Outermost Termination via Contextual Dependency Pairs

Bernhard Gramlich
TU Vienna, gramlich@logic.at

Felix Schernhammer*
TU Vienna, felixs@logic.at

1 Introduction and Overview

Recently, the problem of proving outermost termination has been addressed mainly by methods relying on transformations ([4, 1, 3]). Here we describe a more direct approach inspired by the dependency pair (DP) framework of [2]. The basic idea is to enrich dependency pairs by an additional component, namely the calling context of the corresponding recursive function call. Then one can use the additional contextual information to model DP chains adhering to certain strategies (e.g. the outermost strategy). Additionally, existing methods of the ordinary DP approach can partly be reused.

Building upon this framework of contextual DPs, we describe a DP processor exploiting the additional contextual information. Basically, this processor analyzes nested contexts accumulated by consecutive DPs on DP chain candidates for (certain) redexes. If such a redex is found, the chain candidate is not a proper chain. Finally, we provide some empirical evaluation of our approach.

2 Contextual Dependency Pairs

The central observation of the (ordinary) dependency pair approach is that, given a non-terminating rewrite system $\mathscr{R}$, there exists an infinite reduction sequence (starting w.l.o.g. with a root reduction step), such that no redex contracted in this sequence contains a non-terminating proper subterm. Such reduction sequences roughly correspond to minimal dependency pair chains whose (non-)existence is analyzed in the DP framework. In the case of outermost rewriting the above observation does not hold.

Example 1. *Consider the TRS $\mathscr{R}$ consisting of the rules*

$$a \to f(a) \qquad f(x) \to g(x)\ .$$

$\mathscr{R}$ is not outermost terminating: $a \to f(a) \to g(a) \to g(f(a)) \to g(g(a)) \to \dots$ Here, it is crucial to reduce the term $f(a)$ at the root position, although it contains the term a as proper subterm which is not outermost terminating.

Example 1 shows that in the case of outermost rewriting it is sometimes crucial to allow reductions whose redexes properly contain outermost non-terminating terms. Hence, we restrict our attention to a restricted form of outermost termination, namely outermost termination in a context.

Definition 1 (outermost termination in a context)**.** *Let $\mathscr{R}$ be a TRS. A term s is* outermost terminating in context $C[\Box]_p$ *if $C[s]_p$ does not admit an infinite outermost reduction sequence where each redex contracted occurs at, below or parallel to p and where infinitely many of these steps are at or below p.*

Indeed, given a rewrite system $\mathscr{R}$, there exists an infinite outermost reduction sequence (starting w.l.o.g. with a root reduction step) contracting only redexes whose proper subterms are all outermost terminating in their respective contexts, whenever $\mathscr{R}$ is outermost non-terminating.

In order to analyze outermost termination in an adapted dependency pair framework, we enrich the dependency pairs by an additional component, namely the context in which the corresponding recursive

*The author has been supported by the Austrian Academy of Sciences under grant 22.361.

function call takes place. Informally, this amounts to an extended *contextual* version of dependency pairs which incorporates the full information of the given rules (especially the complete right-hand sides) in the form of associated contexts, but which still enables the typical DP-based reasoning.

Definition 2 (*contextual* dependency pairs)**.** *Let $(\mathscr{F},R)$ be a TRS where the signature is partitioned into defined symbols $\mathscr{D}$ and constructors. The set of (extended)* contextual dependency pairs (CDPs) *$CDP(\mathscr{R})$ is given by $DP_c(\mathscr{R}) \uplus V_c(\mathscr{R}) \uplus A_c(\mathscr{R}) \uplus S_c(\mathscr{R})$, where*

