Rewriting techniques, 6: orthogonality, decreasing diagrams

19–12–2019

Exercise 1:
Are the rewrite relations of the following TRSs confluent?
(a) \(a(x, x) \rightarrow m, \quad r \rightarrow s(r), \quad a(x, s(x)) \rightarrow p:\)

Solution:
The TRS is locally confluent (no non-trivial critical pairs), yet it is not confluent. For example the term \(a(r, r)\) has two distinct normal forms \(m\) and \(p\).

(b) \(a(x, x) \rightarrow m, \quad r \rightarrow s(r), \quad s(x) \rightarrow a(x, s(x))\)

Solution:
First, we notice that in 4 steps we can reduce \(r\) to \(m\):
\[
r \rightarrow s(r) \rightarrow a(r, s(r)) \rightarrow a(s(r), s(r)) \rightarrow m
\]
So now we know that we can replace \(r\) with \(m\) in any term. Let’s therefore consider the term \(a(r, s(r))\), i.e. the second element in the chain of reductions shown above. This element can be reduced to \(a(m, s(m))\). This term cannot be reduced to \(m\) since the only reduction path is the infinite sequence
\[
a(m, s(m)) \rightarrow a(m, a(m, s(m))) \rightarrow a(m, a(m, a(m, s(m)))) \rightarrow \ldots
\]
We conclude that the TRS is not confluent since from \(r\) i can reach both \(a(m, s(m))\) and \(m\).

There are many automated tools dealing with rewriting systems. A descriptive list can be found at https://www.jaist.ac.jp/~hirokawa/tool. We can mention

- TTT2 http://cl-informatik.uibk.ac.at/software/ttt2, termination checker,
- mkbTT http://cl-informatik.uibk.ac.at/software/mkbtt performs Knuth-Bendix completion.
- AProVE http://aprove.informatik.rwth-aachen.de is a system for automated termination (can prove termination of Prolog programs, TRSs, Haskell, . . .)
- Dedukti https://deducteam.github.io, a logical framework using rewriting as a computation model.

Termination tools often (at least AProVE and TTT2 do) use the syntax of the termination competition for input TRSs which can be found at https://www.lri.fr/~marche/tpdb/format.html. The main components are, where NL is the newline character,

```
spec ::= ( decl ) spec | ( decl )
decl ::= VAR idlist | RULES listofrules
idlist ::= id | id idlist
listofrules ::= rule | rule NL listofrules
rule ::= term -> term
term ::= id | id( ) | id(termlist)
termlist ::= term | term, termlist
```
Exercise 2:
Consider the following rewriting system

\[
\begin{align*}
0 + n & \rightarrow n \quad (1) \\
s + m + n & \rightarrow s(m + n) \quad (2) \\
m + 0 & \rightarrow m \quad (3) \\
m + s n & \rightarrow s(m + n) \quad (4) \\
\triangle 0 & \rightarrow 0 \quad (5) \\
\triangle (s n) & \rightarrow n + (\triangle n) \quad (6)
\end{align*}
\]

in Dedukti3 syntax,

\begin{verbatim}
constant symbol Nat : TYPE
symbol s : Nat => Nat
constant symbol z : Nat
set builtin "0" := z
set builtin "+1" := s

symbol + : Nat => Nat => Nat
set infix left 6 "+" := add

rule 0 + &m \rightarrow &m
and (s &m) + &m \rightarrow s (&m + &m)
and &m + 0 \rightarrow &m
and &m + (s &m) \rightarrow s (&m + &m)

symbol triangle : Nat => Nat
rule triangle 0 \rightarrow 0
and triangle (s &m) \rightarrow &m + (triangle &m)
compute triangle 45
\end{verbatim}

(a) What happens if lines 13 and 14 are placed before line 11 and 12 when the option keep-rule-order is used?

(b) Compute the complexity of triangle \( n \) in each case.

Solution:

1. + is defined by 3 and 4. First, \( \triangle n \rightarrow n + (n - 1 + (n - 2 + \cdots + (1 + 0) \cdots)) \) in \( n \) operation using 6 and 5.

The next rule to be used is 3 to transform \( 1 + 0 \rightarrow 1 \). The rightmost term is now \( 2 + 1 \). Rule 4 yields \( 2 + 1 \rightarrow s(2 + 0) \rightarrow s2 \). Using \( 4 \) \( n - 2 \) times will allow the generated \( s \) to traverse the term from right to left, resulting in \( s(n + (n - 1 + (n - 2 + \cdots + (3 + 2) \cdots))) \). Re iterating twice this last operation will exhaust the rightmost 2. The rightmost term is then 3, which will be exhausted calling 4 \( 3(n - 2) \) times.

Rigorously, we should prove by induction that if the rightmost term is \( j \), then reducing this term to zero costs \( j(n - j) \) operations approximately.

Assuming this, we deduce that

\[
T(\triangle n) = O \left( \sum_{j=0}^{n} j(n - j) \right) = O(n^3)
\]

2. + is now defined with 1 and 2. The first rule called is 6, \( \triangle n \rightarrow n + \triangle (n - 1) \).

