma_{\mathsf{te}}$. If no confusion arises, we refer to the CE-theory by $\langle M, E\rangle$, without stating its signature explicitly. We also say that a CE-theory $\langle M, E\rangle$ is defined over the signature $\Sigma = \langle S_{\mathsf{th}}, S_{\mathsf{te}}, \mathcal{F}_{\mathsf{th}}, \mathcal{F}_{\mathsf{te}}\rangle$ in order to make the signature explicit.

Definition 6.12 (CE-validity)**.** Let $\mathfrak{T} = \langle M, E\rangle$ be a CE-theory. Then a CE $\Pi X.\, s \approx t\ [\varphi]$ is said to be a *constrained equational consequence* (CE-consequence, for short) *of* $\mathfrak{T}$ or *valid* (CE-valid, for clarity), written as $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$, if $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$ for all X-valued substitutions σ such that $\models_M \varphi\sigma$. We write $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$ if M is known from the context.

We conclude this subsection with an example on CE-validity.

Example 6.13. Consider integer arithmetic as the underlying model M. We take the term signature $\mathcal{F}_{\mathsf{te}} = \{\mathsf{abs}\colon \mathsf{Int} \to \mathsf{Int}, \mathsf{max}\colon \mathsf{Int} \times \mathsf{Int} \to \mathsf{Int}\}$ the set of CEs E consisting of

$$\begin{array}{rcll@{\qquad}rcll}
\mathsf{abs}(x) & \approx & -x & [x < 0] & \mathsf{abs}(x) & \approx & x & [x \geqslant 0] \\
\mathsf{max}(x, y) & \approx & x & [x \geqslant y] & \mathsf{max}(x, y) & \approx & y & [x < y]
\end{array}$$

The following are valid CE-consequences:

$$\mathfrak{T} \models_{\mathsf{cec}} \Pi\{x\}.\, \mathsf{abs}(x) \approx \mathsf{abs}(-x) \qquad \mathfrak{T} \models_{\mathsf{cec}} \Pi\{x, y\}.\, \mathsf{max}(x, y) \approx \mathsf{max}(y, x)$$
$$\mathfrak{T} \models_{\mathsf{cec}} \Pi\{x, y\}.\, \mathsf{abs}(\mathsf{max}(x, y)) \approx \mathsf{max}(\mathsf{abs}(x), \mathsf{abs}(y))\ [0 \leqslant x \land 0 \leqslant y]$$

On the other hand, the CE $\Pi\varnothing.\, \mathsf{abs}(x) \approx \mathsf{abs}(-x)$ is not a valid CE-consequence: For the $\varnothing$-valued identity substitution σ, we have that $\mathsf{abs}(x)\sigma = \mathsf{abs}(x) \not\overset{*}{\leftrightarrow}_E \mathsf{abs}(-x) = \mathsf{abs}(-x)\sigma$.

Properties of CE-Validity

This subsection covers important properties related to CE-validity, for example, we show that validity forms an equivalence and a congruence relation. Furthermore, we cover in which way it is closed under substitutions and contexts, and how equality can be induced from constraints.

Our first two lemmas follow immediately from the definition of the CE-validity.

Lemma 6.14. *Let $\mathfrak{T} = \langle M, E\rangle$ be a CE-theory. Then for any $\Pi X.\, s \approx t\ [\varphi] \in E$, we have $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$.*

Lemma 6.15 (congruence)**.** *Let $\mathfrak{T} = \langle M, E\rangle$ be a CE-theory. For any set $X \subseteq \mathcal{V}_{\mathsf{th}}$ and logical constraint φ such that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq X$, the binary relation $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, \cdot \approx \cdot\ [\varphi]$ over terms is a congruence relation over Σ.*

For stability under substitutions, we differentiate two kinds; for each CE $\Pi X.\, s \approx t\ [\varphi]$, the first one considers substitutions instantiating variables in X; the second one considers substitutions instantiating variables not in X.

Lemma 6.16 (stability of theory terms)**.** *Let $\mathfrak{T} = \langle M, E\rangle$ be a CE-theory. Let $X, Y \subseteq \mathcal{V}_{\mathsf{th}}$ be sets of theory variables and σ a substitution such that $\sigma(y) \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$ for any $y \in Y$. If $\mathfrak{T} \models_{\mathsf{cec}} \Pi Y.\, s \approx t\ [\varphi]$, then $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, s\sigma \approx t\sigma\ [\varphi\sigma]$.*

Proof (Sketch). Take any X-valued substitution θ with $\models_M (\varphi\sigma)\theta$. This gives a Y-valued substitution ξ by defining $\xi(y) = [\![(\theta \circ \sigma)(y)]\!]$ for each $y \in Y$. From Lemma 6.3, we know $(\theta \circ \sigma)(y) \overset{*}{\leftrightarrow}_E \xi(y)$ for any $y \in Y$. We also have $\models_M \varphi\xi$ by Lemma 6.1, and hence $s\xi \overset{*}{\leftrightarrow}_E t\xi$ by assumption. Thus, using Lemma 6.10, we obtain $(s\sigma)\theta = s(\theta \circ \sigma) \overset{*}{\leftrightarrow}_E s\xi \overset{*}{\leftrightarrow}_E t\xi \overset{*}{\leftrightarrow}_E t(\theta \circ \sigma) = (t\sigma)\theta$. □

Lemma 6.17 (general stability)**.** *Let $\langle M, E\rangle$ be a CE-theory and σ a substitution such that $\mathcal{D}\mathsf{om}(\sigma) \cap X = \varnothing$. Then, if $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$ then $E \models_{\mathsf{cec}} \Pi X.\, s\sigma \approx t\sigma\ [\varphi]$.*

Proof (Sketch). Take any X-valued substitution δ such that $\models_M \varphi\delta$. Take $\gamma = \delta \circ \sigma$. From $\mathcal{D}\mathsf{om}(\sigma) \cap X = \varnothing$, we have $\varphi\gamma = \varphi\delta$, and therefore, $s\sigma\delta \overset{*}{\leftrightarrow}_E t\sigma\delta$ holds. □

One may expect that $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$ holds for equivalent terms s, t such that $\varphi \Rightarrow s = t$ is valid. In fact, a more general result can be obtained.

Lemma 6.18 (model consequence)**.** *Let $\langle M, E\rangle$ be a CE-theory, $X \subseteq \mathcal{V}_{\mathsf{th}}$ a set of theory variables, $s, t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$, and φ a logical constraint over X. If $\models_M (\varphi\sigma \Rightarrow s\sigma = t\sigma)$ holds for all X-valued substitutions σ, then $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$.*

Proof (Sketch). For any X-valued substitution σ with $\models_M \varphi\sigma$, we have $[\![s\sigma]\!]_M = [\![t\sigma]\!]_M$. Then, use Lemma 6.3 to obtain $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$. □

Corollary 6.19. *Let $\langle M, E\rangle$ be a CE-theory, $X \subseteq \mathcal{V}_{\mathsf{th}}$ a set of theory variables, and $\varphi \Rightarrow s = t$ a logical constraint over X such that $\models_M (\varphi \Rightarrow s = t)$. Then, $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$.*

Relations to Conversion of Rewrite Steps

In this subsection, we present characterizations of CE-validity from the perspective of logically constrained rewriting with respect to equations.

Theorem 6.20. *For a CE-theory $\langle M, E\rangle$, $s \overset{*}{\leftrightarrow}_E t$ if and only if $E \models_{\mathsf{cec}} \Pi\varnothing.\, s \approx t\ [\mathsf{true}]$.*

Proof (Sketch). We have $E \models_{\mathsf{cec}} \Pi\varnothing.\, s \approx t\ [\mathsf{true}]$ if and only if $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$ for any $\varnothing$-valued substitution σ such that $\sigma \models \mathsf{true}$ if and only if $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$ for any substitution σ. Thus, the claim follows by Lemma 6.10. □

We consider now the general case with a possibly non-empty set X of theory variables and a non-trivial constraint $\varphi \neq \mathsf{true}$ (also $\neg\varphi$) for the CE $\Pi X.\, s \approx t\ [\varphi]$. In this case, the following partial characterization can be made by using the notion of trivial CEs [64]. We can naturally extend the notion of trivial CEs in [64] to our setting as follows: a CE $\Pi X.\, s \approx t\ [\varphi]$ is said to be *trivial* if $s\sigma = t\sigma$ for any X-valued substitution σ such that $\models_M \varphi\sigma$.

Theorem 6.21. *Let $\langle M, E\rangle$ be a CE-theory, and $\Pi X.\, s \approx t\ [\varphi]$ a CE. Suppose $s \overset{*}{\leftrightarrow}_E s'$ and $t \overset{*}{\leftrightarrow}_E t'$ for some s', t' such that $\Pi X.\, s' \approx t'\ [\varphi]$ is trivial. Then, $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$.*

Proof (Sketch). Take an arbitrary X-valued substitution σ such that $\models_M \varphi\sigma$. Then, it follows from our assumptions that $s\sigma \overset{*}{\leftrightarrow}_E s'\sigma = t'\sigma \overset{*}{\leftrightarrow}_E t\sigma$. □

Unfortunately, none of the CE-consequences in Example 6.13 can be handled by Theorems 6.20 and 6.21.

6.4 Proving CE-Validity

As mentioned in the previous section, CE-validity of a CE $\Pi X.\, s \approx t\ [\varphi]$ with respect to a CE-theory $\langle M, E\rangle$ is very tedious to be shown by the convertibility of s and t in E. This motivates us to introduce another approach to reason about CE-validity. In this section, we introduce an inference system for proving CE-validity of CEs together with a discussion on its applicability and generality.

$$\textit{Refl}: \quad \frac{}{E \vdash \Pi X.\, s \approx s\ [\varphi]}\ \mathcal{V}\mathsf{ar}(\varphi) \subseteq X \qquad \textit{Trans}: \quad \frac{E \vdash \Pi X.\, s \approx t\ [\varphi] \quad E \vdash \Pi X.\, t \approx u\ [\varphi]}{E \vdash \Pi X.\, s \approx u\ [\varphi]}$$

$$\textit{Sym}: \quad \frac{E \vdash \Pi X.\, t \approx s\ [\varphi]}{E \vdash \Pi X.\, s \approx t\ [\varphi]} \qquad \textit{Cong}: \quad \frac{E \vdash \Pi X.\, s_1 \approx t_1\ [\varphi] \quad \dots \quad E \vdash \Pi X.\, s_n \approx t_n\ [\varphi]}{E \vdash \Pi X.\, f(s_1, \dots, s_n) \approx f(t_1, \dots, t_n)\ [\varphi]}$$

$$\textit{Rule}: \quad \frac{}{E \vdash \Pi X.\, \ell \approx r\ [\varphi]}\ (\Pi X.\, \ell \approx r\ [\varphi]) \in E$$

$$\textit{Theory Instance}: \quad \frac{E \vdash \Pi Y.\, s \approx t\ [\varphi]}{E \vdash \Pi X.\, s\sigma \approx t\sigma\ [\varphi\sigma]}\ \forall x \in Y.\ x\sigma \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$$

$$\textit{General Instance}: \quad \frac{E \vdash \Pi X.\, s \approx t\ [\varphi]}{E \vdash \Pi X.\, s\sigma \approx t\sigma\ [\varphi]}\ \mathcal{D}\mathsf{om}(\sigma) \cap X = \varnothing$$

$$\textit{Weakening}: \quad \frac{E \vdash \Pi X.\, s \approx t\ [\psi]}{E \vdash \Pi X.\, s \approx t\ [\varphi]}\ \models_M (\varphi \Rightarrow \psi), \mathcal{V}\mathsf{ar}(\varphi) \subseteq X$$

$$\textit{Split}: \quad \frac{E \vdash \Pi X.\, s \approx t\ [\varphi] \quad E \vdash \Pi X.\, s \approx t\ [\psi]}{E \vdash \Pi X.\, s \approx t\ [\varphi \vee \psi]}$$

$$\textit{Axiom}: \quad \frac{}{E \vdash \Pi X.\, s \approx t\ [\varphi]}\ \begin{array}{l} \models_M (\varphi\sigma \Rightarrow s\sigma = t\sigma) \text{ for all } \sigma \text{ s.t. } X \subseteq \mathcal{VD}\mathsf{om}(\sigma), \\ \mathcal{V}\mathsf{ar}(\varphi) \subseteq X \end{array}$$

$$\textit{Abst}: \quad \frac{E \vdash \Pi X.\, s\sigma \approx t\sigma\ [\varphi\sigma]}{E \vdash \Pi X.\, s \approx t\ [\varphi]}\ \begin{array}{l} \models_M (\varphi \Rightarrow \bigwedge_{x \in X} x = \sigma(x)), \\ \mathcal{V}\mathsf{ar}(\varphi) \subseteq X, (\bigcup_{x \in X} \mathcal{V}\mathsf{ar}(\sigma(x))) \subseteq X \end{array}$$

$$\textit{Enlarge}: \quad \frac{E \vdash \Pi Y.\, s \approx t\ [\varphi]}{E \vdash \Pi X.\, s \approx t\ [\varphi]}\ \mathcal{V}\mathsf{ar}(s, t) \cap (Y \setminus X) = \varnothing,\ \mathcal{V}\mathsf{ar}(\varphi) \subseteq X$$

Figure 6.2: Inference rules of $\mathbf{CEC}_0$.

Inference System CEC_0 and Its Soundness

In this subsection, we introduce an inference system $\mathbf{CEC}_0$ (*Constrained Equational Calculus for elementary steps*) for proving CE-validity of CEs. We prove soundness of it, by which it is guaranteed that all CEs $\Pi X.\, s \approx t\ [\varphi]$ derivable from E in the system $\mathbf{CEC}_0$ are valid, i.e. $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$.

Definition 6.22 (derivation of $\mathbf{CEC}_0$)**.** Let $\mathfrak{T} = \langle M, E \rangle$ be a CE-theory. The inference system $\mathbf{CEC}_0$ consists of the inference rules given in Figure 6.2. We assume in the rules that X, Y range over a (possibly infinite) set of theory variables, φ ranges over logical constraints. Let $\Pi X.\, s \approx t\ [\varphi]$ be a CE. We say that $\Pi X.\, s \approx t\ [\varphi]$ is derivable in $\mathbf{CEC}_0$ from E (or $\Pi X.\, s \approx t\ [\varphi]$ is a consequence of E), written by $E \vdash_{\mathbf{CEC}_0} \Pi X.\, s \approx t\ [\varphi]$, if there exists a derivation of $E \vdash \Pi X.\, s \approx t\ [\varphi]$ in the system $\mathbf{CEC}_0$. When no confusion arises, $E \vdash_{\mathbf{CEC}_0} \Pi X.\, s \approx t\ [\varphi]$ is abbreviated as $E \vdash \Pi X.\, s \approx t\ [\varphi]$.

We proceed with intuitive explanations of each rule. The rules *Refl*, *Trans*, *Sym*, *Cong*, and *Rule* are counterparts of the inference rules used in equational logic.

In order to handle instantiations, we consider two cases, namely *Theory Instance* and *General Instance*. The former rule covers instantiations affecting the logical constraint whereas the latter covers the case not affecting it.

The *Weakening* and *Split* rules handle logical reasoning in constraints. The *Weakening* rule logically weakens the constraint equation by strengthening its constraint. Note here the direction of the entailment $\varphi \Rightarrow \psi$ in the side condition: the rule is sound because the constrained equation is valid under the stronger constraint φ if the equation is valid under the weaker constraint ψ. Since some rules, like *Cong* and *Trans*, have premises with equality constraints, it may be required to first apply the *Weakening* rule to synchronize the constraints before using these rules. On the other hand, in the *Split* rule, the constraint of the conclusion $\varphi \vee \psi$ is logically weaker than the independent ones, φ and ψ, in each premise. The inference is still sound as it only joins two premises. Using the *Split* rule, one can perform reasoning based on case analysis.

The *Axiom* rule makes it possible to use equational consequences entailed in the constraint part of equational reasoning. The *Abst* rule incorporates consequences entailed in the constraint part in a different way, that is, to infer a possible abstraction of the equation.

The rules *Enlarge* and *Restrict* are used to adjust the set of instantiated variables (the "ΠX" part of CEs), with the proviso that it does not affect the validity. Note that, in *Enlarge*, $Y \subseteq X$ implies the side condition $\mathcal{V}\mathsf{ar}(s,t) \cap (Y \setminus X) = \varnothing$. We also want to remark that despite its name, the restriction $X \subseteq Y$ can be added to *Enlarge*, provided that removed variables are not used in s, t (the side condition $\mathcal{V}\mathsf{ar}(s,t) \cap (Y \setminus X) = \varnothing$ has to be satisfied).

Lemma 6.23. *Let $\langle M, E\rangle$ be a CE-theory. If $E \vdash_{\mathbf{CEC}_0} \Pi X.\, s \approx t\ [\varphi]$, then $\Pi X.\, s \approx t\ [\varphi]$ is a CE.*

Proof (Sketch). The proof proceeds by induction on the derivation of $E \vdash \Pi X.\, s \approx t\ [\varphi]$ that (1) $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ have the same sort, (2) φ is a logical constraint, (3) $X \subseteq \mathcal{V}_{\mathsf{th}}$, and (4) $\mathcal{V}\mathsf{ar}(\varphi) \subseteq X$. □

Below we present some examples of derivations which cover all of our inference rules at least once. In the following example we denote *Theory Instance* by *TInst* and *General Instance* by *GInst* accordingly.

Example 6.24. Let $\langle M, E\rangle$ be the CE-theory given in Example 6.8. Below, we present a derivation of $E \vdash \Pi\{n\}.\, \mathsf{exp}(x,n) * \mathsf{exp}(x,-n) \approx \mathsf{e}$.

$$\dfrac{\dfrac{\dfrac{}{E \vdash \Pi\{n,m\}.\, \mathsf{exp}(x,n) * \mathsf{exp}(x,m) \approx \mathsf{exp}(x,n+m)}\,\mathit{Rule}}{E \vdash \Pi\{n\}.\, \mathsf{exp}(x,n) * \mathsf{exp}(x,-n) \approx \mathsf{exp}(x,n+(-n))}\,\mathit{TInst} \qquad (1)}{E \vdash \Pi\{n\}.\, \mathsf{exp}(x,n) * \mathsf{exp}(x,-n) \approx \mathsf{e}}\,\mathit{Trans}$$

where (1) is

$$\dfrac{\dfrac{\dfrac{\dfrac{}{E \vdash x \approx x}\,\mathit{Refl}}{E \vdash \Pi\{n\}.\, x \approx x}\,\mathit{Enlarge} \quad \dfrac{}{E \vdash \Pi\{n\}.\, n + (-n) \approx 0}\,\mathit{Axiom}}{E \vdash \Pi\{n\}.\, \mathsf{exp}(x, n+(-n)) \approx \mathsf{exp}(x,0)}\,\mathit{Cong} \quad \dfrac{\dfrac{}{E \vdash \mathsf{exp}(x,0) \approx \mathsf{e}}\,\mathit{Rule}}{E \vdash \Pi\{n\}.\, \mathsf{exp}(x,0) \approx \mathsf{e}}\,\mathit{Enlarge}}{E \vdash \Pi\{n\}.\, \mathsf{exp}(x, n+(-n)) \approx \mathsf{e}}\,\mathit{Trans}$$

Here, $E \vdash \Pi\{n\}.\, n + (-n) \approx 0$ is derived by the *Axiom* rule, because of $\models_M \sigma(n + (-n)) = 0$ holds for all $\{n\}$-valued substitutions σ.

Example 6.25. Let $\langle M, E\rangle$ be the CE-theory given in Example 6.9 where $::$ is used for cons. Below, we present a derivation of $E \vdash \Pi\{n\}.\, \mathsf{nth}(x{::}y{::}zs, n{+}2) \approx \mathsf{nth}(zs, n)\ [n > 0]$.

$$\dfrac{(2) \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{E \vdash \Pi\{n\}.\, \mathsf{nth}(x{::}xs, n) \approx \mathsf{nth}(xs, n-1)\ [n > 0]}\,\textit{Rule}}{E \vdash \Pi\{n\}.\, \mathsf{nth}(x{::}xs, n+1) \approx \mathsf{nth}(xs, (n+1)-1)\ [n+1 > 0]}\,\textit{TInst} \qquad (1)}{E \vdash \Pi\{n\}.\, \mathsf{nth}(x{::}xs, n+1) \approx \mathsf{nth}(xs, n)\ [n+1 > 0]}\,\textit{Trans}}{E \vdash \Pi\{n\}.\, \mathsf{nth}(x{::}xs, n+1) \approx \mathsf{nth}(xs, n)\ [n > 0]}\,\textit{Weaken}}{E \vdash \Pi\{n\}.\, \mathsf{nth}(y{::}zs, n+1) \approx \mathsf{nth}(zs, n)\ [n > 0]}\,\textit{GInst}}{E \vdash \Pi\{n\}.\, \mathsf{nth}(x{::}y{::}zs, n+2) \approx \mathsf{nth}(zs, n)\ [n > 0]}\,\textit{Trans}$$

where (1) is

$$\dfrac{\dfrac{\dfrac{}{E \vdash \Pi\{n\}.\, xs \approx xs}\,\textit{Refl} \qquad \dfrac{}{E \vdash \Pi\{n\}.\, (n+1) - 1 \approx n}\,\textit{Axiom}}{E \vdash \Pi\{n\}.\, \mathsf{nth}(xs, ((n+1)-1) \approx \mathsf{nth}(xs, n)}\,\textit{Cong}}{E \vdash \Pi\{n\}.\, \mathsf{nth}(xs, ((n+1)-1) \approx \mathsf{nth}(xs, n)\ [n+1 > 0]}\,\textit{Weaken}$$

and (2) is

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{E \vdash \Pi\{n\}.\, \mathsf{nth}(x :: xs, n) \approx \mathsf{nth}(xs, n-1)\ [n > 0]}\,\textit{Rule}}{E \vdash \Pi\{n\}.\, \mathsf{nth}(x :: xs, n+2) \approx \mathsf{nth}(xs, (n+2)-1)\ [n+2 > 0]}\,\textit{TInst} \qquad (3)}{E \vdash \Pi\{n\}.\, \mathsf{nth}(x :: xs, n+2) \approx \mathsf{nth}(xs, n+1)\ [n+2 > 0]}\,\textit{Trans}}{E \vdash \Pi\{n\}.\, \mathsf{nth}(x :: ys, n+2) \approx \mathsf{nth}(ys, n+1)\ [n > 0]}\,\textit{Weaken}}{E \vdash \Pi\{n\}.\, \mathsf{nth}(x :: y :: zs, n+2) \approx \mathsf{nth}(y :: zs, n+1)\ [n > 0]}\,\textit{GInst}$$

where (3) is

$$\dfrac{\dfrac{\dfrac{}{E \vdash \Pi\{n\}.\, xs \approx xs}\,\textit{Refl} \qquad \dfrac{}{E \vdash \Pi\{n\}.\, (n+2) - 1 \approx n+1}\,\textit{Axiom}}{E \vdash \Pi\{n\}.\, \mathsf{nth}(xs, ((n+2)-1) \approx \mathsf{nth}(xs, n+1)}\,\textit{Cong}}{E \vdash \Pi\{n\}.\, \mathsf{nth}(xs, (n+2)-1) \approx \mathsf{nth}(xs, n+1)\ [n+2 > 0]}\,\textit{Weaken}$$

In the next example, we illustrate applications of the *Split* and the *Abst* rule.

Example 6.26. Let $\mathsf{f} \colon \mathsf{Bool} \to \mathsf{Int} \in \mathcal{F}_{\mathsf{te}}$. Let $E = \{\Pi\varnothing.\, \mathsf{f}(\mathsf{true}) \approx 0\ [\mathsf{true}], \Pi\varnothing.\, \mathsf{f}(\mathsf{false}) \approx 0\ [\mathsf{true}]\}$. Then we have $\mathsf{f}(\mathsf{true}) \leftrightarrow_E 0$ and $\mathsf{f}(\mathsf{false}) \leftrightarrow_E 0$. Thus, for all $\{x\}$-valued substitutions σ, we have $\mathsf{f}(x)\sigma \overset{*}{\leftrightarrow}_E 0\sigma$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{E \vdash \Pi\varnothing.\, \mathsf{f}(\mathsf{true}) \approx 0}\,\textit{Rule}}{E \vdash \Pi\varnothing.\, \mathsf{f}(\mathsf{true}) \approx 0\ [\mathsf{true} = \mathsf{true}]}\,\textit{Weaken}}{E \vdash \Pi\{x\}.\, \mathsf{f}(x) \approx 0\ [x = \mathsf{true}]}\,\textit{Abst} \qquad \dfrac{\dfrac{\dfrac{}{E \vdash \Pi\varnothing.\, \mathsf{f}(\mathsf{false}) \approx 0}\,\textit{Rule}}{E \vdash \Pi\varnothing.\, \mathsf{f}(\mathsf{false}) \approx 0\ [\mathsf{false} = \mathsf{false}]}\,\textit{Weaken}}{E \vdash \Pi\{x\}.\, \mathsf{f}(x) \approx 0\ [x = \mathsf{false}]}\,\textit{Abst}}{E \vdash \Pi\{x\}.\, \mathsf{f}(x) \approx 0\ [x = \mathsf{true} \lor x = \mathsf{false}]}\,\textit{Split}}{E \vdash \Pi\{x\}.\, \mathsf{f}(x) \approx 0\ [\mathsf{true}]}\,\textit{Weaken}$$

We now present soundness of the system $\mathbf{CEC}_0$ with respect to CE-validity.

Theorem 6.27 (soundness of the system $\mathbf{CEC}_0$)**.** *Let $\langle M, E\rangle$ be a CE-theory and $\Pi X.\, s \approx t\ [\varphi]$ a CE. If $E \vdash_{\mathbf{CEC}_0} \Pi X.\, s \approx t\ [\varphi]$, then $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$.*

Proof (Sketch). The proof proceeds by induction on the derivation. The cases for all rules except for *Abst* and *Enlarge* follow by Lemmas 6.14–6.18 and well-known properties in logic. For the case *Abst*, suppose that $E \vdash \Pi X.\, s \approx t\ [\varphi]$ is derived from $E \vdash \Pi X.\, s\sigma \approx t\sigma\ [\varphi\sigma]$ as given in Figure 6.2. Let ρ be an X-valued substitution such that $\models_M \varphi\rho$. Then by the side conditions we have $[\![\rho(x)]\!] = [\![\rho(\sigma(x))]\!]$ for any $x \in X$. Hence, $\models_M \varphi\sigma\rho$, and $s\sigma\rho \overset{*}{\leftrightarrow}_E t\sigma\rho$ by the induction hypothesis. Then by $\mathcal{V}\mathsf{ar}(s,t) \subseteq X$, we have $s\rho \overset{*}{\leftrightarrow}_E s\sigma\rho \overset{*}{\leftrightarrow}_E t\sigma\rho \overset{*}{\leftrightarrow}_E t\rho$. For the case *Enlarge*, suppose that $E \vdash \Pi X.\, s \approx t\ [\varphi]$ is derived from $E \vdash \Pi Y.\, s \approx t\ [\varphi]$ as given in Figure 6.2. Let δ be an X-valued substitution such that $\models_M \varphi\delta$. Define a substitution δ' as follows: $\delta'(x^\tau) = c^\tau$ if $x \in Y \setminus X$, and $\delta'(x) = \delta(x)$ otherwise, where c^τ is (arbitrarily) taken from $\mathcal{V}\mathsf{al}^\tau$. Clearly, δ' is Y-valued. We also have $s\delta' = s\delta$, $t\delta' = t\delta$, and $\varphi\delta' = \varphi\delta$ by the side conditions. Thus, $s\delta = s\delta' \overset{*}{\leftrightarrow}_E t\delta' = t\delta$ using the induction hypothesis. □

Remark 6.28. We remark that some of our inference rules utilize validity in the model. Thus, $\mathbf{CEC}_0$ does not have convenient properties like recursive enumerability of its theorems like we are used to from other formal systems.

Partial Completeness of CEC_0

In this subsection, we present some results regarding the completeness property of $\mathbf{CEC}_0$.

Lemma 6.29. *Let $\langle M, E\rangle$ be a CE-theory and $\Pi X.\, s \approx t\ [\varphi]$ a CE such that φ is satisfiable. Suppose $s\sigma = t\sigma$ for all X-valued substitutions σ such that $\models_M \varphi\sigma$. Then $E \vdash \Pi X.\, s \approx t\ [\varphi]$.*

Proof (Sketch). The case $s, t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$ follows by the assumption using the *Axiom* rule. Next, we consider the case $s = x \in \mathcal{V}$ with $x \notin X$. In this case, we can derive $t = x$ using the assumption, and the case follows using the *Refl* rule. For the general case, from the assumption, one can let $s = C[s_1, \dots, s_n]$ and $t = C[t_1, \dots, t_n]$ for a multi-hole context C and terms s_i, t_i $(1 \leqslant i \leqslant n)$ such that either one of s_i or t_i is a variable. Thus, by the previous cases, we know that $E \vdash \Pi X.\, s_i \approx t_i\ [\varphi]$ for each $1 \leqslant i \leqslant n$, possibly using the *Sym* rule. Thus, the claim is obtained using *Refl*, *Trans* and *Cong* rules. □

Lemma 6.30. *Let $\langle M, E\rangle$ be a CE-theory and $\Pi X.\, s \approx t\ [\varphi]$ a CE such that φ is satisfiable. Suppose $s\sigma \leftrightarrow^{=}_{\mathsf{calc}} t\sigma$ for all X-valued substitutions σ such that $\mathcal{D}\mathsf{om}(\sigma) = X$ and $\models_M \varphi\sigma$. Then $E \vdash \Pi X.\, s \approx t\ [\varphi]$.*

Proof (Sketch). First of all, the claim for the case $s, t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$ follows using the *Axiom* rule. If $s\sigma = t\sigma$ for all X-valued substitutions σ, the claim follows from Lemma 6.29. Thus, it remains to consider the case that there exists an X-valued substitution σ such that $s\sigma \leftrightarrow_{\mathsf{calc}} t\sigma$, $\models_M \varphi\sigma$ and $X = \mathcal{D}\mathsf{om}(\sigma)$. Suppose that $s\sigma = C[f(v_1, \dots, v_n)]_q$ and $t\sigma = C[v_0]_q$ with $f \in \mathcal{F}_{\mathsf{th}}$ and $v_0, \dots, v_n \in \mathcal{V}\mathsf{al}$ such that $\mathcal{I}(f)(v_1, \dots, v_n) = v_0$. As $\mathcal{D}\mathsf{om}(\sigma) = X$ and σ is X-valued, we have $s|_q = f(s_1, \dots, s_n)$ with $s_1, \dots, s_n \in \mathcal{V}\mathsf{al} \cup X$

and $t|_q \in \mathcal{V}\mathsf{al} \cup X$; hence $s|_q, t|_q \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$, and thus, $E \vdash \Pi X.\, s|_q \approx t|_q\ [\varphi]$ holds as we mentioned above. Let $s = C_1[f(s_1, \dots, s_n)]_q$ and $t = C_2[t_0]_q$ with $C_1\sigma = C_2\sigma$ for any X-valued substitution σ such that $\mathcal{D}\mathsf{om}(\sigma) = X$ and $\models_M \varphi\sigma$. Then, $E \vdash \Pi X.\, s \approx t\ [\varphi]$ follows using $E \vdash \Pi X.\, s|_q \approx t|_q\ [\varphi]$, Lemma 6.29, and the *Cong* and *Trans* rules. □

From Lemma 6.30, the partial completeness for at most one calculation step follows.

Theorem 6.31. *Let $\langle M, E\rangle$ be a CE-theory and $\Pi X.\, s \approx t\ [\varphi]$ a CE such that φ is satisfiable. Suppose $s\sigma \leftrightarrow^{=}_{\mathsf{calc}} t\sigma$ for all X-valued substitutions σ such that $\models_M \varphi\sigma$. Then $E \vdash_{\mathbf{CEC}_0} \Pi X.\, s \approx t\ [\varphi]$.*

One may expect that Theorem 6.31 can be extended to the general completeness theorem for arbitrary E in such a way that $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$ implies $E \vdash \Pi X.\, s \approx t\ [\varphi]$ (full completeness). Rephrasing this, we have: if $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$ for all X-valued substitutions σ such that $\models_M \varphi\sigma$, then $E \vdash \Pi X.\, s \approx t\ [\varphi]$. The partial completeness result above is far from this formulation of full completeness in that the assumption does not assume arbitrary conversions $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$ but only $s\sigma \leftrightarrow^{=}_{\mathsf{calc}} t\sigma$ (i.e. at most one calculation step). However, full completeness does not seem to hold for the system $\mathbf{CEC}_0$, as witnessed by the following example.