$$\begin{aligned}
DP_c(\mathscr{R}) &= \{l^\# \to r|_p^\# \ [c] \mid l \to r \in R, p \in Pos_{\mathscr{D}}(r), c = r[\Box]_p\} \\
V_c(\mathscr{R}) &= \{l^\# \to T(r|_p) \ [c] \mid l \to r \in R, r|_p = x \in Var, c = r[\Box]_p\} \\
A_c(\mathscr{R}) &= \{T(f(x_1,\dots,x_{ar(f)})) \to f^\#(x_1,\dots,x_{ar(f)}) \ [\Box] \mid l \to r \in R, root(r|_p) = f \in \mathscr{D}\} \\
S_c(\mathscr{R}) &= \{T(f(\vec{x})) \to T(x_i)[f(\vec{x})[\Box]_i] \mid \vec{x} = x_1,\dots,x_{ar(f)}, l \to r \in R, root(r|_p) = f, i \in \{1,\dots,ar(f)\}\}\,.
\end{aligned}$$

Here, T is a new auxiliary function symbol (the token *symbol for "shifting attention"). We call $V_c(\mathscr{R})$* variable descent *CDPs, $S_c(\mathscr{R})$* shift *CDPs and $A_c(\mathscr{R})$* activation *CDPs.* Contextual *rules of the shape $l \to r\ [c]$ can be interpreted as $l \to c[r]$ when used as rewrite rules. Slightly abusing notation, for a set $\mathscr{P}$ of such contextual rewrite rules (i.e. a* contextual TRS*) we denote by $\to_{\mathscr{P}}$ the corresponding induced ordinary rewrite relation.*

Example 2. *Consider the TRS $\mathscr{R}$ of Example 1; $CDP(\mathscr{R})$ consists of:*

$$\begin{array}{cccc}
a^\# \to a^\#[f(\Box)] & a^\# \to f^\#(a)[\Box] & f^\#(x) \to T(x)[g(\Box)] & T(a) \to a^\#[\Box] \\
T(f(x)) \to f^\#(x)[\Box] & T(g(x)) \to g^\#(x)[\Box] & T(f(x)) \to T(x)[f(\Box)] & T(g(x)) \to T(x)[g(\Box)]\,.
\end{array}$$

Note that the restrictions imposed by the outermost strategy are not reflected in the definition of $CDP(\mathscr{R})$. Instead the strategy plays an important role in the definition of outermost CDP chains.

Definition 3 (outermost CDP problem)**.** *An* outermost contextual dependency pair problem *(O-CDP problem) is a triple $(\mathscr{P},\mathscr{R},T)$ where $\mathscr{P}$ is a contextual TRS, $\mathscr{R}$ a TRS and T is a designated function symbol not occurring in the signature of $\mathscr{R}$.*

Definition 4 (outermost CDP chain)**.** *Let $(\mathscr{P},\mathscr{R},T)$ be an O-CDP problem. A sequence $s_1 \to t_1\ [c_1], s_2 \to t_2\ [c_2], \dots$ of CDPs is an outermost $(\mathscr{P},\mathscr{R},T)$-CDP chain if there exists a substitution σ, such that*

$$\begin{array}{rcl}
s_1\sigma & \to_{\mathscr{P}} & c_1[t_1\sigma]_{p_1} = c_1'[t_1\sigma]_{p_1'} \\
\xrightarrow{\not\leq p_1' *}_{\mathscr{R}} c_1''[s_2\sigma]_{p_1'} & \to_{\mathscr{P}} & c_1''[c_2[t_2\sigma]_{p_2}]_{p_1'} = c_2'[t_2\sigma]_{p_2'} \\
\xrightarrow{\not\leq p_2' *}_{\mathscr{R}} c_2''[s_3\sigma]_{p_2'} & \to_{\mathscr{P}} & c_2''[c_3[t_3\sigma]_{p_3}]_{p_2'} = c_3'[t_3\sigma]_{p_3'} \ \dots
\end{array}$$

