Recording Completion for Certificates in Equational Reasoning *

Thomas Sternagel Sarah Winkler Harald Zankl
Institute of Computer Science, University of Innsbruck, Innsbruck, Austria
{thomas.sternagel,sarah.winkler,harald.zankl}@uibk.ac.at

Abstract

We introduce recording completion, a variant of Knuth-Bendix completion which facilitates the construction of certificates for various equational logic proofs (completion proofs, entailment proofs and dis-proofs). The approach generalizes to more powerful variants of completion such as ordered completion and AC completion. We implemented recording completion in the tools KBCV and MKBTT. Both tools allow to choose among different formats of proof certificates, namely conversions, proof trees, and conversions with history. We report on experimental results in which all generated certificates have been verified by the trustable checker CeTA.

Keywords equational logic, rewriting, completion, certification

1. Introduction

Solving the word problem requires to decide whether an equation $s \approx t$ follows from an equational system $\mathcal{E}$. It is well known that the word problem is undecidable (see, e.g., [1]). Birkhoff's theorem [3] in combination with Knuth-Bendix completion [10] (if successful) gives a decision procedure for the word problem. If an equational system $\mathcal{E}$ can be transformed into an equivalent terminating and confluent rewrite system $\mathcal{R}$, then $s \approx t$ follows from $\mathcal{E}$ if and only if the $\mathcal{R}$-normal forms of s and t coincide.

Example 1. *For $\mathcal{E}$ consisting of the equations $\{\mathsf{ff} \approx \mathsf{f}, \mathsf{ggf} \approx \mathsf{g}\}$ (where f and g are unary function symbols, for which we find it convenient to abbreviate $\mathsf{f}(\mathsf{g}(\mathsf{f}(x)))$ to fgf etc.) a possible choice of $\mathcal{R}$ is $\{\mathsf{ff} \to \mathsf{f}, \mathsf{gf} \to \mathsf{g}, \mathsf{gg} \to \mathsf{g}\}$. Since $\mathsf{fgf} \to^*_{\mathcal{R}} \mathsf{fg} \;{}^*_{\mathcal{R}}\!\leftarrow \mathsf{fgg}$, the equation $\mathsf{fgf} \approx \mathsf{fgg}$ follows from $\mathcal{E}$.*

An alternative way to show that $s \approx t$ follows from $\mathcal{E}$ is to find a proof tree using the inference rules of *equational logic* (see Figure 1(a)). A proof tree showing that $\mathsf{fgf} \approx \mathsf{fgg}$ follows from $\mathcal{E}$ is depicted in Figure 2 on page 4. Note that in the case of completion such proof trees are not constructed explicitly.

When we want to answer/certify whether $s \approx t$ follows from $\mathcal{E}$ we are faced with the following situation:

- It is hard to find a conversion or proof tree showing that $s \approx t$ follows from $\mathcal{E}$ but easy to check validity of such a certificate.
- Under the assumption that Knuth-Bendix completion is successful, it is easy to decide the existence of a conversion or proof tree for $s \approx t$ (just rewrite s and t to $\mathcal{R}$-normal forms) but hard to certify this decision. In particular, it is typically far from obvious that $\mathcal{E}$ and $\mathcal{R}$ are equivalent.

In this paper we introduce *recording completion*, which overcomes both problems. Recording completion augments Knuth Bendix completion by a history component that allows to reconstruct how the rules in $\mathcal{R}$ have been derived from the equations in $\mathcal{E}$. Then, from a join $s \to^*_{\mathcal{R}} \cdot \;{}^*_{\mathcal{R}}\!\leftarrow t$ a conversion $s \leftrightarrow^*_{\mathcal{E}} t$ can be reconstructed, which is the basis for building an explicit proof tree. As a consequence, recording completion can also come up with an equivalence proof for $\mathcal{R}$ and $\mathcal{E}$, and hence facilitates certification of a completion proof.

We have implemented recording completion into the tools KBCV [16] (based on standard Knuth-Bendix completion) and MKBTT [21] (also admitting ordered and AC completion) such that they can generate certificates for equational logic proofs as well as completion proofs. These certificates can then be checked by CeTA [15, 18], an executable and trustable proof checker generated via the code export functionality of Isabelle, based on the *Isabelle Formalization of Rewriting* (IsaFoR).

The remainder of the paper is organized as follows: In the next section we recall preliminaries before Section 3 introduces recording completion. We first describe our approach for Knuth-Bendix completion before outlining extensions to ordered and AC completion. The various kinds of proof certificates enabled by recording completion are described in Section 4. The tools and their features are presented in Section 5 before experimental results are discussed in Section 6. Section 7 concludes.

2. Preliminaries

We assume familiarity with term rewriting, equational logic, and completion [1].

A *signature* $\mathcal{F}$ is a set of *function symbols* (denoted by f, g, etc.). The set of *terms* over a signature $\mathcal{F}$ and a set of variables $\mathcal{V}$ is denoted by $\mathcal{T}(\mathcal{F}, \mathcal{V})$. A term is *ground* if it does not contain variables. Let p be a position in a term t. Then $t|_p$ denotes the subterm of t at position p and $t[s]_p$ denotes the result of replacing the subterm of t at position p by the term s. For terms s and t from $\mathcal{T}(\mathcal{F}, \mathcal{V})$ we call $s \approx t$ an *equation*. An *equational system* (ES) $\mathcal{E}$ is a set of equations. Sometimes we orient an equation $s \approx t$, write $s \to t$, and call it a *rule*. Then for an ES $\mathcal{E}$ we denote by $\to_{\mathcal{E}}$ the smallest relation that contains $\mathcal{E}$ and is closed under contexts and substitutions. Let $\to$ be a relation. We write $(\to)^{-1}$ (or simply $\leftarrow$)

* This research is supported by the Austrian Science Fund (FWF) project I963 and a grant of the Hypo Tirol Bank.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

CPP '15, January 13–14, 2015, Mumbai, India.
Copyright is held by the owner/author(s).
ACM 978-1-4503-3300-9/15/01.
http://dx.doi.org/10.1145/10.1145/2676724.2693171

$$\frac{}{t \approx t}\ (\mathsf{refl})$$

$$\frac{t \approx s}{s \approx t}\ (\mathsf{sym})$$

$$\frac{s \approx t \quad t \approx u}{s \approx u}\ (\mathsf{trans})$$

$$\frac{s \approx t \in \mathcal{E}}{s\sigma \approx t\sigma}\ (\mathsf{assm})$$

$$\frac{s_1 \approx t_1 \quad \cdots \quad s_n \approx t_n}{f(s_1,\ldots,s_n) \approx f(t_1,\ldots,t_n)}\ (\mathsf{cong})$$

(a) Inference rules of *equational logic*.