Example 6.32. Consider the following LCES:

$$E = \left\{ \begin{array}{ll} \Pi\{x\}.\, \mathsf{nneg}(x) \approx \mathsf{true}\ [x = 0] & (1) \\ \Pi\{x, y\}.\, \mathsf{nneg}(x) \approx \mathsf{nneg}(y)\ [x + 1 = y] & (2) \end{array} \right.$$

For each $n \geqslant 0$, we have $\mathsf{nneg}(n) \overset{*}{\leftrightarrow}_E \mathsf{true}$:

$$\mathsf{nneg}(n) \leftrightarrow_E \mathsf{nneg}(n-1) \leftrightarrow_E \cdots \leftrightarrow_E \mathsf{nneg}(0) \leftrightarrow_E \mathsf{true}$$

Thus, for the CE $\Pi\{x\}.\, \mathsf{nneg}(x) \approx \mathsf{true}\ [x \geqslant 0]$, we have for all σ such that $\models_M \sigma(x) \geqslant 0$, $\mathsf{nneg}(x)\sigma \overset{*}{\leftrightarrow}_E \mathsf{true}\sigma = \mathsf{true}$. On the other hand, for each $n \geqslant 0$, one can give a derivation of $\Pi\varnothing.\, \mathsf{nneg}(n) \approx \mathsf{true}$—for example, for $n = 2$,

$$\dfrac{\dfrac{\dfrac{}{(2)}\,\mathit{Rule}}{\Pi\varnothing.\, \mathsf{nneg}(2) \approx \mathsf{nneg}(1)}\,\mathit{TInst} \qquad \dfrac{\dfrac{\dfrac{}{(2)}\,\mathit{Rule}}{\Pi\varnothing.\, \mathsf{nneg}(1) \approx \mathsf{nneg}(0)}\,\mathit{I} \quad \dfrac{\dfrac{}{(1)}\,\mathit{Rule}}{\Pi\varnothing.\, \mathsf{nneg}(0) \approx \mathsf{true}}\,\mathit{TInst}}{\Pi\varnothing.\, \mathsf{nneg}(1) \approx \mathsf{true}}\,\mathit{Trans}}{\Pi\varnothing.\, \mathsf{nneg}(2) \approx \mathsf{true}}\,\mathit{Trans}$$

However, these derivations differ for each $n \geqslant 0$, and are hardly merged. As a conclusion, it seems that the CE $\Pi\{x\}.\, \mathsf{nneg}(x) \approx \mathsf{true}\ [x \geqslant 0]$ is beyond the derivability of $\mathbf{CEC}_0$.

Remark 6.33. After looking at Example 6.32, it might seem reasonable that adding some kind of induction reasoning is required for our proof system. However, rules for the induction on positive integers, etc. are only possible when working with a specific model. Such rules have clearly a different nature than rules in our calculus that work with any underlying model. Our calculus $\mathbf{CEC}_0$ intends to be a general calculus that is free from specific underlying models and does not include model-specific rules.

Remark 6.34. We remark a difficulty to extend Theorem 6.31 to multiple (calculation) steps, i.e. to have a statement like $E \vdash_{\mathbf{CEC}_0} \Pi X.\, s \approx t\ [\varphi]$ whenever $s\sigma \leftrightarrow^*_{\mathsf{calc}} t\sigma$ for all X-valued substitutions σ such that $\models_M \varphi\sigma$. Or even to obtain a slightly weaker statement like $E \vdash_{\mathbf{CEC}_0} \Pi X.\, s \approx t\ [\varphi]$ whenever there exists a natural number n such that $s\sigma \leftrightarrow^n_{\mathsf{calc}} t\sigma$ for all X-valued substitutions σ such that $\models_M \varphi\sigma$. It might look that induction on the length of $s\sigma \leftrightarrow^*_{\mathsf{calc}} t\sigma$ (or the one on n in the case of the latter) can be applied. However, to apply Theorem 6.31 to each step, we need the form $s_0\sigma \leftrightarrow_{\mathsf{calc}} s_1\sigma \leftrightarrow_{\mathsf{calc}} \cdots \leftrightarrow_{\mathsf{calc}} s_n\sigma$ for each σ which is not generally implied by $s\sigma \leftrightarrow^*_{\mathsf{calc}} t\sigma$ or $s\sigma \leftrightarrow^n_{\mathsf{calc}} t\sigma$, as intermediate terms may vary depending on the substitution σ. Extending Theorem 6.31 to multiple steps remains as our future work.

Remark 6.35. Our final remark deals with the difficulty to extend Theorem 6.31 to a single rule step, i.e. to have a statement like $E \vdash_{\mathbf{CEC}_0} \Pi X.\, s \approx t\ [\varphi]$ whenever $s\sigma \leftrightarrow_{\mathsf{rule},E} t\sigma$ for all X-valued substitutions σ such that $\models_M \varphi\sigma$. For the rule step, there may be multiple choices of positions and multiple choices of CEs to be applied for the step $s\sigma \leftrightarrow_{\mathsf{rule},E} t\sigma$. Thus, we have to divide X-valued substitutions satisfying φ depending on each position p that a CE is applied and each applied CE $\Pi X_i.\, \ell_i \approx r_i\ [\varphi_i] \in E$, and combine the obtained consequences. However, it is in general not guaranteed that such a division of substitutions can be characterized by a constraint. Note that the set of sets of substitutions is in general not countable but the set of constraints is countable. Thus, it may be necessary to assume some assumption on the expressiveness of constraints to obtain the extension for the single rule step. On the other hand, we conjecture that the (full) completeness would hold for CE-theories with a finite underlying model.

6.5 Algebraic Semantics for CE-Validity

In this section, we explore algebraic semantics for CE-validity. In this approach, CE-validity is characterized by validity in models in a class of algebras, which we call *CE-algebras*. We show that this characterization is sound and complete, in the sense that CE-validity can be fully characterized.

CE-Algebras

In this subsection, we introduce a notion of CE-algebras and validity in them. After presenting basic properties of our semantics, we prove its soundness with respect to the CE-validity.

Definition 6.36 (CE-$\langle\Sigma, M\rangle$-algebra)**.** Let $\Sigma = \langle S_{\mathsf{th}}, S_{\mathsf{te}}, \mathcal{F}_{\mathsf{th}}, \mathcal{F}_{\mathsf{te}}\rangle$ be a signature and $M = \langle \mathcal{I}, \mathcal{J}\rangle$ be a model over S_{th} and $\mathcal{F}_{\mathsf{th}}$. A *constrained equational* $\langle\Sigma, M\rangle$*-algebra* (CE-$\langle\Sigma, M\rangle$-algebra, for short) is a pair $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J}\rangle$ where $\mathfrak{I}$ assigns each $\tau \in S$ a non-empty set $\mathfrak{I}(\tau)$, specifying its domain, and $\mathfrak{J}$ assigns each $f \colon \tau_1 \times \cdots \times \tau_n \to \tau_0 \in \mathcal{F}$ an interpretation function $\mathfrak{J}(f) \colon \mathfrak{I}(\tau_1) \times \cdots \times \mathfrak{I}(\tau_n) \to \mathfrak{I}(\tau_0)$ that extends the model $M = \langle \mathcal{I}, \mathcal{J}\rangle$, that is, $\mathfrak{I}(\tau) \supseteq \mathcal{I}(\tau)$ for all $\tau \in S_{\mathsf{th}}$ and $\mathfrak{J}(f) \restriction_{\mathcal{I}(\tau_1)\times\cdots\times\mathcal{I}(\tau_n)} = \mathcal{J}(f)$ for all $f \in \mathcal{F}_{\mathsf{th}}$ (or more generally there exists an injective homomorphism $\iota \colon M \to \mathfrak{M}$).

Let $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J} \rangle$ be a CE-$\langle \Sigma, M \rangle$-algebra. A valuation ρ over $\mathfrak{M}$ is defined similarly to M, but S instead of S_{th}, $\mathcal{F}$ instead of $\mathcal{F}_{\mathsf{th}}$, etc. Similarly, a valuation ρ over $\mathfrak{M}$ satisfies a logical constraint φ, denoted by $\models_{\mathfrak{M}} \varphi$, if $[\![\varphi]\!]_{\rho,\mathfrak{M}} = \mathsf{true}$.

Careful readers may wonder why the interpretation functions for the theory part are not the same but an extension of the underlying model $M = \langle \mathcal{I}, \mathcal{J} \rangle$. Indeed, in the definition of CE-$\langle \Sigma, M \rangle$-algebras $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J} \rangle$ above, we only demand that $\mathfrak{I}(\tau) \supseteq \mathcal{I}(\tau)$ for all $\tau \in S_{\mathsf{th}}$ and not $\mathfrak{I}(\tau) = \mathcal{I}(\tau)$ for all $\tau \in S_{\mathsf{th}}$. In fact, this is required to obtain the completeness result; however, this explanation is postponed until Example 6.53. We continue to present some basic properties of our semantics which are proven in a straightforward manner.

Lemma 6.37. *Let $\mathfrak{T} = \langle M, E \rangle$ be a CE-theory over a signature Σ, and $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J} \rangle$ a CE-$\langle \Sigma, M \rangle$-algebra. Then, the binary relation over terms given by $[\![\cdot]\!]_{\mathfrak{M},\rho} = [\![\cdot]\!]_{\mathfrak{M},\rho}$ for any valuation ρ on $\mathfrak{M}$, is closed under substitutions and contexts.*

Lemma 6.38. *Let $\mathfrak{T} = \langle M, E \rangle$ be a CE- theory over a signature Σ such that $M = \langle \mathcal{I}, \mathcal{J} \rangle$, $\mathfrak{M}$ a CE-$\langle \Sigma, M \rangle$-algebra, and $X \subseteq \mathcal{V}_{\mathsf{th}}$ a set of theory variables and suppose $t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$. Then, for any valuation ρ on $\mathfrak{M}$ such that $\rho(x) \in \mathcal{I}(\tau)$ for all $x^{\tau} \in X$, we have $[\![t]\!]_{\mathfrak{M},\rho} = [\![t]\!]_{M,\rho}$.*

Next, we extend the definition of validity on CE-algebras to CEs, by which we can give a notion of models of CE-theories, and the semantic consequence relation.

Definition 6.39 (model of constrained equational theory)**.** Let $\mathfrak{T} = \langle M, E \rangle$ be a CE-theory over a signature Σ such that $M = \langle \mathcal{I}, \mathcal{J} \rangle$, and $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J} \rangle$ a CE-$\langle \Sigma, M \rangle$-algebra.

1. A CE $\Pi X.\, \ell \approx r\ [\varphi]$ is said to be *valid* in $\mathfrak{M}$, denoted by $\models_{\mathfrak{M}} \Pi X.\, \ell \approx r\ [\varphi]$, if for all valuations ρ over $\mathfrak{M}$ satisfying the constraint φ (i.e. $[\![\varphi]\!]_{\mathfrak{M},\rho} = \mathsf{true}$ holds) and $\rho(x) \in \mathcal{I}(\tau)$ holds for all $x^{\tau} \in X$, we have $[\![\ell]\!]_{\mathfrak{M},\rho} = [\![r]\!]_{\mathfrak{M},\rho}$.

2. A CE-$\langle \Sigma, M \rangle$-algebra $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J} \rangle$ is said to be a *model* of the CE-theory $\mathfrak{T}$ if $\models_{\mathfrak{M}} E$. Here, $\models_{\mathfrak{M}} E$ denotes that $\models_{\mathfrak{M}} \Pi X.\, \ell \approx r\ [\varphi]$ for all $\Pi X.\, \ell \approx r\ [\varphi] \in E$.

3. Let $\Pi X.\, \ell \approx r\ [\varphi]$ be a CE. We write $\mathfrak{T} \models \Pi X.\, \ell \approx r\ [\varphi]$ (or $E \models \Pi X.\, \ell \approx r\ [\varphi]$ if no confusion arises) if $\models_{\mathfrak{M}} \Pi X.\, \ell \approx r\ [\varphi]$ holds for all CE-$\langle \Sigma, M \rangle$-algebras $\mathfrak{M}$ that are models of $\mathfrak{T}$.

We remark that, in item 1., as $\varphi \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, \mathcal{V}_{\mathsf{th}})$, we have $[\![\varphi]\!]_{\mathfrak{M},\rho} = \mathsf{true}$ if and only if $[\![\varphi]\!]_{M,\rho} = \mathsf{true}$ by Lemma 6.38. Based on the preceding lemmas, soundness of our semantics with respect to conversion is not difficult to obtain.

Lemma 6.40 (soundness with respect to conversion)**.** *Let $\mathfrak{T} = \langle M, E \rangle$ be a CE-theory over a signature Σ, and $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J} \rangle$ a CE-$\langle \Sigma, M \rangle$-algebra such that $\models_{\mathfrak{M}} E$. If $s \overset{*}{\leftrightarrow}_E t$ then $[\![s]\!]_{\mathfrak{M},\rho} = [\![t]\!]_{\mathfrak{M},\rho}$ for any valuation ρ on $\mathfrak{M}$.*

Proof (Sketch). It suffices to consider the case $s \leftrightarrow_E t$ with a root step; the claim easily follows from Lemma 6.37. Let $\Sigma = \langle S_{\mathsf{th}}, S_{\mathsf{te}}, \mathcal{F}_{\mathsf{th}}, \mathcal{F}_{\mathsf{te}} \rangle$ and $M = \langle \mathcal{I}, \mathcal{J} \rangle$. Let

$s \leftrightarrow_{\mathsf{calc}} t$. Then, $s, t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}})$, and hence $[\![s]\!]_M = [\![t]\!]_M$. Thus, $[\![s]\!]_{\mathfrak{M}} = [\![t]\!]_{\mathfrak{M}}$ by Lemma 6.38. Otherwise, let $s \leftrightarrow_{\mathsf{rule},E} t$. Then, $s = \ell\sigma$ and $t = r\sigma$ for some $\Pi X.\, \ell \approx r\ [\varphi] \in E$ and an X-valued substitution σ such that $\models_M \varphi\sigma$. We have a valuation $[\![\sigma]\!]_{\mathfrak{M},\rho}$ on $\mathfrak{M}$ by $[\![\sigma]\!]_{\mathfrak{M},\rho}(y) = [\![\sigma(y)]\!]_{\mathfrak{M},\rho}$ for any $y \in \mathcal{V}$. Then, similarly to Lemma 6.1, we have $[\![u\sigma]\!]_{\mathfrak{M},\rho} = [\![u]\!]_{\mathfrak{M},[\![\sigma]\!]_{\mathfrak{M},\rho}}$ for any term $u \in \mathcal{T}(\Sigma, \mathcal{V})$. Furthermore, for $x \in X$, $[\![\sigma]\!]_{\mathfrak{M},\rho}(x) = [\![\sigma(x)]\!]_{\mathfrak{M},\rho} = \sigma(x)$ holds. Hence, by Lemma 6.38, $[\![\varphi]\!]_{\mathfrak{M},[\![\sigma]\!]_{\mathfrak{M},\rho}} = \mathsf{true}$. Thus, $[\![s]\!]_{\mathfrak{M},\rho} = [\![\ell]\!]_{\mathfrak{M},[\![\sigma]\!]_{\mathfrak{M},\rho}} = [\![r]\!]_{\mathfrak{M},[\![\sigma]\!]_{\mathfrak{M},\rho}} = [\![t]\!]_{\mathfrak{M},\rho}$. □

Now we present the soundness of our semantics with respect to the CE-validity.

Theorem 6.41 (soundness with respect to CE-validity). *Let $\mathfrak{T}$ be a CE-theory. If $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$, then $\mathfrak{T} \models \Pi X.\, s \approx t\ [\varphi]$.*

Proof (Sketch). Let $\mathfrak{T} = \langle M, E\rangle$ and $M = \langle \mathcal{I}, \mathcal{J}\rangle$. Suppose $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J}\rangle$ is a CE-$\langle \Sigma, M\rangle$-algebra such that $\models_{\mathfrak{M}} E$. Let ρ be a valuation over $\mathfrak{M}$ satisfying the constraints φ and $\rho(x) \in \mathcal{I}(\tau)$ holds for all $x^\tau \in X$. Now, let $\hat{\rho}$ be a valuation that is obtained from ρ by restricting its domain to X. Then, $\models_M \varphi\hat{\rho}$ by Lemma 6.38, and thus $s\hat{\rho} \overset{*}{\leftrightarrow}_E t\hat{\rho}$ holds. Hence, by Lemma 6.40, $[\![s\hat{\rho}]\!]_{\mathfrak{M},\tau} = [\![t\hat{\rho}]\!]_{\mathfrak{M},\tau}$ holds for any valuation τ. This means that $[\![s]\!]_{\mathfrak{M},\tau'} = [\![t]\!]_{\mathfrak{M},\tau'}$ for any extension τ' of $\hat{\rho}$. In particular, one obtains $[\![s]\!]_{\mathfrak{M},\rho} = [\![t]\!]_{\mathfrak{M},\rho}$. □

The combination of Theorem 6.27 and Theorem 6.41 implies the following corollary.

Corollary 6.42 (soundness of $\mathbf{CEC}_0$ with respect to algebraic semantics). *Let $\mathfrak{T}$ be a CE-theory. If $\mathfrak{T} \vdash_{\mathbf{CEC}_0} \Pi X.\, s \approx t\ [\varphi]$, then $\mathfrak{T} \models \Pi X.\, s \approx t\ [\varphi]$.*

Example 6.43. Consider integer arithmetic for the underlying model M. Take a term signature $\mathcal{F}_{\mathsf{te}} = \{\mathsf{a}\colon \mathsf{Int}\}$. Consider the LCES $E = \{\mathsf{a} \approx \mathsf{0}, \mathsf{a} \approx \mathsf{1}\}$ with $\mathsf{0}, \mathsf{1} \in \mathcal{V}\mathsf{al}$ and $0, 1 \in \mathbb{Z}$, hence $\mathcal{J}(\mathsf{0}) = 0$ and $\mathcal{J}(\mathsf{1}) = 1$. Then, for any valuation ρ on a CE-$\langle \Sigma, M\rangle$-algebra $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J}\rangle$ we have $\rho(\mathsf{0}) = \mathsf{0}$ and $\rho(\mathsf{1}) = \mathsf{1}$. Thus, if $\mathfrak{M}$ is a model of E then it follows that $0 = [\![\mathsf{0}]\!] = [\![\mathsf{a}]\!] = [\![\mathsf{1}]\!] = 1$, which is a contradiction. Therefore, there is no CE-$\langle \Sigma, M\rangle$-algebra $\mathfrak{M}$ which validates E.

This example motivates us to introduce the following definition of consistency for CE-theories.

Definition 6.44 (consistency). A CE-theory is said to be *consistent* if it has a model.

Our definition of consistent CE-theories does not exclude any theory that has only an almost trivial model such that $\mathcal{I}(\tau) = \{\bullet\}$ for all $\tau \in \mathcal{S}_{\mathsf{te}}$.

Completeness with Respect to CE-Validity

In this subsection, we prove the completeness of algebraic semantics with respect to the CE-validity. That is, if a CE is valid in all models of a CE-theory then it is a CE-consequence of the CE-theory. We start by defining congruence relations, quotient algebras and term algebras that suit our formalism, incorporating standard notions for example the first-order equational logic, and then present basic results on them.

Let $\Sigma = \langle S_{\mathsf{th}}, S_{\mathsf{te}}, \mathcal{F}_{\mathsf{th}}, \mathcal{F}_{\mathsf{te}} \rangle$ be a signature, $M = \langle \mathcal{I}, \mathcal{J} \rangle$ a model over S_{th} and $\mathcal{F}_{\mathsf{th}}$, and $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J} \rangle$ a CE-$\langle \Sigma, M \rangle$-algebra. A *congruence relation* on $\mathfrak{M}$ is an S-indexed family of relations $\sim = (\sim^\tau)_{\tau \in S}$ that satisfies all of the following:

1. $\sim^\tau$ is an equivalence relation on $\mathfrak{I}(\tau)$,
2. $\sim^\tau \cap\, \mathcal{I}(\tau)^2$ is the identity relation for $\tau \in S_{\mathsf{th}}$, and
3. for each $f\colon \tau_1 \times \cdots \times \tau_n \to \tau_0 \in \mathcal{F}$, if $a_i \sim^{\tau_i} b_i$ for all $1 \leqslant i \leqslant n$ then $\mathfrak{J}(f)(a_1, \ldots, a_n) \sim^{\tau_0} \mathfrak{J}(f)(b_1, \ldots, b_n)$.

We note here that the difference from the standard notion of congruence relation on algebras is located in item 2. Given a CE-$\langle \Sigma, M \rangle$-algebra $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J} \rangle$ and a congruence relation $\sim$ on it, the quotient CE-$\langle \Sigma, M \rangle$-algebra $\mathfrak{M}/{\sim} = \langle \mathfrak{I}', \mathfrak{J}' \rangle$ is defined as follows: $\mathfrak{I}'(\tau) = \mathfrak{I}(\tau)/{\sim^\tau} = \{[a]_{\sim^\tau} \mid a \in \mathfrak{I}(\tau)\}$ where $[a]_{\sim^\tau}$ is the equivalence class of $a \in \mathfrak{I}(\tau)$, i.e. $[a]_{\sim^\tau} = \{b \in \mathfrak{I}(\tau) \mid a \sim^\tau b\}$, and $\mathfrak{J}'(f)([a_1]_{\sim^{\tau_1}}, \ldots, [a_n]_{\sim^{\tau_n}}) = [\mathfrak{J}(f)(a_1, \ldots, a_n)]_{\sim^{\tau_0}}$. It is easy to see that $\mathfrak{J}'$ is well-defined provided that $\sim$ is a congruence. When no confusion occurs, we omit the superscript τ from $\sim^\tau$.

Lemma 6.45 (quotient algebra)**.** *Let $\mathfrak{M}$ be a CE-$\langle \Sigma, M \rangle$-algebra, and $\sim$ a congruence on it. Then $\mathfrak{M}/{\sim}$ is a CE-$\langle \Sigma, M \rangle$-algebra.*

Next, we define the term algebra. In contrast to the usual construction, for term CE-algebras we need to take care of identification induced by underlying models.

Definition 6.46 (term algebra)**.** Let $\Sigma = \langle S_{\mathsf{th}}, S_{\mathsf{te}}, \mathcal{F}_{\mathsf{th}}, \mathcal{F}_{\mathsf{te}} \rangle$ be a signature, $M = \langle \mathcal{I}, \mathcal{J} \rangle$ a model over S_{th} and $\mathcal{F}_{\mathsf{th}}$, and U a set of variables. The *term algebra generated from U with M* (denoted by $T[M](\Sigma, U)$) is a pair $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J} \rangle$ where

- $\mathfrak{I}(\tau) = \mathcal{T}(\mathcal{F}, U)^\tau/{\sim_{\mathrm{c}}}$, and
- $\mathfrak{J}(f)([t_1]_{\mathrm{c}}, \ldots, [t_n]_{\mathrm{c}}) = [f(t_1, \ldots, t_n)]_{\mathrm{c}}$ for any $f \in \mathcal{F}$.

Here, $\mathcal{F} = \mathcal{F}_{\mathsf{th}} \cup \mathcal{F}_{\mathsf{te}}$, $\sim_{\mathrm{c}} = \leftrightarrow^*_{\mathsf{calc}}$, and $[t]_{\mathrm{c}}$ denotes the $\sim_{\mathrm{c}}$-equivalence class containing a term t. Since $\leftrightarrow^*_{\mathsf{calc}}$ is sort preserving, we regard $\sim_{\mathrm{c}}$ as the sum of the τ-indexed family of relations $\sim^\tau_{\mathrm{c}}$ with $\tau \in S$. Clearly, $\mathfrak{J}(f)$ is well-defined, since $\leftrightarrow^*_{\mathsf{calc}}$ is closed under contexts.

Lemma 6.47. *Let $\Sigma = \langle S_{\mathsf{th}}, S_{\mathsf{te}}, \mathcal{F}_{\mathsf{th}}, \mathcal{F}_{\mathsf{te}} \rangle$ be a signature, M a model over S_{th} and $\mathcal{F}_{\mathsf{th}}$, and U a set of variables. Then, the term algebra $T[M](\Sigma, U)$ is a CE-$\langle \Sigma, M \rangle$-algebra.*

We introduce a syntactic counter part of the notion of consistency of CE-theories for which equivalence of the two notions will be proved only briefly.

Definition 6.48 (consistency with respect to values)**.** A CE-theory $\mathfrak{T} = \langle M, E \rangle$ is said to be *consistent with respect to values* (value-consistent, for short) if for any $u, v \in \mathcal{V}\mathsf{al}^\tau$, $u \xleftrightarrow{*}_E v$ implies $u = v$.

Based on the preparations so far, we now proceed to construct canonical models of CE-theories. The first step is to show that $\overset{*}{\leftrightarrow}_E$ is a congruence relation on the term algebra for each CE-theory $\mathfrak{T} = \langle M, E\rangle$; special attention on $\sim_{\mathsf{c}}$ is required.

Lemma 6.49. *Let $\mathfrak{T} = \langle M, E\rangle$ be a value-consistent CE-theory over a signature Σ, and U a set of variables. For any $[s]_{\mathsf{c}}, [t]_{\mathsf{c}} \in T[M](\Sigma, U)$, let $\sim_E = \{\langle [s]_{\mathsf{c}}, [t]_{\mathsf{c}}\rangle \mid s \overset{*}{\leftrightarrow}_E t\}$. Then, $\sim_E$ is a congruence relation on the term algebra $T[M](\Sigma, U)$.*

Proof (Sketch). Note first that $\sim_E$ is well-defined because one has always $\leftrightarrow^*_{\mathsf{calc}} \subseteq \overset{*}{\leftrightarrow}_E$. Let $\Sigma = \langle \mathcal{S}_{\mathsf{th}}, \mathcal{S}_{\mathsf{te}}, \mathcal{F}_{\mathsf{th}}, \mathcal{F}_{\mathsf{te}}\rangle$, $M = \langle \mathcal{I}, \mathcal{J}\rangle$, and $T[M](\Sigma, U) = \langle \mathfrak{I}, \mathfrak{J}\rangle$. We only present a proof that $\sim^\tau_E \cap \mathcal{I}(\tau)^2$ equals the identity relation for $\tau \in \mathcal{S}_{\mathsf{th}}$ here. Let $\tau \in \mathcal{S}_{\mathsf{th}}$ and suppose $[u]_{\mathsf{c}} \sim^\tau_E [v]_{\mathsf{c}}$ with $u, v \in \mathcal{I}(\tau) \cong \mathcal{V}\mathsf{al}^\tau$. Then, we have $u \overset{*}{\leftrightarrow}_E v$ by the definition of $\sim_E$, and by consistency with respect to values of the theory $\mathfrak{T}$, we obtain $u = v$ as $u, v \in \mathcal{V}\mathsf{al}$. Therefore, $[u]_{\mathsf{c}} = [v]_{\mathsf{c}}$. □

We give a construction of canonical models for each CE-theory $\mathcal{T}$.

Lemma 6.50. *Let $\mathfrak{T} = \langle M, E\rangle$ be a value-consistent CE-theory over a signature Σ. Then, the quotient $\mathcal{T}_E = T[M](\Sigma, \mathcal{V})/{\sim_E}$ of the term algebra is a CE-$\langle \Sigma, M\rangle$-algebra. Furthermore, both of the following hold:*

1. *$\models_{\mathcal{T}_E} \Pi X.\, s \approx t\ [\varphi]$ if and only if $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$, and*
2. *$\models_{\mathcal{T}_E} E$.*

Proof (Sketch). That $\mathcal{T}_E$ is a CE-$\langle \Sigma, M\rangle$-algebra follows from Lemmas 6.45 and 6.49. Let us abbreviate $[[t]_{\mathsf{c}}]_{\sim_E}$ as $[t]_E$. First we claim that $[u\sigma]_E = [\![u]\!]_{\mathcal{T}_E,\rho}$ holds for any term u, for any substitution σ and valuation ρ on $\mathcal{T}_E$ such that $\rho(x) = [\sigma(x)]_E$, using induction on u. *1.* ($\Rightarrow$) Let σ be an X-valued substitution such that $\models_M \varphi\sigma$. Take a valuation ρ on $\mathcal{T}_E$ as $\rho(x) = [\sigma(x)]_E$. Then, $\rho(x) \in \mathcal{I}(\tau)$ for all $x^\tau \in X$ and $\models_M \varphi\rho$. Thus, $[\![s]\!]_{\mathcal{T}_E,\rho} = [\![t]\!]_{\mathcal{T}_E,\rho}$. Hence, $[s\sigma]_E = [t\sigma]_E$ by the claim, and therefore, $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$. ($\Leftarrow$) Let ρ be a valuation over $\mathcal{T}_E$ satisfying the constraints φ and $\rho(x) \in \mathcal{I}(\tau)$ for all $x \in X$. Take a substitution σ in such a way that $\sigma(x) = v_x$ for each $x \in X$, where $v_x \in \mathcal{V}\mathsf{al}^\tau$ such that $[v_x]_E = \rho(x)$. By Lemma 6.38, $[\![\varphi]\!]_{M,\rho} = \mathsf{true}$, and thus, $\models_M \varphi\sigma$ by Lemma 6.1. Hence $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$, and thus $[s\sigma]_E = [t\sigma]_E$. Therefore $[\![s]\!]_{\mathcal{T}_E,\rho} = [\![t]\!]_{\mathcal{T}_E,\rho}$ by the claim. Item *2.* follows from item *1.* □

Before proceeding to the completeness theorem, we connect the two notions related to consistency (Definitions 6.44 and 6.48).

Lemma 6.51. *A CE-theory $\mathfrak{T}$ is consistent if and only if it is consistent with respect to values.*

Proof (Sketch). ($\Rightarrow$) Suppose that $\mathfrak{T} = \langle M, E\rangle$ is a CE-theory over a signature Σ and let $M = \langle \mathcal{I}, \mathcal{J}\rangle$. Let $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J}\rangle$ be a CE-$\langle \Sigma, M\rangle$-algebra such that $\models_{\mathfrak{M}} E$. Suppose $u, v \in \mathcal{V}\mathsf{al}^\tau$ with $u \overset{*}{\leftrightarrow}_E v$. Then $[\![u]\!]_{\mathfrak{M}} = [\![v]\!]_{\mathfrak{M}}$ by Lemma 6.40. Therefore, by $u, v \in \mathcal{V}\mathsf{al}^\tau \cong \mathcal{I}(\tau) \subseteq \mathfrak{I}(\tau)$, we have $u = [\![u]\!]_M = [\![u]\!]_{\mathfrak{M}} = [\![v]\!]_{\mathfrak{M}} = [\![v]\!]_M = v$. ($\Leftarrow$) By Lemma 6.50, $\mathcal{T}_E$ is a model of $\mathfrak{T}$. This witnesses that $\mathfrak{T}$ is consistent. □

We now arrive at the main theorem of this section.

Theorem 6.52 (completeness)**.** *Let $\mathfrak{T} = \langle M, E \rangle$ be a consistent CE-theory. Then, we have $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$ if and only if $E \models \Pi X.\, s \approx t\ [\varphi]$.*

Proof. The *only if* part follows from Theorem 6.41. Thus, it remains to show the *if* part. Suppose contrapositively that $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$ does not hold. Then, by Lemma 6.50 *1*, $\not\models_{\mathcal{T}_E} \Pi X.\, s \approx t\ [\varphi]$. Since $\models_{\mathcal{T}_E} E$, by Lemma 6.50 *2*, this witnesses that there exists a CE-$\langle \Sigma, M \rangle$-algebra $\mathfrak{M}$ such that $\models_{\mathfrak{M}} E$ but not $\models_{\mathfrak{M}} \Pi X.\, s \approx t\ [\varphi]$. This means $E \not\models \Pi X.\, s \approx t\ [\varphi]$. This completes the proof of the *if* part. $\square$

To conclude this section, we explain the postponed question from the beginning of the section on the definition of CE-algebras. The question was on why it is required to include those equipped with underlying extended models—if such models would not be allowed, one does not obtain the completeness result as witnessed by the following example.

Example 6.53. Consider integer arithmetic for the underlying model M. Take a term signature $\mathcal{F}_{\mathsf{te}} = \{\mathsf{f}\colon \mathsf{Ints} \to \mathsf{Bool}, \mathsf{g}\colon \mathsf{Ints} \to \mathsf{Bool}\}$. Consider the LCES $E = \{\mathsf{f}(x) \approx \mathsf{true}\ [x \geqslant \mathsf{0}], \mathsf{f}(x) \approx \mathsf{true}\ [x < \mathsf{0}], \mathsf{g}(x) \approx \mathsf{true}\}$. By orienting the equations in an obvious way, we obtain a complete LCTRS (e.g. [45, 88]). Then as $\mathsf{g}(x){\downarrow} = \mathsf{true} \neq \mathsf{f}(x) = \mathsf{f}(x){\downarrow}$, it turns out that no conversion holds between $\mathsf{g}(x)$ and $\mathsf{f}(x)$. It follows from Theorem 6.20 that $E \not\models_{\mathsf{cec}} \Pi\varnothing.\, \mathsf{g}(x) \approx \mathsf{f}(x)$. Now, from Theorem 6.52, we have $E \not\models \Pi\varnothing.\, \mathsf{g}(x) \approx \mathsf{f}(x)$, i.e. one should find a model that witnesses this invalidity. Indeed, one can take a CE-$\langle \Sigma, M \rangle$-algebra $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J} \rangle$ with $\mathfrak{I}(\mathsf{Ints}) = \mathbb{Z} \cup \{\bullet\}$, where $\bullet \notin \mathbb{Z}$, with the interpretations: $\mathfrak{J}(\mathsf{f})(\bullet) = \mathsf{false}$, $\mathfrak{J}(\mathsf{f})(a) = \mathsf{true}$ for all $a \in \mathbb{Z}$, and $\mathfrak{J}(\mathsf{g})(x) = \mathsf{true}$ for all $x \in \mathbb{Z} \cup \{\bullet\}$. On the other hand, if we would require to take $\mathfrak{I}(\mathsf{Ints}) = \mathbb{Z}$, then we do not get any model that invalidates this CE.

6.6 Related Work

Constrained rewriting began to be popular around 1990, which has been initiated by the motivation to achieve a tractable solution for completion modulo equations (such as AC, ACI, etc.), by separating off the (intractable) equational solving part as constraints. These constraints mainly consist of (dis)equality of built-in equational theories such as $x * y \approx_{AC} y * x$. A constrained completion procedure in such a framework is given in [35]; it is well-known that the specification language Maude also deals with such built-in theories [51]. This line of research was extended to a framework of rewriting with constraints of an arbitrary first-order formula in [35], where various completion methods have been developed for this. However, they, similar to us, mainly considered term algebras as the underlying models, because the main motivation was to deal with a wide range of completion problems by separating off some parts of the equational theory as constraints.