where $c_i' = c_{i-1}''[c_i]$, $p_i' = p_{i-1}'.p_i$ and the $\mathscr{R}$-reduction $c_i'[t_i\sigma]_{p_i'} \xrightarrow{\not\leq p_i' *}_{\mathscr{R}} c_i''[s_{i+1}\sigma]_{p_i'}$ is empty (i.e., $c_i'[t_i\sigma]_{p_i'} = c_i''[s_{i+1}\sigma]_{p_i'}$) whenever $root(t_i) = T$ (i.e., the token symbol) for all $i \geq 1$. Moreover, for each single reduction $s \xrightarrow{q}_{\mathscr{P}} t$ or $s \xrightarrow{q}_{\mathscr{R}} t$ $erase(s)$ may not contain a redex above q. Here $erase(s)$ is obtained by replacing all marked dependency pair symbols $f^\#$ by their unmarked versions f and by replacing terms $T(s')$ by s'.[1] *An O-CDP chain is* minimal *if for every $i \geq 0$ every subterm of $c_i'[t_i\sigma]_{p_i'}$ at position $q > p_i'$ is outermost terminating in its context (here: $s_1\sigma = c_0''[s_1\sigma]_{p_0'}$ with $p_0' = \varepsilon$, $c_0''[\Box] = \Box$).*

We say an O-CDP problem is *finite* if it does not admit an infinite minimal O-CDP chain.

Theorem 1. *A TRS $\mathscr{R}$ is outermost terminating iff the O-CDP problem $(CDP(\mathscr{R}),\mathscr{R},T)$ is finite.*

[1] Formally, this definition of *erase* is not compatible with the strict modularity of the DP framework. However, to restore full modularity the *erase* function could be part of an O-CDP problem. We refrain from doing so for notational simplicity. Moreover, note that position q exists in *erase*(s) since there cannot be a T symbol above q in s.

Example 3. *Consider the TRS $\mathscr{R}$ from Example 1 ($CDP(\mathscr{R})$ is given in Example 2) and the corresponding O-CDP problem $P = (CDP(\mathscr{R}), \mathscr{R}, T)$. P admits an infinite O-CDP chain:*

$$a^{\#} \to f^{\#}(a)\ [\square], f^{\#}(x) \to T(x)\ [g(\square)], T(a) \to a^{\#}\ [\square], \ldots$$

In the following we use the notions of sound resp. complete CDP processors analogously to corresponding notions of [2].

3 Analyzing Contexts

In this section we develop a method to prove the absence of minimal O-CDP chains (for a given O-CDP problem $(\mathscr{P}, \mathscr{R}, T)$) by inspecting the nested contexts of consecutive CDPs of candidates for infinite O-CDP chains.

The main problem here is that these contexts are not constant according to Definition 4. However, the (nested) contexts are stable modulo reduction parallel to the position of the hole, i.e. they are only altered through reductions parallel to the hole position. Hence, if the nested contexts contain redexes (strictly) above the hole position that are oblivious to this kind of parallel reductions, then the corresponding sequence of CDPs does not form an O-CDP chain.

To characterize (or rather approximate) these redexes we consider only those $\mathscr{R}$-rules whose left-hand sides are linear and not overlapped by any other $\mathscr{R}$-rule (strictly) below the root. This left-linear overlay sub-system $\mathscr{R}_{ioc}$ of $\mathscr{R}$ will be used for Invalidating (potential) Outermost CDP Chains.

Given an O-CDP problem $(\mathscr{P}, \mathscr{R}, T)$ and a sequence of *CDPs* S: $s_1 \to t_1[c_1], \ldots, s_n \to t_n[c_n]$, if $c_1[\ldots c_n[erase(t_n)]_{p_n} \ldots]_{p_1}$ contains a redex w.r.t. $\mathscr{R}_{ioc}$ strictly above $p_1.\cdots.p_n$, S is not an O-CDP chain.