$$\frac{(\mathcal{E},\mathcal{R},\mathcal{H})}{(\mathcal{E}\cup\{m\colon s\approx t\},\mathcal{R},\mathcal{H}\cup\{m\colon s \overset{j}{\leftarrow} u \overset{k}{\rightarrow} t\})}\ (\mathsf{deduce}) \qquad \text{if } s\ {}_{\mathcal{R}}\overset{j}{\leftarrow} u \overset{k}{\rightarrow}_{\mathcal{R}}\ t$$

$$\frac{(\mathcal{E}\cup\{i\colon s\approx t\},\mathcal{R},\mathcal{H})}{(\mathcal{E},\mathcal{R}\cup\{i\colon s\to t\},\mathcal{H})}\ (\mathsf{orient_l}) \qquad \text{if } s > t$$

$$\frac{(\mathcal{E}\cup\{i\colon s\approx t\},\mathcal{R},\mathcal{H}\cup\{i\colon s\ \overset{j}{\circ_1}\ u\ \overset{k}{\circ_2}\ t\})}{(\mathcal{E},\mathcal{R}\cup\{i\colon t\to s\},\mathcal{H}\cup\{i\colon t\,(\overset{k}{\circ_2})^{-1}\,u\,(\overset{j}{\circ_1})^{-1}\,s\})}\ (\mathsf{orient_r}) \qquad \text{if } t > s$$

$$\frac{(\mathcal{E}\cup\{i\colon s\approx t\},\mathcal{R},\mathcal{H})}{(\mathcal{E}\cup\{m\colon u\approx t\},\mathcal{R},\mathcal{H}\cup\{m\colon u \overset{l}{\leftarrow} s \overset{i}{\rightarrow} t\})}\ (\mathsf{simplify_l}) \qquad \text{if } s \overset{l}{\rightarrow}_{\mathcal{R}}\ u$$

$$\frac{(\mathcal{E}\cup\{i\colon s\approx t\},\mathcal{R},\mathcal{H})}{(\mathcal{E}\cup\{m\colon s\approx u\},\mathcal{R},\mathcal{H}\cup\{m\colon s \overset{i}{\rightarrow} t \overset{l}{\rightarrow} u\})}\ (\mathsf{simplify_r}) \qquad \text{if } t \overset{l}{\rightarrow}_{\mathcal{R}}\ u$$

$$\frac{(\mathcal{E}\cup\{i\colon s\approx s\},\mathcal{R},\mathcal{H}\cup\{i\colon s\circ_1 v\circ_2 s\})}{(\mathcal{E},\mathcal{R},\mathcal{H})}\ (\mathsf{delete})$$

$$\frac{(\mathcal{E},\mathcal{R}\cup\{i\colon s\to t\},\mathcal{H})}{(\mathcal{E},\mathcal{R}\cup\{m\colon s\to u\},\mathcal{H}\cup\{m\colon s \overset{i}{\rightarrow} t \overset{j}{\rightarrow} u\})}\ (\mathsf{compose}) \qquad \text{if } t \overset{j}{\rightarrow}_{\mathcal{R}}\ u$$

$$\frac{(\mathcal{E},\mathcal{R}\cup\{i\colon s\to t\},\mathcal{H})}{(\mathcal{E}\cup\{m\colon u\approx t\},\mathcal{R},\mathcal{H}\cup\{m\colon u \overset{j}{\leftarrow} s \overset{i}{\rightarrow} t\})}\ (\mathsf{collapse}) \qquad \text{if } s \overset{j}{\rightarrow}_{\mathcal{R}}\ u$$

(b) Inference rules of *recording completion*.

Figure 1: Inference rules.

for the inverse of $\to$, $\leftrightarrow$ for $\to \cup \leftarrow$, and $\to^*$ for the reflexive and transitive closure of $\to$.

An ES $\mathcal{E}$ is called a *term rewrite system* (TRS) if all equations are rules. A TRS $\mathcal{R}$ is *terminating* if $\to_{\mathcal{R}}$ is well-founded, *locally confluent* if ${}_{\mathcal{R}}\!\leftarrow \cdot \to_{\mathcal{R}} \subseteq \to^*_{\mathcal{R}} \cdot {}^*_{\mathcal{R}}\!\leftarrow$, and *confluent* if ${}^*_{\mathcal{R}}\!\leftarrow \cdot \to^*_{\mathcal{R}} \subseteq \to^*_{\mathcal{R}} \cdot {}^*_{\mathcal{R}}\!\leftarrow$. A terminating and confluent TRS is called *complete*. A TRS is *ground confluent* if $s\ {}^*_{\mathcal{R}}\!\leftarrow u \to^*_{\mathcal{R}} t$ implies $s \to^*_{\mathcal{R}} \cdot {}^*_{\mathcal{R}}\!\leftarrow t$ for all ground terms u, and ground complete if it is terminating and ground confluent. A term s is in *($\mathcal{R}$-)normal form* if there is no term t with $s \to_{\mathcal{R}} t$. We write $s{\downarrow_{\mathcal{R}}}$ for an $\mathcal{R}$-normal form of a term s, i.e., some term t such that $s \to^*_{\mathcal{R}} t$ and t is in normal form. Terms s and t are *($\mathcal{R}$-)joinable* if $s \to^*_{\mathcal{R}} \cdot {}^*_{\mathcal{R}}\!\leftarrow t$, and *($\mathcal{E}$-)convertible* if $s \leftrightarrow^*_{\mathcal{E}} t$. A TRS $\mathcal{R}$ and an ES $\mathcal{E}$ are *equivalent* if $\leftrightarrow^*_{\mathcal{E}} = \leftrightarrow^*_{\mathcal{R}}$, i.e., their respective equational theories coincide.

We call (s,t) a *critical pair* of a TRS $\mathcal{R}$ if and only if there are two (not necessarily distinct) variable renamed rules $\ell_i \to r_i$, $i = 1, 2$ (without common variables) and a position p in ℓ_1 such that θ is a most general unifier of $\ell_1|_p$ and ℓ_2, $\ell_1|_p$ is not a variable, $s = r_1\theta$, and $t = \ell_1\theta[r_2\theta]_p$. The *critical pair theorem* [8, 10] states that a TRS $\mathcal{R}$ is locally confluent if and only if all its critical pairs are joinable.

An *$\mathcal{F}$-algebra* $\mathcal{A}$ consists of a non-empty set A (the *carrier*) and an *interpretation* $I\colon \mathcal{F} \to A^* \to A$. For each $\mathcal{F}$-algebra $\mathcal{A} = (A, I)$, and each *variable assignment* $\alpha\colon \mathcal{V} \to A$, we define the *term evaluation* $[\cdot]^{\mathcal{A}}_{\alpha}\colon \mathcal{T}(\mathcal{F},\mathcal{V}) \to A$ as $[x]^{\mathcal{A}}_{\alpha} = \alpha(x)$ and $[f(t_1,\ldots,t_n)]^{\mathcal{A}}_{\alpha} = I(f)([t_1]^{\mathcal{A}}_{\alpha},\ldots,[t_n]^{\mathcal{A}}_{\alpha})$. We drop $\mathcal{A}$ whenever it is clear from the context.

An $\mathcal{F}$-algebra $\mathcal{A}$ is a *model* of an equation $s \approx t$ if for each variable assignment α the equality $[s]^{\mathcal{A}}_{\alpha} = [t]^{\mathcal{A}}_{\alpha}$ is satisfied. If every $\mathcal{F}$-algebra that is a model of all equations in $\mathcal{E}$ also is a model of $s \approx t$ we write $\mathcal{E} \models s \approx t$ and say that $s \approx t$ *follows from* $\mathcal{E}$. The relation $\models$ is called *semantic entailment*. *Syntactic entailment*, written $\mathcal{E} \vdash s \approx t$, means that we can prove $s \approx t$ with the inference rules from Figure 1(a) using equations from $\mathcal{E}$.

For any ES $\mathcal{E}$ and terms s and t we have $\mathcal{E} \vdash s \approx t$ if and only if $s \leftrightarrow^*_{\mathcal{E}} t$ if and only if $\mathcal{E} \models s \approx t$. The latter follows from Birkhoff's theorem [3].

3. Recording Completion

In this section we present the method of recording completion. In Section 3.1 we consider the case of classical Knuth-Bendix completion. To this end we extend the well-known inference rules (see, e.g., [1]) by a *history component* that allows to infer how the rules in $\mathcal{R}$ have been derived from the equations in $\mathcal{E}$.[1] We discuss extensions to ordered and AC completion in Section 3.2.