Another well-known type of constraints studied in the context of constrained rewriting is membership constraints of regular tree languages. This type of constraints is motivated by dealing with terms over an (order-)sorted signature and representing an infinite number of terms that obeys regular patterns obtained from divergence of theorem proving procedures. In this line of research, [14,15] give a dedicated completion method for constrained rewrite systems with membership constraints of regular tree languages. Further a method for inductive theorem proving for conditional and constrained systems, which is based on tree grammars with constraints, has been proposed in [11]. We also want to mention [85] as a formalism with more abstract constraints—confluence of term rewrite systems with membership constraints over an arbitrary term set has been considered there.

The work in this era which is in our opinion closest to the LCTRSs formalism is the one given in [16]. This is also motivated by giving a link between (symbolic) equational deduction and constraint solving. Thus, they considered constraints of an arbitrary theory such as linear integer arithmetic, similarly to LCTRSs. Based on the initial model of this framework, they gave an operational semantics of constraint equational logic programming.

The introduction of the LCTRS framework is more recent, and was initiated by the motivation to deal with built-in data structures such as integers, bit-vectors etc. in order to verify programs written by real-world programming languages with the help of SMT-solvers. A detailed comparison to the works in this line of research has been given in [45].

All in all, to the best of our knowledge, there does not exist anything in the literature on algebraic semantics of constrained rewriting and Birkhoff style completeness, as considered in this paper.

6.7 Conclusion

With the goal to establish a semantic formalism of logically constrained rewriting, we have introduced the notions of constrained equations and CE-theories. For this, we have extended the form of these constrained equations by specifying explicitly the variables, which need to be instantiated by values, in order to treat equational properties in a uniform way. Then we have introduced a notion of CE-validity to give a uniform foundation from a semantic point of view for the LCTRS formalism. After establishing basic properties of the introduced validity, we have shown the relation to the conversion of rewriting. Then we presented a sound inference system $\mathbf{CEC}_0$ to prove validity of constrained equations in CE-theories. We have demonstrated its ability to establish validity via some examples. A partial completeness result and a discussion on why only partial completeness is obtained followed. Finally, we devised sound and complete algebraic semantics, which enables one to show invalidity of constrained equations in CE-theories. Furthermore, we have derived an important characterization of CE-theories, namely, consistency of CE-theories, for which the completeness theorem holds. Thus, we have established a basis for CE-theories and their validity by showing its fundamental properties and giving methods for proving and disproving the validity of constrained equations in CE-theories.

The question whether there exists a sound and complete proof system that captures

CE-validity remains open. Part of our future work is the automation of proving validity of constrained equations.

This paper uses the initial formalism of LCTRSs given in [45]. However, there exists a variant which incorporates the interpretation of user-defined function symbols by the term algebra [13]. This variant is incomparable to the initial one. Nevertheless, to investigate the semantic side of LCTRSs, the initial formalism was a reasonable starting point. The adaptation of the current work to the extended formalism is also a part of our future work.

Chapter 7

Automated Analysis of Logically Constrained Rewrite Systems using crest

Publication Details

The content of this chapter is taken from the camera-ready version of the following publication:

[66] Jonas Schöpf and Aart Middeldorp. Automated analysis of logically constrained rewrite systems using crest. In Arie Gurfinkel and Marijn Heule, editors, *Proceedings of the 31st International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 15696 of *Lecture Notes in Computer Science*, pages 124–144, 2025. doi: 10.1007/978-3-031-90643-5_7

Adjustments were made to the presentation of the content to align with the format of this thesis and obvious typos were corrected. We also corrected a typo in the explanation of confluence in the first paragraph of Section 7.3: instead of $s \downarrow t$ as stated in [66], it should read $t \downarrow u$. An appendix including missing proofs can be found in Appendix D. A full version of this paper is available via arXiv [72].

The content of this publication, including the implementation in crest and the the accompanying artifact [71] were developed primarily by me in collaboration with my supervisor Aart Middeldorp.

Abstract

We present crest, a tool for automatically proving (non-)confluence and termination of logically constrained rewrite systems. We compare crest to other tools for logically constrained rewriting. Extensive experiments demonstrate the promise of crest.

7.1 Introduction

Term rewriting is a simple Turing-complete model of computation. Properties like confluence and termination are of key interest and numerous powerful tools have been developed for their analysis. Logically constrained term rewrite systems (LCTRSs for short) constitute a natural extension of term rewrite systems (TRSs) in which rules are equipped with logical constraints that are handled by SMT solvers, thereby avoiding the cumbersome encoding of operations on e.g. integers and bit vectors in term rewriting. LCTRSs, introduced by Kop and Nishida in 2013 [45], are useful for program analysis [13, 23, 40, 89]. They also developed Ctrl [46, 47], a tool for LCTRSs specializing in termination analysis and equivalence testing. Later, techniques for completion [88] and non-termination [58] analysis were added.

In this paper we describe crest, the Constrained REwriting Software Tool. The tool crest was first announced in [64] with support for a small number of confluence techniques. The new version described here includes numerous extensions:

- more advanced confluence techniques (introduced in [68]),
- automated non-confluence and termination analysis,
- support for fixed-sized bit vectors,
- transformation techniques based on splitting critical pairs and merging constrained rewrite rules, to further boost the confluence proving power.

Extensive experiments show the strength of crest. The tool is open-source and available from http://cl-informatik.uibk.ac.at/software/crest.

The remainder of the paper is organized as follows. In the next section we recall important definitions pertaining to LCTRSs. Section 7.3 summarizes the main confluence and termination techniques implemented in crest. Automation details are presented in Section 7.4. The new transformation techniques are described in Section 7.5. In Section 7.6 we present our experiments, before concluding in Section 7.7 with suggestions for future extensions. We conclude this introductory section with mentioning other tools for LCTRSs.

Related Tools. We already mentioned Ctrl[1] which until 2023 was the only tool capable of analyzing confluence and termination of LCTRSs. It supports termination analysis [44], completion techniques [88], rewriting induction for equivalence testing of LCTRSs [23], and basic confluence analysis [47]. Unfortunately, it is neither actively maintained nor very well documented, which is one reason why the development of crest was started. Moreover, a branch[2] of the automated resource analysis tool TcT [8] performs complexity analysis on LCTRSs based on [89]. RMT by Ciobâcă et al. [12, 13] is a newer tool for program analysis based on a variation of LCTRSs.

In the 2024 edition of the Confluence Competition[3] the tool CRaris,[4] developed by Nishida and Kojima, made its appearance. The tool implements weak orthogonality [45] and the Knuth–Bendix criterion for terminating LCTRSs [64]. For termination, it implements the dependency pair framework [44] and the singleton self-looping removal processor [50] for LCTRSs with bit vectors.

[1]http://cl-informatik.uibk.ac.at/software/ctrl/
[2]https://github.com/bytekid/tct-lctrs
[3]https://ari-cops.uibk.ac.at/CoCo/2024/competition/LCTRS/
[4]https://www.trs.css.i.nagoya-u.ac.jp/craris/

Also in 2024 Guo et al. [26, 27] announced Cora, a new open-source tool for termination analysis of logically constrained *simply-typed* term rewrite systems, which serve as a high-order generalization of LCTRSs. It employs static dependency pairs [49] with several base methods, including a variant of the higher-order recursive path order [33].

7.2 Logically Constrained Term Rewriting

Familiarity with the basic notions of term rewriting [9] is assumed. We assume a many-sorted signature $\mathcal{F} = \mathcal{F}_{\mathsf{te}} \cup \mathcal{F}_{\mathsf{th}}$ consisting of term and theory symbols together with a countably infinite set of variables $\mathcal{V}$. For every sort ι in $\mathcal{F}_{\mathsf{th}}$ we have a non-empty set $\mathcal{V}\mathsf{al}_\iota \subseteq \mathcal{F}_{\mathsf{th}}$ of value symbols, such that all $c \in \mathcal{V}\mathsf{al}_\iota$ are constants of sort ι. We demand $\mathcal{F}_{\mathsf{te}} \cap \mathcal{F}_{\mathsf{th}} \subseteq \mathcal{V}\mathsf{al}$ where $\mathcal{V}\mathsf{al} = \bigcup_\iota \mathcal{V}\mathsf{al}_\iota$. The set of terms constructed from function symbols in $\mathcal{F}$ and variables in $\mathcal{V}$ is by $\mathcal{T}(\mathcal{F}, \mathcal{V})$. A term in $\mathcal{T}(\mathcal{F}_{\mathsf{th}}, \mathcal{V})$ is called a *logical* term. Ground logical terms are mapped to values by an interpretation $\mathcal{J}$: $[\![f(t_1, \ldots, t_n)]\!] = f_{\mathcal{J}}([\![t_1]\!], \ldots, [\![t_n]\!])$. Logical terms of sort bool are called *constraints*. A constraint φ is *valid* if $[\![\varphi\gamma]\!] = \top$ for all substitutions γ such that $\gamma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$. Positions are sequences of positive integers to indicate subterms. The root of a term is denoted by the empty string ϵ. For a term s, its subterm at position p is given by $s|_p$. The set of positions in $s \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ is denoted by $\mathcal{P}\mathsf{os}(s)$ whereas $\mathcal{P}\mathsf{os}_{\mathcal{F}}(s)$ is restricted to positions with function symbols in s. We write $\mathcal{V}\mathsf{ar}(s)$ for the set of variables in s. A *constrained rewrite rule* is a triple $\rho\colon \ell \to r\ [\varphi]$ where $\ell, r \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ are terms of the same sort such that $\mathsf{root}(\ell) \in \mathcal{F}_{\mathsf{te}} \setminus \mathcal{F}_{\mathsf{th}}$ and φ is a constraint. We denote the set $\mathcal{V}\mathsf{ar}(\varphi) \cup (\mathcal{V}\mathsf{ar}(r) \setminus \mathcal{V}\mathsf{ar}(\ell))$ of *logical* variables in ρ by $\mathcal{LV}\mathsf{ar}(\rho)$. We write $\mathcal{EV}\mathsf{ar}(\rho)$ for the set $\mathcal{V}\mathsf{ar}(r) \setminus (\mathcal{V}\mathsf{ar}(\ell) \cup \mathcal{V}\mathsf{ar}(\varphi))$ of *extra* variables. A set of constrained rewrite rules is called an LCTRS. A substitution σ *respects* a rule $\rho\colon \ell \to r\ [\varphi]$, denoted by $\sigma \vDash \rho$, if $\mathcal{D}\mathsf{om}(\sigma) \subseteq \mathcal{V}\mathsf{ar}(\rho)$, $\sigma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$, and $\varphi\sigma$ is valid. Moreover, a constraint φ is respected by σ, denoted by $\sigma \vDash \varphi$, if $\sigma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$ and $\varphi\sigma$ is valid. We call $f(x_1, \ldots, x_n) \to y\ [y = f(x_1, \ldots, x_n)]$ with a fresh variable y and $f \in \mathcal{F}_{\mathsf{th}} \setminus \mathcal{V}\mathsf{al}$ a *calculation rule*. The set of all calculation rules induced by the signature $\mathcal{F}_{\mathsf{th}}$ of an LCTRS $\mathcal{R}$ is denoted by $\mathcal{R}_{\mathsf{ca}}$ and we abbreviate $\mathcal{R} \cup \mathcal{R}_{\mathsf{ca}}$ to $\mathcal{R}_{\mathsf{rc}}$. A rewrite step $s \to_{\mathcal{R}} t$ satisfies $s|_p = \ell\sigma$ and $t = s[r\sigma]_p$ for some position p, constrained rewrite rule $\rho\colon \ell \to r\ [\varphi]$ in $\mathcal{R}_{\mathsf{rc}}$, and substitution σ such that $\sigma \vDash \rho$.

A *constrained term* is a pair $s\ [\varphi]$ consisting of a term s and a constraint φ. Two constrained terms $s\ [\varphi]$ and $t\ [\psi]$ are *equivalent*, denoted by $s\ [\varphi] \sim t\ [\psi]$, if for every substitution $\gamma \vDash \varphi$ with $\mathcal{D}\mathsf{om}(\gamma) = \mathcal{V}\mathsf{ar}(\varphi)$ there is some substitution $\delta \vDash \psi$ with $\mathcal{D}\mathsf{om}(\delta) = \mathcal{V}\mathsf{ar}(\psi)$ such that $s\gamma = t\delta$, and vice versa. Let $s\ [\varphi]$ be a constrained term. If $s|_p = \ell\sigma$ for some constrained rewrite rule $\rho\colon \ell \to r\ [\psi] \in \mathcal{R}_{\mathsf{rc}}$, position p, and substitution σ such that $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}\mathsf{ar}(\varphi)$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$, φ is satisfiable and $\varphi \Rightarrow \psi\sigma$ is valid then $s\ [\varphi] \to_{\mathcal{R}} s[r\sigma]_p\ [\varphi]$. The rewrite relation $\xrightarrow{\sim}_{\mathcal{R}}$ on constrained terms is defined as $\sim \cdot \to_{\mathcal{R}} \cdot \sim$ and $s\ [\varphi] \xrightarrow{\sim}{}^{p}_{\mathcal{R}} t\ [\psi]$ indicates that the rewrite step in $\xrightarrow{\sim}_{\mathcal{R}}$ takes place at position p in s. Similarly, we write $s\ [\varphi] \xrightarrow{\sim}_{\geqslant p} t\ [\psi]$ if the position in the rewrite step is below position p. We illustrate some of these concepts by means of a simple example which models the computation of the maximum of two integers.

Example 7.1. Consider the LCTRS $\mathcal{R}$ over the theory Ints with the rules

$$\alpha\colon \mathsf{max}(x,y) \to x\ [x \geqslant y] \qquad\qquad \beta\colon \mathsf{max}(x,y) \to y\ [y \geqslant x]$$

Here x and y are logical variables in both rules. There are no extra variables. The symbol max is the only term symbol. The theory symbols depend on the definition of Ints. As the goal is automation this usually consists of non-linear integer arithmetic as specified in the respective SMT-LIB theory.[5]

By applying the calculation rule $x_1 + x_2 \to y\ [y = x_1 + x_2]$ with substitution $\{x_1 \mapsto \mathsf{3}, x_2 \mapsto \mathsf{2}, y \mapsto \mathsf{5}\}$ followed by rule α we obtain

$$\mathsf{max}(\mathsf{3}+\mathsf{2},\mathsf{3}) \to \mathsf{max}(\mathsf{5},\mathsf{3}) \to \mathsf{5}$$

An example of constrained rewriting is given by

$$\begin{aligned}\mathsf{max}(\mathsf{3},\mathsf{3}+x)\ [x \geqslant \mathsf{0}] &\xrightarrow{\sim} \mathsf{max}(\mathsf{3},z)\ [x \geqslant \mathsf{0} \wedge z = \mathsf{3}+x]\\ &\xrightarrow{\sim} z\ [x \geqslant \mathsf{0} \wedge z = \mathsf{3}+x]\end{aligned}$$

One-step rewriting, i.e., rewriting a term using a single rule, was introduced above. The sufficient criteria for confluence, highlighted in the next section, heavily rely on the notation of parallel ($\mathrel{+\!\!\!\!\!\rightarrow}$) and multi-step ($\mathrel{\circ\!\!\!\rightarrow}$) rewriting following [68, Definition 3] and [64, Definition 8]. The former is capable of applying several rules at parallel positions in a step while the latter additionally allows recursive steps within the used matching substitutions of rules. A rewrite sequence consists of consecutive rewrite steps, independent of which kind. The reflexive and transitive closure of $\to$ is denoted by $\to^*$. Moreover, for arbitrary terms s and t we write $s \leftrightarrow t$ if $s\ (\leftarrow \cup \to)\ t$ and $s \downarrow t$ if there exists a term u such that $s \to^* u \mathrel{{}^*{\leftarrow}} t$.

7.3 Confluence and Termination

Termination and confluence are well-known properties in static program analysis. Both properties are in general undecidable. With respect to (logically constrained) term rewriting, a program is terminating whenever it does not admit an infinite rewrite sequence. Confluence states that $t \downarrow u$ whenever $t \mathrel{{}^*{\leftarrow}} s \to^* u$, for all terms s, t and u. Naively checking the properties is obviously not feasible. In (logically constrained) term rewriting there exist sufficient criteria that guarantee that these properties are satisfied for a given program. In the following we highlight key components of confluence and termination analysis for logically constrained rewrite systems.

The confluence methods implemented in crest are based on (parallel) critical pairs. These are defined as follows. Given a constrained rewrite rule ρ, we write $\mathcal{EC}_\rho$ for $\bigwedge\{x = x \mid x \in \mathcal{EV}\mathsf{ar}(\rho)\}$. An *overlap* of an LCTRS $\mathcal{R}$ is a triple $\langle \rho_1, p, \rho_2\rangle$ with rules $\rho_1\colon \ell_1 \to r_1\ [\varphi_1]$ and $\rho_2\colon \ell_2 \to r_2\ [\varphi_2]$, satisfying the following conditions: (1) ρ_1 and ρ_2 are variable-disjoint variants of rewrite rules in $\mathcal{R}_{\mathsf{rc}}$, (2) $p \in \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell_2)$, (3) ℓ_1 and $\ell_2|_p$

[5]https://smtlib.cs.uiowa.edu/Theories/Ints.smt2

unify with mgu σ such that $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2)$, (4) $\varphi_1\sigma \wedge \varphi_2\sigma$ is satisfiable, and (5) if $p = \epsilon$ then ρ_1 and ρ_2 are not variants, or $\mathcal{V}\mathsf{ar}(r_1) \nsubseteq \mathcal{V}\mathsf{ar}(\ell_1)$. In this case we call $\ell_2\sigma[r_1\sigma]_p \approx r_2\sigma\ [\varphi_1\sigma \wedge \varphi_2\sigma \wedge \psi\sigma]$ a *constrained critical pair* (CCP) obtained from the overlap $\langle \rho_1, p, \rho_2 \rangle$. Here $\psi = \mathcal{EC}_{\rho_1} \wedge \mathcal{EC}_{\rho_2}$. The peak

$$\ell_2\sigma[r_1\sigma]_p\ [\Phi] \leftarrow \ell_2\sigma\ [\Phi] \rightarrow^{\epsilon} r_2\sigma\ [\Phi]$$

with $\Phi = (\varphi_1 \wedge \varphi_2 \wedge \psi)\sigma$, from which the constrained critical pair originates, is called a *constrained critical peak*. The set of all constrained critical pairs of $\mathcal{R}$ is denoted by $\mathsf{CCP}(\mathcal{R})$. A constrained equation $s \approx t\ [\varphi]$ is *trivial* if $s\sigma = t\sigma$ for every substitution σ with $\sigma \vDash \varphi$. The trivial equations from $\mathcal{EC}_\rho$ are used in order to prevent loosing the information which (extra) variables are logical variables in the underlying rules of a CCP.

Example 7.2. Let us extend the LCTRS $\mathcal{R}$ from Example 7.1 with an additional rule modeling the commutativity of max:

$$\mathsf{max}(x,y) \rightarrow x\ [x \geqslant y] \qquad \mathsf{max}(x,y) \rightarrow y\ [y \geqslant x] \qquad \mathsf{max}(x,y) \rightarrow \mathsf{max}(y,x)$$

There are six constrained critical pairs, including the following two:

$$x \approx y\ [x \geqslant y \wedge y \geqslant x] \qquad\qquad x \approx \mathsf{max}(y,x)\ [x \geqslant y]$$

The left one is trivial, the one on the right becomes trivial after one rewrite step: $x \approx \mathsf{max}(y,x)\ [x \geqslant y] \rightarrow x \approx x\ [x \geqslant y]$. The remaining four pairs can be rewritten similarly.

Restricting the way in which constrained critical peaks are rewritten into trivial ones, yields different sufficient conditions for confluence of (left-)linear LCTRSs. We state the conditions below but refer to [45, 64, 68] for precise definitions:

(C1) (weak) orthogonality ([45, Theorem 4]),

(C2) joinable critical pairs for terminating LCTRSs ([64, Corollary 4]).

(C3) strong closedness for linear LCTRSs ([64, Theorem 2]),

(C4) (almost) parallel closedness for left-linear LCTRSs ([64, Theorem 4]),

(C5) (almost) development closedness for left-linear LCTRSs ([68, Corollary 1]).

The final confluence criterion implemented in crest is based on parallel critical pairs. Let $\mathcal{R}$ be an LCTRS, $\rho\colon \ell \rightarrow r\ [\varphi]$ a rule in $\mathcal{R}_{\mathsf{rc}}$, and $P \subseteq \mathcal{P}\mathsf{os}_{\mathcal{F}}(\ell)$ a non-empty set of parallel positions. For every $p \in P$ let $\rho_p\colon \ell_p \rightarrow r_p\ [\varphi_p]$ be a variant of a rule in $\mathcal{R}_{\mathsf{rc}}$. Let $\psi = \mathcal{EC}_\rho \wedge \bigwedge_{p\in P} \mathcal{EC}_{\rho_p}$ and $\Phi = \varphi\sigma \wedge \psi\sigma \wedge \bigwedge_{p \in P} \varphi_p\sigma$. The peak $\ell\sigma[r_p\sigma]_{p\in P}\ [\Phi] \mathrel{+\!\!\!\!\leftarrow\!\!\!\!+} \ell\sigma\ [\Phi] \rightarrow^{\epsilon}_{\mathcal{R}} r\sigma\ [\Phi]$ forms a *constrained parallel critical pair* $\ell\sigma[r_p\sigma]_{p\in P} \approx r\sigma\ [\Phi]$ if the following conditions are satisfied:

1. $\mathcal{V}\mathsf{ar}(\rho_1) \cap \mathcal{V}\mathsf{ar}(\rho_2) = \varnothing$ for different rules ρ_1 and ρ_2 in $\{\rho\} \cup \{\rho_p \mid p \in P\}$,

2. σ is an mgu of $\{\ell_p = \ell|_p \mid p \in P\}$ such that $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho) \cup \bigcup_{p\in P} \mathcal{LV}\mathsf{ar}(\rho_p)$,
3. $\varphi\sigma \wedge \bigwedge_{p\in P} \varphi_p\sigma$ is satisfiable, and
4. if $P = \{\epsilon\}$ then ρ_ϵ is not a variant of ρ or $\mathcal{V}\mathsf{ar}(r) \nsubseteq \mathcal{V}\mathsf{ar}(\ell)$.

A constrained peak forming a constrained parallel critical pair is called a *constrained parallel critical peak*. The set of all constrained parallel critical pairs of $\mathcal{R}$ is denoted by $\mathsf{CPCP}(\mathcal{R})$. The following sufficient condition for confluence is reported in ([68, Corollary 2]):

(C6) parallel closedness of parallel critical pairs for left-linear LCTRSs.

Conditions (C5) and (C6) do not subsume each other. Both generalize conditions (C1) – (C4). All these confluence criteria try to find a specific closing rewrite sequence starting from a constrained (parallel) critical pair—which is seen as a constrained equation—to a trivial constrained equation. For example, parallel closedness in (C4) involves showing that each constrained critical pair $s \approx t\ [\varphi]$ can be rewritten into a trivial constrained equation using a single parallel step $s \approx t\ [\varphi] \mathrel{+\!\!\!\!\!\twoheadrightarrow}_{\geqslant 1} s' \approx t\ [\varphi]$. Note that only the left part (s) is rewritten here and $s' \approx t\ [\varphi]$ is a trivial constrained equation. For (finite) terminating TRSs, confluence is decided by rewriting critical pairs to normal form [37]. For terminating LCTRSs confluence—even for a decidable theory—is undecidable [68], but rewriting constrained critical pairs to normal forms is still of value. This is used in (C2) above. We need however to adapt the notion of normal form for constrained terms.

Example 7.3. The LCTRS $\mathcal{R}$ over the theory Ints with rewrite rules

$$\begin{array}{lll}
\mathsf{f}(x) \to \mathsf{g}(x)\ [x \geqslant 1] & \mathsf{g}(1) \to \mathsf{a} & \mathsf{h}(x) \to \mathsf{a}\ [x \leqslant 1] \\
\mathsf{f}(x) \to \mathsf{h}(x)\ [x \leqslant 2] & \mathsf{g}(x) \to \mathsf{b}\ [x \geqslant 2] & \mathsf{h}(x) \to \mathsf{b}\ [x > 1] \\
 & \mathsf{g}(x) \to \mathsf{c}\ [x < 1] &
\end{array}$$

admits one (modulo symmetry) constrained critical pair:

$$\mathsf{g}(x) \approx \mathsf{h}(x)\ [x \geqslant 1 \wedge x \leqslant 2]$$

None of the rules above are applicable, so this non-trivial constrained critical pair is in normal form with respect to $\to_{\mathcal{R}}$, but it would be wrong to conclude that $\mathcal{R}$ is not confluent; all substitutions σ that satisfy the constraint $x \geqslant 1 \wedge x \leqslant 2$ allow us to rewrite $(\mathsf{g}(x) \approx \mathsf{h}(x))\sigma$ to the trivial equations $\mathsf{a} \approx \mathsf{a}$ or $\mathsf{b} \approx \mathsf{b}$.

Definition 7.4. Given an LCTRS $\mathcal{R}$, a constrained term $s\ [\varphi]$ is in *normal form* if and only if for all substitutions σ with $\sigma \vDash \varphi$ we have $s\sigma \to_{\mathcal{R}} t$ for no term t.

Note that the constrained critical pair in Example 7.3 is not in normal form according to this definition. We present a simple sufficient condition for non-confluence. The easy proof can be found in the appendix of [72].

Lemma 7.5. *An* LCTRS *is* non-confluent *if there exists a constrained critical pair that rewrites to a non-trivial constrained equation in normal form.*

We will resume the analysis of Example 7.3 in Section 7.5. Termination plays an important role in the analysis of LCTRSs. crest implements the following methods reported in the papers by Kop and Nishida [44, 45]:

(T1) dependency graph ([44, Theorems 4 & 5]),

(T2) recursive path order ([45, Theorem 5]),

(T3) value criterion ([44, Theorem 10]),

(T4) reduction pairs ([44, Theorem 12]).

Method (T1) computes the strongly connected components in the dependency graph, and transforms the input LCTRS into so-called DP problems, which can be analyzed independently. It lies at the heart of the dependency pair framework [24] implemented in most termination tools for TRSs. Methods (T2) and (T4) are LCTRS variants of well-known methods for TRSs [7, 18]. Two further methods implemented in Ctrl are ported to crest:

(T5) subterm criterion

(T6) special value criterion

While (T5) is a well-known termination method for DP problems originating from TRSs [30], (T3) and (T6) are specific to LCTRSs. Method (T5) operates on the syntactic structure of dependency pairs and ignores the constraints. In method (T3) dependency pair symbols are also projected to a direct argument but then a strict decrease with respect to the constraint is required. For example, the rule $\mathsf{f}(x) \to \mathsf{f}(x-1)\ [x > 0]$ cannot be handled by (T5), but as $x > 0$ implies the strict decrease $x \succ x - 1$ for a suitable well-founded relation $\succ$, (T3) applies. Method (T6) is an extension of (T3) in which linear combinations of arguments are considered. Methods (T3) and (T6) are adapted to the higher-order LCTRS setting in [26, Sections 4.2 & 4.3].

7.4 Automation

Our tool crest is written in Haskell and the current version consists of roughly 12000 lines of code. Core modules like SMT solving use a fork of the `simple-smt` package[6] and the rewriting modules are inspired by the `term-rewriting` package.[7] In the following we provide some details of the key components.

[6]https://hackage.haskell.org/package/simple-smt
[7]https://hackage.haskell.org/package/term-rewriting

```
(format LCTRS :smtlib 2.6)
(theory Ints)
(fun f (-> Int Int Int))
(fun g (-> Int Int Int))
(fun c (-> Int Int Int))
(fun h (-> Int Int))
(rule (f x y) (h (g y (* 2 2))) :guard (and (<= x y) (= y 2)))
(rule (f x y) (c 4 x) :guard (<= y x))
(rule (g x y) (g y x))
(rule (c x y) (g 4 2) :guard (not (= x y)))
(rule (h x) x)
```

Figure 7.1: ARI file 1528 (without sort annotations and meta information).

Input Format. crest operates on LCTRSs in the new ARI format[8] [3] adopted by the Confluence Competition (CoCo) and also partly by the Termination Competition.[9] Problems in the ARI database are given a unique number which we will use throughout this paper to address specific LCTRSs. An example problem is given in Figure 7.1. The ARI database format requires sort annotations for variables appearing as an argument of a polymorphic predicate. If this sort can be inferred at a different position then this can be ignored for crest. For example, consider the rule $\mathsf{f}(x = y, x) \to z\ [z = x + 1]$ with $\mathsf{f}\colon \mathsf{Bool} \to \mathsf{Int} \to \mathsf{Int}$, $+\colon \mathsf{Int} \to \mathsf{Int} \to \mathsf{Int}$ and $=\colon A \to A \to \mathsf{Bool}$ with a polymorphic sort A. In the ARI database all variables need concrete sort annotation. For crest no sort annotation is necessary as all the sorts of variables can be inferred from the sort of f.

Theory symbols are those that are defined in a specific SMT-LIB theory, however, for fixed-sized bit vectors crest additionally supports function symbols defined in the SMT-LIB logic `QF_BV`.[10] In addition to LCTRSs also plain TRSs and many-sorted TRSs are supported.

Pre-Processing. After parsing its input and assigning already known sorts to function symbols and variables we apply a basic type inference algorithm. Some function symbols in the core theory, which provides basic boolean functions, like "=" have a polymorphic sort. Therefore we need to infer unknown sorts in order to obtain a fully sorted LCTRS. This is required as sort information must be present for the declaration of variables in the SMT solver. During the parsing phase crest parses the respective theory from an internal representation of the SMT-LIB specification. Currently the theory of integers, reals, fixed-sized bit vectors and a combination of integers and reals are supported. Subsequently crest preprocesses the LCTRS by moving values in the left-hand sides of the rewrite rules into the constraints (by applying the transformation described in [64, Definition 13]). Afterwards it merges as many rules as possible following Definition 7.12 in Section 7.5.

[8] https://project-coco.uibk.ac.at/ARI/lctrs.php
[9] https://termination-portal.org/wiki/Termination_Portal
[10] https://smt-lib.org/logics-all.shtml#QF_BV

Rewriting. One of the key components is the rewriting module which provides functionality to perform rewriting on constrained terms. This module computes rewrite sequences of arbitrary length, using single steps, parallel rewrite steps [68, Definition 7] and multisteps [68, Definition 5]. Calculation steps are modeled in an obvious way; whenever we have a term $s[f(s_1,\ldots,s_n)]\ [\varphi]$ with $s_1,\ldots,s_n \in \mathcal{V}\mathsf{al} \cup \mathcal{V}\mathsf{ar}(\varphi)$ and $f \in \mathcal{F}_{\mathsf{th}}$, then we produce $s[x]\ [\varphi \wedge x = f(s_1,\ldots,s_n)]$ for a fresh variable x. In some cases single rule steps need more care because of the lack of equivalence steps in rewrite sequences. For rules with variables that do not occur in the left-hand side, the matching substitution of the left-hand side does not provide an instantiation. However, those variables are logical and need to be instantiated with values. This is achieved by adding the constraint of the rule and its extra variables to the resulting constrained term after it has been confirmed that for those variables an instantiation exists. We illustrate this in the following example.

Example 7.6. Consider the constrained rule $\rho\colon \mathsf{f}(x) \to y\ [x \geqslant 0 \wedge x > y]$, the constrained term $\mathsf{f}(z)\ [z = 2]$ and the matching substitution $\{x \mapsto z\}$ between the left-hand side of ρ and $\mathsf{f}(z)$. The variable y is not part of the matching substitution and thus crest rewrites $\mathsf{f}(z)\ [z = 2]$ to $y\ [z = 2 \wedge z \geqslant 0 \wedge z > y]$. Using the constrained rule $\rho'\colon \mathsf{f}(x) \to y\ [x \geqslant 0]$ from the same constrained term would give $y\ [z = 2 \wedge z \geqslant 0 \wedge y = y]$.

SMT. SMT solving is a key component in the analysis of LCTRSs and SMT solvers are heavily used during the analysis. In order for SMT solving to not form a bottleneck some care is needed. Again, each different analysis method is equipped with its own SMT solver instance started at the beginning of the analysis. Afterwards such an instance runs until the method has finished. In between, it waits for SMT queries, hence we avoid several restarts of this instance. Constraints are modeled as regular terms of sort boolean and can be checked for satisfiability and validity. Each of those checks runs in its own context (using push and pop commands) in order to avoid any interference with previous queries. Currently crest utilizes Z3 [17] as the default SMT solver, as it turned out to be the most reliable during development. Nevertheless, crest provides the (experimental) possibility to use Yices [19] and cvc5 [10].

Confluence. The computation of constrained critical pairs follows the definition and constrained parallel critical pairs are computed in a bottom up fashion by collecting all possible combinations of parallel steps. Then the various methods to conclude confluence are applied on those pairs. If a method fails on a constrained critical pair then, using Definition 7.7, the constrained critical pair is split. The logical constraint used in splitting is taken from a matching rule. The various methods run concurrently in order to prevent starvation of methods because of pending SMT solver queries. The first method which succeeds returns the result and all others, including their SMT solver instances, are terminated. We adopt heuristics to bound the number of rewrite steps in the closing sequences. The method that posed the biggest challenge to automation is the 2-parallel closedness [68, Definition 11] needed for (C6) as we cannot simply use an arbitrary parallel step starting from the right-hand side but need to synthesize a parallel step over a set of parallel positions that adheres to the variable condition present in the definition.

Termination. The choices in the parameters of the subterm criterion (T5) and the recursive path order (T2) are modeled in the SMT encoding. Similarly, for the value criterion (T3) first all possible projections are computed. Then an SMT encoding based on the given rules and theory is constructed and a model of the encoding (if it exists) delivers suitable projections that establish termination. An explicit boolean flag in the SMT encoding determines if a strict or weak decrease is achieved. The special variant with projections to suitable linear combinations (T6) encodes this by attaching unknown constants to the projected arguments and summing them up. Those unknowns are then determined by the SMT solver. The (special) value criterion is currently restricted to the theory of integers as suitable well-founded orderings are required. For the integer theory we use $n \succ m$ if $n > m \wedge n \geqslant 0$ holds.