Example 4. *Consider the TRS of Example 1. There is a CDP α: $a^{\#} \to a^{\#}[f(\square)]$. Here, $\mathscr{R}_{ioc} = \mathscr{R}$. Considering the sequence of CDPs consisting only of α (i.e. a CDP-sequence of length 1), the term $f(erase(a^{\#})) = f(a)$ is matched by the lhs $f(x)$ (strictly) above position 1 (namely at ε) and hence this sequence of CDPs (i.e. the CDP α) cannot be part of any infinite O-CDP chain.*

Now, in order to effectively check whether the nested contexts of a sequence of CDPs contain a redex above the hole position, we describe the possible nested contexts by a tree automaton. This is to say that, given a sequence of dependency pairs $s_1 \to s_n[c_1], \ldots, s_n \to t_n[c_n]$, we construct a tree automaton $\mathscr{A}$ accepting all terms $c_1[c_2[\ldots c_n[erase(t_n)]\ldots]]\sigma$ where σ is some substitution.

Moreover, we construct another tree automaton $\mathscr{B}$ that accepts *all* terms that are matched by any rule from $\mathscr{R}_{ioc}$. Then, the idea roughly is to check whether the language accepted by $\mathscr{A}$ is a sublanguage of $\mathscr{B}$ and thus to conclude that the sequence of CDPs is not a proper O-CDP chain.

We discuss the construction of the involved automata in more detail, starting with the one that accepts a term if some subterm of it is matched by the left-hand side of some rewrite rule. Starting from an O-CDP problem $(\mathscr{P}, \mathscr{R}, T)$, the signature Σ' of the automaton we are going to construct is the signature of $\mathscr{R}$ plus $\{H, A\}$ where H is a new function symbol of arity one and A is a new constant. The role of the symbol H is to indicate the positions in accepted terms where no reductions may take place (because there is a more outer redex), and A is needed since the automata we use work on ground terms only.

Definition 5 (FP automaton)**.** *Let $\mathscr{R}$ be a left-linear TRS over a signature Σ and let $\Sigma' = \Sigma \uplus \{H, A\}$ where H is unary and A is a constant. The* forbidden pattern automaton *$FPA(\mathscr{R})$ is given by $\bigcup_{l \to r \in \mathscr{R}} \mathscr{A}^l$ where $\mathscr{A}^l$ is the automaton accepting ground terms having subterms of the shape $l\sigma[H(l\sigma|_p)]_p$ ($p > \varepsilon$).*

Example 5. *Consider the TRS $\mathscr{R}$ given by*

$$a \to f(b) \qquad b \to g(b) \qquad b \to g(a) \qquad f(g(x)) \to \bot\,.$$

$FPA(\mathscr{R})$ consists of the following transitions.

$$\begin{array}{ccccccc} a \to q_1 & A \to q_1 & b \to q_1 & f(q_1) \to q_1 & g(q_1) \to q_1 & H(q_1) \to q_2 & f(q_2) \to q_2 \\ g(q_2) \to q_2 & g(q_2) \to q_3 & g(q_1) \to q_3' & H(q_3') \to q_3 & f(q_3) \to q_{end} & f(q_{end}) \to q_{end} & g(q_{end}) \to q_{end} \end{array}$$

The state q_{end} is the unique end state. Here, state q_1 corresponds to arbitrary terms without any occurrence of H and q_2 corresponds to arbitrary terms with an occurrence of H. Moreover, q_3 either corresponds to g-rooted terms containing an H or instances of $H(g(x))$ while q_3' corresponds to g-rooted terms not containing an H.

Next, we discuss the construction of an automaton accepting terms obtained by nesting contexts of subsequent CDPs in sequences of CDPs.

Definition 6 (CDP automaton). *Let $(\mathscr{P},\mathscr{R},T)$ be an O-CDP problem for a given finite $\mathscr{R} = (\Sigma,R)$ and let $S\colon s_1 \to t_1,[c_1]\dots,s_n \to t_n[c_n]$ be a sequence of CDPs. The CDP automaton $DPA((c_1,\dots,c_n),\Sigma,t_n)$ corresponding to this CDP sequence is given by $\mathscr{A}_{c_1[\dots c_n[H(erase(t_n))]]}$. Here, $\mathscr{A}_t$ is the automaton accepting ground terms $t\sigma$ over the signature $\Sigma \cup \{H,A\}$ where $x\sigma = A$ for all $x \in Var(t)$.*

Replacing variables by just one new constant is justified, since we are only interested in matches by the left-hand sides of $\mathscr{R}_{ioc}$ (which is left-linear).