3.1 Knuth-Bendix Completion

Recording completion basically aims to construct entailment proofs, which may be represented by conversions or proof trees. The construction of an entailment proof for $\mathcal{E} \vdash s \approx t$ (if possible at all) proceeds in three phases to obtain a conversion, plus an additional phase to get a proof tree:

record The inference rules of recording completion (see Figure 1(b)) are applied to the equational system $\mathcal{E}$. Upon success, a confluent and terminating TRS $\mathcal{R}$ (equivalent to $\mathcal{E}$) and a history $\mathcal{H}$ (recording how the rules in $\mathcal{R}$ have been derived) are computed.

compare If the previous phase is successful, the test for $s \to^*_{\mathcal{R}} \cdot {}^*_{\mathcal{R}}\!\leftarrow t$ is performed by rewriting s and t to $\mathcal{R}$-normal form.

[1] Note that our implementations (see Section 5) are based on the slightly more involved rules of [20], for which the extension works alike. For reasons of readability this is not reflected in our presentation.

recall If the previous phase is successful, we construct $s \leftrightarrow_{\mathcal{E}}^* t$ from $s \rightarrow_{\mathcal{R}}^* \cdot {}_{\mathcal{R}}^*\!\leftarrow t$, using the information recorded in the history $\mathcal{H}$.

plant & grow Based on the conversion $s \leftrightarrow_{\mathcal{E}}^* t$ a proof tree for $\mathcal{E} \vdash s \approx t$ is constructed.

In the sequel we discuss the four phases in detail.

3.1.1 Record

The *record* phase uses the inference rules from Figure 1(b), which are similar to the standard rules except for the following two differences: In the collapse-rule we dropped the condition of strict encompassment. Since we only consider finite runs, this condition is no longer required for soundness (cf. [15, Theorem 5.2]). Furthermore, there is a new history component $\mathcal{H}$ whose entries are of the form $i\colon s \overset{j}{\circ_1} u \overset{k}{\circ_2} t$ where i is the index of the history entry, j and k are indices of rules in $\mathcal{R}$, s, u and t are terms, and $\circ_1, \circ_2 \in \{\leftarrow, \rightarrow\}$.

Let us take a closer look at the extended inference rules. For deduce the peak $s \overset{j}{\leftarrow} u \overset{k}{\rightarrow} t$ that triggers the new equation $s \approx t$ is stored in a history entry (where m is assumed to be a fresh index that is larger than every earlier index). By $\mathsf{orient}_\mathsf{l}$ we orient an equation from left to right and the corresponding history entry remains unchanged, whereas by $\mathsf{orient}_\mathsf{r}$ we orient an equation from right to left and thus have to "mirror" the corresponding history entry. Here $>$ is a reduction order, provided as part of the input. The rules $\mathsf{simplify}_\mathsf{l}$ and $\mathsf{simplify}_\mathsf{r}$ are used to $\mathcal{R}$-rewrite a left- or right-hand side of an equation. With delete we remove trivial equations from $\mathcal{E}$ and the corresponding history entry from $\mathcal{H}$. Finally, compose rewrites a right-hand side of a rule in $\mathcal{R}$ while collapse does the same for left-hand sides.

We write $(\mathcal{E}_i, \mathcal{R}_i, \mathcal{H}_i) \rightsquigarrow (\mathcal{E}_{i+1}, \mathcal{R}_{i+1}, \mathcal{H}_{i+1})$ for the application of an arbitrary inference rule to the triple $(\mathcal{E}_i, \mathcal{R}_i, \mathcal{H}_i)$ resulting in $(\mathcal{E}_{i+1}, \mathcal{R}_{i+1}, \mathcal{H}_{i+1})$.

Definition 1. *A* run *of recording completion for $\mathcal{E}$ is a finite sequence*

$$(\mathcal{E}_0, \mathcal{R}_0, \mathcal{H}_0) \rightsquigarrow^n (\mathcal{E}_n, \mathcal{R}_n, \mathcal{H}_n)$$

of rule applications, where $\mathcal{E}_0 = \mathcal{E}$, $\mathcal{R}_0 = \varnothing$, and $\mathcal{H}_0 = \varnothing$. A run is successful *if $\mathcal{E}_n = \varnothing$ and all critical pairs of $\mathcal{R}_n$ that are not contained in $\bigcup_{i \le n} \mathcal{E}_i$ are joinable by $\mathcal{R}_n$.*

If $(\mathcal{E}, \varnothing, \varnothing) \rightsquigarrow^n (\varnothing, \mathcal{R}, \mathcal{H})$ is a successful run for $\mathcal{E}$, then $\mathcal{R}$ is confluent, terminating, and equivalent to $\mathcal{E}$ (cf. Theorem 1). Note that the requirement on critical pairs for a successful run can be replaced by local confluence of $\mathcal{R}$, since local confluence implies that all critical pairs of $\mathcal{R}$ are joinable.

If recording completion was successful $\mathcal{H}_n$ contains important information about the intermediate steps which we will need later to construct the conversion $s \leftrightarrow_{\mathcal{E}}^* t$ from the join $s \rightarrow_{\mathcal{R}}^* \cdot {}_{\mathcal{R}}^*\!\leftarrow t$.

The *record* phase is sound, the proof has been formalized in IsaFoR (see [15]).

Theorem 1. *If $(\mathcal{E}, \varnothing, \varnothing) \rightsquigarrow^n (\varnothing, \mathcal{R}, \mathcal{H})$ is a successful run of recording completion then $\mathcal{R}$ is confluent, terminating, and equivalent to $\mathcal{E}$.*

Next we demonstrate the record phase by an example.

Example 2. *Recall $\mathcal{E}$ from Example 1. We start with the triple depicted in Table 1(a) and perform recording completion. Note that LPO with empty precedence orients all emerging rules in the desired direction. After orienting rules* 1 *and* 2 *from left to right we deduce a critical pair between rules 2 and 1, resulting in the equation* $3\colon \mathsf{ggf} \approx \mathsf{gf}$ *and the history entry* $3\colon \mathsf{gf} \overset{1}{\leftarrow} \mathsf{ggff} \overset{2}{\rightarrow} \mathsf{gf}$*. Next we simplify the left-hand side of equation* 3 *by an application of rule* 2 *and obtain the equation* $4\colon \mathsf{g} \approx \mathsf{gf}$ *with corresponding history entry* $4\colon \mathsf{g} \overset{2}{\leftarrow} \mathsf{ggf} \overset{3}{\rightarrow} \mathsf{gf}$*. Orienting rule* 4 *from right to left causes the history entry to be mirrored, i.e.,* $4\colon \mathsf{gf} \overset{3}{\leftarrow} \mathsf{ggf} \overset{2}{\rightarrow} \mathsf{g}$*. Rules* 2 *and* 4 *allow to deduce equation* $5\colon \mathsf{gg} \approx \mathsf{g}$ *with history* $5\colon \mathsf{gg} \overset{4}{\leftarrow} \mathsf{ggf} \overset{2}{\rightarrow} \mathsf{g}$*, which we orient from left to right. Collapsing the left-hand side of rule* 2 *with rule* 5 *yields* $6\colon \mathsf{gf} \approx \mathsf{g}$ *with* $6\colon \mathsf{gf} \overset{5}{\leftarrow} \mathsf{ggf} \overset{2}{\rightarrow} \mathsf{g}$*. Now rule* 4 *simplifies equation* 6 *into* $7\colon \mathsf{g} \approx \mathsf{g}$ *with* $7\colon \mathsf{g} \overset{4}{\leftarrow} \mathsf{gf} \overset{6}{\rightarrow} \mathsf{g}$*, which is immediately deleted afterwards. Finally, $\mathcal{E}_n$ is empty and as all remaining critical pairs of $\mathcal{R}_n$ are joinable, the procedure can be stopped. Since there is no rule with index* 6 *the history entry* 6 *can be dropped. Hence, we obtain the result depicted in Table 1(b) where $\mathcal{R}_n$ is terminating, confluent, and equivalent to $\mathcal{E}$.*

$\mathcal{E}_0$	$\mathcal{R}_0$	$\mathcal{H}_0$
$1\colon \mathsf{ff} \approx \mathsf{f}$	$\varnothing$	$\varnothing$
$2\colon \mathsf{ggf} \approx \mathsf{g}$		

(a) Initial state.