Method (T4) receives a DP problem as input and tries to transform it into a smaller one by orienting strictly as many dependency pairs as possible. It is parameterized by a list of termination methods which are applied on the DP problem. The first one which succeeds determines the remaining problem to be solved. Before trying to solve the latter, (T1) is used to decompose it into smaller problems.

Features. Via the command-line arguments several features of crest can be accessed. This includes control over the number of threads in the concurrent setup, the overall timeout of the analysis, or if proof output and debug output should be printed. Furthermore, (parallel) critical pairs or the dependency graph approximation of a given LCTRS problem can be computed. The interface also offers a way to transform an LCTRS into a fully sorted LCTRS in the ARI format. In order to alter the default strategy for the analysis, crest offers a very basic strategy language to specify which methods should be used. Detailed information is provided in the usage information of the supplemented artifact.

7.5 Improving the Analysis via Transformations

In this section we present new transformations which are especially useful for confluence analysis. These transformations operate on either rules or constrained critical pairs and split or unify those based on their constraints.

Splitting Constrained Critical Pairs

If a constrained critical pair has more than one instance, which is almost always the case, and they cannot all be rewritten by a single rule, then we are not able to perform any rewrite step. To overcome this problem we propose a simple method to split constrained critical pairs.

Definition 7.7. Given an LCTRS $\mathcal{R}$, a constrained critical pair $\rho\colon s \approx t\ [\varphi] \in \mathsf{CCP}(\mathcal{R})$ and a constraint $\psi \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, \mathcal{V}\mathsf{ar}(\varphi))$, the set $\mathsf{CCP}(\mathcal{R})^{\psi}_{\rho}$ is defined as $(\mathsf{CCP}(\mathcal{R}) \setminus \{\rho\}) \cup \{s \approx t\ [\varphi \wedge \psi], s \approx t\ [\varphi \wedge \neg\psi]\}$.

The following key lemma states that after splitting critical pairs, all confluence methods are still available. The proof is given in the appendix of [72].

Lemma 7.8. *If $t\ {}_{\mathcal{R}}\!\leftarrow s \rightarrow_{\mathcal{R}} u$ then $t \downarrow_{\mathcal{R}} u$ or $t \leftrightarrow_{\mathsf{CCP}^{\psi}_{\rho}(\mathcal{R})} u$.*

We illustrate the lemma on the LCTRS in Example 7.3.

Example 7.9. Consider the CCP $\mathsf{g}(x) \approx \mathsf{h}(x)\ [\varphi]$ with $\varphi\colon x \geqslant 1 \wedge x \leqslant 2$ from Example 7.3. It is neither in normal form nor trivial. Since the subterm $\mathsf{g}(x)$ matches the left-hand side of the rule $\mathsf{g}(x) \to \mathsf{a}\ [x = 1]$ (which is how crest renders the rule $\mathsf{g}(1) \to \mathsf{a}$), and the combined constraint $\varphi \wedge x = 1$ is satisfiable, the CCP is split into

$$\mathsf{g}(x) \approx \mathsf{h}(x)\ [\varphi \wedge x = 1] \qquad \text{and} \qquad \mathsf{g}(x) \approx \mathsf{h}(x)\ [\varphi \wedge x \neq 1]$$

The left one rewrites to the trivial constrained equation $\mathsf{a} \approx \mathsf{a}\ [\varphi \wedge x = 1]$ using the rules $\mathsf{g}(x) \to \mathsf{a}\ [x = 1]$ and $\mathsf{h}(x) \to \mathsf{a}\ [x \leqslant 1]$. The right one is rewritten to $\mathsf{b} \approx \mathsf{b}\ [\varphi \wedge x \neq 1]$ using the rules $\mathsf{g}(x) \to \mathsf{b}\ [x \leqslant 2]$ and $\mathsf{h}(x) \to \mathsf{b}\ [x > 1]$. Hence the LCTRS $\mathcal{R}$ is locally confluent by Lemma 7.8. Using RPO with the precedence $\mathsf{f} > \mathsf{g} > \mathsf{h} > \mathsf{a} > \mathsf{b} > \mathsf{c}$, termination of $\mathcal{R}$ is easily shown and hence $\mathcal{R}$ is confluent.

The following example shows that constrained critical pairs may be split infinitely often before local confluence can be verified.

Example 7.10. Consider the LCTRS $\mathcal{R}$ over the theory Ints consisting of the rules

$$\begin{array}{rlll}
\mathsf{a} \to \mathsf{f}(n) & [n \geqslant 0] & \qquad \mathsf{a} \to \mathsf{g}(n) & [n \geqslant 0] \\
\mathsf{f}(n) \to \mathsf{b} & [n = 0] & \qquad \mathsf{g}(n) \to \mathsf{b} & [n = 0] \\
\mathsf{f}(n) \to \mathsf{f}(m) & [n > 0 \wedge 2 * m = n] & \qquad \mathsf{g}(n) \to \mathsf{g}(m) & [n > 0 \wedge 2 * m = n] \\
\mathsf{f}(n) \to \mathsf{f}(m) & [n > 0 \wedge 2 * m + 1 = n] & \qquad \mathsf{g}(n) \to \mathsf{g}(m) & [n > 0 \wedge 2 * m + 1 = n]
\end{array}$$

This LCTRS has a constrained critical pair $\mathsf{f}(n) \approx \mathsf{g}(m)\ [n \geqslant 0 \wedge m \geqslant 0 \wedge n = n \wedge m = m]$ originating from a. To show confluence of $\mathcal{R}$ we would need to split the pair in order to make rules applicable for joining a specific instance. However, there are infinitely many instances with pairwise different joining sequences.

The next example shows that splitting also helps to prove non-confluence.

Example 7.11. Consider the LCTRS $\mathcal{R}$ in Example 7.3. By changing the constraint of the rule $\mathsf{f}(x) \to \mathsf{g}(x)\ [x \geqslant 1]$ to $[x \geqslant 0]$ we obtain a non-confluent LCTRS. This is shown by splitting the constrained critical pair $\mathsf{g}(x) \approx \mathsf{h}(x)\ [x \geqslant 0 \wedge x \leqslant 2]$, and subsequently showing that $\mathsf{g}(x) \approx \mathsf{h}(x)\ [x < 1 \wedge x \geqslant 0 \wedge x \leqslant 2]$ rewrites to the non-trivial normal form $\mathsf{c} \approx \mathsf{a}\ [x < 1 \wedge x \geqslant 0 \wedge x \leqslant 2]$.

Merging Constrained Rewrite Rules

Next we discuss the merging of constrained rewrite rules. The idea here is that rewrite steps may become possible after merging similar rules.

Definition 7.12. Let $\rho_i \colon \ell_i \to r_i\ [\varphi_i]$ for $i = 1, 2$ be variable-disjoint rewrite rules in an LCTRS $\mathcal{R}$. Suppose there exists a renaming σ such that $\ell_1 = \ell_2\sigma$, $r_1 = r_2\sigma$ and $\mathcal{V}\mathsf{ar}(\varphi_1) = \mathcal{V}\mathsf{ar}(\varphi_2\sigma)$. The LCTRS $\mathcal{R}^{\rho_2}_{\rho_1}$ is defined as

$$(\mathcal{R} \setminus \{\rho_1, \rho_2\}) \cup \{\ell_1 \to r_1\ [\varphi_1 \vee \varphi_2\sigma]\}$$

The easy proof of the following lemma is omitted.

Lemma 7.13. *The relations $\to_{\mathcal{R}}$ and $\to_{\mathcal{R}^{\rho_2}_{\rho_1}}$ coincide.*

Example 7.14. The LCTRS $\mathcal{R}$ over the theory Ints consisting of the rewrite rules

$$\begin{array}{lll} \mathsf{f}(x) \to \mathsf{2} \quad [\mathsf{1} \leqslant x \wedge x \leqslant \mathsf{3}] & \mathsf{g}(x) \to \mathsf{h}(x) & \mathsf{h}(x) \to y\ [x = \mathsf{2} \wedge y = x] \\ \mathsf{f}(x) \to \mathsf{g}(x)\ [\mathsf{2} \leqslant x \wedge x \leqslant \mathsf{4}] & & \mathsf{h}(x) \to y\ [x = \mathsf{3} \wedge y = \mathsf{2}] \end{array}$$

admits the constrained critical pair $\mathsf{2} \approx \mathsf{g}(x)\ [\mathsf{1} \leqslant x \wedge x \leqslant \mathsf{3} \wedge \mathsf{2} \leqslant x \wedge x \leqslant \mathsf{4}]$. After rewriting the subterm $\mathsf{g}(x)$ to $\mathsf{h}(x)$, no further step is possible because the rewrite rules for h are not applicable. However, if we merge the two rules for h into

$$\mathsf{h}(x) \to y\ [(x = \mathsf{2} \wedge y = x) \vee (x = \mathsf{3} \wedge y = \mathsf{2})]$$

we can proceed as $\mathsf{1} \leqslant x \wedge x \leqslant \mathsf{3} \wedge \mathsf{2} \leqslant x \wedge x \leqslant \mathsf{4}$ implies $((x = \mathsf{2} \wedge y = x) \vee (x = \mathsf{3} \wedge y = \mathsf{2}))\sigma$ for $\sigma(y) = \mathsf{2}$. This is exactly how crest operates.

7.6 Experimental Evaluation

In this section we show the progress of crest since the start of its development in early 2023. Initial experiments of an early prototype of crest were reported in [64]. In the following tables the prototype of [64] is denoted by prototype. Since then more criteria for (non-)confluence and termination were added, and parts of the tool infrastructure were completely revised. Detailed results are available from the website of crest and the artifact of the experiments at the Zenodo repository [71].

All experiments were performed using the `benchexec`[11] benchmarking framework which is also used in the StarExec cluster. The benchmark hardware consists of several Intel Xeon E5-2650 v4 CPUs having a base clock speed of 2.20GHz, amounting in total to 64 cores and 128 GB of RAM. As benchmarks we use the problems in the new ARI database[12] in addition to the examples from this paper.

Tool Setup. Each tool receives an ARI benchmark as input and should return either "YES" (property was proved), "NO" (property was disproved) or "MAYBE" (don't know) as the first line of its output. In the tables we represent those with ✓, ✗ and ?, respectively. The fourth category depicted by † denotes that the translation from the ARI format to the input format of the respective tool failed. In order to have a realistic setup, a tool

[11] https://github.com/sosy-lab/benchexec/
[12] https://ari-cops.uibk.ac.at/ARI/

Table 7.1: Confluence analysis of examples.

tool	7.1	7.2	7.3	7.10	7.11	7.14
crest	✓	✓	✓	?	✗	✓
CRaris	?	?	?	?	?	?
Ctrl	✓	?	?	?	?	?
prototype	✓	✓	?	?	?	?

has 4 cores including 8 GB of RAM available for each run. Each tool has 60 seconds to solve a problem before it is killed. Since we have no information about how many threads the other tools use, in the experiments we use CPU time over wall-clock time in order to have a fair comparison.

Our tool crest is split into different binaries depending on the analysis. The most important ones are `crest-cr` for confluence and `crest-sn` for termination. We use those two including an additional flag to allow at most 8 threads for the concurrent setup. The default strategy for confluence uses all methods concurrently and where specific methods are tested we restrict to those using our strategy flag. For the default termination setup we use reduction pairs including dependency graph analysis, recursive path order, (special) value criterion and subterm criterion.

Cora, Ctrl and the prototype of [64] do not accept the ARI format as input. We have developed transformation tools which (try to) transform an ARI benchmark into their respective input. This might not always be possible, hence the transformation tool might fail, which is the reason why we do not distinguish tool (parse) errors from "MAYBE".

Examples. In Table 7.1 we compare the LCTRS confluence tools on the examples in this paper. crest only fails on Example 7.10, which is a confluent LCTRS, but for which no automatable method is known.

Confluence Competition. The last (2024) Confluence Competition[13] hosted the first LCTRS category, with crest and CRaris as participants. The former achieved 67 confluence and 26 non-confluence proofs on a total of 100 selected problems from the ARI database. CRaris, which does not (yet) implement techniques for non-confluence achieved 54 confluence proofs. Currently, crest is the only tool utilizing a criterion for non-confluence of LCTRSs.

Confluence. All confluence criteria implemented in crest, except (C2), require left-linearity. For (C3) right-linearity is also required. Left- and right-linearity is checked only on the non-logical variables. Table 7.2 presents a summary of the confluence methods implemented in crest. The full set of benchmarks consists of the 107 problems in the

[13]https://project-coco.uibk.ac.at/2024/index.php

Table 7.2: Confluence analysis using methods in crest on 107 LCTRSs.

criterion	solved	time (AVG)	time (total)
termination and joinable critical pairs (C2)	50	4.55 s	487 s
orthogonality (C1)	62	0.10 s	11 s
weak orthogonality (C1)	65	0.12 s	13 s
strongly closed critical pairs (C3)	56	1.21 s	129 s
parallel closed critical pairs (C4)	66	0.44 s	47 s
almost parallel closed critical pairs (C4)	70	11.03 s	1180 s
development closed critical pairs (C5)	66	0.39 s	42 s
almost development closed critical pairs (C5)	71	2.06 s	220 s
parallel closed parallel critical pairs (C6)	71	13.93 s	1490 s
all confluence methods (C1)–(C6)	72	8.40 s	899 s
non-confluence (Lemma 7.5)	26	1.96 s	210 s
methods (C1)–(C6) + (Lemma 7.5)	98	1.84 s	197 s
total solved	98	—	—

ARI database. crest can prove in a full run with all methods enabled 72 confluent and 26 non-confluent. Of the remaining 9 problems, 2 result in "MAYBE" and 7 in a timeout. Interesting to observe is that (almost) development closedness is way faster than (almost) parallel closedness, which may be due to the fact that less multi-steps than parallel steps are needed to turn a constrained critical pair into a trivial one. The number 72 is explained by the fact that (C5) and (C6) are incomparable: (C5) succeeds on the problem in Figure 7.1 but fails on the one in Figure 7.2, while the opposite holds for (C6).

In Table 7.3 we compare all confluence tools on the same 107 LCTRS problems. Ctrl supports only weak orthogonality and CRaris in addition the Knuth–Bendix criterion.

```
(format LCTRS :smtlib 2.6)
(theory Ints)
(fun f (-> Int Int))
(fun g (-> Int Int Int))
(fun a Int)
(rule (f a) (g 4 4))
(rule a (g (+ 1 1) (+ 3 1)))
(rule (g x y) (f (g z y)) :guard (= z (- x 2)))
```

Figure 7.2: ARI file 1529 (without sort annotations and meta information).

Table 7.3: Confluence analysis of LCTRS tools on 107 LCTRSs.

tool	✓	✗	?	†	solved	time (AVG)	time (total)
CRaris	58	0	49	—	54 %	0.13 s	14 s
crest	72	26	9	—	92 %	1.84 s	197 s
Ctrl	54	0	49	4	50 %	0.17 s	18 s
prototype	67	0	37	3	63 %	1.14 s	122 s
total solved	72	26	—	—	92 %	—	—

Overall crest is able to solve 92 % of the LCTRS problems in the current ARI database and this percentage is reached even if the timeout is restricted to 10 seconds. The prototype of [64] supports the methods (C1), (C3), (C4) and proves 67 (63 %) confluent within 122 seconds.

Termination. In Table 7.4 we compare the different termination methods in crest. The "dependency graph" method corresponds to (T1) with a check for the absence of SCCs, "recursive path order" corresponds to (T2), "subterm criterion" to (T5), "(special) value criterion" to (T3) ((T6)) and "reduction pairs" to (T4). The methods annotated with (T1) work on DP problems and are applied after an initial dependency graph analysis. The method "reduction pairs no SVC" uses (T2), (T3) and (T5) and "default" includes additionally (T6). The latter constitutes the current default setup in crest.

We continue the evaluation by comparing crest to other termination tools for LCTRSs. For this comparison we use the higher-order tool Cora and Ctrl. The experiments in Table 7.5 show that the tools are comparable in strength on the LCTRS benchmarks in the ARI database, which is not that surprising as the implemented methods are similar. All tools together prove 73 % of the LCTRSs in the ARI database terminating. All those tools fail on the bit vector problem in Figure 7.3 whereas CRaris is able to prove termination (Naoki Nishida, personal communication). A fork of the official

```
(format LCTRS :smtlib 2.6)
(theory FixedSizeBitVectors)
(fun cnt (-> (_ BitVec 4) (_ BitVec 4)))
(fun u1 (-> (_ BitVec 4) (_ BitVec 4) (_ BitVec 4) (_ BitVec 4)))
(rule (cnt x) (u1 x #b0000 #b0000) )
(rule (u1 x i z) (u1 x (bvadd i #b0001) (bvadd z #b0001))
  :guard (bvult i x)))
(rule (u1 x i z) z :guard (not (bvult i x))))
```

Figure 7.3: ARI file 1605 (without sort annotations and meta information).

Table 7.4: Termination analysis using methods in crest on 107 LCTRSs.

method	solved	time (AVG)	time (total)
DP graph (T1)	9	0.08 s	9 s
recursive path order (T2)	27	0.11 s	12 s
recursive path order (T1), (T2)	28	0.11 s	12 s
subterm criterion (T1), (T5)	12	0.12 s	13 s
value criterion (T1), (T3)	34	0.13 s	14 s
special value criterion (T1), (T6)	70	0.12 s	13 s
reduction pairs no SVC (T1)–(T5)	37	0.14 s	15 s
default (T1)–(T6)	74	0.15 s	16 s
total solved	74	—	—

version of Ctrl[14] implements the technique of [58] for non-termination of LCTRSs. Initial experiments reveal that it succeeds to prove non-termination of 8 problems in Table 7.5.

Term Rewrite Systems. In the final experiment we compare crest with the state-of-the-art in automated confluence proving for TRSs. After parsing an input TRS, crest attaches a single sort to all function symbols and variables, and adds an empty constraint to all rules. At this point the TRS can be analyzed as an LCTRS. We compare crest to the latest winner of the TRS category in the Confluence Competition, CSI [55], on the 566 TRS benchmarks in the ARI database. The results can be seen in Table 7.6. Keeping in mind that there is some overhead in the analysis of crest on TRSs as all its methods are geared towards the constrained setting, the 31 % mark is not a bad result. Here it is important to note that CSI has been actively developed over a ten-year period and utilizes many more confluence methods—there is several decades of research on confluence analysis of TRSs while LCTRS confluence analysis is still in its infancy.

[14]https://github.com/bytekid/ctrl

Table 7.5: Termination analysis of LCTRS tools on 107 LCTRSs.

tool	✓	?	†	solved	time (AVG)	time (total)
Cora	71	30	6	66 %	2.47 s	264 s
crest	74	33	—	69 %	0.15 s	16 s
Ctrl	74	29	4	69 %	0.96 s	103 s
total solved	78	—	—	73 %	—	—

Table 7.6: Confluence analysis of crest and CSI on 566 TRSs.

tool	✓	✗	?	solved	time (AVG)	time (total)
crest	100	73	393	31 %	15.87 s	8980 s
CSI	259	192	115	80 %	6.25 s	3540 s
total solved	259	192	—	80 %	—	—

7.7 Conclusion and Future Work

In this paper we presented crest, an open-source tool for automatically proving termination and (non-)confluence of LCTRSs. Detailed experiments were provided to show the power of crest.

In order to further strengthen the (non-)confluence analysis in crest we plan to adapt powerful methods like order-sorted decomposition [22] and redundant rules [54, 74] for plain term rewriting to the constrained setting. Labeling techniques [92] are also on the agenda. The same holds for termination analysis. Natural candidates are matrix interpretations [21] as well as the higher-order methods in [26]. Especially termination problems on real values, like the one in Figure 7.4, should be supported in future. Also non-termination analysis of LCTRSs [58] is of interest. Completion, which is supported in Ctrl [88], is another topic for a future release of crest. In a recent paper [4] the semantics of LCTRSs is investigated. In that context, concepts like checking consistency of constrained theories are relevant, which are worthy to investigate from an automation viewpoint.

Since constrained rewriting is highly complex [68, Section 3], a formalization of the implemented techniques in a proof assistant like Isabelle/HOL is important. The recent advances in the formalization and subsequent certification of advanced confluence techniques [29, 38, 39] for plain rewriting in connection with the transformation in [68, Section 4] make this a realistic goal.

Finally, to improve the user experience we aim at a convenient web interface and a richer command-line strategy.

```
(format LCTRS :smtlib 2.6)
(theory Reals)
(fun sumroot (-> Real Real))
(fun sqrt (-> Real Real))
(rule (sumroot x) 0.0 :guard (>= 0.0 x))
(rule (sumroot x) (+ (sqrt x) (sumroot  (- x 1.0)))
  :guard (not (>= 0.0 x)))
```

Figure 7.4: ARI file 1549 (without meta information).

Code and Availability Statement. The source code and data that support the contributions of this work are freely available in the Zenodo repository “crest - Constrained REwriting Software Tool: Artifact for TACAS 2025” at https://doi.org/10.5281/zenodo.13969852 [71]. The authors confirm that the data supporting the findings of this study are available within the paper and the artifact.

Chapter 8

Conclusion and Future Work

The content of this thesis extends (confluence) research on the formalism of LCTRSs in several directions. This includes the development of various sufficient confluence criteria based on closing (parallel) critical pairs, as well as an attempt for some formal semantics of LCTRSs. In our view, the transformation of an LCTRS into a TRS—including the outlined properties—offers a valuable foundation for future adaptations from TRSs. This thesis serves as an entry point for everybody that aims to further extend LCTRSs.

Many aspects of LCTRSs are still in an early stage of their development, leaving plenty of room for future work. For TRSs, there exist significantly more results on confluence than for LCTRSs. Therefore, adapting techniques such as labeling methods [1, 56, 92], hot-decreasingness and critical pair closing methods [31], as well as other compositional confluence criteria from [74] is definitely of interest. Furthermore, stronger techniques for non-confluence analysis of LCTRSs are still needed, e.g., see [2]. While we have considered constrained parallel critical pairs, there is still no constrained notion of simultaneous critical pairs, despite existing confluence criterion for TRSs based on them [36,60]. Finally, efficiently implementing the reduction method [67] into crest remains an open task.

Termination analysis for LCTRSs clearly needs more advanced techniques. In particular, it would be worthwhile to investigate whether the higher-order methods in [26, 27] can be adapted effectively to the first-order setting. Equally interesting is the integration of established unconstrained termination methods for TRSs into LCTRSs, for example: matrix interpretations [21], arctic interpretations [48], the weighted path order [63, 91], and the tuple interpretation method [90]. For each of these, the key challenge is how to efficiently incorporate the theory part of an LCTRS. Another interesting direction is extending non-termination analysis for LCTRSs: for instance, extending [58] to support loop-preserving transformations or even developing more advanced loop detection methods. This, in turn, may requires more research on narrowing of constrained terms.

Many proofs in the LCTRS literature are quite involved, and verifying their correctness using a proof assistant is highly desirable. This would make it possible to certify proofs—similar to what is done with CeTA [75]—produced by automated tools. This is a first stepping stone towards establishing a certified category for LCTRSs in the CoCo. To support this, it is necessary to extend the ARI database with additional LCTRS confluence problems. Moreover, it remains an open question how rewriting induction [23, 34] can be used for non-terminating LCTRSs. Finally, it would be highly desirable to develop a property preserving transformation from real-world code to LCTRSs, potentially using the LLVM intermediate representation.

Bibliography

[1] Takahito Aoto. Automated confluence proof by decreasing diagrams based on rule-labelling. In Christopher Lynch, editor, *Proceedings of the 21st International Conference on Rewriting Techniques and Applications (RTA)*, volume 6 of *Leibniz International Proceedings in Informatics*, pages 7–16, 2010. doi: 10.4230/LIPIcs.RTA.2010.7.

[2] Takahito Aoto. Disproving confluence of term rewriting systems by interpretation and ordering. In Pascal Fontaine, Christophe Ringeissen, and Renate A. Schmidt, editors, *Proceedings of the 9th International Symposium on Frontiers of Combining Systems (FroCoS)*, volume 8152 of *Lecture Notes in Computer Science*, pages 311–326, 2013. doi: 10.1007/978-3-642-40885-4_22.

[3] Takahito Aoto, Nao Hirokawa, Dohan Kim, Misaki Kojima, Aart Middeldorp, Fabian Mitterwallner, Naoki Nishida, Teppei Saito, Jonas Schöpf, Kiraku Shintani, René Thiemann, and Akihisa Yamada. A new format for rewrite systems. In *Proceedings of the 12th International Workshop on Confluence (IWC)*, pages 32–37, 2023. Available at http://cl-informatik.uibk.ac.at/iwc/iwc2023.pdf.

[4] Takahito Aoto, Naoki Nishida, and Jonas Schöpf. Equational theories and validity for logically constrained term rewriting. In Jakob Rehof, editor, *Proceedings of the 9th International Conference on Formal Structures for Computation and Deduction (FSCD)*, volume 299 of *Leibniz International Proceedings in Informatics*, pages 31:1–31:21, 2024. doi: 10.4230/LIPIcs.FSCD.2024.31.

[5] Takahito Aoto, Naoki Nishida, and Jonas Schöpf. Equational theories and validity for logically constrained term rewriting (full version). *CoRR*, abs/2405.01174, 2024. doi: 10.48550/arXiv.2405.01174.

[6] Takahito Aoto, Junichi Yoshida, and Yoshihito Toyama. Proving confluence of term rewriting systems automatically. In Ralf Treinen, editor, *Proceedings of the 20th International Conference on Rewriting Techniques and Applications (RTA)*, volume 5595 of *Lecture Notes in Computer Science*, pages 93–102, 2009. doi: 10.1007/978-3-642-02348-4_7.

[7] Thomas Arts and Jürgen Giesl. Termination of term rewriting using dependency pairs. *Theoretical Computer Science*, 236(1-2):133–178, 2000. doi: 10.1016/S0304-3975(99)00207-8.

[8] Martin Avanzini, Georg Moser, and Michael Schaper. TcT: Tyrolean Complexity Tool. In Marsha Chechik and Jean-François Raskin, editors, *Proceedings of the 22nd*

International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS), volume 9636 of *Lecture Notes in Computer Science*, pages 407–423, 2016. doi: 10.1007/978-3-662-49674-9_24.

[9] Franz Baader and Tobias Nipkow. *Term Rewriting and All That*. Cambridge University Press, 1998. doi: 10.1017/CBO9781139172752.

[10] Haniel Barbosa, Clark W. Barrett, Martin Brain, Gereon Kremer, Hanna Lachnitt, Makai Mann, Abdalrhman Mohamed, Mudathir Mohamed, Aina Niemetz, Andres Nötzli, Alex Ozdemir, Mathias Preiner, Andrew Reynolds, Ying Sheng, Cesare Tinelli, and Yoni Zohar. cvc5: A versatile and industrial-strength SMT solver. In Dana Fisman and Grigore Rosu, editors, *Proceedings of the 28th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 13243 of *Lecture Notes in Computer Science*, pages 415–442, 2022. doi: 10.1007/978-3-030-99524-9_24.

[11] Adel Bouhoula and Florent Jacquemard. Sufficient completeness verification for conditional and constrained TRS. *Journal of Applied Logic*, 10(1):127–143, 2012. doi: 10.1016/J.JAL.2011.09.001.

[12] Stefan Ciobâcă, Dorel Lucanu, and Andrei Sebastian Buruiană. Operationally-based program equivalence proofs using LCTRSs. *Journal of Logical and Algebraic Methods in Programming*, 135:100894, 2023. doi: 10.1016/j.jlamp.2023.100894.

[13] Stefan Ciobâcă and Dorel Lucanu. A coinductive approach to proving reachability properties in logically constrained term rewriting systems. In Didier Galmiche, Stephan Schulz, and Roberto Sebastiani, editors, *Proceedings of the 9th International Joint Conference on Automated Reasoning (IJCAR)*, volume 10900 of *Lecture Notes in Artificial Intelligence*, pages 295–311, 2018. doi: 10.1007/978-3-319-94205-6_20.

[14] Hubert Comon. Completion of rewrite systems with membership constraints. Part I: deduction rules. *Journal of Symbolic Computation*, 25(4):397–419, 1998. doi: 10.1006/JSCO.1997.0185.

[15] Hubert Comon. Completion of rewrite systems with membership constraints. Part II: constraint solving. *Journal of Symbolic Computation*, 25(4):421–453, 1998. doi: 10.1006/JSCO.1997.0186.

[16] John Darlington and Yike Guo. Constrained equational deduction. In Stéphane Kaplan and Mitsuhiro Okada, editors, *Proceedings of the 2nd International Workshop on Conditional and Typed Rewriting Systems (CTRS)*, volume 516 of *Lecture Notes in Computer Science*, pages 424–435, 1991. doi: 10.1007/3-540-54317-1_111.

[17] Leonardo de Moura and Nikolaj Bjørner. Z3: An efficient SMT solver. In C. R. Ramakrishnan and Jakob Rehof, editors, *Proceedings of the 14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 4963 of *Lecture Notes in Computer Science*, pages 337–340, 2008. doi: 10.1007/978-3-540-78800-3_24.

[18] Nachum Dershowitz. Orderings for term-rewriting systems. *Theoretical Computer Science*, 17(3):279–301, 1982. doi: 10.1016/0304-3975(82)90026-3.

[19] Bruno Dutertre. Yices 2.2. In Armin Biere and Roderick Bloem, editors, *Proceedings of the 26th International Conference on Computer Aided Verification (CAV)*, volume 8559 of *Lecture Notes in Computer Science*, pages 737–744, 2014. doi: 10.1007/978-3-319-08867-9_49.

[20] Heinz-Dieter Ebbinghaus, Jörg Flum, and Wolfgang Thomas. *Mathematical logic*. Springer, 2021. doi: 10.1007/978-3-030-73839-6.

[21] Jörg Endrullis, Johannes Waldmann, and Hans Zantema. Matrix interpretations for proving termination of term rewriting. *Journal of Automated Reasoning*, 40(2-3):195–220, 2008. doi: 10.1007/s10817-007-9087-9.

[22] Bertram Felgenhauer, Aart Middeldorp, Harald Zankl, and Vincent van Oostrom. Layer systems for proving confluence. *ACM Transactions on Computational Logic*, 16(2):14:1–14:32, 2015. doi: 10.1145/2710017.

[23] Carsten Fuhs, Cynthia Kop, and Naoki Nishida. Verifying procedural programs via constrained rewriting induction. *ACM Transactions on Computational Logic*, 18(2):14:1–14:50, 2017. doi: 10.1145/3060143.

[24] Jürgen Giesl, René Thiemann, and Peter Schneider-Kamp. The dependency pair framework: Combining techniques for automated termination proofs. In Franz Baader and Andrei Voronkov, editors, *Proceedings of the 11th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR)*, volume 3452 of *Lecture Notes in Computer Science*, pages 301–331, 2005. doi: 10.1007/978-3-540-32275-7_21.

[25] Bernhard Gramlich. Confluence without termination via parallel critical pairs. In Hélène Kirchner, editor, *Proceedings of the 21st International Colloquium on Trees in Algebra and Programming (CAAP)*, volume 1059 of *Lecture Notes in Computer Science*, pages 211–225, 1996. doi: 10.1007/3-540-61064-2_39.

[26] Liye Guo, Kasper Hagens, Cynthia Kop, and Deivid Vale. Higher-order constrained dependency pairs for (universal) computability. In Rastislav Královič and Antonín Kučera, editors, *Proceedings of the 49th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 306 of *Leibniz International Proceedings in Informatics*, pages 57:1–57:15, 2024. doi: 10.4230/LIPIcs.MFCS.2024.57.

[27] Liye Guo and Cynthia Kop. Higher-order LCTRSs and their termination. In Stephanie Weirich, editor, *Proceedings of the 33rd European Symposium on Programming (ESOP)*, volume 14577 of *Lecture Notes in Computer Science*, pages 331–357, 2024. doi: 10.1007/978-3-031-57267-8_13.

[28] Kasper Hagens and Cynthia Kop. Rewriting induction for higher-order constrained term rewriting systems. In Juliana Bowles and Harald Søndergaard, editors, *Proceedings of the 34th International Symposium on Logic-Based Program Synthesis and Transformation (LOPSTR)*, volume 14919 of *Lecture Notes in Computer Science*, pages 202–209, 2024. doi: 10.1007/978-3-031-71294-4_12.

[29] Nao Hirokawa, Dohan Kim, Kiraku Shintani, and René Thiemann. Certification of confluence- and commutation-proofs via parallel critical pairs. In Sandrine Blazy, Brigitte Pientka, Amin Timany, and Dmitriy Traytel, editors, *Proceedings of the 13th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP)*, pages 147–161, 2024. doi: 10.1145/3636501.3636949.

[30] Nao Hirokawa and Aart Middeldorp. Dependency pairs revisited. In Vincent van Oostrom, editor, *Proceedings of the 15th International Conference on Rewriting Techniques and Applications (RTA)*, volume 3091 of *Lecture Notes in Computer Science*, pages 249–268, 2004. doi: 10.1007/978-3-540-25979-4_18.

[31] Nao Hirokawa, Julian Nagele, Vincent van Oostrom, and Michio Oyamaguchi. Confluence by critical pair analysis revisited. In Pascal Fontaine, editor, *Proceedings of the 27th International Conference on Automated Deduction (CADE)*, volume 11716 of *Lecture Notes in Artificial Intelligence*, pages 319–336, 2019. doi: 10.1007/978-3-030-29436-6_19.

[32] Gérard Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of the ACM*, 27(4):797–821, 1980. doi: 10.1145/322217.322230.

[33] Jean-Pierre Jouannaud and Albert Rubio. The higher-order recursive path ordering. In *Proceedings of the 14th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 402–411, 1999. doi: 10.1109/LICS.1999.782635.