Theorem 2 (analyzing nested contexts). *Let $(\mathscr{P}, \mathscr{R},T)$ be an O-CDP problem for a given finite $\mathscr{R} = (\Sigma,R)$ and let $S\colon s_1 \to t_1,[c_1]\dots,s_n \to t_n[c_n]$ be a sequence of CDPs. If $L(DPA((c_1,\dots,c_n),\Sigma,t_n)) \subseteq L(FPA(\mathscr{R}_{ioc}))$, then S cannot be part of an infinite O-CDP chain.*

In order to use Theorem 2 in termination proofs and in particular in the DP framework, we have to consider candidates for CDP chains in a complete way. This is to say that for one CDP π we consider all potential sequences of CDPs that start from π and eventually use it again. If no such sequence can be part of an infinite CDP chain, then it is sound to delete π.

The rest of this section is concerned with describing the (in general infinitely many) candidate chains, or, rather, the corresponding nested contexts, in a finite way. Starting from existing dependency graph approximations, we identify cycles in this graph and describe them as combinations of *minimal cycles*. Here, a cycle is a sequence of nodes with identical start and end node, which is minimal if each node except the start node occurs at most once and the start node occurs only at the beginning and at the end.

By $node(C)$ we denote the start node of cycle C. Arbitrary cycles are combinations of minimal cycles. These combinations can be represented by trees called *minimal cycle combinations*.

Definition 7 (minimal cycle combination). *Let $\mathscr{G}$ be a (finite) graph and let $C_1,\dots,C_n$ be the set of minimal cycles of $\mathscr{G}$. A* minimal cycle combination *(MCC) is a tree whose nodes are minimal cycles and whose edges $C_i - C_j$ are labeled by a positive integer i from $\{1,\dots,|C_i|\}$ if C_j is a (minimal) cycle for the i^{th} node of C_i. A minimal cycle combination is* hierarchical *if $node(C_i) \notin C_j$ whenever C_j is (in the tree) below C_i.*

We say an MCC *is for node n* if its root node is a cycle with start node n. The depth of a hierarchical MCC is bounded by the number of nodes in the graph, and thus finite in most interesting cases when considering DP graph approximations. Every MCC corresponds to a set $Cyc(M)$ of cycles obtained by recursively combining the contained minimal cycles.

Example 6. *Consider the TRS $\mathscr{R}$ of Example 5 and the subset $\{\alpha\colon a^\# \to b^\#[f(\Box)], \beta\colon b^\# \to b^\#[g(\Box)], \gamma\colon b^\# \to a^\#[g\Box]\}$ of CDP($\mathscr{R}$). A dependency graph approximation might contain the edges (α,β), (α,γ), (β,γ), (β,β), (γ,α). The minimal cycles in this graph are $C_1 = (\alpha,\beta,\gamma,\alpha)$, $C_2 = (\alpha,\gamma,\alpha)$ and $C_3 = (\beta)$ (modulo rotations). For α there are two MCCs, namely M_1 consisting of only one (root) node C_2 and M_2 consisting of the node C_1 with one child C_3 connected by an edge with label 2. We have $Cyc(M_1) = \{C_2\}$ and $Cyc(M_2) = \{(\alpha,\beta^n,\gamma,\alpha) \mid n \geq 1\}$.*

For a finite graph $\mathscr{G}$ there is only a finite number of hierarchical MCCs having different sets of associated cycles. Moreover, for every given cycle C, there is a hierarchical MCC M such that $C \in Cyc(M)$. Since a cycle in a DP graph approximation corresponds to a sequence of CDPs, it also corresponds to a sequence of nested contexts. We denote the set of nested contexts corresponding to the cycles $Cyc(M)$ of an MCC M as $ctxs(M)$. Slightly abusing notation we write $ctxs(M)[t]$ for the set of terms given by $\{c[t] \mid c \in ctxs(M)\}$. Now, given a hierarchical MCC M w.r.t. a CDP graph and a term t, we can construct a tree automaton $MCCA(M,t)$ accepting all terms from $ctxs(M)[H(t)]$ (where the variables have been replaced by A). This is the crucial step of describing the infinite set $ctxs(M)[H(t)]$ finitely by $MCCA(M,t)$.