$\mathcal{E}_n$	$\mathcal{R}_n$	$\mathcal{H}_n$
$\varnothing$	$1\colon \mathsf{ff} \to \mathsf{f}$	$3\colon \mathsf{ggf} \overset{1}{\leftarrow} \mathsf{ggff} \overset{2}{\rightarrow} \mathsf{gf}$
	$4\colon \mathsf{gf} \to \mathsf{g}$	$4\colon \mathsf{gf} \overset{3}{\leftarrow} \mathsf{ggf} \overset{2}{\rightarrow} \mathsf{g}$
	$5\colon \mathsf{gg} \to \mathsf{g}$	$5\colon \mathsf{gg} \overset{4}{\leftarrow} \mathsf{ggf} \overset{2}{\rightarrow} \mathsf{g}$

(b) Final state.

Table 1: Example of recording completion.

3.1.2 Compare

Let $(\mathcal{E}, \varnothing, \varnothing) \rightsquigarrow^n (\varnothing, \mathcal{R}, \mathcal{H})$ be a successful run of recording completion. In the *compare* phase we test joinability of the terms in $s \approx t$ with respect to $\mathcal{R}$. If the two terms are joinable, then $s \approx t$ follows from $\mathcal{E}$ and the next phase constructs an $\mathcal{E}$-conversion $s \leftrightarrow_{\mathcal{E}}^* t$. In the other case $s \approx t$ does not follow from $\mathcal{E}$.

Example 3 (Continued from Example 2)**.** *For* fgf *and* fgg *we obtain joinability since* $\mathsf{fgf} \overset{4}{\rightarrow}_{\mathcal{R}} \mathsf{fg} \; {}_{\mathcal{R}}\!\overset{5}{\leftarrow} \mathsf{fgg}$*. Hence* $\mathsf{fgf} \approx \mathsf{fgg}$ *follows from $\mathcal{E}$.*

The *compare* phase is sound since $\mathcal{R}$ and $\mathcal{E}$ are equivalent (cf. Theorem 1).

3.1.3 Recall

Let $(\mathcal{E}, \varnothing, \varnothing) \rightsquigarrow^n (\mathcal{E}', \mathcal{R}, \mathcal{H})$ be a run of recording completion. Then the *recall* phase transforms a join $s \rightarrow_{\mathcal{R}}^* \cdot {}_{\mathcal{R}}^*\!\leftarrow t$ into a conversion $s \leftrightarrow_{\mathcal{E}}^* t$ as follows. For each step $t_1 \overset{i}{\rightarrow} t_2$ where the index i is not in $\mathcal{E}$ the corresponding history entry is *inserted*. Let $i\colon \ell \to r$ be the rule with index i. Then there must be a history entry $i\colon \ell \overset{j}{\circ_1} u \overset{k}{\circ_2} r$, a position p, and a substitution σ such that $t_1|_p = \ell\sigma$ and $t_2|_p = {}_k r\sigma$. The step $t_1 \overset{i}{\rightarrow} t_2$ is replaced by the conversion $t_1 \overset{j}{\circ_1} t_1[u\sigma]_p \overset{k}{\circ_2} t_2$. This process terminates since $i > j, k$, i.e., the index of any history entry depends on smaller indices only and finally we arrive at a conversion using indices from $\mathcal{E}$.

Example 4 (Continued from Example 3)**.** *The* recall *phase starts with the join* $\mathsf{fgf} \overset{4}{\rightarrow}_{\mathcal{R}} \mathsf{fg} \; {}_{\mathcal{R}}\!\overset{5}{\leftarrow} \mathsf{fgg}$ *and replaces the step* $\underline{\mathsf{fgf}} \overset{4}{\rightarrow} \underline{\mathsf{fg}}$ *(at position* 1*) using history entry* 4 *by* $\underline{\mathsf{fgf}} \overset{3}{\leftarrow} \underline{\mathsf{fggf}} \overset{2}{\rightarrow} \underline{\mathsf{fg}}$*, etc. until we arrive at the following conversion using indices from $\mathcal{E}$ only*

$$\mathsf{fgf} \overset{2}{\leftarrow} \mathsf{fggff} \overset{1}{\rightarrow} \mathsf{fggf} \overset{2}{\rightarrow} \mathsf{fg} \overset{2}{\leftarrow} \mathsf{fggf} \overset{2}{\leftarrow} \mathsf{fgggff} \overset{1}{\rightarrow} \mathsf{fgggf} \overset{2}{\rightarrow} \mathsf{fgg} \tag{1}$$

The arrows in this conversion indicate if the step is performed via an equation from $\mathcal{E}$ or a symmetric version of an equation from $\mathcal{E}$.

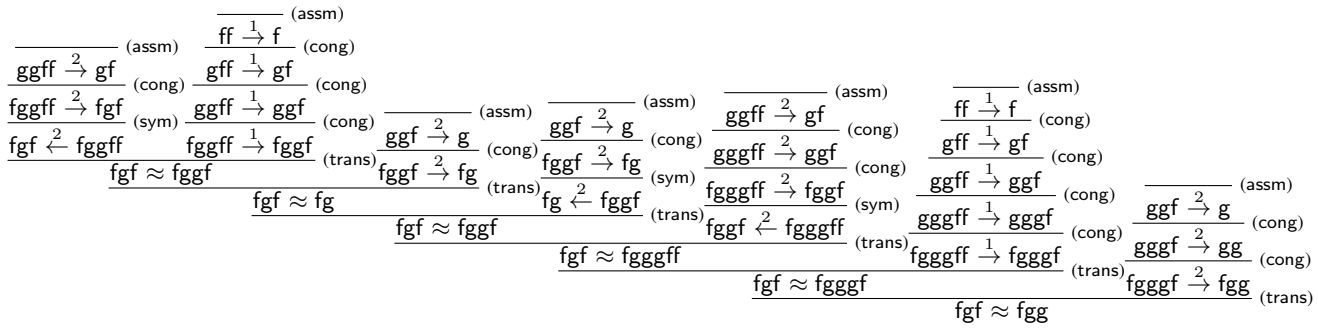

Figure 2: Result of the plant & grow phase.

The next lemma states the desired property of the *recall* phase. Note that we do not need a *successful* run of recording completion but any join $s \to_{\mathcal{R}}^* \cdot \,{}_{\mathcal{R}}^*\!\leftarrow t$ can be transformed into $s \leftrightarrow_{\mathcal{E}}^* t$.

Lemma 1. *Let $(\mathcal{E}, \varnothing, \varnothing) \rightsquigarrow^n (\mathcal{E}_n, \mathcal{R}_n, \mathcal{H}_n)$ be a run of recording completion. Then the recall phase transforms any join using rules from $\mathcal{R}_n$ into a conversion using rules from $\mathcal{E}$.*

Proof. Let I_0 be the set of all indices of $\mathcal{E}$ and I be the set of all indices of all $\mathcal{E}_i$ and $\mathcal{R}_i$ with $0 \leqslant i \leqslant n$. We show the slightly stronger claim that any conversion over I is transformed into a conversion over I_0.