Next, instead of reducing \( \triangle \) as above, calling 2 yields \( n + \triangle (n - 1) \rightarrow s((n - 1) + \triangle (n - 1)) \). Calling again the same rule \( n - 1 \) times results in

\[
s(s(\cdots s(\triangle (n - 1)) \cdots))
\]
Re-iterating this procedure \( n - 1 \) times yields the final result in

\[
T(\text{triangle } n) = O \left( \sum_{k=0}^{n} k \right) = O(n^2)
\]

**Exercise 3:**

**Definition 1 (Strong joinability).** Two terms \( t \) and \( u \) are strongly joinable if there are terms \( v \) and \( w \) such that \( t \rightarrow^* v \leftarrow^* u \) and \( t \rightarrow^* w \leftarrow^* u \).

(a) Give a TRS that is locally confluent but not confluent.

**Solution:**

\[
0 \rightarrow 1; 1 \rightarrow 0; 0 \rightarrow a; 1 \rightarrow b
\]

(b) Why does the critical pairs lemma give only local confluence?

**Solution:**

In the proof, the case of variable overlap needs several rewriting steps if the rule used is non right linear, we can thus only deduce local confluence.

(c) Prove that if every critical pair of a TRS \( R \) is strongly joinable, then \( R \) is strongly confluent.

**Solution:**

The proof is similar to the one of the critical pair lemma. The case where the two reductions are parallel is unchanged.

The second case of variable overlap is simplified by the linearity restriction (and allows to have confluence rather than local confluence).

Instances of critical pairs become strongly joinable because all critical pairs are strongly joinable (and the rewrite relation is stable by context and substitution).

(d) Is the following rewrite system confluent?

\[
\begin{align*}
\text{f} (\text{g} \, \text{x} \, \text{y}) & \rightarrow \text{g} (\text{f} \, (\text{j} \, \text{x}) \, \text{y}) \\
\text{g} \, \text{x}(\text{k} \, \text{y} \, \text{z}) & \rightarrow \text{i} (\text{l} \, \text{y}) \, (\text{h} \, \text{z} \, \text{x}) \\
\text{f} (\text{h} \, \text{x} \, \text{y}) & \rightarrow \text{h} \, \text{x}(\text{f} \, (\text{j} \, \text{y})) \\
\text{h} \, \text{x} \, (\text{f} \, (\text{j} \, \text{y})) & \rightarrow \text{f} \, (\text{h} \, \text{x} \, \text{y})
\end{align*}
\]

**Solution:**

There are two critical pairs

\[
f (\text{i} \, (\text{l} \, \text{y})(\text{h} \, \text{z} \, \text{x})) \quad \text{g} \, (\text{f} \, (\text{j} \, \text{x}) \, (\text{k} \, \text{y} \, \text{z})), \quad \text{f} \, (\text{f} \, (\text{h} \, \text{x} \, \text{y})), \quad \text{h} \, \text{x}(\text{f} \, (\text{j} \, \text{y})))
\]

and we have

\[
f (\text{f} \, (\text{h} \, \text{x} \, \text{y})) \rightarrow \text{f}(\text{h} \,(\text{f}(\text{j} \, \text{y})))) \leftarrow \text{h} \,(\text{f}(\text{j}(\text{j} \, \text{y}))))
\]

and

\[
f(\text{i} \,(\text{l} \, \text{y})(\text{h} \, \text{z} \, \text{x})) \rightarrow \text{i} \,(\text{l} \, \text{y})(\text{h} \,(\text{f}(\text{j} \, \text{x}))) \leftarrow \text{i}(\text{l} \, \text{y})(\text{h} \,(\text{f}(\text{j} \, \text{x})))) \leftarrow \text{g}(\text{f}(\text{j} \, \text{x}))(\text{k} \, \text{y} \, \text{z})
\]

(e) Is the system

\[
\begin{align*}
\text{h} \, \text{f} \, \text{a} \, \text{a} & \rightarrow \text{g} \, \text{a} \, \text{a} \\
\text{a} & \rightarrow \text{b} \\
\text{h} \, \text{g} \, \text{a} \, \text{a} & \rightarrow \text{h} \, \text{f} \, \text{a} \, \text{a} \\
\text{h} \, \text{x} \, \text{b} \, \text{y} & \rightarrow \text{h} \, \text{x} \, \text{y} \, \text{y} \\
\text{h} \, \text{x} \, \text{y} \, \text{b} & \rightarrow \text{h} \, \text{x} \, \text{y} \, \text{y}
\end{align*}
\]
confluent?

Solution:
No since \( h f b b \leftrightarrow^* h g b b \) but \( h f b b \downarrow h g b b \) does not hold (even though we have \( s \rightarrow^* \cdot \leftarrow^* t \) and \( s \rightarrow^* \cdot \leftarrow^* t \) for every critical pair).