[34] Kentaro Kikuchi, Takahito Aoto, and Isao Sasano. Inductive theorem proving in non-terminating rewriting systems and its application to program transformation. In Ekaterina Komendantskaya, editor, *Proceedings of the 21st International Symposium on Principles and Practice of Declarative Programming (PPDP)*, pages 13:1–13:14, 2019. doi: 10.1145/3354166.3354178.

[35] Claude Kirchner and Hélène Kirchner. Constrained equational reasoning. In Gastón H Gonnet, editor, *Proceedings of the 25th ACM-SIGSAM International Symposium on Symbolic and Algebraic Computation (ISSAC)*, pages 382–389, 1989. doi: 10.1145/74540.74585.

[36] Christina Kirk and Aart Middeldorp. Formalizing simultaneous critical pairs for confluence of left-linear rewrite systems. In Sandrine Blazy, Kathrin Stark, Nicolas Tabareau, and Amin Timany, editors, *Proceedings of the 14th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP)*, pages 156–170, 2025. doi: 10.1145/3703595.3705881.

[37] Donald E. Knuth and Peter B. Bendix. Simple word problems in universal algebras. In John Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, 1970. doi: 10.1016/B978-0-08-012975-4.50028-X.

[38] Christina Kohl and Aart Middeldorp. A formalization of the development closedness criterion for left-linear term rewrite systems. In Robbert Krebbers, Brigitte Pientka, Dmitriy Traytel, and Steve Zdancewic, editors, *Proceedings of the 12th ACM SIGPLAN International Conference on Certified Programs and Proofs (CPP)*, pages 197–210, 2023. doi: 10.1145/3573105.3575667.

[39] Christina Kohl and Aart Middeldorp. Formalizing almost development closed critical pairs. In Adam Naumowicz and René Thiemann, editors, *Proceedings of the 14th International Conference on Interactive Theorem Proving (ITP)*, volume 268 of *Leibniz International Proceedings in Informatics*, pages 38:1–38:8, 2023. doi: 10.4230/LIPIcs.ITP.2023.38.

[40] Misaki Kojima and Naoki Nishida. From starvation freedom to all-path reachability problems in constrained rewriting. In Michael Hanus and Daniela Inclezan, editors, *Proceedings of the 25th International Symposium on Practical Aspects of Declarative Languages (PADL)*, volume 13880 of *Lecture Notes in Computer Science*, pages 161–179, 2023. doi: 10.1007/978-3-031-24841-2_11.

[41] Misaki Kojima and Naoki Nishida. Reducing non-occurrence of specified runtime errors to all-path reachability problems of constrained rewriting. *Journal of Logical and Algebraic Methods in Programming*, 135:1–19, 2023. doi: 10.1016/j.jlamp.2023.100903.

[42] Cynthia Kop. Termination of LCTRSs. In *Proceedings of the 13th International Workshop on Termination (WST)*, pages 59–63, 2013.

[43] Cynthia Kop. Quasi-reductivity of logically constrained term rewriting systems. *CoRR*, abs/1702.02397, 2016. Proceedings of WST 2013 (doi: 10.48550/ARXIV.1702.02397).

[44] Cynthia Kop. Termination of LCTRSs. *CoRR*, abs/1601.03206, 2016. Proceedings of WST 2013 (doi: 10.48550/ARXIV.1601.03206).

[45] Cynthia Kop and Naoki Nishida. Term rewriting with logical constraints. In Pascal Fontaine, Christophe Ringeissen, and Renate A. Schmidt, editors, *Proceedings of the 9th International Symposium on Frontiers of Combining Systems (FroCoS)*, volume 8152 of *Lecture Notes in Artificial Intelligence*, pages 343–358, 2013. doi: 10.1007/978-3-642-40885-4_24.

[46] Cynthia Kop and Naoki Nishida. Automatic constrained rewriting induction towards verifying procedural programs. In Jacques Garrigue, editor, *Proceedings of the 12th Asian Symposium on Programming Languages and Systems (APLAS)*, volume 8858 of *Lecture Notes in Computer Science*, pages 334–353, 2014. doi: 10.1007/978-3-319-12736-1_18.

[47] Cynthia Kop and Naoki Nishida. Constrained Term Rewriting tooL. In Martin Davis, Ansgar Fehnker, Annabelle McIver, and Andrei Voronkov, editors, *Proceedings of the 20th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR)*, volume 9450 of *Lecture Notes in Computer Science*, pages 549–557, 2015. doi: 10.1007/978-3-662-48899-7_38.

[48] Adam Koprowski and Johannes Waldmann. Arctic termination ... below zero. In Andrei Voronkov, editor, *Proceedings of the 19th International Conference on Rewriting Techniques and Applications (RTA)*, volume 5117 of *Lecture Notes in Computer Science*, pages 202–216, 2008. doi: 10.1007/978-3-540-70590-1_14.

[49] Keiichirou Kusakari and Masahiko Sakai. Enhancing dependency pair method using strong computability in simply-typed term rewriting. *Applicable Algebra in Engineering, Communication and Computing*, 18(5):407–431, 2007. doi: 10.1007/S00200-007-0046-9.

[50] Ayuka Matsumi, Naoki Nishida, Misaki Kojima, and Donghoon Shin. On singleton self-loop removal for termination of LCTRSs with bit-vector arithmetic. *CoRR*, abs/2307.14094, 2023. doi: 10.48550/arXiv.2307.14094.

[51] José Meseguer. Twenty years of rewriting logic. *Journal of Logic and Algebraic Programming*, 81(7-8):721–781, 2012. doi: 10.1016/J.JLAP.2012.06.003.

[52] Florian Meßner, Julian Parsert, Jonas Schöpf, and Christian Sternagel. A formally verified solver for homogeneous linear diophantine equations. In Jeremy Avigad and Assia Mahboubi, editors, *Proceedings of the 9th International Conference on Interactive Theorem Proving (ITP)*, volume 10895 of *Lecture Notes in Computer Science*, pages 441–458, 2018. doi: 10.1007/978-3-319-94821-8_26.

[53] Fabian Mitterwallner, Jonas Schöpf, and Aart Middeldorp. Reducing confluence of LCTRSs to confluence of TRSs. In *Proceedings of the 12th International Workshop on Confluence (IWC)*, pages 3–8, 2023. Available at http://cl-informatik.uibk.ac.at/iwc/iwc2023.pdf.

[54] Julian Nagele, Bertram Felgenhauer, and Aart Middeldorp. Improving automatic confluence analysis of rewrite systems by redundant rules. In Maribel Fernández, editor, *Proceedings of the 26th International Conference on Rewriting Techniques and Applications (RTA)*, volume 36 of *Leibniz International Proceedings in Informatics*, pages 257–268, 2015. doi: 10.4230/LIPIcs.RTA.2015.257.

[55] Julian Nagele, Bertram Felgenhauer, and Aart Middeldorp. CSI: New evidence — a progress report. In Leonardo de Moura, editor, *Proceedings of the 26th International Conference on Automated Deduction (CADE)*, volume 10395 of *Lecture Notes in Artificial Intelligence*, pages 385–397, 2017. doi: 10.1007/978-3-319-63046-5_24.

[56] Julian Nagele, Bertram Felgenhauer, and Harald Zankl. Certifying confluence proofs via relative termination and rule labeling. *Logical Methods in Computer Science*, 13(2), 2017. doi: 10.23638/LMCS-13(2:4)2017.

[57] Julian Nagele and Aart Middeldorp. Certification of classical confluence results for left-linear term rewrite systems. In Jasmin Christian Blanchette and Stephan Merz, editors, *Proceedings of the 7th International Conference on Interactive Theorem Proving (ITP)*, volume 9807 of *Lecture Notes in Computer Science*, pages 290–306, 2016. doi: 10.1007/978-3-319-43144-4_18.

[58] Naoki Nishida and Sarah Winkler. Loop detection by logically constrained term rewriting. In Ruzica Piskac and Philipp Rümmer, editors, *Proceedings of the 10th Working Conference on Verified Software: Theories, Tools, and Experiments (VSSTE)*, volume 11294 of *Lecture Notes in Computer Science*, pages 309–321, 2018. doi: 10.1007/978-3-030-03592-1_18.

[59] Enno Ohlebusch. *Advanced Topics in Term Rewriting*. Springer, 2002. doi: 10.1007/978-1-4757-3661-8.

[60] Satoshi Okui. Simultaneous critical pairs and Church-Rosser property. In Tobias Nipkow, editor, *Proceedings of the 9th International Conference on Rewriting Techniques and Applications (RTA)*, volume 1379 of *Lecture Notes in Computer Science*, pages 2–16, 1998. doi: 10.1007/BFb0052357.

[61] Cecilia Pasquini and Rainer Böhme. Trembling triggers: exploring the sensitivity of backdoors in DNN-based face recognition. *EURASIP Journal on Information Security*, 12(1), 2020. doi: 10.1186/s13635-020-00104-z.

[62] Emil L. Post. A variant of a recursively unsolvable problem. *Bulletin of the American Mathematical Society*, 52:264–268, 1946. doi: 10.1090/S0002-9904-1946-08555-9.

[63] Teppei Saito and Nao Hirokawa. Weighted path orders are semantic path orders. In Uli Sattler and Martin Suda, editors, *Proceedings of the 14th International Symposium on Frontiers of Combining Systems (FroCoS)*, volume 14279 of *Lecture Notes in Artificial Intelligence*, pages 63–80, 2023. doi: 10.1007/978-3-031-43369-6_4.

[64] Jonas Schöpf and Aart Middeldorp. Confluence criteria for logically constrained rewrite systems. In Brigitte Pientka and Cesare Tinelli, editors, *Proceedings of the 29th International Conference on Automated Deduction (CADE)*, volume 14132 of *Lecture Notes in Artificial Intelligence*, pages 474–490, 2023. doi: 10.1007/978-3-031-38499-8_27.

[65] Jonas Schöpf and Aart Middeldorp. Confluence criteria for logically constrained rewrite systems (full version). *CoRR*, abs/2309.12112, 2023. doi: 10.48550/arXiv.2309.12112.

[66] Jonas Schöpf and Aart Middeldorp. Automated analysis of logically constrained rewrite systems using crest. In Arie Gurfinkel and Marijn Heule, editors, *Proceedings of the 31st International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, volume 15696 of *Lecture Notes in Computer Science*, pages 124–144, 2025. doi: 10.1007/978-3-031-90643-5_7.

[67] Jonas Schöpf and Aart Middeldorp. Improving confluence analysis for LCTRSs. In *Proceedings of the 14th International Workshop on Confluence (IWC)*, 2025. Available at https://iwc2025.github.io/.

[68] Jonas Schöpf, Fabian Mitterwallner, and Aart Middeldorp. Confluence of logically constrained rewrite systems revisited. In Christoph Benzmüller, Marijn J.H. Heule, and Renate A. Schmidt, editors, *Proceedings of the 12th International Joint Conference on Automated Reasoning (IJCAR)*, volume 14740 of *Lecture Notes in Artificial Intelligence*, pages 298–316, 2024. doi: 10.1007/978-3-031-63501-4_16.

[69] Jonas Schöpf, Fabian Mitterwallner, and Aart Middeldorp. Confluence of logically constrained rewrite systems revisited. *CoRR*, abs/2402.13552, 2024. doi: 10.48550/ARXIV.2402.13552.

[70] Jonas Schöpf and Christian Sternagel. $\mathsf{T_TT_2}$ with termination templates for teaching. *CoRR*, abs/1806.050401702.02397, 2018. Proceedings of WST 2018 (doi: 10.48550/ARXIV.1806.05040).

[71] Jonas Schöpf and Aart Middeldorp. crest - Constrained REwriting Software Tool: Artifact for TACAS 2025, October 2024. doi: 10.5281/zenodo.13969852.

[72] Jonas Schöpf and Aart Middeldorp. Automated analysis of logically constrained rewrite systems using crest. *CoRR*, abs/2501.05240, 2025. doi: 10.48550/arXiv.2501.05240.

[73] Kiraku Shintani and Nao Hirokawa. Compositional confluence criteria. In Amy P. Felty, editor, *Proceedings of the 7th International Conference on Formal Structures for Computation and Deduction (FSCD)*, volume 228 of *Leibniz International Proceedings in Informatics*, pages 28:1–28:19, 2022. doi: 10.4230/LIPICS.FSCD.2022.28.

[74] Kiraku Shintani and Nao Hirokawa. Compositional confluence criteria. *Logical Methods in Computer Science*, 20(1), 2024. doi: 10.46298/lmcs-20(1:6)2024.

[75] Christian Sternagel and René Thiemann. A framework for developing stand-alone certifiers. *Electronic Notes in Theoretical Computer Science*, 312:51–67, 2015. doi: 10.1016/j.entcs.2015.04.004.

[76] Kanta Takahata, Jonas Schöpf, Naoki Nishida, and Takahito Aoto. Characterizing equivalence of logically constrained terms via existentially constrained terms. In Santiago Escobar and Laura Titolo, editors, *Proceedings of the 35th International Symposium on Logic-Based Program Synthesis and Transformation (LOPSTR)*, Lecture Notes in Computer Science, 2025. to appear.

[77] Kanta Takahata, Jonas Schöpf, Naoki Nishida, and Takahito Aoto. Characterizing equivalence of logically constrained terms via existentially constrained terms (full version). *CoRR*, abs/2505.21986, 2025. doi: 10.48550/arXiv.2505.21986.

[78] Kanta Takahata, Jonas Schöpf, Naoki Nishida, and Takahito Aoto. Most general constrained rewriting in logically constrained term rewriting systems. In Soichiro Hidaka and Takeshi Tsukada, editors, *Proceedings of the 27th JSSST Workshop on Programming and Programming Languages (PPL)*, 2025. In Japanese. Gamagori, Japan, March 5–7, 2025.

[79] Kanta Takahata, Jonas Schöpf, Naoki Nishida, and Takahito Aoto. Recovering commutation of logically constrained rewriting and equivalence transformations. In Małgorzata Biernacka and Carlos Olarte, editors, *Proceedings of the 27th International Symposium on Principles and Practice of Declarative Programming (PPDP)*. Association of the Computing Machinery, 2025. to appear.

[80] Kanta Takahata, Jonas Schöpf, Naoki Nishida, and Takahito Aoto. Recovering commutation of logically constrained rewriting and equivalence transformations (full version). *CoRR*, abs/2507.09326, 2025. doi: 10.48550/arXiv.2507.09326.

[81] Terese, editor. *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2003.

[82] René Thiemann, Jonas Schöpf, Christian Sternagel, and Akihisa Yamada. Certifying the weighted path order. In Zena M. Ariola, editor, *Proceedings of the 5th International Conference on Formal Structures for Computation and Deduction (FSCD)*, volume 167 of *Leibniz International Proceedings in Informatics*, pages 4:1–4:20, 2020. doi: 10.4230/LIPIcs.FSCD.2020.4.

[83] Yoshihito Toyama. On the Church–Rosser property of term rewriting systems. NTT ECL Technical Report 17672, NTT ECL, 1981. In Japanese.

[84] Yoshihito Toyama. Commutativity of term rewriting systems. In *Programming of Future Generation Computers II*, pages 393–407. North-Holland, 1988.

[85] Yoshihito Toyama. Membership conditional term rewriting systems. *IEICE Transactions*, E72(11):1224–1229, 1989.

[86] Dirk van Dalen. *Logic and Structure*. Springer, 2004. doi: 10.1007/978-3-540-85108-0.

[87] Vincent van Oostrom. Developing developments. *Theoretical Computer Science*, 175(1):159–181, 1997. doi: 10.1016/S0304-3975(96)00173-9.

[88] Sarah Winkler and Aart Middeldorp. Completion for logically constrained rewriting. In Hélène Kirchner, editor, *Proceedings of the 3rd International Conference on Formal Structures for Computation and Deduction (FSCD)*, volume 108 of *Leibniz International Proceedings in Informatics*, pages 30:1–30:18, 2018. doi: 10.4230/LIPIcs.FSCD.2018.30.

[89] Sarah Winkler and Georg Moser. Runtime complexity analysis of logically constrained rewriting. In Maribel Fernández, editor, *Proceedings of the 30th International Symposium on Logic-Based Program Synthesis and Transformation (LOPSTR)*,

volume 12561 of *Lecture Notes in Computer Science*, pages 37–55, 2021. doi: 10.1007/978-3-030-68446-4_2.

[90] Akihisa Yamada. Tuple interpretations for termination of term rewriting. *Journal of Automated Reasoning*, 66(4):667–688, 2022. doi: 10.1007/s10817-022-09640-4.

[91] Akihisa Yamada, Keiichirou Kusakari, and Toshiki Sakabe. A unified ordering for termination proving. *Science of Computer Programming*, 111:110–134, 2015. doi: 10.1016/j.scico.2014.07.009.

[92] Harald Zankl, Bertram Felgenhauer, and Aart Middeldorp. Labelings for decreasing diagrams. *Journal of Automated Reasoning*, 54(2):101–133, 2015. doi: 10.1007/s10817-014-9316-y.

Appendices

A Appendix to Chapter 4

In this appendix, we give the missing proofs of the paper [64] which are only available in the full version on arXiv [65].

Proof of Theorems 4.25 and 4.28.

Since Theorem 4.25 is a special case of Theorem 4.28, we prove the latter. Our proof is based on the proof that has been formalized in Isabelle/HOL for plain TRSs reported in [57]. The following definitions and results from [57] are needed in the proof.

Recall that a context is a term with exactly one hole ($\Box$). A multihole context is a term with an arbitrary number of holes. We write $C[t_1, \ldots, t_n]$ for filling the n holes present in the multihole context C from left to right with the terms $t_1, \ldots, t_n$. In order to save space we abbreviate $a_1, \ldots, a_n$ to $\overline{a}$.

Definition A.1 ([57, Definition 4])**.** We write $s \mathrel{\parallel\!\!\!\!\rightarrow}_{C|\overline{a}} t$ if $s = C[a_1, \ldots, a_n]$ and $t = C[b_1, \ldots, b_n]$ for some $b_1, \ldots, b_n$ with $a_i \rightarrow_\epsilon b_i$ for all $1 \leqslant i \leqslant n$.

It is easy to show that $s \mathrel{\parallel\!\!\!\!\rightarrow} t$ if and only if $s \mathrel{\parallel\!\!\!\!\rightarrow}_{C|\overline{s}} t$ for some multihole context C and terms $\overline{s}$. In order to measure the overlap between two parallel steps starting at the same term, the overlapping redexes are collected in a multiset.

Definition A.2 ([57, Definition 5])**.** The overlap between two co-initial parallel rewrite steps is defined by the following equations

$$
\begin{aligned}
\blacktriangle(\, {}_{\Box|a}\!\mathrel{\leftarrow\!\!\!\!\parallel}\, s \mathrel{\parallel\!\!\!\!\rightarrow}_{\Box|b} \,) &= \{s\} \\
\blacktriangle(\, {}_{C|a_1,\ldots,a_c}\!\mathrel{\leftarrow\!\!\!\!\parallel}\, s \mathrel{\parallel\!\!\!\!\rightarrow}_{\Box|b} \,) &= \{a_1, \ldots, a_c\} \\
\blacktriangle(\, {}_{\Box|a}\!\mathrel{\leftarrow\!\!\!\!\parallel}\, s \mathrel{\parallel\!\!\!\!\rightarrow}_{C|b_1,\ldots,b_c} \,) &= \{b_1, \ldots, b_c\} \\
\blacktriangle(\, {}_{f(C_1,\ldots,C_n)|\overline{a}}\!\mathrel{\leftarrow\!\!\!\!\parallel}\, s \mathrel{\parallel\!\!\!\!\rightarrow}_{f(D_1,\ldots,D_n)|\overline{b}} \,) &= \bigcup_{i=1}^{n} \blacktriangle(\, {}_{C_i|\overline{a}_i}\!\mathrel{\leftarrow\!\!\!\!\parallel}\, s_i \mathrel{\parallel\!\!\!\!\rightarrow}_{D_i|\overline{b}_i} \,)
\end{aligned}
$$

Here $\overline{a}_1, \ldots, \overline{a}_n = \overline{a}$ and $\overline{b}_1, \ldots, \overline{b}_n = \overline{b}$ are partitions of $\overline{a}$ and $\overline{b}$ such that the length of $\overline{a}_i$ and $\overline{b}_i$ matches the number of holes in C_i and D_i for $1 \leqslant i \leqslant n$.

The properties in the next lemma are crucial for the proof of Theorem 4.

Lemma A.3 ([57, Lemma 6])**.** *For a peak* ${}_{C|\overline{a}}\!\mathrel{\leftarrow\!\!\!\!\parallel}\, s \mathrel{\parallel\!\!\!\!\rightarrow}_{D|\overline{b}}$ *the following properties of* $\blacktriangle(\cdot)$ *hold.*

- *If we have* $s = f(s_1, \ldots, s_n)$ *with* $C = f(C_1, \ldots, C_n)$ *and* $D = f(D_1, \ldots, D_n)$ *then* $\blacktriangle(\, {}_{C_i|\overline{a}_i}\!\mathrel{\leftarrow\!\!\!\!\parallel}\, s_i \mathrel{\parallel\!\!\!\!\rightarrow}_{D_i|\overline{b}_i} \,) \subseteq \blacktriangle(\, {}_{C|\overline{a}}\!\mathrel{\leftarrow\!\!\!\!\parallel}\, s \mathrel{\parallel\!\!\!\!\rightarrow}_{D|\overline{b}} \,)$ *for all* $1 \leqslant i \leqslant n$.

- *The overlap is bounded by* $\overline{a}$*:* $\{a_1, \ldots, a_c\} \rhd^{=}_{\mathsf{mul}} \blacktriangle(\, {}_{C|\overline{a}}\!\mathrel{\leftarrow\!\!\!\!\parallel}\, s \mathrel{\parallel\!\!\!\!\rightarrow}_{D|\overline{b}} \,)$.

- *The overlap is symmetric:* $\blacktriangle(\, {}_{D|\overline{b}}\!\mathrel{\leftarrow\!\!\!\!\parallel}\, s \mathrel{\parallel\!\!\!\!\rightarrow}_{C|\overline{a}} \,) = \blacktriangle(\, {}_{C|\overline{a}}\!\mathrel{\leftarrow\!\!\!\!\parallel}\, s \mathrel{\parallel\!\!\!\!\rightarrow}_{D|\overline{b}} \,)$.

The following technical lemma is needed in the third case of the proof of Theorem 4.28.

Lemma A.4 ([57, Lemma 7])**.** *Let s be a linear term. If $s\sigma \mathrel{+\!\!\!\!\!\rightarrow}_{C|\bar{s}} t$ then either $t = s\tau$ for some substitution τ such that $x\sigma \mathrel{+\!\!\!\!\!\rightarrow} x\tau$ for all $x \in \mathcal{V}\mathsf{ar}(s)$ or there exist a context D, a non-variable subterm s' of s, a rule $\rho\colon \ell \to r\ [\varphi] \in \mathcal{R} \cup \mathcal{R}_{\mathsf{ca}}$, a substitution $\tau \vDash \rho$, and a multihole context C' such that $s = D[s']$, $s'\sigma = \ell\tau$, $D\sigma[r\tau] = C'[s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n]$ and $t = C'[t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n]$ for some $1 \leqslant i \leqslant n$.* □

Lemma A.5. *Suppose $s \approx t\ [\varphi] \mathrel{\tilde{+\!\!\!\!\!\rightarrow}}_{\geqslant p} u \approx v\ [\psi]$ and $\gamma \vDash \varphi$. If $p = 1$ then $s\gamma \mathrel{+\!\!\!\!\!\rightarrow} u\delta$ and $t\gamma = v\delta$ for some substitution δ with $\delta \vDash \psi$. If $p = 2$ then $s\gamma = u\delta$ and $t\gamma \mathrel{+\!\!\!\!\!\rightarrow} v\delta$ for some substitution δ with $\delta \vDash \psi$.*

Proof. The statements are obtained by repeated applications of Lemma 4.20. The key observation, which follows from the proof of [88, Lemma 8], is that positions are preserved when lifting rewrite steps on constrained terms to the unconstrained setting. □

The lemma above also holds for rewrite sequences $s \approx t\ [\varphi] \mathrel{\tilde{\rightarrow}}^*_{\geqslant p} u \approx v\ [\psi]$.

Proof of Theorem 4.28. Consider an arbitrary peak

$$t \ {}_{C|\bar{a}}\!\mathrel{\leftarrow\!\!\!\!\!+} s \mathrel{+\!\!\!\!\!\rightarrow}_{D|\bar{b}} u$$

using rewrite rules from $\mathcal{R} \cup \mathcal{R}_{\mathsf{ca}}$. We show $t \mathrel{+\!\!\!\!\!\rightarrow} \cdot {}^*\!\!\mathrel{\leftarrow\!\!\!\!\!+} u$ and $t \mathrel{+\!\!\!\!\!\rightarrow}^* \cdot \mathrel{\leftarrow\!\!\!\!\!+} u$ by well-founded induction on the amount of overlap between the co-initial parallel steps using the order $\rhd_{\mathsf{mul}}$ and continue with a second induction on the term s using $\rhd$. If $s = x \in \mathcal{V}$ then $t = u = x$. Suppose $s = f(s_1, \ldots, s_n)$. We consider the following four cases.

1. If we have $C = f(C_1, \ldots, C_n)$ and $D = f(D_1, \ldots, D_n)$ then $t = f(t_1, \ldots, t_n)$, $u = f(u_1, \ldots, u_n)$ and we obtain partitions $\bar{a}_1, \ldots, \bar{a}_n = \bar{a}$ and $\bar{b}_1, \ldots, \bar{b}_n = \bar{b}$ of $\bar{a}$ and $\bar{b}$ with $t_i \ {}_{C_i|\bar{a}_i}\!\mathrel{\leftarrow\!\!\!\!\!+} s_i \mathrel{+\!\!\!\!\!\rightarrow}_{D_i|\bar{b}_i} u_i$ for all $1 \leqslant i \leqslant n$. The amount of overlap for each argument position is contained in the amount of overlap of the initial peak according to Lemma A.3:

$$\blacktriangle\big(\ {}_{C_i|\bar{a}_i}\!\mathrel{\leftarrow\!\!\!\!\!+} s_i \mathrel{+\!\!\!\!\!\rightarrow}_{D_i|\bar{b}_i}\big) \ \subseteq\ \blacktriangle\big(\ {}_{C|\bar{a}}\!\mathrel{\leftarrow\!\!\!\!\!+} s \mathrel{+\!\!\!\!\!\rightarrow}_{D|\bar{b}}\big)$$

   and thus

$$\blacktriangle\big(\ {}_{C|\bar{a}}\!\mathrel{\leftarrow\!\!\!\!\!+} s \mathrel{+\!\!\!\!\!\rightarrow}_{D|\bar{b}}\big) \ \rhd^{=}_{\mathsf{mul}}\ \blacktriangle\big(\ {}_{C_i|\bar{a}_i}\!\mathrel{\leftarrow\!\!\!\!\!+} s_i \mathrel{+\!\!\!\!\!\rightarrow}_{D_i|\bar{b}_i}\big)$$

   The inner induction hypothesis yields terms $v_1, \ldots, v_n, w_1, \ldots, w_n$ such that $t_i \mathrel{+\!\!\!\!\!\rightarrow} v_i \ {}^*\!\!\mathrel{\leftarrow\!\!\!\!\!+} u_i$ and $t_i \mathrel{+\!\!\!\!\!\rightarrow}^* w_i \mathrel{\leftarrow\!\!\!\!\!+} u_i$ for all $1 \leqslant i \leqslant n$. Hence $t \mathrel{+\!\!\!\!\!\rightarrow} f(v_1, \ldots, v_n) \ {}^*\!\!\mathrel{\leftarrow\!\!\!\!\!+} u$ and $t \mathrel{+\!\!\!\!\!\rightarrow}^* f(w_1, \ldots, w_n) \mathrel{\leftarrow\!\!\!\!\!+} u$.

2. If $C = D = \square$ then both steps are root steps and thus single rewrite steps. Hence the given peak can be written as

$$t = r_1\sigma_1 \leftarrow_{\epsilon|\rho_1|\sigma_1} s \rightarrow_{\epsilon|\rho_2|\sigma_2} r_2\sigma_2 = u$$

with rewrite rules $\rho_1\colon \ell_1 \to r_1\ [\varphi_1]$ and $\rho_2\colon \ell_2 \to r_2\ [\varphi_2]$ from $\mathcal{R} \cup \mathcal{R}_{\mathsf{ca}}$ and substitutions σ_1 and σ_2 that respect these rules. We have $s = \ell_1\sigma_1 = \ell_2\sigma_2$. We may assume that ρ_1 and ρ_2 have no variables in common, and thus $\mathcal{D}\mathsf{om}(\sigma_1) \cap \mathcal{D}\mathsf{om}(\sigma_2) = \varnothing$. From $\ell_1\sigma_1 = \ell_2\sigma_2$ we infer that $\sigma' = \sigma_1 \cup \sigma_2$ is a unifier of ℓ_1 and ℓ_2. Let σ be an mgu of ℓ_1 and ℓ_2. Since $\sigma_1 \vDash \rho_1$ and $\sigma_2 \vDash \rho_2$, $\sigma'(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2)$, and thus $\sigma(x) \in \mathcal{V} \cup \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2)$. Since $\varphi_1\sigma_1$ and $\varphi_2\sigma_2$ are valid, the constraint $\varphi_1\sigma \wedge \varphi_2\sigma$ is satisfiable. Hence conditions 1, 2, 3 and 4 in Definition 4.4 hold for $\langle \rho_1, \epsilon, \rho_2 \rangle$ and thus we obtain a critical pair, provided condition 5 also holds. If condition 5 is *not* fulfilled then ρ_1 and ρ_2 are variants and $\mathcal{V}\mathsf{ar}(r_1) \subseteq \mathcal{V}\mathsf{ar}(\ell_1)$ (and thus also $\mathcal{V}\mathsf{ar}(r_2) \subseteq \mathcal{V}\mathsf{ar}(\ell_2)$). Hence $r_1\sigma_1 = r_2\sigma_2$ and thus $t = u$. In the remaining case condition 5 holds and $\langle \rho_1, \epsilon, \rho_2 \rangle$ does form an overlap. The induced constrained critical pair is $r_2\sigma \approx r_1\sigma\ [\varphi_1\sigma \wedge \varphi_2\sigma \wedge \psi\sigma]$ with

$$\psi = \bigwedge \{x = x \mid x \in \mathcal{EV}\mathsf{ar}(\rho_1) \cup \mathcal{EV}\mathsf{ar}(\rho_2)\}$$

To simplify the notation, we abbreviate $r_1\sigma$ to t', $r_2\sigma$ to u', and $\varphi_1\sigma \wedge \varphi_2\sigma \wedge \psi\sigma$ to φ'. By assumption critical pairs are almost parallel closed and as we have an overlay we have both

a) $t' \approx u'\ [\varphi']\ \overset{\sim}{\mathrel{-\!\!\|\!\!\rightarrow}}_{\geqslant 1} \cdot \overset{\sim}{\rightarrow}{}^{*}_{\geqslant 2}\ v \approx w\ [\psi']$ for some trivial $v \approx w\ [\psi']$, and

b) $t' \approx u'\ [\varphi']\ \overset{\sim}{\mathrel{-\!\!\|\!\!\rightarrow}}_{\geqslant 2} \cdot \overset{\sim}{\rightarrow}{}^{*}_{\geqslant 1}\ v' \approx w'\ [\psi'']$ for some trivial $v' \approx w'\ [\psi'']$.

Let γ be the substitution such that $\sigma\gamma = \sigma'$. We claim that γ respects φ'. So let $x \in \mathcal{V}\mathsf{ar}(\varphi') = \mathcal{V}\mathsf{ar}(\varphi_2\sigma \wedge \varphi_1\sigma \wedge \psi\sigma)$. We have

$$\mathcal{LV}\mathsf{ar}(\rho_1) = \mathcal{V}\mathsf{ar}(\varphi_1) \cup \mathcal{EV}\mathsf{ar}(\rho_1) \qquad \mathcal{LV}\mathsf{ar}(\rho_2) = \mathcal{V}\mathsf{ar}(\varphi_2) \cup \mathcal{EV}\mathsf{ar}(\rho_2)$$

Together with $\mathcal{V}\mathsf{ar}(\psi) = \mathcal{EV}\mathsf{ar}(\rho_1) \cup \mathcal{EV}\mathsf{ar}(\rho_2)$ we obtain

$$\mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2) = \mathcal{V}\mathsf{ar}(\varphi_1) \cup \mathcal{V}\mathsf{ar}(\varphi_2) \cup \mathcal{V}\mathsf{ar}(\psi)$$

Since $\sigma'(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho_1) \cup \mathcal{LV}\mathsf{ar}(\rho_2)$, we obtain $\gamma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi')$ and thus $\gamma \vDash \varphi'$. At this point applications of Lemma A.5 to the constrained rewrite sequence in item 1 yields a substitution δ respecting ψ' such that $t'\gamma \mathrel{-\!\!\|\!\!\rightarrow} v\delta$ and $u'\gamma \to^* w\delta$. Since $v \approx w\ [\psi']$ is trivial, $v\delta = w\delta$ and hence $t'\gamma \mathrel{-\!\!\|\!\!\rightarrow} \cdot {}^{*}\!\!\leftarrow u'\gamma$. The sequence $t'\gamma \to^* \cdot \mathrel{\leftarrow\!\!\|\!\!-} u'\gamma$ is obtained in the same way from item 2. Hence

$$t'\gamma = (r_1\sigma)\gamma = r_1\sigma' = r_1\sigma_1 = t \qquad\qquad u'\gamma = r_2\sigma' = r_2\sigma_2 = u$$

an thus $t \mathrel{-\!\!\|\!\!\rightarrow} \cdot {}^{*}\!\!\leftarrow u$ and $t \to^* \cdot \mathrel{\leftarrow\!\!\|\!\!-} u$ as desired.