To this end, consider a conversion $t_1 \overset{i_1}{\leftrightarrow} t_2 \overset{i_2}{\leftrightarrow} \cdots \overset{i_{m-1}}{\leftrightarrow} t_m$ where indices $i_1, \ldots, i_{m-1}$ are in I, i.e., $\overset{i}{\to}$ if and only if $\overset{i}{\to}$ or $\overset{i}{\leftarrow}$ is in $\mathcal{R}_j$ or $s \overset{i}{\approx} t$ or $t \overset{i}{\approx} s$ is in $\mathcal{E}_j$ for some $0 \leqslant j \leqslant n$. Let S be the multiset of indices occurring in this conversion, S_0 the submultiset of S with indices from $\mathcal{E}_0$, and $S' = S \setminus S_0$. We show the claim by induction on the multiset S' where multisets are ordered by the multiset extension of $>_{\mathbb{N}}$. In the base case S' is empty and hence $t_1 \overset{I_0}{\leftrightarrow}{}^* t_m$ which shows the result. In the step case there exists an index $l \in S'$ such that $t_1 \overset{I_0}{\leftrightarrow}{}^* t_j \overset{I_0}{\leftrightarrow} t_{j+1} \overset{I_0}{\leftrightarrow}{}^* t_m$ for some $1 \leqslant j < m$. Now assume that l does not appear in $\mathcal{H}_n$. Then it must have been deleted by the rule delete and hence $t_j = t_{j+1}$. This case is finished by applying the induction hypothesis to $t_1 \overset{I_0}{\leftrightarrow}{}^* t_j \overset{I_0}{\leftrightarrow}{}^* t_m$. In the other case let $l\colon s \overset{p}{\circ_1} u \overset{k}{\circ_2} t \in \mathcal{H}_n$. Since $l >_{\mathbb{N}} p, k$ (by construction of the inference rules of recording completion) the induction hypothesis applies to $t_1 \overset{I_0}{\leftrightarrow}{}^* t_j \overset{p}{\circ_1} \cdot \overset{k}{\circ_2} t_{j+1} \overset{I_0}{\leftrightarrow}{}^* t_m$ which concludes the proof. □

3.1.4 Plant and Grow

The *plant and grow* phase constructs a proof tree for $\mathcal{E} \vdash s \approx t$ from a conversion $s \leftrightarrow_{\mathcal{E}} \cdot \leftrightarrow_{\mathcal{E}} \cdots \leftrightarrow_{\mathcal{E}} t$ as follows: The tree is *planted* by applications of trans (downwards) while it *grows* (upwards) using rules cong (to chop off common contexts), sym (to optionally change the direction of an equation), and assm to finish the branch of the proof.

Example 5. *For the conversion* (1) *from Example 4 the corresponding proof tree is shown in Figure 2 (note how the left assumption of the first trans-rule together with the right assumptions of all trans-rules directly correspond to conversion* (1)*).*

3.2 Ordered and Normalized Completion

Ordered completion was dubbed *unfailing completion* when introduced by Bachmair *et al.* [2]. Though it will in general only produce a ground complete system, this turns out to be sufficient for many applications including theorem proving. Consequently, various refutational theorem proving tools are based on variants of ordered completion [6, 7].

Ordered completion can be described by an inference system [2] which differs from the one for standard completion by only two features: (1) deduce steps allow to compute critical pairs involving not only rewrite rules but also equations, and (2) orientable instances of equations are admitted instead of rules in the rewrite steps of $\mathsf{simplify}_\mathsf{l}$, $\mathsf{simplify}_\mathsf{r}$, compose, and collapse inferences.

In an ordered recording completion procedure the history component $\mathcal{H}$ thus contains entries $i\colon s \overset{j}{\circ_1} u \overset{k}{\circ_2} t$ where $\circ_1, \circ_2 \in \{\leftarrow, \rightarrow, \approx\}$. We then extend the inference rules of Figure 1(b) with an additional deduce° rule as well as $\mathsf{simplify}^\circ_\mathsf{l}$, $\mathsf{simplify}^\circ_\mathsf{r}$, $\mathsf{compose}^\circ$, and $\mathsf{collapse}^\circ$ rules that apply orientable instances of equations in rewrite steps. The rules deduce° and $\mathsf{compose}^\circ$ are displayed in Figure 3(a). All remaining rules are obtained by modifying their counterpart in Figure 1(b) in a similar way as done for $\mathsf{compose}^\circ$. In deduce° the relation $\overset{j}{\circ_1}$ denotes $\overset{j}{\leftarrow}$ if j is an index of a rule in $\mathcal{R}$, and $\overset{j}{\approx}$ otherwise. Similarly, $\overset{k}{\circ_2}$ denotes $\overset{k}{\to}$ if k is an index of a rule in $\mathcal{R}$, and $\overset{k}{\approx}$ otherwise. Inferences using the deduce° rule can in practice be restricted to *extended critical pairs* [2].

Example 6. *Consider the system GRP466-1 in TPTP:*[2]

$$x_2 \approx (x_0/x_0)/(x_1/((x_2/(x_3/x_1))/\mathsf{i}(x_3)))$$
$$x_0/\mathsf{i}(x_1) \approx x_0 \cdot x_1$$

Using ordered recording completion, the tool MKBTT can produce a proof of the equation $\mathsf{i}(\mathsf{a}) \cdot \mathsf{a} \approx \mathsf{i}(\mathsf{b}) \cdot \mathsf{b}$ *which is certified by CeTA. Note that standard completion does not succeed in this case.*

In the tool MKBTT we also support recording completion in the setting of normalized completion [11], which covers AC completion [13] as a subcase. These methods extend standard completion by reasoning modulo built-in theories, and are useful in presence of, e.g., associative and commutative operators where standard completion fails. Though the implementation of MKBTT is actually based on normalized completion, we restrict to AC completion in the sequel in order to simplify the presentation. Figure 3(b) shows two example rules.

Here the signature $\mathcal{F}$ is supposed to include a subset of binary AC operators $\mathcal{F}_{\mathsf{AC}}$, and the ES AC consists of all equations $f(x, y) \approx f(y, x)$ and $f(x, f(y, z)) \approx f(f(x, y), z)$ such that $f \in \mathcal{F}_{\mathsf{AC}}$. The reduction order $>$ has to satisfy AC compatibility, i.e., $\leftrightarrow_{\mathsf{AC}}^* \cdot > \cdot \leftrightarrow_{\mathsf{AC}}^* \subseteq >$. We write $s \overset{j}{\to}_{\mathcal{R}/\mathsf{AC}} t$ whenever $s \leftrightarrow_{\mathsf{AC}}^* \cdot \overset{j}{\to}_{\mathcal{R}} \cdot \leftrightarrow_{\mathsf{AC}}^* t$.