Example 7. *Consider the TRS, CDPs and MCCs of Example 6. $MCCA(M_2,a)$ consists of the following transitions (the contexts of $ctxs(M_2)[a]$ have the shape $f(g^+(g(f(a))))$ where the end state is q_{end}.*

$$\begin{array}{ccccc} A \to q_1 & a \to q_1 & H(q_1) \to q_2 & f(q_2) \to q_3 & g(q_3) \to q_4 \\ g(q_4) \to q_5 & g(q_5) \to q_5 & f(q_5) \to q_{end} & & \end{array}$$

Theorem 3 (Context Processor)**.** *Let $\Pi = (\{s \to t\ [c]\} \uplus \mathscr{P},\ \mathscr{R}, T)$ be a CDP problem with $\mathscr{R} = (\Sigma, R)$ finite. If for every hierarchical MCC M for $s \to t$ in some CDP graph approximation we have $L(MCCA(M),t) \subseteq L(FPA(\mathscr{R}_{ioc}))$, then Π is finite if and only if $(\mathscr{P}, \mathscr{R}, T)$ is finite.*

Example 8. *Consider the TRS $\mathscr{R}$, CDPs and MCCs of Example 6, the automata $MCCA(M_2,a)$ and $MCCA(M_1,a)$ (the former is given in Example 7) and the automaton $FPA(\mathscr{R}_{ioc})$ from Example 5 (note that here $\mathscr{R}_{ioc} = \mathscr{R}$). We have $L(MCCA(M_i),a) \subseteq L(FPA(\mathscr{R}_{ioc}))$ for $i \in \{1,2\}$. Hence, it is sound to delete α. Note that β cannot be deleted and, indeed, the infinite sequence of CDPs consisting of only β CDPs is an infinite O-CDP chain.*

4 Evaluation and Conclusion

In a prototypical implementation of the CDP framework we performed some benchmark tests on the TRSs of the outermost section of the TPDB. In addition to the context processor of Section 3 we also used various standard DP processors not relying on minimality of DP problems, such as the polynomial ordering processors without usable rules. Out of 133 potentially terminating TRSs in the outermost category of the TPDB our implementation was able to prove outermost termination of 60 systems. To put this in perspective, in the termination competition of 2009 the three participating tools obtained 72 (Jambox), 46 (TrafO) and 27 (AProVE) successful proofs of outermost termination.

The modularity of the CDP framework is inherited from the ordinary DP framework. Hence, also in this extended framework new processors based on new ideas and tailored for specific classes of problems can be modularly added. Regarding future work, among generalizations of standard DP processors for the adapted framework an outermost usable rules criterion based on the restricted minimality property of minimal O-CDP chains would be very interesting.

References

[1] J. Endrullis and D. Hendriks. From outermost to context-sensitive rewriting. In *Proc. 20th International Conference on Rewriting Techniques and Applications, RTA'09*, pages 305–319, 2009. Springer-Verlag.

[2] J. Giesl, R. Thiemann, P. Schneider-Kamp, and S. Falke. Mechanizing and improving dependency pairs. *J. Autom. Reason.*, 37(3):155–203, 2006.

[3] M. Raffelsieper and H. Zantema. A transformational approach to prove outermost termination automatically. *Electron. Notes Theor. Comput. Sci.*, 237:3–21, 2009.

[4] R. Thiemann. From outermost termination to innermost termination. In *Proc 35th Conference on Current Trends in Theory and Practice of Computer Science, SOFSEM'09*, pages 533–545, 2009. Springer-Verlag.