3. If $C = f(C_1, \ldots, C_n)$ and $D = \Box$ then the step to the right is a single root step and we have $t = f(t_1, \ldots, t_n)\ {}_{C|\overline{a}}\!\mathrel{\leftarrow\!\!\|\!\!-}\ s = \ell\sigma \to_{\epsilon|\rho|\sigma} r\sigma = u$ with rule $\rho\colon \ell \to r\ [\varphi]$ from $\mathcal{R} \cup \mathcal{R}_{\mathsf{ca}}$. Since ℓ is linear by assumption, we can apply Lemma A.4, which gives rise to the following two cases.

a) In the first case $t = \ell\tau$ for some substitution τ with $x\sigma \twoheadrightarrow\!\!\!\!\!\!+\; x\tau$ for all $x \in \mathcal{V}\mathsf{ar}(\ell)$. Define
$$\delta(x) = \begin{cases} \tau(x) & \text{if } x \in \mathcal{V}\mathsf{ar}(\ell) \\ \sigma(x) & \text{otherwise} \end{cases}$$
We have $t = \ell\tau = \ell\delta$ and claim that $t \to_\epsilon r\delta$. So we need to show that $\delta \vDash \rho$. If $x \in \mathcal{LV}\mathsf{ar}(\ell \to r\ [\varphi])$ then $\sigma(x) \in \mathcal{V}\mathsf{al}$ and thus $x\sigma = x\tau = x\delta$. Hence also $\delta(x) \in \mathcal{V}\mathsf{al}$. Since $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{LV}\mathsf{ar}(\ell \to r\ [\varphi])$, we obtain $\gamma\sigma = \gamma\delta$. Hence $\delta \vDash \rho$ as desired. Moreover, $u = r\sigma \twoheadrightarrow\!\!\!\!\!\!+\; r\delta$ since $x\sigma \twoheadrightarrow\!\!\!\!\!\!+\; x\delta$ for all variables $x \in \mathcal{V}\mathsf{ar}(r)$. The latter holds because $x\sigma \twoheadrightarrow\!\!\!\!\!\!+\; x\tau = x\delta$ if $x \in \mathcal{V}\mathsf{ar}(\ell)$ and $x\sigma = x\delta$ if $x \notin \mathcal{V}\mathsf{ar}(\ell)$.

b) In the second case we obtain a context E, a non-variable term ℓ'', a rule $\rho'\colon \ell' \to r'\ [\varphi']$, a substitution τ respecting ρ', and a multihole context C' such that $\ell = E[\ell'']$, $\ell''\sigma = \ell'\tau$, $E\sigma[r'\tau] = C'[s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_c]$ and $t = C'[t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_c]$ for some $1 \leqslant i \leqslant c$. Since we may assume without loss of generality that ρ and ρ' are variable disjoint, from $\ell''\sigma = \ell'\tau$ we infer that $\sigma' = \sigma \cup \tau$ is a unifier for ℓ'' and ℓ'. Let μ be an mgu of ℓ'' and ℓ'. We distinguish two cases, depending on ρ'.

First suppose $\rho' \in \mathcal{R}$. Since $C \neq \square$, we obtain a constrained critical pair $E\mu[r'\mu] \approx r\mu\ [\varphi'\mu \wedge \varphi\mu \wedge \psi\mu]$ with
$$\psi = \bigwedge \{x = x \mid x \in \mathcal{EV}\mathsf{ar}(\rho) \cup \mathcal{EV}\mathsf{ar}(\rho')\}$$
To simplify the notation, we abbreviate $E\mu[r'\mu]$ to t', $r\mu$ to u', and $\varphi'\mu \wedge \varphi\mu \wedge \psi\mu$ to φ''. By assumption critical pairs are almost parallel closed and, since we deal with an inner critical pair, we obtain
$$t' \approx u'\ [\varphi''] \mathrel{\tilde{\twoheadrightarrow\!\!\!\!\!\!+}}_{\geqslant 1} v \approx w\ [\psi'] \tag{A1}$$
for some trivial constrained equation $v \approx w\ [\psi']$. Let γ be the substitution such that $\mu\gamma = \sigma'$. We claim that γ respects φ''. So let $x \in \mathcal{V}\mathsf{ar}(\varphi'') = \mathcal{V}\mathsf{ar}(\varphi'\mu \wedge \varphi\mu \wedge \psi\mu)$. We have
$$\mathcal{LV}\mathsf{ar}(\rho') = \mathcal{V}\mathsf{ar}(\varphi') \cup \mathcal{EV}\mathsf{ar}(\rho') \qquad \mathcal{LV}\mathsf{ar}(\rho) = \mathcal{V}\mathsf{ar}(\varphi) \cup \mathcal{EV}\mathsf{ar}(\rho)$$
Together with $\mathcal{V}\mathsf{ar}(\psi) = \mathcal{EV}\mathsf{ar}(\rho') \cup \mathcal{EV}\mathsf{ar}(\rho)$ we obtain
$$\mathcal{LV}\mathsf{ar}(\rho') \cup \mathcal{LV}\mathsf{ar}(\rho) = \mathcal{V}\mathsf{ar}(\varphi') \cup \mathcal{V}\mathsf{ar}(\varphi) \cup \mathcal{V}\mathsf{ar}(\psi)$$
Since $\sigma'(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho') \cup \mathcal{LV}\mathsf{ar}(\rho)$, we obtain $\gamma(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi'')$ and thus $\gamma \vDash \varphi''$. We now apply Lemma A.5 to (A1). This yields a substitution δ respecting ψ' such that $t'\gamma \twoheadrightarrow\!\!\!\!\!\!+\; v\delta$ and $u'\gamma = w\delta$. Since $v \approx w\ [\psi']$ is trivial, $v\delta = w\delta$ and hence $t'\gamma \twoheadrightarrow\!\!\!\!\!\!+\; u'\gamma$. From $t'\gamma = (E\mu[r'\mu])\gamma = E\sigma'[r'\sigma'] = E\sigma[r'\tau]$ and $u'\gamma = (r\mu)\gamma = r\sigma' = r\sigma = u$ we obtain $E\sigma[r'\tau] \twoheadrightarrow\!\!\!\!\!\!+\; u$. Hence we have a new peak
$$t \;{}_{C'|\overline{a}'}\!\twoheadleftarrow\!\!\!\!\!\!+\; E\sigma[r'\tau] \twoheadrightarrow\!\!\!\!\!\!+\; u$$

with $\overline{a}' = a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_c$. We have

$$\begin{aligned} \blacktriangle(\,{}_{C|\overline{a}}{\mathrel{\leftarrow\!\!\!\!\!\!+\!\!\!+}}\ s\ \mathrel{+\!\!\!+\!\!\!\!\!\!\rightarrow}_{\square|\ell\sigma}\,) = \{a_1, \dots, a_c\} &\ \rhd_{\mathsf{mul}}\ \{a_1, \dots, a_{i-1}, a_{i+1}, \dots, a_c\} \\ &\ \rhd^{=}_{\mathsf{mul}}\ \blacktriangle(\,{}_{C'|\overline{a}'}{\mathrel{\leftarrow\!\!\!\!\!\!+\!\!\!+}}\ E\sigma[r'\tau]\ \mathrel{+\!\!\!+\!\!\!\!\!\!\rightarrow}\,) \end{aligned}$$

by Lemma A.3. Hence we can apply the outer induction hypothesis, which yields $t \mathrel{+\!\!\!+\!\!\!\!\!\!\rightarrow} \cdot {}^{*}\!\!\leftarrow u$ and $t \rightarrow^{*} \cdot \mathrel{\leftarrow\!\!\!\!\!\!+\!\!\!+} u$.

Next suppose $\rho' \in \mathcal{R}_{\mathsf{ca}}$. In this case $\ell''\sigma = \ell'\tau = f(s_1, \dots, s_n)$ with $f \in \mathcal{F}_{\mathsf{th}} \setminus \mathcal{F}_{\mathsf{te}}$, $s_1, \dots, s_n \in \mathcal{V}\mathsf{al}$ and $r' = [\![f(s_1, \dots, s_n)]\!]_{\mathcal{J}}$. So $r'\tau = r'$ and from this point on we can reason as in the preceding case.

4. The final case $D = f(D_1, \dots, D_n)$ and $C = \square$ is obtained from the previous one by symmetry.

We have completed the proof of the strong confluence of $\mathrel{+\!\!\!+\!\!\!\!\!\!\rightarrow}_{\mathcal{R}}$, which implies the confluence of $\rightarrow_{\mathcal{R}}$. □

B Appendix to Chapter 5

In this appendix, we give the missing proofs of the paper [68] which are only available in the full version on arXiv [69].

Proofs for Section 5.5

The following lemma is from [88].

Lemma B.1. *If $s\ [\varphi] \rightarrow_p t\ [\varphi]$ for a position p and substitution σ with $\sigma \vDash \varphi$ then $s\sigma \rightarrow_p t\sigma$.* □

Lemma B.2. *Suppose $s \approx t\ [\varphi] \overset{\sim}{\rightarrow}{}^{*}_{\geqslant p} u \approx v\ [\psi]$ with $\gamma \vDash \varphi$, $\mathcal{D}\mathsf{om}(\gamma) = \mathcal{V}\mathsf{ar}(\varphi)$ and position p. If $p = 1q$ for a position q then $s\gamma \rightarrow^{*}_{\geqslant q} u\delta$ and $t\gamma = v\delta$ for some substitution δ with $\delta \vDash \psi$ and $\mathcal{D}\mathsf{om}(\delta) = \mathcal{V}\mathsf{ar}(\psi)$. If $p = 2q$ for a position q then $s\gamma = u\delta$ and $t\gamma \rightarrow^{*}_{\geqslant q} v\delta$ for some substitution δ with $\delta \vDash \psi$ and $\mathcal{D}\mathsf{om}(\delta) = \mathcal{V}\mathsf{ar}(\psi)$.*

Proof. We prove the lemma for a single step $s \approx t\ [\varphi] \overset{\sim}{\rightarrow}_{\geqslant p} u \approx v\ [\psi]$ with $\gamma \vDash \varphi$, $\mathcal{D}\mathsf{om}(\gamma) = \mathcal{V}\mathsf{ar}(\varphi)$ and position p. The general statement follows then by an obvious induction argument. Consider the case $p = 1q$. By definition we can split the step into $s \approx t\ [\varphi] \sim s' \approx t'\ [\varphi'] \rightarrow_{\geqslant p} u' \approx t'\ [\varphi'] \sim u \approx v\ [\psi]$. Since $\gamma \vDash \varphi$, by definition of $\sim$, there exists a substitution $\sigma \vDash \varphi'$ with $\mathcal{D}\mathsf{om}(\sigma) = \mathcal{V}\mathsf{ar}(\varphi')$ such that $(s \approx t)\gamma = (s' \approx t')\sigma$. Lemma B.1 yields $(s' \approx t')\sigma \rightarrow_{\geqslant p} (u' \approx t')\sigma$. Again by definition of $\sim$ we have $(u' \approx t')\sigma = (u \approx v)\tau$ for a substitution τ with $\tau \vDash \psi$ and $\mathcal{D}\mathsf{om}(\tau) = \mathcal{V}\mathsf{ar}(\psi)$. Hence $s\gamma = s'\sigma \rightarrow_{\geqslant q} u'\sigma = u\tau$ and $t\gamma = t'\sigma = v\tau$. The case $p = 2q$ is similar. □

Proof of Lemma 5.16. By separating the domain of the substitution to logical and non-logical variables in Lemma B.2 and applying closure under substitution, we obtain the result. □

Lemma B.3. *If $s\ [\varphi] \mathrel{\circ\!\!\!\!\to} t\ [\varphi]$ then $s\delta \mathrel{\circ\!\!\!\!\to} t\delta$ for all substitutions $\delta \vDash \varphi$.*

Proof. We proceed by induction on $\mathrel{\circ\!\!\!\!\to}$. In case 1 we have $x\ [\varphi] \mathrel{\circ\!\!\!\!\to} x\ [\varphi]$, and $x\delta \mathrel{\circ\!\!\!\!\to} x\delta$ holds trivially. In case 2 we have $s = f(s_1, \ldots, s_n)$, $t = f(t_1, \ldots, t_n)$ and $s_i\ [\varphi] \mathrel{\circ\!\!\!\!\to} t_i\ [\varphi]$ for $1 \leqslant i \leqslant n$. From the induction hypothesis we obtain $s_i\delta \mathrel{\circ\!\!\!\!\to} t_i\delta$ for all $1 \leqslant i \leqslant n$, which further implies $s\delta \mathrel{\circ\!\!\!\!\to} t\delta$. In case 3 we have $s = \ell\sigma$ and $t = r\sigma$ for some rule $\rho\colon \ell \to r\ [\psi] \in \mathcal{R}_{\mathsf{rc}}$, $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}\mathsf{ar}(\varphi)$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$, φ is satisfiable, $\varphi \Rightarrow \psi\sigma$ is valid, and $\sigma(x)\ [\varphi] \mathrel{\circ\!\!\!\!\to} \tau(x)\ [\varphi]$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$. From $\mathcal{LV}\mathsf{ar}(\rho) \subseteq \mathcal{D}\mathsf{om}(\sigma)$ and $s = \ell\sigma$ we infer $\mathcal{V}\mathsf{ar}(\rho) \subseteq \mathcal{D}\mathsf{om}(\sigma)$. Without loss of generality we assume $(\mathcal{V}\mathsf{ar}(s) \cup \mathcal{V}\mathsf{ar}(t) \cup \mathcal{V}\mathsf{ar}(\varphi)) \cap \mathcal{V}\mathsf{ar}(\rho) = \varnothing$ and restrict the domain of σ to $\mathcal{V}\mathsf{ar}(\rho)$ as other variables are irrelevant. From the induction hypothesis we obtain $\sigma(x)\delta \mathrel{\circ\!\!\!\!\to} \tau(x)\delta$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$. Moreover, since $\delta \vDash \varphi$ we have $\delta \vDash \psi\sigma$ and thus also $\sigma\delta \vDash \psi$. Therefore $s\delta = \ell\sigma\delta \mathrel{\circ\!\!\!\!\to} r\tau\delta = t\delta$ as desired. ☐

Lemma B.4. *If $s \approx t\ [\varphi] \mathrel{\widetilde{\circ\!\!\!\!\to}}_{\geqslant 1} u \approx v\ [\psi]$ then for all substitutions $\sigma \vDash \varphi$ with $\mathcal{D}\mathsf{om}(\sigma) = \mathcal{V}\mathsf{ar}(\varphi)$ there exists $\delta \vDash \psi$ with $\mathcal{D}\mathsf{om}(\delta) = \mathcal{V}\mathsf{ar}(\psi)$ such that $s\sigma \mathrel{\circ\!\!\!\!\to} u\delta$ and $t\sigma = v\delta$.*

Proof. By unfolding the definition of $\mathrel{\widetilde{\circ\!\!\!\!\to}}$ we obtain $s \approx t\ [\varphi] \sim s' \approx t'\ [\varphi'] \mathrel{\circ\!\!\!\!\to}_{\geqslant 1} u' \approx v'\ [\varphi'] \sim u \approx v\ [\psi]$. Let σ be a substitution with $\sigma \vDash \varphi$. From the definition of $\sim$ we obtain a substitution τ such that $\tau \vDash \varphi'$, $\mathcal{D}\mathsf{om}(\tau) = \mathcal{V}\mathsf{ar}(\varphi')$, $s\sigma = s'\tau$ and $t\sigma = t'\tau$. As all contracted redexes in the multi-step $s' \approx t'\ [\varphi']$ are below the position 1, this corresponds to case 2 in Definition 5.13 with s' and t' being the first and second argument of $\approx$. Hence $s'\ [\varphi'] \mathrel{\circ\!\!\!\!\to} u'\ [\varphi']$ and $t' = v'$. We therefore obtain $t'\tau = v'\tau$ and $s'\tau \mathrel{\circ\!\!\!\!\to} u'\tau$ from Lemma B.3. From the equivalence $u' \approx v'\ [\varphi'] \sim u \approx v\ [\psi]$ together with $\tau \vDash \varphi'$ and $\mathcal{D}\mathsf{om}(\tau) = \varphi'$ we obtain a substitution γ such that $\gamma \vDash \psi$, $\mathcal{D}\mathsf{om}(\gamma) = \mathcal{V}\mathsf{ar}(\psi)$, $u'\tau = u\gamma$ and $v'\tau = v\gamma$. Hence $s\sigma = s'\tau \mathrel{\circ\!\!\!\!\to} u'\tau = u\gamma$ and $t\sigma = t'\tau = v'\tau = v\gamma$. ☐

Proof of Lemma 5.17. This follows by dropping the restriction on the substitution in Lemma B.4, exactly like in the proof of Lemma 5.16. ☐

Proofs for Section 5.6

Lemma B.5. *If $s\ [\varphi] \mathrel{+\!\!\!\!\to}^{P} t\ [\varphi]$ then $s\delta \mathrel{+\!\!\!\!\to}^{P} t\delta$ for all substitutions δ with $\delta \vDash \varphi$.*

Proof. We proceed by induction on $\mathrel{+\!\!\!\!\to}$. In case 1 we have $x\ [\varphi] \mathrel{+\!\!\!\!\to}^{\varnothing} x\ [\varphi]$, and $x\delta \mathrel{+\!\!\!\!\to}^{\varnothing} x\delta$ holds trivially. In case 2 we have $s = f(s_1, \ldots, s_n)$, $t = f(t_1, \ldots, t_n)$, $s_i\ [\varphi] \mathrel{+\!\!\!\!\to}^{P_i} t_i\ [\varphi]$ for $1 \leqslant i \leqslant n$ and $P = \{ip \mid 1 \leqslant i \leqslant n \text{ and } p \in P_i\}$. From the induction hypothesis we obtain $s_i\delta \mathrel{+\!\!\!\!\to}^{P_i} t_i\delta$ for all $1 \leqslant i \leqslant n$ and thus also $s\delta \mathrel{+\!\!\!\!\to}^{P} t\delta$. In case 3 we have $s \mathrel{+\!\!\!\!\to}^{\{\epsilon\}} t$ with $s = \ell\sigma$ and $t = r\sigma$ for some rule $\rho\colon \ell \to r\ [\psi] \in \mathcal{R}_{\mathsf{rc}}$ and substitution σ such that $\sigma(x) \in \mathcal{V}\mathsf{al} \cup \mathcal{V}\mathsf{ar}(\varphi)$ for all $x \in \mathcal{LV}\mathsf{ar}(\rho)$, φ is satisfiable and $\varphi \Rightarrow \psi\sigma$ is valid. From $\mathcal{LV}\mathsf{ar}(\rho) \subseteq \mathcal{D}\mathsf{om}(\sigma)$ and $s = \ell\sigma$ we infer $\mathcal{V}\mathsf{ar}(\rho) \subseteq \mathcal{D}\mathsf{om}(\sigma)$. Without loss of generality we assume $(\mathcal{V}\mathsf{ar}(s) \cup \mathcal{V}\mathsf{ar}(t) \cup \mathcal{V}\mathsf{ar}(\varphi)) \cap \mathcal{V}\mathsf{ar}(\rho) = \varnothing$ and restrict the domain of σ to $\mathcal{V}\mathsf{ar}(\rho)$. Moreover, from $\delta \vDash \varphi$ we infer $\delta \vDash \psi\sigma$ and thus also $\sigma\delta \vDash \psi$. Therefore $s\delta = \ell\sigma\delta \mathrel{+\!\!\!\!\to}^{\{\epsilon\}} r\sigma\delta = t\delta$ as desired. ☐

The following lemma is proved by replaying the proof of Lemma B.4, using Definition 5.23 and Lemma B.5 with the respective adaptions to the parallel rewrite relation.

Lemma B.6. *If $s \approx t\ [\varphi] \overset{\sim}{\mathrel{+\!\!\!\!\!\twoheadrightarrow}}{}_{\geqslant 1}^{P}\ u \approx v\ [\psi]$ then for all substitutions $\sigma \vDash \varphi$ with $\mathcal{D}\mathsf{om}(\sigma) = \mathcal{V}\mathsf{ar}(\varphi)$ there exists a $\delta \vDash \psi$ with $\mathcal{D}\mathsf{om}(\delta) = \mathcal{V}\mathsf{ar}(\psi)$ such that $s\sigma \mathrel{+\!\!\!\!\!\twoheadrightarrow}^{P} u\delta$ and $t\sigma = v\delta$.* □

Proof of Lemma 5.31. This follows from Lemma B.6, similar to the proof of Lemma 5.16. □

Proof of Lemma 5.32. Let $s \approx t\ [\varphi]$ be a 1-parallel closed constrained critical pair. So

$$s \approx t\ [\varphi] \overset{\sim}{\mathrel{+\!\!\!\!\!\twoheadrightarrow}}{}_{\geqslant 1}\ s' \approx t'\ [\varphi'] \overset{\sim}{\rightarrow}{}_{\geqslant 2}^{*}\ u \approx v\ [\psi] \tag{B1}$$

with trivial $u \approx v\ [\psi]$. Let $\sigma \vDash \varphi$. To the first part of (B1) we apply Lemma B.6. This yields a substitution $\delta \vDash \varphi'$ such that $s\sigma \mathrel{+\!\!\!\!\!\twoheadrightarrow}^{P} s'\delta$ and $t\sigma = t'\delta$. According to Lemma 5.16 the second part of (B1) yields $s'\delta = u\gamma$ and $t'\delta \to^{*} v\gamma$ for some substitution γ with $\gamma \vDash \psi$. We obtain $u\gamma = v\gamma$ from the triviality of $u \approx v\ [\psi]$. Combining the above facts yields the desired $s\sigma \mathrel{+\!\!\!\!\!\twoheadrightarrow}^{P} u\gamma = v\gamma\ {}^{*}\!\leftarrow t\sigma$. □

Proof of Lemma 5.33. Let $s = \ell\sigma'[r_p\sigma']_{p \in P} \approx r\sigma' = t\ [\varphi]$ be a 2-parallel closed constrained critical pair. So

$$s \approx t\ [\varphi] \overset{\sim}{\mathrel{+\!\!\!\!\!\twoheadrightarrow}}{}_{\geqslant 2}^{Q}\ s' \approx t'\ [\varphi'] \overset{\sim}{\rightarrow}{}_{\geqslant 1}^{*}\ u' \approx v'\ [\psi] \tag{B2}$$

with trivial $u' \approx v'\ [\psi]$ and $\mathcal{TV}\mathsf{ar}(v', \psi, Q) \subseteq \mathcal{TV}\mathsf{ar}(\ell\sigma', \varphi, P)$. Consider a substitution σ and let $\sigma \vDash \varphi$. We split σ into $\sigma = \sigma_{\mathsf{th}} \cup \sigma_{\mathsf{te}} = \sigma_{\mathsf{th}}\sigma_{\mathsf{te}}$ such that $\mathcal{D}\mathsf{om}(\sigma_{\mathsf{th}}) \cap \mathcal{D}\mathsf{om}(\sigma_{\mathsf{te}}) = \varnothing$ and $\mathcal{D}\mathsf{om}(\sigma_{\mathsf{th}}) = \mathcal{V}\mathsf{ar}(\varphi)$. From $\sigma \vDash \varphi$ we obtain $\sigma_{\mathsf{th}} \vDash \varphi$. To the first part of (B2) we apply Lemma B.6. This yields a substitution $\delta \vDash \varphi'$ with $\mathcal{D}\mathsf{om}(\delta) = \mathcal{V}\mathsf{ar}(\varphi')$ such that $s\sigma_{\mathsf{th}} = s'\delta$ and $t\sigma_{\mathsf{th}} \mathrel{+\!\!\!\!\!\twoheadrightarrow}^{Q} t'\delta$. Applying Lemma B.2 to the second part of (B2) yields $s'\delta \to^{*} u'\gamma$ and $t'\delta = v'\gamma$ for some substitution γ with $\gamma \vDash \psi$ and $\mathcal{D}\mathsf{om}(\gamma) = \mathcal{V}\mathsf{ar}(\psi)$. We obtain $u'\gamma = v'\gamma$ from the triviality of $u' \approx v'\ [\psi]$. Combining the above facts yields $s\sigma = s\sigma_{\mathsf{th}}\sigma_{\mathsf{te}} \to^{*} u'\gamma\sigma_{\mathsf{te}} = v'\gamma\sigma_{\mathsf{te}}\ {}^{Q}\!\twoheadleftarrow\!\!\!\!\!+\ t\sigma_{\mathsf{th}}\sigma_{\mathsf{te}} = t\sigma$ by closure of rewriting under substitution. It remains to show the variable condition $\mathcal{V}\mathsf{ar}(v'\gamma\sigma_{\mathsf{te}}, Q) \subseteq \mathcal{V}\mathsf{ar}(\ell\sigma'\sigma, P)$. Clearly

$$\begin{aligned}
\mathcal{TV}\mathsf{ar}(\ell\sigma', \varphi, P) &= \bigcup_{p \in P} \mathcal{TV}\mathsf{ar}(\ell\sigma'|_p, \varphi) = \bigcup_{p \in P} \mathcal{V}\mathsf{ar}(\ell\sigma'|_p) \setminus \mathcal{V}\mathsf{ar}(\varphi) \\
&= \bigcup_{p \in P} \mathcal{V}\mathsf{ar}(\ell\sigma'\sigma_{\mathsf{th}}|_p) = \mathcal{V}\mathsf{ar}(\ell\sigma'\sigma_{\mathsf{th}}, P)
\end{aligned}$$

and similarly $\mathcal{TV}\mathsf{ar}(v', \psi, Q) = \mathcal{V}\mathsf{ar}(v'\gamma, Q)$. Hence $\mathcal{V}\mathsf{ar}(v'\gamma, Q) \subseteq \mathcal{V}\mathsf{ar}(\ell\sigma'\sigma_{\mathsf{th}}, P)$ follows from $\mathcal{TV}\mathsf{ar}(v', \psi, Q) \subseteq \mathcal{TV}\mathsf{ar}(\ell\sigma', \varphi, P)$. Therefore,

$$\mathcal{V}\mathsf{ar}(v'\gamma\sigma_{\mathsf{te}}, Q) \subseteq \mathcal{V}\mathsf{ar}(\ell\sigma'\sigma_{\mathsf{th}}\sigma_{\mathsf{te}}, P) = \mathcal{V}\mathsf{ar}(\ell\sigma'\sigma, P)$$

as desired. □

C Appendix to Chapter 6

In this appendix, we present the proofs which are omitted in [4]. However, they are also available from the full version on arXiv [5].

Full Proofs of Section 6.2

Lemma 6.1. *Let $t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, \mathcal{V}_{\mathsf{th}})$, ρ a valuation, and σ a substitution.*

1. *Suppose $\sigma(x) \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, \mathcal{V}_{\mathsf{th}})$ for all $x \in \mathcal{V}_{\mathsf{th}}$. Let $[\![\sigma]\!]_{M,\rho}$ be a valuation defined as $[\![\sigma]\!]_{M,\rho}(x) = [\![\sigma(x)]\!]_{M,\rho}$. Then, $[\![t]\!]_{M,[\![\sigma]\!]_{M,\rho}} = [\![t\sigma]\!]_{M,\rho}$.*
2. *Suppose that $\mathcal{V}\mathsf{ar}(t) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$. Then, $[\![t]\!]_{M,\hat{\sigma}} = [\![t\sigma]\!]_M$, where the valuation $\hat{\sigma}$ is defined by $\hat{\sigma}(x^\tau) = \xi(\sigma(x)) \in \mathcal{I}(\tau)$ for $x \in \mathcal{VD}\mathsf{om}(\sigma)$, where ξ is a bijection $\mathcal{V}\mathsf{al}^\tau \cong \mathcal{I}(\tau)$.*

Proof. 1. The proof proceeds by structural induction on $t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, \mathcal{V}_{\mathsf{th}})$. Suppose $t = x \in \mathcal{V}_{\mathsf{th}}$. Then, $[\![x]\!]_{[\![\sigma]\!]_\rho} = [\![\sigma]\!]_\rho(x) = [\![\sigma(x)]\!]_\rho$ by definition. Suppose $t = f(t_1, \ldots, t_n)$. Then, by using the induction hypothesis, we obtain $[\![t]\!]_{[\![\sigma]\!]_\rho} = [\![f(t_1, \ldots, t_n)]\!]_{[\![\sigma]\!]_\rho} = \mathcal{J}(f)([\![t_1]\!]_{[\![\sigma]\!]_\rho}, \ldots, [\![t_n]\!]_{[\![\sigma]\!]_\rho}) = \mathcal{J}(f)([\![t_1\sigma]\!]_\rho, \ldots, [\![t_n\sigma]\!]_\rho) = [\![f(t_1\sigma, \ldots, t_n\sigma)]\!]_\rho = [\![t\sigma]\!]_\rho$. *2.* Take an arbitrary valuation ρ. Then, as $\mathcal{V}\mathsf{ar}(t) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$, we have $[\![\sigma]\!]_{M,\rho}(x) = [\![\sigma(x)]\!]_{M,\rho} = [\![\sigma(x)]\!]_M = \sigma(x) = \hat{\sigma}(x)$ for $x \in \mathcal{V}\mathsf{ar}(t)$. Thus, similar to *1.*, it follows by induction on t that $[\![t]\!]_{M,\hat{\sigma}} = [\![t\sigma]\!]_{M,\rho}$. Since $\mathcal{V}\mathsf{ar}(t) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$, we have $t\sigma \in \mathcal{T}(\mathcal{F}_{\mathsf{th}})$. Thus $[\![t]\!]_{M,\hat{\sigma}} = [\![t\sigma]\!]_M$. □

Lemma 6.2. *Let φ be a logical constraint.*

1. *$\models_M \varphi$ if and only if $\models_M \varphi\sigma$ for all substitutions σ such that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$.*
2. *If $\models_M \varphi$, then $\models_M \varphi\sigma$ for all substitutions σ such that $\sigma(x) \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, \mathcal{V}_{\mathsf{th}})$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi) \cap \mathcal{D}\mathsf{om}(\sigma)$.*
3. *The following statements are equivalent: (1) $\models_M \neg\varphi$, (2) $\not\models_M \varphi\sigma$ for all substitutions σ such that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$, and (3) $\sigma \models_M \varphi$ for no substitution σ.*

Here, $\sigma \models_M \varphi$ denotes that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$ and $\models_M \varphi\sigma$ hold.

Proof.

1. ($\Rightarrow$) Suppose $\models_M \varphi$. Then $[\![\varphi]\!]_{M,\rho} = \mathsf{true}$ for any valuation ρ over M. Thus, for any substitution σ such that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$, $[\![\varphi]\!]_{M,\hat{\sigma}} = \mathsf{true}$, where $\hat{\sigma}$ is defined as in Lemma 6.1. Hence, by Lemma 6.1, $[\![\varphi\sigma]\!]_M = \mathsf{true}$ for any substitution σ such that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$. Therefore, $\models_M \varphi\sigma$ holds for all substitution σ such that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$. ($\Leftarrow$) Suppose $\models_M \varphi\sigma$ for all substitutions σ such that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$. Let ρ be a valuation over a model $M = \langle \mathcal{I}, \mathcal{J} \rangle$. Then $\rho = (\rho_\tau)_{\tau \in S_{\mathsf{th}}}$ for mappings $\rho_\tau \colon \mathcal{V}^\tau \to \mathcal{I}(\tau)$. Now, in the view of $\mathcal{V}\mathsf{al}_\tau \cong \mathcal{I}(\tau)$, we can take a substitution $\check{\rho}$ given by $\check{\rho}(x) = \rho(x) \in \mathcal{V}\mathsf{al}$ for all $x \in \mathcal{V}\mathsf{ar}(\varphi)$.

Then, as $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{VD}\mathsf{om}(\check{\rho})$, we have $\models_M \varphi\check{\rho}$ by our assumption. This means $[\![\varphi\check{\rho}]\!]_M = \mathsf{true}$. Hence, it follows from Lemma 6.1 that $[\![\varphi]\!]_{M,\hat{\check{\rho}}} = \mathsf{true}$. Moreover, we have $\hat{\check{\rho}}(x) = \rho(x)$ for $x \in \mathcal{V}\mathsf{ar}(\varphi)$ by definition, and thus $[\![\varphi]\!]_{M,\rho} = \mathsf{true}$. Therefore, $[\![\varphi]\!]_{M,\rho} = \mathsf{true}$ for any valuation ρ over M. This shows $\models_M \varphi$.

2. Suppose $\models_M \varphi$. Then $[\![\varphi]\!]_{M,\rho} = \mathsf{true}$ for any valuation ρ over M. Suppose that σ is a substitution such that $\mathcal{V}\mathsf{ar}(\varphi) \cap \mathcal{D}\mathsf{om}(\sigma) \subseteq \mathcal{T}(\mathcal{F}_{\mathsf{th}}, \mathcal{V}_{\mathsf{th}})$. Let σ' be a substitution such that $\sigma'(x) = \sigma(x)$ for $x \in \mathcal{V}\mathsf{ar}(\varphi)$ and $\sigma'(x) = x$ otherwise. Then $\sigma'(x) \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, \mathcal{V}_{\mathsf{th}})$ for any $x \in \mathcal{V}_{\mathsf{th}}$. Then, using Lemma 6.1, we have $[\![\varphi]\!]_{[\![\sigma]\!]_\rho} = [\![\varphi]\!]_{[\![\sigma']\!]_\rho} = [\![\varphi\sigma']\!]_\rho = [\![\varphi\sigma]\!]_\rho$. Thus, $[\![\varphi\sigma]\!]_\rho = \mathsf{true}$ for any ρ. Therefore, $\models_M \varphi\sigma$.

3. Using *1*, we have $\models_M \neg\varphi$ iff $\models_M \neg\varphi\sigma$ for all substitution σ such that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$ iff $\not\models_M \varphi\sigma$ for all substitution σ such that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$ iff there is no substitution σ such that $\models_M \varphi\sigma$ and $\mathcal{V}\mathsf{ar}(\varphi) \subseteq \mathcal{VD}\mathsf{om}(\sigma)$ iff $\sigma \models \varphi$ for no substitution σ.