Example 7. *When given Abelian group theory as specified by* $\mathcal{E} = \{x \cdot (y \cdot z) \approx (x \cdot y) \cdot z,\ x \cdot y \approx y \cdot x,\ x \cdot 1 \approx x,\ x \cdot \mathsf{i}(x) \approx 1\}$ *as input, MKBTT can produce a proof of the equation* $\mathsf{i}(x) \cdot \mathsf{i}(y) \approx \mathsf{i}(x \cdot y)$ *which is certified by CeTA. As CeTA does currently not support rewriting modulo AC, all AC steps must be specified explicitly.*

[2] http://www.cs.miami.edu/~tptp/

$$\frac{(\mathcal{E},\mathcal{R},\mathcal{H})}{(\mathcal{E}\cup\{m\colon s\approx t\},\mathcal{R},\mathcal{H}\cup\{m\colon s \overset{j}{\circ_1} u \overset{k}{\circ_2} t\})}\ (\mathsf{deduce}^{\circ}) \qquad \text{if } s\ {}_{\mathcal{S}}\overset{j}{\leftarrow} u \overset{k}{\rightarrow}_{\mathcal{S}} t \text{ for } \mathcal{S}=\mathcal{R}\cup\mathcal{E}\cup\mathcal{E}^{-1}$$

$$\frac{(\mathcal{E},\mathcal{R}\cup\{i\colon s\to t\},\mathcal{H})}{(\mathcal{E},\mathcal{R}\cup\{m\colon s\to u\},\mathcal{H}\cup\{m\colon s \overset{i}{\to} t \overset{j}{\approx} u\})}\ (\mathsf{compose}^{\circ}) \qquad \text{if } v \overset{j}{\approx} w\in\mathcal{E}\cup\mathcal{E}^{-1}, v\sigma > w\sigma \text{ and } t\to_{v\sigma\to w\sigma} u$$

(a) Ordered completion.

$$\frac{(\mathcal{E},\mathcal{R},\mathcal{H})}{(\mathcal{E}\cup\{m\colon s\approx t\},\mathcal{R},\mathcal{H}\cup\{m\colon s\ {}_{\mathcal{R}/\mathsf{AC}}\overset{j}{\leftarrow} u \overset{k}{\rightarrow}_{\mathcal{R}/\mathsf{AC}} t\})}\ (\mathsf{deduce}^{\mathsf{AC}}) \qquad \text{if } s\ {}_{\mathcal{R}/\mathsf{AC}}\overset{j}{\leftarrow} u \overset{k}{\rightarrow}_{\mathcal{R}/\mathsf{AC}} t$$

$$\frac{(\mathcal{E},\mathcal{R}\cup\{i\colon s\to t\},\mathcal{H})}{(\mathcal{E},\mathcal{R}\cup\{m\colon s\to u\},\mathcal{H}\cup\{m\colon s \overset{i}{\to} t \overset{j}{\to}_{\mathcal{R}/\mathsf{AC}} u\})}\ (\mathsf{compose}^{\mathsf{AC}}) \qquad \text{if } t \overset{j}{\to}_{\mathcal{R}/\mathsf{AC}} u$$

(b) AC completion.

Figure 3: Some additional inference rules for variants of completion.

4. Certificates for Equational Proofs

Recording completion enables the generation of various kinds of proof certificates, which can all be checked by CeTA 2.18. CeTA accepts certificates in the *certification proof format* (CPF),[3] which is a common format for rewrite proofs in XML. Clearly the equational system must always be part of the certificate. We discuss the additional content for different kinds of certificates below.

Entailment Proofs Any certificate for a proof of $\mathcal{E}\vdash s\approx t$ requires the equational system $\mathcal{E}$ and the goal equation $s\approx t$. The construction of such a proof has been outlined in the previous section. Concerning the representation of the single proof steps in the certificate, one can think of different levels of verbosity.

- Conversion: In the simplest case recording completion omits the plant and grow phase and simply describes a conversion for the goal equation $s\approx t$ using equations in $\mathcal{E}$, such as in Example 4.
- Proof trees: In this case the certificate contains a proof tree for $s\approx t$ using equations in $\mathcal{E}$, as exemplified in Figure 2 (Example 5).
- History: One can ensure $\mathcal{E}\vdash s\approx t$ by showing that (i) the TRS $\mathcal{R}_i$ associated with the current completion state admits a join $s\to^*_{\mathcal{R}_i}\cdot\ {}^*_{\mathcal{R}_i}\!\leftarrow t$, and (ii) all history entries $m\colon s'\overset{j}{\circ_1}u\overset{k}{\circ_2}t'$ are consequences of $\mathcal{E}$ (i.e., $s'\leftrightarrow^*_{\mathcal{E}} t'$) and can thus be used as auxiliary equations. For our running example the data of a dedicated certificate is shown in Table 2. To avoid cyclic references, history entries are processed in order of their indices. This approach requires the certifier to support such auxiliary equations. In return, proofs become much shorter as the history itself is the proof of $\leftrightarrow^*_{\mathcal{R}_i}\subseteq\leftrightarrow^*_{\mathcal{E}}$ which obviously has linear size. In contrast, the recall phase might produce certificates where the conversion $s\leftrightarrow^*_{\mathcal{E}} t$ is exponentially larger than the join $s\to^*_{\mathcal{R}_i}\cdot\ {}^*_{\mathcal{R}_i}\!\leftarrow t$.

Note that in order to establish a proof tree for $\mathcal{E}\vdash s\approx t$ it is not necessarily required to perform a successful run of completion. Interleaving the record and compare phase the procedure can be stopped once $s\to^*_{\mathcal{R}_i}\cdot\ {}^*_{\mathcal{R}_i}\!\leftarrow t$ can be established. Still, a proof tree (or conversion) can be constructed (cf. Lemma 1), which can be checked by CeTA. This is different for entailment dis-proofs, which are discussed next.

[3] `http://cl-informatik.uibk.ac.at/software/cpf/`

$\mathcal{E}$:	$\mathsf{ff}\approx\mathsf{f}$
	$\mathsf{ggf}\approx\mathsf{g}$
goal equation:	$\mathsf{fgf}\approx\mathsf{fgg}$
$\mathcal{H}$:	$\mathsf{ggf}\leftarrow\mathsf{ggff}\rightarrow\mathsf{gf}$
	$\mathsf{gf}\leftarrow\mathsf{ggf}\rightarrow\mathsf{g}$
	$\mathsf{gg}\leftarrow\mathsf{ggf}\rightarrow\mathsf{g}$
join:	$\mathsf{fgf}\rightarrow\mathsf{fg}\leftarrow\mathsf{fgg}$

Table 2: Entailment proof using history.

Entailment Dis-Proofs If the record phase is successful in deriving a complete TRS $\mathcal{R}$ but in the compare phase s and t cannot be joined then it follows that $\mathcal{E}\not\vdash s\approx t$. The certificates for entailment dis-proofs $\mathcal{E}\not\vdash s\approx t$ thus require a certificate for a completion proof (see below), besides the goal equation $s\approx t$ and the ES $\mathcal{E}$. In this case CeTA verifies that s and t have different normal forms with respect to $\mathcal{R}$ automatically.

Completion Proofs For completion proofs CeTA does not certify that each application of an inference rule is correct but rather that the output is correct, i.e., the rewrite system $\mathcal{R}$ is (i) terminating, (ii) confluent, and (iii) equivalent to the input ES $\mathcal{E}$.

For (i) the termination proof generated in the course of (recording) completion is added to the certificate while (ii) can be checked by Newman's lemma, stating that in the presence of termination local confluence and confluence coincide. For checking local confluence CeTA does not require any information but supports checking given joining sequences for the critical pairs if they are provided in the certificate. CeTA certifies (iii) by establishing $\leftrightarrow^*_{\mathcal{E}}\subseteq\leftrightarrow^*_{\mathcal{R}}$ and $\leftrightarrow^*_{\mathcal{R}}\subseteq\leftrightarrow^*_{\mathcal{E}}$. It automatically verifies the former by checking that $s\to^*_{\mathcal{R}}\cdot\ {}^*_{\mathcal{R}}\!\leftarrow t$ holds for all $s\approx t$ in $\mathcal{E}$, because due to completeness of $\mathcal{R}$ all such terms s and t must have a common normal form. To establish the latter, CeTA needs additional input, thereby supporting similar possibilities as for entailment proofs:

- Proof trees: In this case the certificate contains a proof tree for each rule $l\to r\in\mathcal{R}$ showing $\mathcal{E}\vdash l\approx r$. This establishes $\to_{\mathcal{R}}\subseteq\leftrightarrow^*_{\mathcal{E}}$ and hence $\leftrightarrow^*_{\mathcal{R}}\subseteq\leftrightarrow^*_{\mathcal{E}}$.
- History: Alternatively, one can guarantee $\leftrightarrow^*_{\mathcal{R}}\subseteq\leftrightarrow^*_{\mathcal{E}}$ by showing that all history entries $i\colon s'\overset{j}{\circ_1}u\overset{k}{\circ_2}t'$ are consequences of $\mathcal{E}$ (i.e., $s'\leftrightarrow^*_{\mathcal{E}} t'$) and can thus be used as auxiliary equations.

The disadvantage of the first approach is that the recall phase gives rise to an exponential blowup in the proof size. That this is problematic in practice is highlighted in Section 6.

5. Tools and Features

We implemented recording completion in KBCV, a tool designed for interactive completion proofs based on Knuth-Bendix completion and MKBTT, an automatic completion tool employing multi-completion based on either standard, ordered or normalized completion. Both tools offer a command-line and a web-interface, KBCV additionally offers a graphical user interface. The sources and web-interfaces are available from

`http://cl-informatik.uibk.ac.at/research/software`

First we present KBCV, which admits user interaction. The tool displays the current sets of equations and rules and the user can select the desired inference rule from Figure 1(b) to be applied (in addition the tool features a fully automatic mode). Internally KBCV uses a combination of the inference system of Figure 1(b) for recording completion and the inference system of [20, Figure 2] for completion using termination tools. Hence, it performs the necessary termination checks automatically. However, the constraints needed for the termination checks (component C in [20, Figure 2]) and the history (component $\mathcal{H}$ in Figure 1(b)) are hidden from the user and not displayed by default.

At any stage of the process the user can try to construct an equivalence proof for two terms s and t using the current set of rules $\mathcal{R}_i$ since $s \to^*_{\mathcal{R}_i} \cdot \ {}^*_{\mathcal{R}_i}\!\leftarrow t$ implies $s \leftrightarrow^*_{\mathcal{E}} t$ by Lemma 1. Note that any $\mathcal{R}_i$ is terminating but need not be confluent. Hence, disproving s and t equivalent is only possible after a successful run of (recording) completion.

To certify its output, KBCV can extract equivalence (dis-)proofs and completion proofs in CPF format, which we have suitably extended. Next we discuss how this can be done via the applet version of KBCV. By default the axioms of group theory ($\mathsf{f}(\mathsf{f}(x,y),z) \approx \mathsf{f}(x,\mathsf{f}(y,z))$, $\mathsf{f}(x,\mathsf{c}) \approx x$, $\mathsf{f}(x,\mathsf{g}(x)) \approx \mathsf{c}$) are loaded and an automatic completion attempt (button *Completion*) is successful within a few seconds on a modern computer. The tool then can construct a proof (having several hundred lines) that $\mathsf{g}(\mathsf{f}(x,y)) \approx \mathsf{f}(\mathsf{g}(y),\mathsf{g}(x))$ follows from the axioms of group theory (*File* $\to$ *Equational Proof*) also within a few seconds. Because these proofs are typically fairly large they are not depicted as trees but in a linear shape. After exporting and saving this proof (button *Export* or *File* $\to$ *Export Equational Proof*) it can be certified by CeTA within a fraction of a second. The workflow for equational disproofs, e.g., that $\mathsf{g}(\mathsf{f}(x,y)) \approx \mathsf{f}(\mathsf{g}(y),\mathsf{g}(x))$ does not follow from the axioms of group theory, is similar.

In addition to the export of equational proofs also the completion proof itself can be exported by KBCV (*File* $\to$ *Export Completion Proof*). Again CeTA can certify it within one second.

To run KBCV in automatic completion mode the flag `-a` is used while `-e` runs the automatic equational logic proof mode. The `-p` flag triggers certifiable output. In equational logic proof mode the default is to output a conversion proof using the history. To output a proof tree and a full conversion the flags `-pt` and `-cv` are used, respectively.

The tool MKBTT runs standard multi-completion by default, using termination tools as backend. Alternatively, it can be set to ordered completion (option `-o`) or normalized completion mode (option `-n`). Note that the latter is a generalization of AC completion. The option `-cert` triggers certifiable output. In case of standard completion this means that MKBTT outputs a completion proof upon success. If the tool is run in ordered completion mode and the input contains a goal which MKBTT can prove, it outputs an entailment proof. The format of this proof can be controlled by the option `-p` followed by one of `cpfconv`, `cpftree`, and `cpfsub` for conversions, proof trees, and history proofs, respectively. In each of these settings an entailment disproof is output if a complete system was derived and the goal equation is not joinable. When MKBTT is run in normalized completion mode with an input containing a goal, the option `-p cpfsub` triggers the output of an entailment proof if the goal could be proved. Note that CeTA certifies neither ground completeness nor AC convergence. Thus, when run in ordered or normalized completion mode with the certification flag, MKBTT only produces entailment proofs but no (ground, AC) completeness proofs and consequently also no dis-proofs.

6. Experiments

The experiments described in this section were conducted with help of the StarExec[4] project. Each tool plus problem instance was given to a single 64 bit GNU/Linux machine with an Intel® Xeon® E5-2609 processor with four cores clocked at 2.40 GHz and 256 GB of memory. The timeout for each problem was set to 10 minutes.

First we performed experiments for completion proofs involving the tools KBCV as well as MKBTT based on 115 problems from the literature (see the website). The following flags were set for the two tools

```
./kbcv -a -p -s 600 -m "./ttt2 -cpf -ext xml - 1" file
./mkbtt -cert -yesno -t 600 file
```

The detailed results are available at

`http://cl-informatik.uibk.ac.at/experiments/2015/cpp`

Both tools were using internal termination methods together with calls to T$_2$ 1.16 running the certified strategy. KBCV could complete 89 systems while MKBTT obtained a score of 86 systems, both tools together succeeded on 99 systems. All proofs could be certified by CeTA 2.18.

In our first tests we considered two possibilities to ensure $\leftrightarrow^*_{\mathcal{R}} \subseteq \leftrightarrow^*_{\mathcal{E}}$: While KBCV 1.6 performs the recall phase to explicitly construct $\ell \leftrightarrow^*_{\mathcal{E}} r$ for each $\ell \to r \in \mathcal{R}$, from version 1.7 onward KBCV just exports the relevant history entries, which are used as auxiliary equations. Hence it is not surprising that not all systems completed by KBCV 1.6 could be certified. For two systems (`TPTP_GRP487-1_theory` and `TPTP_GRP_490-1_theory`) the recall phase did not terminate within the time limit and for other systems (`LS94_P1`, `TPTP_GRP_481-1_theory`, `TPTP_GRP_486-1_theory`, `TPTP_GRP_490-1_theory`) the certificate was too large (365 MB, 230 MB, 406 MB, 581 MB) for CeTA. However, when using auxiliary equations all proofs could be computed and certified (typically within a second) for KBCV as well as MKBTT. Hence further optimization of the proof format seems dispensable.

For experiments on entailment (dis-)proofs we considered the 195 systems from the UEQ category of CASC.[5] In our experiments we also interpret negated conjectures as goal equations. We set the following flags for KBCV and MKBTT:

```
./kbcv -e -p -s 600 -m "./ttt2 -cpf -ext xml - 1" file
./mkbtt -o -p cpfsub -cert -yesno -t 600 file
```

In MKBTT we chose ordered completion for these experiments since this mode is best suited to entailment proofs. The detailed results are available online. KBCV was able to find 106 entailment proofs and 3 entailment dis-proofs while MKBTT found 117 entailment proofs and one entailment dis-proof. Both tools together could solve 142 instances out of the 195 problems. Because unabridged proof trees and conversion proofs turned out to easily

[4] `http://www.starexec.org`

[5] `http://www.cs.miami.edu/~tptp/CASC/`

reach sizes of several hundred MB resulting in more timeouts and hence less answers both tools use the history (as described in Section 4) to keep the proofs as small as possible. Again all of the found proofs have been certified by CeTA.