□

Lemma 6.3. *Let $s, t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}})$. Then, all of the following holds: 1. $[\![t]\!] \in \mathcal{V}\mathsf{al}$, 2. $t \to^!_{\mathsf{ca}} [\![t]\!]$, 3. $s \to^*_{\mathsf{ca}} t$ implies $[\![s]\!] = [\![t]\!]$, and 4. $s \leftrightarrow^*_{\mathsf{calc}} t$ if and only if $[\![s]\!] = [\![t]\!]$.*

Proof. *1.* This claim follows as $[\![t^\tau]\!] \in \mathcal{I}(\tau) \cong \mathcal{V}\mathsf{al}^\tau$. *2.* To show this, it suffices to show $t \to^*_{\mathsf{ca}} [\![t]\!]$, since values are normal forms with respect to calculation steps. We show $t \to^*_{\mathsf{ca}} [\![t]\!]$ by structural induction on t. Suppose $t \in \mathcal{V}\mathsf{al}$, then $[\![t]\!] = t$ and hence $t \to^*_{\mathsf{ca}} [\![t]\!]$. Suppose $t \notin \mathcal{V}\mathsf{al}$. Then $t = f(t_1, \dots, t_n)$ for some $f \in \mathcal{F}_{\mathsf{th}} \setminus \mathcal{V}\mathsf{al}$ and $t_1, \dots, t_n \in \mathcal{T}(\mathcal{F}_{\mathsf{th}})$. By the induction hypothesis, we have $t_i \to^*_{\mathsf{ca}} [\![t_i]\!]$ for all $1 \leqslant i \leqslant n$. Furthermore, by the definition of the calculation steps we have $f([\![t_1]\!], \dots, [\![t_n]\!]) \to \mathcal{I}(f)([\![t_1]\!], \dots, [\![t_n]\!]) = [\![f(t_1, \dots, t_n)]\!]$. Therefore, $t = f(t_1, \dots, t_n) \to^*_{\mathsf{ca}} f([\![t_1]\!], \dots, [\![t_n]\!]) \to_{\mathsf{ca}} [\![t]\!]$. *3.* First, note that the set of calculation rules $\mathcal{R}_{\mathsf{ca}} = \{ f(x_1, \dots, x_n) \to x_0\ [x_0 = f(x_1, \dots, x_n)] \mid f \colon \tau_1 \times \cdots \times \tau_n \to \tau_0 \in \mathcal{F}_{\mathsf{th}} \setminus \mathcal{V}\mathsf{al},\ x_i \in \mathcal{V}\mathsf{ar}^{\tau_i} \text{ for all } 0 \leqslant i \leqslant n \}$ is terminating and has no overlaps, and hence is confluent. [45] Suppose $s \to^*_{\mathsf{ca}} t$. Then it follows from *2.* that $s \to^!_{\mathsf{ca}} [\![s]\!]$ and $t \to^!_{\mathsf{ca}} [\![t]\!]$. Hence, $[\![s]\!] = [\![t]\!]$ by confluence. This shows the desired claim. *4.* The *only-if* part of this claim follows immediately from *3.* The *if* part follows from *1.* as $s \to^*_{\mathsf{ca}} [\![s]\!] = [\![t]\!]\ {}_{\mathsf{calc}}\!\leftarrow^* t$. □

Full Proofs of Section 6.3

Lemma 6.10. *Let E be an LCES over the underlying model $\mathfrak{U} = \langle \mathcal{S}_{\mathsf{th}}, \mathcal{F}_{\mathsf{th}}, M \rangle$ and the term signature $\Sigma_{\mathsf{te}} = \langle \mathcal{S}_{\mathsf{te}}, \mathcal{F}_{\mathsf{te}} \rangle$. Then, all of the following hold: 1. $\leftrightarrow_E$ is symmetric, 2. $\leftrightarrow_E$ is closed under contexts i.e. $s \leftrightarrow_E t$ implies $C[s] \leftrightarrow_E C[t]$ for any context C, and 3. $\leftrightarrow_E$ is closed under substitutions, i.e. $s \leftrightarrow_E t$ implies $s\sigma \leftrightarrow_E t\sigma$ for any substitution σ.*

Proof. *1.* and *2.* are trivial. For *3.* the case $s \leftrightarrow_{\mathsf{calc}} t$ is clear. So, we consider $s \leftrightarrow_{\mathsf{rule},E} t$ and have a CE $\Pi X.\, \ell \approx r\ [\varphi] \in E$, an X-valued substitution ρ such that $\models_M \varphi\rho$, a context C such that $s = C[\ell\rho]$ and $t = C[r\rho]$ (or vice versa). Then, for any $x \in X$,

$\sigma(\rho(x)) = \rho(x)$ as $\rho(x) \in \mathcal{V}\mathsf{al}$. Thus, $\sigma \circ \rho$ is again an X-valued substitution. Moreover, as $\mathcal{V}\mathsf{ar}(\varphi) \subseteq X \subseteq \mathcal{VD}\mathsf{om}(\rho)$, we have $\varphi\rho \in \mathcal{T}(\mathcal{F}_{\mathsf{th}})$. Also, $\varphi\rho = (\varphi\rho)\sigma = \varphi(\sigma \circ \rho)$ and hence $\models_M \varphi(\sigma \circ \rho)$. The substitution $\sigma \circ \rho$ is an X-valued substitution with $\models_M \varphi(\sigma \circ \rho)$. Therefore, we have $s\sigma \leftrightarrow_{\mathsf{rule},E} t\sigma$, because $s\sigma = C[\ell\rho]\sigma = C\sigma[\ell(\sigma \circ \rho)]$ and $t\sigma = C[r\rho]\sigma = C\sigma[r(\sigma \circ \rho)]$. □

Lemma 6.14. *Let $\mathfrak{T} = \langle M, E\rangle$ be a CE-theory. Then for any $\Pi X.\, s \approx t\ [\varphi] \in E$, we have $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$.*

Proof. Consider a CE-theory $\mathfrak{T} = \langle M, E\rangle$ and an arbitrary CE $\Pi X.\, s \approx t\ [\varphi] \in E$. Let σ be an X-valued substitution such that $\models_M \varphi\sigma$. Then, we have $s\sigma \leftrightarrow_{\mathsf{rule},E} t\sigma$ by definition of rewriting with equations. Hence, $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$ and therefore $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$. □

Lemma 6.15 (congruence)**.** *Let $\mathfrak{T} = \langle M, E\rangle$ be a CE-theory. For any set $X \subseteq \mathcal{V}_{\mathsf{th}}$ and logical constraint φ such that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq X$, the binary relation $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, \cdot \approx \cdot\ [\varphi]$ over terms is a congruence relation over Σ.*

Proof. Consider a CE-theory $\mathfrak{T}$. Furthermore, we assume a set $X \subseteq \mathcal{V}_{\mathsf{th}}$ and a logical constraint φ such that $\mathcal{V}\mathsf{ar}(\varphi) \subseteq X$. In order for the CE-consequence to be a binary congruence relation over terms, it needs to be reflexive, symmetric, transitive and congruent. We prove those properties independently:

1. Trivially $s\sigma \overset{*}{\leftrightarrow}_E s\sigma$ holds for all (X-valued) substitutions σ. Hence also $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, s \approx s\ [\varphi]$. (reflexivity)

2. By Lemma 6.10, we have $u \leftrightarrow_E v$ if and only if $v \leftrightarrow_E u$. By repeated application we obtain $u \overset{*}{\leftrightarrow}_E v$ if and only if $v \overset{*}{\leftrightarrow}_E u$. Consider that $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$. Then we have for all X-valued substitutions σ with $\models_M \varphi\sigma$ that $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$, which further implies $t\sigma \overset{*}{\leftrightarrow}_E s\sigma$. Therefore, $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, t \approx s\ [\varphi]$. (symmetry)

3. Consider that $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$ and $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, t \approx u\ [\varphi]$. This implies that for all X-valued substitutions σ with $\models_M \varphi\sigma$ that $s\sigma \overset{*}{\leftrightarrow}_E t\sigma \overset{*}{\leftrightarrow}_E u\sigma$, which gives further that $s\sigma \overset{*}{\leftrightarrow}_E u\sigma$. Therefore, $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, s \approx u\ [\varphi]$ follows. (transitivity)

4. Consider that $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, s_i \approx t_i\ [\varphi]$ for all $1 \leqslant i \leqslant n$. We have for all X-valued substitutions σ with $\models_M \varphi\sigma$ that $s_i\sigma \overset{*}{\leftrightarrow}_E t_i\sigma$ for all $1 \leqslant i \leqslant n$, which implies that $f(s_1, \ldots, s_n)\sigma = f(s_1\sigma, \ldots, s_n\sigma) \overset{*}{\leftrightarrow}_E f(t_1\sigma, \ldots, t_n\sigma) = f(t_1, \ldots, t_n)\sigma$ for an $f \in \Sigma$ by Lemma 6.10. Therefore $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, f(s_1, \ldots, s_n) \approx f(t_1, \ldots, t_n)\ [\varphi]$. (congruence)

All properties in order to form a congruence relation are fulfilled. □

Lemma 6.16 (stability of theory terms)**.** *Let $\mathfrak{T} = \langle M, E\rangle$ be a CE-theory. Let $X, Y \subseteq \mathcal{V}_{\mathsf{th}}$ be sets of theory variables and σ a substitution such that $\sigma(y) \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$ for any $y \in Y$. If $\mathfrak{T} \models_{\mathsf{cec}} \Pi Y.\, s \approx t\ [\varphi]$, then $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, s\sigma \approx t\sigma\ [\varphi\sigma]$.*

Proof. Consider a CE-theory $\mathfrak{T} = \langle M, E\rangle$, the sets $X, Y \subseteq \mathcal{V}_{\mathsf{th}}$ and the substitution σ such that $\sigma(y) \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$ for any $y \in Y$. Moreover, assume $\mathfrak{T} \models_{\mathsf{cec}} \Pi Y.\, s \approx t\ [\varphi]$ which gives $s\gamma \overset{*}{\leftrightarrow}_E t\gamma$ for all Y-valued substitutions γ such that $\models_M \varphi\gamma$. Let θ be an X-valued substitution such that $\models_M (\varphi\sigma)\theta$, i.e. $\models_M \varphi(\theta \circ \sigma)$. Take $y \in Y$. Then, we have $\sigma(y) \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$ by assumption, and thus $\theta(\sigma(y)) \in \mathcal{T}(\mathcal{F}_{\mathsf{th}})$ as θ is X-valued. Hence, we have $(\theta \circ \sigma)(y) \in \mathcal{T}(\mathcal{F}_{\mathsf{th}})$ for all $y \in Y$. Now, take for each $y \in Y$, the value $c_y = [\![(\theta \circ \sigma)(y)]\!]$. This gives a Y-valued substitution ξ by defining $\xi(y) = c_y$ for each $y \in Y$. From this we obtain $(\theta \circ \sigma)(y) \overset{*}{\leftrightarrow}_E \xi(y)$ for any $y \in Y$ by Lemma 6.3, and thus $s(\theta \circ \sigma) \overset{*}{\leftrightarrow}_E s\xi$ and $t(\theta \circ \sigma) \overset{*}{\leftrightarrow}_E t\xi$ by Lemma 6.10. Furthermore, we have $[\![\varphi(\theta \circ \sigma)]\!] = [\![\varphi\xi]\!]$ by Lemma 6.1, and thus, by $\models_M \varphi(\theta \circ \sigma)$, we obtain $\models_M \varphi\xi$. Hence ξ is a Y-valued substitution such that $\models_M \varphi\xi$, and by assumption we obtain $s\xi \overset{*}{\leftrightarrow}_E t\xi$. Thus, $(s\sigma)\theta = s(\theta \circ \sigma) \overset{*}{\leftrightarrow}_E s\xi \overset{*}{\leftrightarrow}_E t\xi \overset{*}{\leftrightarrow}_E t(\theta \circ \sigma) = (t\sigma)\theta$. We have shown for all X-valued substitutions θ with $\models_M (\varphi\sigma)\theta$ that $(s\sigma)\theta \overset{*}{\leftrightarrow}_E (t\sigma)\theta$ from which follows $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, s\sigma \approx t\sigma\ [\varphi\sigma]$. □

Lemma 6.17 (general stability)**.** *Let $\langle M, E\rangle$ be a CE-theory and σ a substitution such that $\mathcal{D}\mathsf{om}(\sigma) \cap X = \varnothing$. Then, if $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$ then $E \models_{\mathsf{cec}} \Pi X.\, s\sigma \approx t\sigma\ [\varphi]$.*

Proof. Assume $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$ and a substitution σ such that $\mathcal{D}\mathsf{om}(\sigma) \cap X = \varnothing$. Let δ be an X-valued substitution such that $\models_M \varphi\delta$. Consider the substitution $\gamma = \delta \circ \sigma$. By $\mathcal{D}\mathsf{om}(\sigma) \cap X = \varnothing$, we have $\gamma(x) = \delta(\sigma(x)) = \delta(x)$ for all $x \in X$. Thus, γ is X-valued and $\varphi\gamma = \varphi\delta$. From our assumption $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$, we obtain $s\gamma \overset{*}{\leftrightarrow}_E t\gamma$ and further $s\sigma\delta \overset{*}{\leftrightarrow}_E t\sigma\delta$. We have that $s\sigma\delta \overset{*}{\leftrightarrow}_E t\sigma\delta$ for any X-valued substitution with $\models_M \varphi\delta$, which concludes $E \models_{\mathsf{cec}} \Pi X.\, s\sigma \approx t\sigma\ [\varphi]$. □

Lemma 6.18 (model consequence)**.** *Let $\langle M, E\rangle$ be a CE-theory, $X \subseteq \mathcal{V}_{\mathsf{th}}$ a set of theory variables, $s, t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$, and φ a logical constraint over X. If $\models_M (\varphi\sigma \Rightarrow s\sigma = t\sigma)$ holds for all X-valued substitutions σ, then $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$.*

Proof. Consider a CE-theory $\mathfrak{T} = \langle M, E\rangle$, a set $X \subseteq \mathcal{V}_{\mathsf{th}}$, terms $s, t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$ and a logical constraint φ with $\mathcal{V}\mathsf{ar}(\varphi) \subseteq X$. Assume that $\models_M (\varphi\sigma \Rightarrow s\sigma = t\sigma)$ for all X-valued substitutions σ. Let σ be an X-valued substitution with $\models_M \varphi\sigma$. By our assumption, $\models_M s\sigma = t\sigma$ holds and thus also $[\![s\sigma]\!]_M = [\![t\sigma]\!]_M$. By Lemma 6.3, we obtain $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$. We have that $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$ for all X-valued substitution σ with $\models_M \varphi\sigma$. Therefore, $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$. □

Corollary 6.19. *Let $\langle M, E\rangle$ be a CE-theory, $X \subseteq \mathcal{V}_{\mathsf{th}}$ a set of theory variables, and $\varphi \Rightarrow s = t$ a logical constraint over X such that $\models_M (\varphi \Rightarrow s = t)$. Then, $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$.*

Proof. Let X be a set of theory variables and $\varphi \Rightarrow s = t$ a logical constraint over X. Suppose $\models_M (\varphi \Rightarrow s = t)$. Let σ be an X-valued substitution. Then, it follows from $\mathcal{V}\mathsf{ar}(\varphi, s, t) \subseteq X$ that $\mathcal{V}\mathsf{ar}(\varphi \Rightarrow s = t) \cap \mathcal{D}\mathsf{om}(\sigma) \subseteq X \cap \mathcal{D}\mathsf{om}(\sigma) \subseteq \mathcal{V}\mathcal{D}\mathsf{om}(\sigma)$. Thus, by Lemma 6.2, $\models_M (\varphi\sigma \Rightarrow s\sigma = t\sigma)$ holds. By Lemma 6.18 we obtain $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$. □

Theorem 6.20. *For a CE-theory $\langle M, E\rangle$, $s \overset{*}{\leftrightarrow}_E t$ if and only if $E \models_{\mathsf{cec}} \Pi\varnothing.\, s \approx t\ [\mathsf{true}]$.*

Proof. Note that $\varnothing$-valued substitutions refer to all substitutions and that $\sigma \models \mathsf{true}$ holds for any substitutions σ. Thus, it follows from the definitions that: $E \models_{\mathsf{cec}} \Pi\varnothing.\, s \approx t\ [\mathsf{true}]$ if and only if $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$ for any $\varnothing$-valued substitution σ such that $\sigma \models \mathsf{true}$ if and only if $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$ for any substitution σ. Now, since the relation $\overset{*}{\leftrightarrow}_E$ is closed under substitutions by Lemma 6.10, $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$ for any substitution σ if and only if $s \overset{*}{\leftrightarrow}_E t$. Therefore, the claim follows. □

Theorem 6.21. *Let $\langle M, E\rangle$ be a CE-theory, and $\Pi X.\, s \approx t\ [\varphi]$ a CE. Suppose $s \overset{*}{\leftrightarrow}_E s'$ and $t \overset{*}{\leftrightarrow}_E t'$ for some s', t' such that $\Pi X.\, s' \approx t'\ [\varphi]$ is trivial. Then, $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$.*

Proof. Let s', t' be terms such that $s \overset{*}{\leftrightarrow}_E s'$, $t \overset{*}{\leftrightarrow}_E t'$, and $\Pi X.\, s' \approx t'\ [\varphi]$ is trivial. Let σ be an arbitrary X-valued substitution such that $\models_M \varphi\sigma$. Then, from $s \overset{*}{\leftrightarrow}_E s'$ and $t \overset{*}{\leftrightarrow}_E t'$, we have $s\sigma \overset{*}{\leftrightarrow}_E s'\sigma$ and $t\sigma \overset{*}{\leftrightarrow}_E t'\sigma$. Furthermore, from the triviality of $\Pi X.\, s' \approx t'\ [\varphi]$, we know $s'\sigma = t'\sigma$. Thus, $s\sigma \overset{*}{\leftrightarrow}_E s'\sigma = t'\sigma \overset{*}{\leftrightarrow}_E t\sigma$. This concludes $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$. □

Full Proofs of Section 6.4

Lemma 6.23. *Let $\langle M, E\rangle$ be a CE-theory. If $E \vdash_{\mathbf{CEC}_0} \Pi X.\, s \approx t\ [\varphi]$, then $\Pi X.\, s \approx t\ [\varphi]$ is a CE.*

Proof. We prove this claim by induction on the derivation. We make a case analysis depending on the last rule that have been applied. We have to check that for the conclusion of the inference rule, say $E \vdash \Pi X.\, s \approx t\ [\varphi]$, the following conditions are met: (1) $s, t \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ have the same sort, (2) φ is a logical constraint, (3) $X \subseteq \mathcal{V}_{\mathsf{th}}$, and (4) $\mathcal{V}\mathsf{ar}(\varphi) \subseteq X$.

- Case where *Refl* is applied. (1) is trivial. (2), (3) are guaranteed by our convention on the inference rules (Definition 6.22). (4) is guaranteed by the side condition of the rule.
- Case where *Trans* is applied. (1) follows from the induction hypotheses. (2)–(4) follow from (one of) the induction hypotheses.
- Case where *Sym* is applied. (1)–(4) follow from the induction hypothesis.
- Case where *Cong* is applied. (1) immediately follows from the induction hypotheses. (2)–(4) follow from (one of) the induction hypotheses.
- Case where *Rule* is applied. (1)–(4) hold because E is an LCES (i.e. a set of CEs.)
- Case where *Theory Instance* is applied. (1) immediately follows from the induction hypothesis. (3) By our convention on the inference rules (Definition 6.22), X is a set of theory variables. (4) We have $\mathcal{V}\mathsf{ar}(\varphi) \subseteq Y$ by induction hypothesis, and by the side condition of the inference rule, we have $x\sigma \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$ for all

$x \in Y$. These together imply that $\mathcal{V}\mathsf{ar}(\varphi\sigma) \subseteq X$. (2) By the previous (3) and (4), $\varphi\sigma \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, \mathcal{V}_{\mathsf{th}})$. It remains to show that the sort of $\varphi\sigma$ is Bool, but hold as so is φ, and the substitution does not change the sort of terms.

- Case where *General Instance* is applied. (1)–(4) follow from the induction hypothesis.
- Case where *Weakening* is applied. (1) and (3) follow from the induction hypothesis. (2) follows from the side condition of the inference rule, as $\models_M (\varphi \Rightarrow \psi)$ guarantees that $\varphi \Rightarrow \psi$ is a logical constraint, hence so is φ. (4) is guaranteed by the side condition of the inference rule.
- Case where *Split* is applied. (1) and (3) follow from (one of) the induction hypotheses. (2) and (4) immediately follow from the induction hypotheses.
- Case where *Axiom* is applied. (1) We have that $s\sigma = t\sigma$ is a logical constraint, which is guaranteed by the side condition. Then, since $=$ is an abbreviation for $=^\tau : \tau \times \tau \to \mathsf{Bool}$ for some τ, the claim follows. (3) By our convention on the inference rules (Definition 6.22), X is a set of theory variables. (4) is guaranteed by the side condition of the inference rule. (2) We have that $\varphi\sigma$ is a logical constraint, which is guaranteed by the side condition. Thus, the only possibility of φ not being a logical constraint, is to have a term variable in it. But this possibility is excluded by (4).
- Case where *Abst* is applied. (1) immediately follows from the induction hypothesis. (2) follows from the side condition of the inference rule. (3) By our convention on the inference rules (Definition 6.22), X is a set of theory variables. (4) is guaranteed by the side condition of the inference rule.
- Case where *Enlarge* is applied. (1) and (2) follow from the induction hypothesis. (3) By our convention on the inference rules (Definition 6.22), X is a set of theory variables. (4) is guaranteed by the side condition of the inference rule.

$\square$

Theorem 6.27 (soundness of the system $\mathbf{CEC}_0$). *Let $\langle M, E\rangle$ be a CE-theory and $\Pi X.\, s \approx t\ [\varphi]$ a CE. If $E \vdash_{\mathbf{CEC}_0} \Pi X.\, s \approx t\ [\varphi]$, then $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$.*

Proof. We prove this claim by induction on the derivation. We make a case analysis depending on which inference rule is applied at the last derivation step.

- Case where *Refl* is applied. Suppose that $E \vdash \Pi X.\, s \approx s\ [\varphi]$ is derived as given in Figure 6.2. By Lemma 6.15, we have $E \models_{\mathsf{cec}} \Pi X.\, s \approx s\ [\varphi]$. This shows the claim.
- Case where *Trans* is applied. Suppose that $E \vdash \Pi X.\, s \approx u\ [\varphi]$ is derived from $E \vdash \Pi X.\, s \approx t\ [\varphi]$ and $E \vdash \Pi X.\, t \approx u\ [\varphi]$ as given in Figure 6.2. By the induction hypotheses, we have $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$ and $E \models_{\mathsf{cec}} \Pi X.\, t \approx u\ [\varphi]$. Then, it follows from Lemma 6.15 that $E \models_{\mathsf{cec}} \Pi X.\, s \approx u\ [\varphi]$.

- Case where *Sym* is applied. Suppose that $E \vdash \Pi X.\, s \approx t\ [\varphi]$ is derived from $E \vdash \Pi X.\, t \approx s\ [\varphi]$ as given in Figure 6.2. By the induction hypotheses, we have $E \models_{\mathsf{cec}} \Pi X.\, t \approx s\ [\varphi]$. Then, it follows from Lemma 6.15 that $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$.

- Case where *Cong* is applied. Suppose that $E \vdash \Pi X.\, f(s_1, \ldots, s_n) \approx f(t_1, \ldots, t_n)\ [\varphi]$ is derived from $E \vdash \Pi X.\, s_1 \approx t_1\ [\varphi], \ldots, E \vdash \Pi X.\, s_n \approx t_n\ [\varphi]$ as given in Figure 6.2. By the induction hypotheses, $E \models_{\mathsf{cec}} \Pi X.\, s_i \approx t_i\ [\varphi]$ for all $1 \leqslant i \leqslant n$. Then, it follows from Lemma 6.15 that $E \models_{\mathsf{cec}} \Pi X.\, f(s_1, \ldots, s_n) \approx f(t_1, \ldots, t_n)\ [\varphi]$.

- Case where *Rule* is applied. Suppose that $E \vdash \Pi X.\, \ell \approx r\ [\varphi]$ is derived as given in Figure 6.2. By the side condition, $(\Pi X.\, \ell \approx r\ [\varphi]) \in E$. Thus, one obtains from Lemma 6.14 that $E \models_{\mathsf{cec}} \Pi X.\, \ell \approx r\ [\varphi]$.

- Case where *Theory Instance* is applied. Suppose that $E \vdash \Pi X.\, s\sigma \approx t\sigma\ [\varphi\sigma]$ is derived from $E \vdash \Pi X.\, s \approx t\ [\varphi]$, as given in Figure 6.2. By the induction hypothesis, we have $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$. Then, because of the side condition, one can apply Lemma 6.16, so that we have $E \models_{\mathsf{cec}} \Pi X.\, s\sigma \approx t\sigma\ [\varphi\sigma]$.

- Case where *General Instance* is applied. Suppose that $E \vdash \Pi X.\, s\sigma \approx t\sigma\ [\varphi]$ is derived from $E \vdash \Pi X.\, s \approx t\ [\varphi]$, as given in Figure 6.2. By the induction hypothesis, we have $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$. Then, because of the side condition, one can apply Lemma 6.17, so that we have $E \models_{\mathsf{cec}} \Pi X.\, s\sigma \approx t\sigma\ [\varphi]$.

- Case where *Weakening* is applied. Suppose that $E \vdash \Pi X.\, s \approx t\ [\psi]$ is derived from $E \vdash \Pi X.\, s \approx t\ [\varphi]$ as given in Figure 6.2. Let σ be an X-valued substitution such that $\models_{\mathcal{M}} \varphi\sigma$. As we have $\models_{\mathcal{M}} (\varphi \Rightarrow \psi)$ by the side condition, we have $\models_{\mathcal{M}} (\varphi\sigma \Rightarrow \psi\sigma)$ (also because of $\mathcal{V}\mathsf{ar}(\varphi) \subseteq X$ and σ is X-valued), and hence $\models_{\mathcal{M}} \psi\sigma$. By the induction hypothesis, we have $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\psi]$, that is, $s\sigma' \xleftrightarrow{*}_E t\sigma'$ holds for any X-valued substitution such that $\models_{\mathcal{M}} \psi\sigma'$. Thus, one obtains $s\sigma \xleftrightarrow{*}_E t\sigma$. We now have shown that $s\sigma \xleftrightarrow{*}_E t\sigma$ holds for any X-valued substitution σ such that $\models_{\mathcal{M}} \varphi\sigma$, and this concludes $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$.

- Case where *Split* is applied. Suppose that $E \vdash \Pi X.\, s \approx t\ [\varphi \vee \psi]$ is derived from $E \vdash \Pi X.\, s \approx t\ [\varphi]$ and $E \vdash \Pi X.\, s \approx t\ [\psi]$ as given in Figure 6.2. Let σ be an X-valued substitution such that $\models_{\mathcal{M}} (\varphi \vee \psi)\sigma$, i.e. $\models_{\mathcal{M}} (\varphi\sigma \vee \psi\sigma)$. Then, $\models_{\mathcal{M}} \varphi\sigma$ or $\models_{\mathcal{M}} \psi\sigma$ holds (note here $\varphi\sigma, \psi\sigma$ are ground theory terms). By the induction hypotheses, we have $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$ and $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\psi]$, and thus, in either case, $s\sigma \xleftrightarrow{*}_E t\sigma$ holds. Therefore, we have that $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi \vee \psi]$.

- Case where *Axiom* is applied. Suppose that $E \vdash \Pi X.\, s \approx t\ [\varphi]$ is derived as given in Figure 6.2. By the side conditions, one can apply Lemma 6.18, so that $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$ holds.

- Case where *Abst* is applied. Suppose that $E \vdash \Pi X.\, s \approx t\ [\varphi]$ is derived from $E \vdash \Pi X.\, s\sigma \approx t\sigma\ [\varphi\sigma]$ as given in Figure 6.2. Let ρ be an X-valued substitution such that $\models_{\mathcal{M}} \varphi\rho$. As we have $\models_{\mathcal{M}} (\varphi \Rightarrow \bigvee_{x \in X} x = \sigma(x))$, we have $\models_{\mathcal{M}} (\varphi\rho \Rightarrow \bigvee_{x \in X} \rho(x) = \rho(\sigma(x)))$, and thus $\models_{\mathcal{M}} \bigvee_{x \in X} \rho(x) = \rho(\sigma(x)))$, i.e. $[\![\rho(x)]\!] = [\![\rho(\sigma(x))]\!]$

for any $x \in X$ (note here that $\rho(\sigma(x))$ is a ground theory term by the side condition $\bigcup_{x\in X} \mathcal{V}\mathsf{ar}(\sigma(x)) \subseteq X$). Thus, it follows from $\models_M \varphi\rho$ that $\models_M \varphi\sigma\rho$, as $\mathcal{V}\mathsf{ar}(\varphi) \subseteq X$. By the induction hypothesis, we have $E \models_{\mathsf{cec}} \Pi X.\, s\sigma \approx t\sigma\ [\varphi\sigma]$, and thus, $s\sigma\rho \xleftrightarrow{*}_E t\sigma\rho$ holds. Hence, from $[\![\rho(x)]\!] = [\![\rho(\sigma(x))]\!]$ for any $x \in X$ (and $\mathcal{V}\mathsf{ar}(s,t) \subseteq X$), one obtains $s\rho \xleftrightarrow{*}_E t\rho$. Therefore, we now have shown that $s\rho \xleftrightarrow{*}_E t\rho$ holds for any X-valued substitution ρ such that $\models_M \varphi\rho$, and this concludes $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$.

- Case where *Enlarge* is applied. Suppose that $E \vdash \Pi X.\, s \approx t\ [\varphi]$ is derived from $E \vdash \Pi Y.\, s \approx t\ [\varphi]$ as given in Figure 6.2. Let δ be an X-valued substitution such that $\models_M \varphi\delta$. Define a substitution δ' as follows: $\delta'(x^\tau) = c^\tau$ if $x \in Y \setminus X$, and $\delta'(x) = \delta(x)$ otherwise, where c^τ is (arbitrarily) taken from $\mathcal{V}\mathsf{al}^\tau$. Apparently, δ' is Y-valued by definition, as δ is X-valued. Furthermore, since $\mathcal{V}\mathsf{ar}(\varphi) \subseteq X$ by the side condition, we have $\varphi\delta' = \varphi\delta$, and thus, $\models_M \varphi\delta'$ also holds. By the induction hypothesis, we have $E \models_{\mathsf{cec}} \Pi Y.\, s \approx t\ [\varphi]$, and thus, $s\delta' \xleftrightarrow{*}_E t\delta'$. From the side condition that $\mathcal{V}\mathsf{ar}(s,t) \cap (Y \setminus X) = \varnothing$, we have $s\delta' = s\delta$ and $t\delta' = t\delta$. Thus, $s\delta \xleftrightarrow{*}_E t\delta$. Therefore, we now have shown that $s\delta \xleftrightarrow{*}_E t\delta$ holds for any X-valued substitution δ such that $\models_M \varphi\delta$, and this concludes $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$.

$\square$

Lemma 6.29. *Let $\langle M, E\rangle$ be a CE-theory and $\Pi X.\, s \approx t\ [\varphi]$ a CE such that φ is satisfiable. Suppose $s\sigma = t\sigma$ for all X-valued substitutions σ such that $\models_M \varphi\sigma$. Then $E \vdash \Pi X.\, s \approx t\ [\varphi]$.*

Proof. First, we consider the case $s, t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$. In this case, the claim is obtained easily from the *Axiom* rule as follows. Suppose σ is an X-valued substitution. If $\models_M \varphi\sigma$, then it follows from the assumption that $s\sigma = t\sigma$. Then, by $s, t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$ and $X \subseteq \mathcal{VD}\mathsf{om}(\sigma)$, we have $[\![s\sigma]\!] = [\![t\sigma]\!]$, and hence, $\models_M (\varphi\sigma \Rightarrow s\sigma = t\sigma)$ holds. Suppose $\not\models_M \varphi\sigma$. In this case, $\models_M (\varphi\sigma \Rightarrow s\sigma = t\sigma)$ holds trivially. All in all, $\models_M (\varphi\sigma \Rightarrow s\sigma = t\sigma)$ for any X-valued substitution σ. Therefore, by *Axiom* rule, $E \vdash \Pi X.\, s \approx t\ [\varphi]$.

Next, we consider the case $s = x \in \mathcal{V}$. The case $x \in X$ is included in the case above. So, assume $x \notin X$. By our assumption, we exclude that case that φ is not satisfiable. Let σ be a X-valued substitution such that $\models_M \varphi\sigma$. Furthermore, take σ_0 as a substitution given by: $\sigma_0(x) = x$ and $\sigma_0(y) = \sigma(y)$ for any $y \neq x$. Clearly, we have $\models_M \varphi\sigma_0$, as σ_0 and σ disagree only on the variable $x \notin X$ and $\mathcal{V}\mathsf{ar}(\varphi) \subseteq X$. Thus, by our condition, $x = x\sigma_0 = t\sigma_0$. Hence we know $t \in \mathcal{V}$. Similarly, one can take a substitution σ_1 as: $\sigma_1(x) = x'$ and $\sigma_0(y) = \sigma(y)$ for any $y \neq x$, where x' is a variable other than x. Then, again by our condition, $x' = x\sigma_1 = t\sigma_1$. Thus, we know from $x' \neq x$ that $t\sigma_1 \neq t\sigma_0$. From this, we obtain $t = x$, as σ_0 and σ_1 disagree only on the term variable x. Since $E \vdash \Pi X.\, x \approx x\ [\varphi]$ by the *Refl* rule, the claim follows.