7. Conclusion

In this paper we have introduced recording completion facilitating the construction of entailment (dis-)proofs and the certification of Knuth-Bendix completion. We have implemented recording completion in KBCV and MKBTT to generate proof certificates which can be checked with CeTA. The need for certified proofs in this context is demonstrated by the following example: The TRS reported in [19] for proof transformation axioms (ES `WS06_proofreduction` in our problem set) was claimed to be complete for this theory but is actually not confluent. We have obtained a (certified) completion proof for this system, making its practical use for algebraic proof mining more reliable.

We are not aware of other completion based provers (besides C*i*ME3 [6]) that yield sufficient details for the certification of their output. C*i*ME3 implements an annotated version of ordered completion [5] similar to the ordered completion method used in MKBTT. The critical difference to our approach is that the history is not saved as a stand-alone component but directly integrated into terms, equations, and rules during the process of completion. Hence a term t comes with an original version t^0, a current version t^*, and a reduction sequence from t^0 to t^*. Similarly an equation $s \approx t$ also contains all intermediate (rewrite) steps that show that both terms are equal. While the approach presented in [5] also supports auxiliary equations to facilitate shorter proofs, this is not the case for the approach from [6]. As illustrated by our experiments, this feature is indeed crucial to success in many cases. On the other hand, [6] can output a proof before a complete rewrite system was derived, which is not the case for [5]. Hence, to the best of our knowledge our approach is the only one to produce certifiable equational proofs from completion runs which combines these two key properties. Moreover, the approaches from [5, 6] are restricted to entailment proofs and cannot certify the results of completion runs, and are not applicable in AC completion runs.

We also mention that resolution based ATPs (automated theorem provers) are already used for finding and certifying proofs. Closest to our approach is Ivy [12], which checks the detailed proof steps reported by Prover9 in ACL2/HOL Light. A slightly different strategy is persued by the *hammer approach* [4, 9], which employs the axioms or proof steps used in a refutation proof (found by an external ATP) and then tries to find a certified proof using a variant of the Metis ATP whose inferences go through Isabelle's kernel. While in our setting a (correct) proof can always be certified, the proof replay in the hammer approach may fail (if incomplete transformations are used). Finally, we mention the GDV verifier [17], which can check TPTP proofs independent of the employed prover. But rather than being based on a trusted proof assistant, it employs cross-verification in that the proof steps are checked by other ATPs.

As future work we want to study the length of (equational) proofs generated by recording completion. Earlier work [14] might be helpful, although it only considers the length of the rewrite proof. Another possible direction is the integration of our findings into the hammer approach.

Acknowledgments We thank Cezary Kaliszyk and René Thiemann for helpful comments and the latter also for the formalization in Isabelle.

References

[1] Franz Baader and Tobias Nipkow. *Term Rewriting and All That*. Cambridge University Press, New York, USA, August 1999.

[2] Leo Bachmair, Nachum Dershowitz, and David Plaisted. Completion without failure. In *Resolution of Equations in Algebraic Structures, Vol. 2: Rewriting Techniques*, pages 1–30, 1989.

[3] Garrett Birkhoff. On the structure of abstract algebras. *Mathematical Proceedings of the Cambridge Philosophical Society*, 31(4):433–454, 1935.

[4] Sascha Böhme and Tobias Nipkow. Sledgehammer: Judgement day. In Jürgen Giesl and Reiner Hähnle, editors, *IJCAR*, volume 6173 of *Lecture Notes in Artificial Intelligence*, pages 107–121. Springer, 2010.

[5] Evelyne Contejean and Pierre Corbineau. Reflecting proofs in first-order logic with equality. In Robert Nieuwenhuis, editor, *CADE*, volume 3632 of *Lecture Notes in Artificial Intelligence*, pages 7–22. Springer, 2005.

[6] Evelyne Contejean, Pierre Courtieu, Julien Forest, Olivier Pons, and Xavier Urbain. Automated certified proofs with C*i*ME3. In Manfred Schmidt-Schauß, editor, *RTA*, volume 10 of *Leibniz International Proceedings in Informatics*, pages 21–30. Schloss Dagstuhl, 2011.

[7] Jean-Marie Gaillourdet, Thomas Hillenbrand, Bernd Löchner, and Hendrik Spies. The new WALDMEISTER loop at work. In *CADE*, volume 2741 of *Lecture Notes in Artificial Intelligence*, pages 317–321, 2003.

[8] Gérard Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of the ACM*, 27(4):797–821, 1980.

[9] Cezary Kaliszyk and Josef Urban. PRocH: Proof reconstruction for HOL light. In Maria Paola Bonacina, editor, *CADE*, volume 7898 of *Lecture Notes in Artificial Intelligence*, pages 267–274. Springer, 2013.

[10] Donald E. Knuth and Peter P. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, New York, 1970.

[11] Claude Marché. Normalized rewriting: An alternative to rewriting modulo a set of equations. *J. Symb. Comp.*, 21(3):253–288, 1996.

[12] William McCune and Olga Shumsky. Ivy: A preprocessor and proof checker for first-order logic. In *Computer-Aided Reasoning: ACL2 Case Studies*, chapter 16. Kluwer Academic Publishers, 2000.

[13] Gerald E. Peterson and Mark E. Stickel. Complete sets of reductions for some equational theories. *Journal of the ACM*, 28(2):233–264, 1981.

[14] David Plaisted and Andrea Sattler-Klein. Proof lengths for equational completion. *Information and Computation*, 125(2):154–170, 1996.

[15] Christian Sternagel and René Thiemann. Formalizing Knuth-Bendix orders and Knuth-Bendix completion. In Femke van Raamsdonk, editor, *RTA*, volume 21 of *Leibniz International Proceedings in Informatics*, pages 287–302. Schloss Dagstuhl, 2013.

[16] Thomas Sternagel and Harald Zankl. KBCV – Knuth-Bendix completion visualizer. In Bernhard Gramlich, Dale Miller, and Ulrike Sattler, editors, *IJCAR*, volume 7364 of *Lecture Notes in Artificial Intelligence*, pages 530–536, 2012.

[17] Geoff Sutcliffe. Semantic derivation verification. *Int. J. Artif. Intell. Tools*, 15(6):1053–1070, 2006.

[18] René Thiemann and Christian Sternagel. Certification of termination proofs using CeTA. In Stefan Berghofer, Tobias Nipkow, Christian Urban, and Makarius Wenzel, editors, *TPHOLs*, volume 5674 of *Lecture Notes in Computer Science*, pages 452–468. Springer, 2009.

[19] Ian Wehrman and Aaron Stump. Mining propositional simplification proofs for small validating clauses. In *Proc. 3rd PDPAR*, volume 144 of *ENTCS*, pages 79–91, 2005.

[20] Ian Wehrman, Aaron Stump, and Edwin M. Westbrook. Slothrop: Knuth-Bendix completion with a modern termination checker. In Frank Pfenning, editor, *RTA*, volume 4098 of *Lecture Notes in Computer Science*, pages 287–296. Springer, 2006.

[21] Sarah Winkler, Haruhiko Sato, Aart Middeldorp, and Masahito Kurihara. Multi-completion with termination tools. *J. Autom. Reasoning*, 50(3):317–354, 2013.