Now we proceed to the general case. By our assumption, we exclude the case where φ is not satisfiable. Let σ be a X-valued substitution such that $\models_M \varphi\sigma$. Then, from the assumption we have $s\sigma = t\sigma$, and hence, we know that s, t are unifiable, and thus, there is a multi-hole context C such that $s = C[s_1, \ldots, s_n]$ and $t = C[t_1, \ldots, t_n]$ such that either one of s_i or t_i is a variable for each $1 \leqslant i \leqslant n$. By $C\sigma[s_1\sigma, \ldots, s_n\sigma] = s\sigma = t\sigma = C\sigma[t_1\sigma, \ldots, t_n\sigma]$, we know $s_i\sigma = t_i\sigma$ for each $1 \leqslant i \leqslant n$. Thus, by our claim above for the case $s \in \mathcal{V}$, and the *Sym* rule, we know that $E \vdash \Pi X.\, s_i \approx t_i\ [\varphi]$ for each $1 \leqslant i \leqslant n$. Then, using *Refl*, *Trans*, *Cong* rules, it is straightforward to obtain the desired $E \vdash \Pi X.\, s \approx t\ [\varphi]$. □

Lemma 6.30. *Let $\langle M, E\rangle$ be a CE-theory and $\Pi X.\, s \approx t\ [\varphi]$ a CE such that φ is satisfiable. Suppose $s\sigma \leftrightarrow^{=}_{\mathsf{calc}} t\sigma$ for all X-valued substitutions σ such that $\mathcal{D}\mathsf{om}(\sigma) = X$ and $\models_M \varphi\sigma$. Then $E \vdash \Pi X.\, s \approx t\ [\varphi]$.*

Proof. We first show the case that $s, t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$. Let σ be any X-valued substitution such that $\models_M \varphi\sigma$. Then, $s\sigma, t\sigma \in \mathcal{T}(\mathcal{F}_{\mathsf{th}})$, and either $s\sigma = t\sigma$ or and $s\sigma = C[f(v_1, \ldots, v_n)]$ and $t\sigma = C[v_0]$ (or vice versa) for some context C and $f \in \mathcal{F}_{\mathsf{th}}$ and $v_0, \ldots, v_n \in \mathcal{V}\mathsf{al}$ such that $\mathcal{I}(f)(v_1, \ldots, v_n) = v_0$. Then, clearly, $\models_M s\sigma = t\sigma$. Thus, $\models_M (\varphi\sigma \Rightarrow s\sigma = t\sigma)$. Hence, using the *Axiom* rule, it follows that $E \vdash \Pi X.\, s \approx t\ [\varphi]$. If $s\sigma = t\sigma$ for all X-valued substitutions σ, then the claim follows from Lemma 6.29. Thus, suppose there exists an X-valued substitution σ such that $s\sigma \leftrightarrow_{\mathsf{calc}} t\sigma$, $\models_M \varphi\sigma$ and $X = \mathcal{D}\mathsf{om}(\sigma)$. Then, we have $s\sigma = C[f(v_1, \ldots, v_n)]_q$ and $t\sigma = C[v_0]_q$ for some context C and $f \in \mathcal{F}_{\mathsf{th}}$ and $v_0, \ldots, v_n \in \mathcal{V}\mathsf{al}$ such that $\mathcal{I}(f)(v_1, \ldots, v_n) = v_0$ (or vice versa). Let $p \in \mathcal{P}\mathsf{os}(s)$ be the position such that $s|_p$ is the minimal subterm of s such that $f(v_1, \ldots, v_n) \trianglelefteq s|_p\sigma$. Then, because $\mathcal{D}\mathsf{om}(\sigma) = X$ and σ is X-valued, we have $s|_p = f(s_1, \ldots, s_n)$ with $s_1, \ldots, s_n \in \mathcal{V}\mathsf{al} \cup X$. Hence, $p = q$. Let $p' \in \mathcal{P}\mathsf{os}(t)$ be the position such that $t|_{p'}$ is the minimal subterm of t such that $v_0 \trianglelefteq t|_{p'}\sigma$. Then, because $\mathcal{D}\mathsf{om}(\sigma) = X$ and σ is X-valued, we have $t|_{p'} \in \mathcal{V}\mathsf{al} \cup X$. Hence $p' = q$. Thus, we have $p = p' = q$. Furthermore, $s|_q, t|_q \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$. Thus, it follows from the previous arguments that $E \vdash \Pi X.\, s|_q \approx t|_q\ [\varphi]$. Now, we have $s = C_1[f(s_1, \ldots, s_n)]_q$ and $t = C_2[t_0]_q$ for some $s_1, \ldots, s_n, t_0 \in \mathcal{V}\mathsf{al} \cup X$ and contexts C_1, C_2. By our assumption, we have that $s\sigma = C_1\sigma[f(s_1\sigma, \ldots, s_n\sigma)]_q \leftrightarrow^{=}_{\mathsf{calc}} C_2\sigma[t_0\sigma]_q$ for all X-valued substitutions σ such that $\mathcal{D}\mathsf{om}(\sigma) = X$ and $\models_M \varphi\sigma$. Since $f(s_1\sigma, \ldots, s_n\sigma) = t_0\sigma$ never holds for such substitutions σ, we have that for each position $q' \not\leqslant q$, $C_1\sigma|_{q'} = C_2\sigma|_{q'}$ holds for any X-valued substitution σ such that $\mathcal{D}\mathsf{om}(\sigma) = X$ and $\models_M \varphi\sigma$. Thus, by Lemma 6.29, $E \vdash \Pi X.\, s|_{q'} \approx t|_{q'}\ [\varphi]$ holds for any $q' \in \mathcal{P}\mathsf{os}(s) \cap \mathcal{P}\mathsf{os}(t)$ such that $q' \not\leqslant q$. Now, using *Cong* and *Trans* rules, we have $E \vdash \Pi X.\, s \approx t\ [\varphi]$. □

Full Proofs of Section 6.5

Lemma 6.40 (soundness with respect to conversion)**.** *Let $\mathfrak{T} = \langle M, E\rangle$ be a CE-theory over a signature Σ, and $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J}\rangle$ a CE-$\langle \Sigma, M\rangle$-algebra such that $\models_{\mathfrak{M}} E$. If $s \overset{*}{\leftrightarrow}_E t$ then $[\![s]\!]_{\mathfrak{M},\rho} = [\![t]\!]_{\mathfrak{M},\rho}$ for any valuation ρ on $\mathfrak{M}$.*

Proof. Clearly, it suffices to show that $s \leftrightarrow_E t$ implies $[\![s]\!]_{\mathfrak{M},\rho} = [\![t]\!]_{\mathfrak{M},\rho}$ for any valuation ρ on $\mathfrak{M}$. Moreover, by Lemma 6.37, it suffices to consider the case where $s \leftrightarrow_E t$ is a root step. Let $\Sigma = \langle \mathcal{S}_{\mathsf{th}}, \mathcal{S}_{\mathsf{te}}, \mathcal{F}_{\mathsf{th}}, \mathcal{F}_{\mathsf{te}}\rangle$ and $M = \langle \mathcal{I}, \mathcal{J}\rangle$. First, we consider the case $s \leftrightarrow_{\mathsf{calc}} t$. Then, $s = f(s_1, \ldots, s_n)$, $s_1, \ldots, s_n, t \in \mathcal{V}\mathsf{al}$, $f \in \mathcal{F}_{\mathsf{th}}$, and $\mathcal{I}(f)(s_1, \ldots, s_n) = t$ (or vice versa). Thus, $s, t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}})$, and we have $[\![s]\!]_M = [\![t]\!]_M$, and hence $[\![s]\!]_{\mathfrak{M}} = [\![t]\!]_{\mathfrak{M}}$

follows by Lemma 6.38. By $s, t \in \mathcal{T}(\mathcal{F}_{\mathsf{th}})$, we have $[\![s]\!]_{\mathfrak{M},\rho} = [\![t]\!]_{\mathfrak{M},\rho}$ for any valuation ρ. Next, we consider the case $s \leftrightarrow_{\mathsf{rule},E} t$. Then, $s = \ell\sigma$ and $t = r\sigma$ for some $\Pi X.\, \ell \approx r\ [\varphi] \in E$ and an (X-valued) substitution σ such that $\sigma(x) \in \mathcal{I}(\tau)$ for $x^\tau \in X$ and $\models_M \varphi\sigma$. Let ξ be a valuation on $\mathfrak{M}$ such that $\xi(x) = \sigma(x)$ for each $x \in X$. Then, as $\varphi \in \mathcal{T}(\mathcal{F}_{\mathsf{th}}, X)$, by Lemma 6.38, we obtain $[\![\varphi]\!]_{\mathfrak{M},\xi} = [\![\varphi]\!]_{M,\sigma} = \mathsf{true}$. Thus, by our assumption that $\models_{\mathfrak{M}} E$, it follows that $[\![\ell]\!]_{\mathfrak{M},\xi} = [\![r]\!]_{\mathfrak{M},\xi}$. Thus, we have $[\![\ell]\!]_{\mathfrak{M},\xi} = [\![r]\!]_{\mathfrak{M},\xi}$ for any valuation ξ such that $\xi(x) = \sigma(x)$ for each $x \in X$. Now, define a valuation $[\![\sigma]\!]_{\mathfrak{M},\rho}$ on $\mathfrak{M}$ by $[\![\sigma]\!]_{\mathfrak{M},\rho}(y) = [\![\sigma(y)]\!]_{\mathfrak{M},\rho}$ for any $y \in \mathcal{V}$. Then, it can be shown by induction on u that $[\![u\sigma]\!]_{\mathfrak{M},\rho} = [\![u]\!]_{\mathfrak{M},[\![\sigma]\!]_{\mathfrak{M},\rho}}$ for any term $u \in \mathcal{T}(\Sigma, \mathcal{V})$, similarly to Lemma 6.1. Furthermore, for $x \in X$, as $\sigma(x) \in \mathcal{I}(\tau)$, $[\![\sigma]\!]_{\mathfrak{M},\rho}(x) = [\![\sigma(x)]\!]_{\mathfrak{M},\rho} = \sigma(x)$ holds. Thus, $[\![s]\!]_{\mathfrak{M},\rho} = [\![\ell]\!]_{\mathfrak{M},[\![\sigma]\!]_{\mathfrak{M},\rho}} = [\![r]\!]_{\mathfrak{M},[\![\sigma]\!]_{\mathfrak{M},\rho}} = [\![t]\!]_{\mathfrak{M},\rho}$. □

Theorem 6.41 (soundness with respect to CE-validity). *Let $\mathfrak{T}$ be a CE-theory. If $\mathfrak{T} \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$, then $\mathfrak{T} \models \Pi X.\, s \approx t\ [\varphi]$.*

Proof. Suppose that $\mathfrak{T} = \langle M, E\rangle$ is a CE-theory over a signature Σ. Let $M = \langle \mathcal{I}, \mathcal{J}\rangle$. Suppose moreover $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$. We show that for any CE-$\langle\Sigma, M\rangle$-algebra $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J}\rangle$ such that $\models_{\mathfrak{M}} E$, we have $\models_{\mathfrak{M}} \Pi X.\, s \approx t\ [\varphi]$. Suppose $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J}\rangle$ is a CE-$\langle\Sigma, M\rangle$-algebra such that $\models_{\mathfrak{M}} E$. Then, by the definition of CE-validity, $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$ holds for any X-valued substitution σ such that $\models_M \varphi\sigma$. Let ρ be a valuation over $\mathfrak{M}$ satisfying the constraints φ and $\rho(x) \in \mathcal{I}(\tau)$ holds for all $x^\tau \in X$. Now, let $\hat{\rho}$ be a valuation that is obtained from ρ by restricting its domain to X. Then since $\rho(x) \in \mathcal{I}(\tau) \cong \mathcal{V}\mathsf{al}^\tau$ for all $x \in X$, $\hat{\rho}$ is a substitution such that $\mathcal{D}\mathsf{om}(\hat{\rho}) = \mathcal{VD}\mathsf{om}(\hat{\rho}) = X$. Furthermore, as $\mathcal{V}\mathsf{ar}(\varphi) \subseteq X$, we have by Lemma 6.38 that $\models_M \varphi\hat{\rho}$. Thus, $\hat{\rho}$ is a X-valued substitution satisfying $\models_M \varphi\hat{\rho}$, and thus, by our assumption, $s\hat{\rho} \overset{*}{\leftrightarrow}_E t\hat{\rho}$ holds. Thus, by Lemma 6.40, $[\![s\hat{\rho}]\!]_{\mathfrak{M},\tau} = [\![t\hat{\rho}]\!]_{\mathfrak{M},\tau}$ holds for any valuation τ. This means that for any extension τ' of $\hat{\rho}$, $[\![s]\!]_{\mathfrak{M},\tau'} = [\![t]\!]_{\mathfrak{M},\tau'}$. In particular, one can take $\tau' := \rho$. Then $[\![s]\!]_{\mathfrak{M},\rho} = [\![t]\!]_{\mathfrak{M},\rho}$ holds. Thus we have now shown that $[\![s]\!]_{\mathfrak{M},\rho} = [\![t]\!]_{\mathfrak{M},\rho}$ holds, for all valuations ρ over $\mathfrak{M}$ satisfying the constraints φ and $\rho(x) \in \mathcal{I}(\tau)$ holds for all $x^\tau \in X$. Therefore, we conclude $\models_{\mathfrak{M}} \Pi X.\, s \approx t\ [\varphi]$. □

Lemma 6.45 (quotient algebra). *Let $\mathfrak{M}$ be a CE-$\langle\Sigma, M\rangle$-algebra, and $\sim$ a congruence on it. Then $\mathfrak{M}/{\sim}$ is a CE-$\langle\Sigma, M\rangle$-algebra.*

Proof. Suppose $\Sigma = \langle S_{\mathsf{th}}, S_{\mathsf{te}}, \mathcal{F}_{\mathsf{th}}, \mathcal{F}_{\mathsf{te}}\rangle$ and $M = \langle \mathcal{I}, \mathcal{J}\rangle$. Let $\mathfrak{M}/{\sim} = \langle \mathfrak{I}', \mathfrak{J}'\rangle$. Then $\mathfrak{I}', \mathfrak{J}'$ are well-defined as we mentioned above. By the definition of $\sim = (\sim^\tau)_{\tau\in S}$, the domain and codomain of $\mathfrak{J}'(f)$ respect the sort of function symbols f. Lastly, since $\sim^\tau \cap\, \mathcal{I}(\tau)^2$ is the identify relation for $\tau \in S_{\mathsf{th}}$, we obtain that $\mathfrak{M}/{\sim}$ restricted to $(\mathcal{I}(\tau))_{\tau\in S_{\mathsf{th}}}$ and $\mathcal{F}_{\mathsf{th}}$ is isomorphic to M. Therefore, $\mathfrak{M}/{\sim}$ is again a CE-$\langle\Sigma, M\rangle$-algebra. □

Lemma 6.47. *Let $\Sigma = \langle S_{\mathsf{th}}, S_{\mathsf{te}}, \mathcal{F}_{\mathsf{th}}, \mathcal{F}_{\mathsf{te}}\rangle$ be a signature, M a model over S_{th} and $\mathcal{F}_{\mathsf{th}}$, and U a set of variables. Then, the term algebra $T[M](\Sigma, U)$ is a CE-$\langle\Sigma, M\rangle$-algebra.*

Proof. Let $M = \langle \mathcal{I}, \mathcal{J}\rangle$ and $T[M](\Sigma, U) = \langle \mathfrak{I}, \mathfrak{J}\rangle$. First, we show $\mathcal{I}(\tau) \subseteq \mathfrak{I}(\tau)$ for any $\tau \in S_{\mathsf{th}}$. Let $u, v \in \mathcal{V}\mathsf{al}^\tau$. Then $u \sim_{\mathrm{c}} v$ implies $u = v$ by Lemma 6.3. Hence for any $u, v \in \mathcal{V}\mathsf{al}^\tau$, $[u]_{\mathrm{c}} = [v]_{\mathrm{c}}$ if and only if $u = v$. Thus, we have $\mathcal{I}(\tau) = \mathcal{V}\mathsf{al}^\tau \cong \{[v]_{\mathrm{c}} \mid v \in \mathcal{V}\mathsf{al}^\tau\} \subseteq$

$\mathcal{T}(\Sigma, U)^\tau/{\sim_\mathrm{c}}$. Here, $A \cong B$ denotes that sets A, B are isomorphic (and isomorphic sets are identified). Thus, $\mathcal{I}(\tau) \subseteq \mathfrak{I}(\tau)$ (up to isomorphism). Next, let $v_1, \ldots, v_n \in \mathcal{V}\mathsf{al}$ and $\mathcal{J}(f)(v_1, \ldots, v_n) = v_0$. Then, by $f(v_1, \ldots, v_n) \leftrightarrow^*_\mathsf{calc} v_0$, we have $f(v_1, \ldots, v_n) \sim_\mathrm{c} v_0$. Therefore, $\mathfrak{J}(f)([v_1]_\mathrm{c}, \ldots, [v_n]_\mathrm{c}) = [f(v_1, \ldots, v_n)]_\mathrm{c} = [v_0]_\mathrm{c} = v_0 = \mathcal{J}(f)(v_1, \ldots, v_n)$. Note that v_0 and $[v_0]_\mathrm{c}$ are identified based on $\mathcal{V}\mathsf{al}^\tau \cong \{[v]_\mathrm{c} \mid v \in \mathcal{V}\mathsf{al}^\tau\}$. □

Lemma 6.49. *Let $\mathfrak{T} = \langle M, E\rangle$ be a value-consistent CE-theory over a signature Σ, and U a set of variables. For any $[s]_\mathrm{c}, [t]_\mathrm{c} \in T[M](\Sigma, U)$, let $\sim_E = \{\langle [s]_\mathrm{c}, [t]_\mathrm{c}\rangle \mid s \overset{*}{\leftrightarrow}_E t\}$. Then, $\sim_E$ is a congruence relation on the term algebra $T[M](\Sigma, U)$.*

Proof. Note first that $\sim_E$ is well-defined because one has always $\leftrightarrow^*_\mathsf{calc} \subseteq \overset{*}{\leftrightarrow}_E$. Let $\Sigma = \langle S_\mathsf{th}, S_\mathsf{te}, \mathcal{F}_\mathsf{th}, \mathcal{F}_\mathsf{te}\rangle$, $M = \langle \mathcal{I}, \mathcal{J}\rangle$, and $T[M](\Sigma, U) = \langle \mathfrak{I}, \mathfrak{J}\rangle$. Since $s \overset{*}{\leftrightarrow}_E t$ implies terms s, t have the same sort, and thus one can regard $\sim_E$ as the sum of τ-indexed $\sim_E^\tau$ with $\tau \in S$. First, since $\overset{*}{\leftrightarrow}_E$ is an equivalence relation, $\sim_E$ is an equivalence relation. Next, we show that $\sim_E^\tau \cap \mathcal{I}(\tau)^2$ equals the identity relation for $\tau \in S_\mathsf{th}$. For this, let $\tau \in S_\mathsf{th}$ and suppose $[u]_\mathrm{c} \sim_E^\tau [v]_\mathrm{c}$ with $u, v \in \mathcal{I}(\tau) \cong \mathcal{V}\mathsf{al}^\tau$. Then, we have $u \overset{*}{\leftrightarrow}_E v$ by the definition of $\sim_E$, and by consistency with respect to values of the theory $\mathfrak{T}$, we obtain $u = v$ as $u, v \in \mathcal{V}\mathsf{al}$. Hence, $[u]_\mathrm{c} = [v]_\mathrm{c}$. Finally, assume $f \in \mathcal{F}$ and $[s_i]_\mathrm{c} \sim_E [t_i]_\mathrm{c}$ for $1 \leqslant i \leqslant n$. Then $s_i \overset{*}{\leftrightarrow}_E t_i$ for $1 \leqslant i \leqslant n$ by the definition of $\sim_E$. Hence $f(s_1, \ldots, s_n) \overset{*}{\leftrightarrow}_E f(t_1, \ldots, t_n)$. Thus, $\mathfrak{J}(f)([s_1]_\mathrm{c}, \ldots, [s_n]_\mathrm{c}) = [f(s_1, \ldots, s_n)]_\mathrm{c} \sim_E [f(t_1, \ldots, t_n)]_\mathrm{c} = \mathfrak{J}(f)([t_1]_\mathrm{c}, \ldots, [t_n]_\mathrm{c})$. □

Lemma 6.50. *Let $\mathfrak{T} = \langle M, E\rangle$ be a value-consistent CE-theory over a signature Σ. Then, the quotient $\mathcal{T}_E = T[M](\Sigma, \mathcal{V})/{\sim_E}$ of the term algebra is a CE-$\langle \Sigma, M\rangle$-algebra. Furthermore, both of the following hold:*

1. *$\models_{\mathcal{T}_E} \Pi X.\, s \approx t\ [\varphi]$ if and only if $E \models_\mathsf{cec} \Pi X.\, s \approx t\ [\varphi]$, and*
2. *$\models_{\mathcal{T}_E} E$.*

Proof. From our assumptions and Lemmas 6.45 and 6.49, it immediately follows that $\mathcal{T}_E$ is a CE-$\langle \Sigma, M\rangle$-algebra. Furthermore, the item *2* follows from item *1*, as $E \models_\mathsf{cec} E$ clearly holds. Thus, it remains to show claim *1.* of this lemma. Let $M = \langle \mathcal{I}, \mathcal{J}\rangle$ and $\mathcal{T}_E = \langle \mathfrak{I}, \mathfrak{J}\rangle$. Let us below also abbreviate $[[t]_\mathrm{c}]_{\sim_E}$ as $[t]_E$. We first show the following claim.

> **Claim**: Let σ, ρ be a substitution and a valuation on $\mathcal{T}_E$, respectively, such that $\rho(x) = [\sigma(x)]_E$. Then, we have $[u\sigma]_E = [\![u]\!]_{\mathcal{T}_E,\rho}$ for any term u.
>
> *Proof of the claim.* The proof proceeds by structural induction on u. The case $u = x \in \mathcal{V}$ follows as $[x\sigma]_E = [\sigma(x)]_E = \rho(x) = [\![x]\!]_\rho$. Consider the remaining case $u \notin \mathcal{V}$. Suppose $u = f(u_1, \ldots, u_n)$ with $f \in \mathcal{F}$. Then, using the induction hypotheses, we have $[f(u_1, \ldots, u_n)\sigma]_E = [f(u_1\sigma, \ldots, u_n\sigma)]_E = \mathfrak{J}(f)([u_1\sigma]_E, \ldots, [u_n\sigma]_E) = \mathfrak{J}(f)([\![u_1]\!]_\rho, \ldots, [\![u_n]\!]_\rho) = [\![f(u_1, \ldots, u_n)]\!]_\rho$. □

We now proceed to show item *1.* ($\Rightarrow$) Suppose $\models_{\mathcal{T}_E} \Pi X.\, s \approx t\ [\varphi]$. That is, $[\![s]\!]_{\mathcal{T}_E,\rho} = [\![t]\!]_{\mathcal{T}_E,\rho}$, for any valuation ρ over $\mathcal{T}_E$ satisfying the constraints φ and $\rho(x) \in \mathcal{I}(\tau)$ holds for all $x \in X$. To show $E \models_\mathsf{cec} \Pi X.\, s \approx t\ [\varphi]$, let us take an X-valued substitution σ

such that $\models_M \varphi\sigma$. We are now going to show $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$ holds. Take a valuation ρ on $\mathcal{T}_E$ as $\rho(x) = [\sigma(x)]_E$. Then, because σ is X-valued, $\rho(x) \in \mathcal{I}(\tau)$ holds for all $x^\tau \in X$. Furthermore, as $\mathcal{V}\mathsf{ar}(\varphi) \subseteq X$, $\models_M \varphi\rho$ follows from $\models_M \varphi\sigma$. Thus, $\llbracket s \rrbracket_{\mathcal{T}_E,\rho} = \llbracket t \rrbracket_{\mathcal{T}_E,\rho}$. Hence, $[s\sigma]_E = [t\sigma]_E$ by the claim above, and therefore, $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$. ($\Leftarrow$) Suppose $E \models_{\mathsf{cec}} \Pi X.\, s \approx t\ [\varphi]$. That is, $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$ holds, for any X-valued substitution σ such that $\models_M \varphi\sigma$. To show $\models_{\mathcal{T}_E} \Pi X.\, s \approx t\ [\varphi]$, let us take a valuation ρ over $\mathcal{T}_E$ satisfying the constraints φ and $\rho(x) \in \mathcal{I}(\tau)$ holds for all $x \in X$. We now going to show $\llbracket s \rrbracket_{\mathcal{T}_E,\rho} = \llbracket t \rrbracket_{\mathcal{T}_E,\rho}$. Since $X \subseteq \mathcal{V}_{\mathsf{th}}$, we have $\tau \in S_{\mathsf{th}}$ for each $x^\tau \in X$. Thus, there exists a value $v_x \in \mathcal{V}\mathsf{al}^\tau$ such that $[v_x]_E = \rho(x) \in \mathcal{I}(\tau)$. Take a substitution σ in such a way that $\sigma(x) = v_x$ for each $x \in X$. Clearly, σ is X-valued. Furthermore, since $\llbracket \varphi \rrbracket_{M,\rho} = \mathsf{true}$ by Lemma 6.38, we have $\models_M \varphi\sigma$ by Lemma 6.1. Thus, by our assumption, $s\sigma \overset{*}{\leftrightarrow}_E t\sigma$ holds. That is, $[s\sigma]_E = [t\sigma]_E$, and therefore $\llbracket s \rrbracket_{\mathcal{T}_E,\rho} = \llbracket t \rrbracket_{\mathcal{T}_E,\rho}$ by the claim above. Therefore, $\mathcal{T}_E \models \Pi X.\, s \approx t\ [\varphi]$. ☐

Lemma 6.51. *A CE-theory $\mathfrak{T}$ is consistent if and only if it is consistent with respect to values.*

Proof. ($\Rightarrow$) Suppose that $\mathfrak{T} = \langle M, E \rangle$ is a CE-theory over a signature Σ and let $M = \langle \mathcal{I}, \mathcal{J} \rangle$. Suppose that $\mathfrak{T}$ is consistent. Then, there exists a CE-$\langle \Sigma, M \rangle$-algebra $\mathfrak{M} = \langle \mathfrak{I}, \mathfrak{J} \rangle$ such that $\models_{\mathfrak{M}} E$. Suppose $u, v \in \mathcal{V}\mathsf{al}^\tau$ with $u \overset{*}{\leftrightarrow}_E v$. By the definition of the CE-$\langle \Sigma, M \rangle$-algebras, we have $\mathcal{I}(\tau) \subseteq \mathfrak{I}(\tau)$. Also, by $\models_{\mathfrak{M}} E$ and $u \overset{*}{\leftrightarrow}_E v$, we have $\llbracket u \rrbracket_{\mathfrak{M}} = \llbracket v \rrbracket_{\mathfrak{M}}$ by Lemma 6.40. Therefore, by $u, v \in \mathcal{V}\mathsf{al}^\tau \cong \mathcal{I}(\tau) \subseteq \mathfrak{I}(\tau)$, we have $u = \llbracket u \rrbracket_M = \llbracket u \rrbracket_{\mathfrak{M}} = \llbracket v \rrbracket_{\mathfrak{M}} = \llbracket v \rrbracket_M = v$. ($\Leftarrow$) By Lemma 6.50, $\mathcal{T}_E$ is a model of $\mathfrak{T}$. This witnesses that $\mathfrak{T}$ is consistent. ☐

D Appendix to Chapter 7

In this appendix we state the closing conditions on (parallel) critical pairs that are used in the confluence results implemented in crest and we present the proofs of Lemmata 7.5 and 7.8.

Definition D.1. A constrained critical pair $s \approx t\ [\varphi]$ is *strongly closed* if

1. $s \approx t\ [\varphi] \ \overset{\sim}{\rightarrow}^{*}_{\geqslant 1} \cdot \overset{\sim}{\rightarrow}^{=}_{\geqslant 2}\ u \approx v\ [\psi]$ for some trivial $u \approx v\ [\psi]$, and

2. $s \approx t\ [\varphi] \ \overset{\sim}{\rightarrow}^{*}_{\geqslant 2} \cdot \overset{\sim}{\rightarrow}^{=}_{\geqslant 1}\ u \approx v\ [\psi]$ for some trivial $u \approx v\ [\psi]$.

An LCTRS is strongly closed if all its constrained critical pairs are strongly closed.

Definition D.2. A constrained critical pair $s \approx t\ [\varphi]$ is *development closed* if $s \approx t\ [\varphi] \ \overset{\sim}{\multimap}_{\geqslant 1}\ u \approx v\ [\psi]$ for some trivial $u \approx v\ [\psi]$. A constrained critical pair is *almost development closed* if it is not an overlay and development closed, or it is an overlay and $s \approx t\ [\varphi] \ \overset{\sim}{\multimap}_{\geqslant 1} \cdot \overset{\sim}{\rightarrow}^{*}_{\geqslant 2}\ u \approx v\ [\psi]$ for some trivial $u \approx v\ [\psi]$. An LCTRS is called (almost) development closed if all its constrained critical pairs are (almost) development closed.

In the following we denote for a term s, a set of parallel positions P in s, and a set of terms $\{t_p\}_{p \in P}$ by $s[t_p]_{p \in P}$ the simultaneous replacement of $s|_p$ in s by t_p for all $p \in P$. The notion $\mathcal{TV}\mathsf{ar}(s, \varphi, P)$ in the variable condition of the next definition expands to $\bigcup_{p \in P} \mathcal{V}\mathsf{ar}(s|_p) \setminus \mathcal{V}\mathsf{ar}(\varphi)$.

Definition D.3. A constrained critical pair $s \approx t\ [\varphi]$ is *1-parallel closed* if $s \approx t\ [\varphi] \overset{\sim}{\mathrel{+\!\!\!\!\!\rightarrow}}_{\geqslant 1} \cdot \overset{\sim}{\rightarrow}^*_{\geqslant 2} u \approx v\ [\psi]$ for some trivial $u \approx v\ [\psi]$. An LCTRS is 1-parallel closed if all its constrained critical pairs are 1-parallel closed. A constrained parallel critical pair $\ell\sigma[r_p\sigma]_{p \in P} \approx r\sigma\ [\varphi]$ is *2-parallel closed* if there exists a set of parallel positions Q such that

$$\ell\sigma[r_p\sigma]_{p \in P} \approx r\sigma\ [\varphi] \overset{\sim}{\mathrel{+\!\!\!\!\!\rightarrow}}^{Q}_{\geqslant 2} \cdot \overset{\sim}{\rightarrow}^*_{\geqslant 1} u \approx v\ [\psi]$$

for some trivial $u \approx v\ [\psi]$ and $\mathcal{TV}\mathsf{ar}(v, \psi, Q) \subseteq \mathcal{TV}\mathsf{ar}(\ell\sigma, \varphi, P)$. An LCTRS is 2-parallel closed if all its constrained parallel critical pairs are 2-parallel closed. An LCTRS is parallel closed if it is 1-parallel closed and 2-parallel closed.

Proof of Lemma 7.5. Assume an LCTRS $\mathcal{R}$ and a constrained critical pair $s \approx t\ [\varphi] \in \mathsf{CCP}(\mathcal{R})$. Further assume that it rewrites to the non-trivial normal form $u \approx v\ [\varphi]$. There exists a substitution $\sigma \vDash \varphi$ such that $t\sigma \neq u\sigma$ and $t\sigma \approx u\sigma$ is a normal form. By definition of constrained critical pair there exists a term s with $t\sigma\ {}^*\!\!\leftarrow s \rightarrow^* u\sigma$, which shows non-confluence of $\mathcal{R}$. □

Proof of Lemma 7.8. Assume an LCTRS $\mathcal{R}$ and let $t\ {}_{\mathcal{R}}\!\!\leftarrow s \rightarrow_{\mathcal{R}} u$. By the critical pair lemma [88, Lemma 20] we have either $t \downarrow_{\mathcal{R}} u$ or $t \leftrightarrow_{\mathsf{CCP}(\mathcal{R})} u$. It remains to show that for terms $s_1, s_2 \in \mathcal{T}(\mathcal{F}, \mathcal{V})$ if $s_1 \leftrightarrow_{s \approx t\, [\varphi]} s_2$ then either $s_1 \leftrightarrow_{s \approx t\, [\varphi \wedge \psi]} s_2$ or $s_1 \leftrightarrow_{s \approx t\, [\varphi \wedge \neg\psi]} s_2$. So suppose $s_1 \leftrightarrow_{s \approx t\, [\varphi]} s_2$. There exist a context C and a substitution σ such that $s_1 = C[s\sigma]$, $s_2 = C[t\sigma]$ and $\sigma \vDash \varphi$. From $\mathcal{V}\mathsf{ar}(\psi) \subseteq \mathcal{V}\mathsf{ar}(\varphi)$ we obtain $\sigma \vDash \varphi \wedge \psi$ or $\sigma \vDash \varphi \wedge \neg\psi$. Hence $t \downarrow_{\mathcal{R}} u$ or $t \leftrightarrow_{\mathsf{CCP}^{\psi}_{\rho}(\mathcal{R})} u$. □

ABSTRACT

Term rewriting serves as a simple but powerful model of computation that is Turing-complete. It offers a rich set of techniques to reason about properties such as termination or confluence, where the latter ensures that the final result is independent of the order in which rewrite rules are applied. Logically constrained term rewriting extends this model by equipping rewrite rules with logical constraints over some theory. This enables built-in support for data structures and thus eases the analysis of real-world code. While plain term rewriting has been extensively studied, logically constrained rewriting—especially confluence analysis—is still in its infancy. In this thesis, we lift several well-known confluence criteria from term rewrite systems to logically constrained rewriting. Additionally, we present the first known non-confluence criterion for logically constrained rewrite systems, along with modular methods that significantly improve the automation of confluence analysis. We also introduce formal semantics for logically constrained rewriting, that serve as the foundation for future research on rewriting induction. Finally, our tool crest presents an automated push-button confluence analysis that implements all of our obtained results.