Rewriting techniques, 5: completion, critical pairs

12–12–2019

Exercise 1:
(a) Find \( r_1 \) and \( r_2 \) such that \{ \( f \ (g \ x) \rightarrow r_1, g \ (h \ x) \rightarrow r_2 \) \} is confluent.

Solution:
There is one critical pair
\[
\begin{align*}
\text{\( r_1 \ [x := h \ x] \)} & \quad \text{f} \\
\text{\( r_2 \)} & \quad \text{f} \ (h \ x)
\end{align*}
\]
This critical pair is joinable if the two terms are convertible. Take for instance \( r_1 = f \ x \) and \( r_2 = h \ x \). Then we must verify that the system \{ \( f \ (g \ x) \rightarrow f \ x, g \ (h \ x) \rightarrow h \ x \) \} is terminating. An LPO with any precedence is enough.

(b) Show that the following string rewrite system is convergent

\[
\begin{align*}
f \ f & \rightarrow f \\
g \ g & \rightarrow \ f \\
f \ g \ f & \rightarrow g \\
g \ g \ g & \rightarrow \ f \ g
\end{align*}
\]
Can you determine the normal form of a term as a function of the numbers of \( f \)s and \( g \)s in it?

Solution:
Some non-trivial critical pairs,
\[
\begin{align*}
g \ g \ g & \\
g \ f & \\
g \ g & \\
g f & \\
g f & \\
g g f & \\
g g f & \\
\end{align*}
\]
The normal form of a term is \( g \) if there is an odd number of \( g \)s in it and \( f \) otherwise.

Exercise 2:
Consider the one-rule TRS \( R = \{ f \ (f \ x) \rightarrow g \ x \} \). Is it confluent?

Solution:
There is a critical pair,
\[
\begin{align*}
f \ (f \ (f \ x)) & \\
g \ (f \ x) & \\
f \ (g \ x)
\end{align*}
\]
which is not joinable. The system is therefore not confluent.
Deduce \[ \frac{E, R}{E \cup \{s \approx t\}, R} \] if \( s \leftarrow u \to t \)

Orient \[ \frac{E \cup \{s \approx t\}, R}{E, R \cup \{s \to t\}} \] if \( s > t \)

Delete \[ \frac{E \cup \{s \approx s\}, R}{E, R} \]

Simplify-identity \[ \frac{E \cup \{s \approx t\}, R}{E \cup \{u \approx t\}, R} \] if \( s \to_R u \)

R-Simplify-rule \[ \frac{E, R \cup \{s \to t\}}{E, R \cup \{s \to u\}} \] if \( t \to_R u \)

L-Simplify-rule \[ \frac{E, R \cup \{s \to t\}}{E \cup \{u \approx t\}, R} \] if \( s \to_R u \)

Figure 1: The inference rules for completion

The inference rules given in figure 1 provide a completion procedure operating on pairs \((E, R)\) where \(E\) is a finite set of identities and \(R\) is a finite set of rewrite rules. The procedures requires a reduction order for the orient rule. We write \( s \to_R u \) to express that \( s \) is reduced by a rule \( \ell \to r \in R \) such that \( \ell \) cannot be reduced by \( s \to t \). The dot in \( s \approx t \) means to indicate that the identity should be seen as an unordered pair \((s \approx t) \in E, R\) such that \( s \approx t \in E \).

A completion procedure is a program that accepts as input a finite set of identities \( E_0 \) and a reduction order \( > \), and uses the rules of Fig. 1 to generate a (finite or infinite) sequence

\[(E_0, R_0) \vdash (E_1, R_1) \vdash \cdots \]

where \( R_0 = \emptyset \). This sequence is called a run of the completion procedure on input \( E_0 \) and \( > \).

Exercise 3:
Complete the following systems with the abstract completion procedure,

(a) \( (x \cdot y) \cdot (y \cdot z) \approx y \)

Solution:
Using abstract completion,

\[
\begin{align*}
(x \cdot y) \cdot (y \cdot z) & \to y \\
y \cdot ((y \cdot w) \cdot z) & \to y \cdot w \\
(x \cdot (y \cdot z)) \cdot z & \to y \cdot z
\end{align*}
\]

\[
\emptyset, \{(x \cdot y) \cdot (y \cdot z) \approx y\}
\]

Deduce

\[
\{(x \cdot y) \cdot (y \cdot z) \approx y\}, \emptyset
\]

Orient

\[
\begin{align*}
(x \cdot y) \cdot (y \cdot z) & \to y \\
y \cdot ((y \cdot w) \cdot z) & \to y \cdot w, \\
(x \cdot (y \cdot z)) \cdot z & \to y \cdot z
\end{align*}
\]

\[
\emptyset
\]

\[
\emptyset, \{(x \cdot y) \cdot (y \cdot z) \approx y, y \cdot ((y \cdot w) \cdot z) \approx y \cdot w, (x \cdot (y \cdot z)) \cdot z \approx y \cdot z\}
\]

Deduce

\[
\begin{align*}
(x \cdot y) \cdot (y \cdot z) & \to y, \\
y \cdot ((y \cdot w) \cdot z) & \to y \cdot w, \\
(x \cdot (y \cdot z)) \cdot z & \to y \cdot z
\end{align*}
\]

\[
\emptyset
\]

\[
\emptyset, \emptyset
\]

\[
\emptyset, \{(x \cdot y) \cdot (y \cdot z) \approx y, y \cdot ((y \cdot w) \cdot z) \approx y \cdot w, (x \cdot (y \cdot z)) \cdot z \approx y \cdot z\}
\]

Delete

\[
\begin{align*}
(x \cdot y) \cdot (y \cdot z) & \to y, \\
y \cdot ((y \cdot w) \cdot z) & \to y \cdot w, \\
(x \cdot (y \cdot z)) \cdot z & \to y \cdot z
\end{align*}
\]

\[
\emptyset
\]

\[
\emptyset
\]

\[
\{(x \cdot y) \cdot (y \cdot z) \approx y, y \cdot ((y \cdot w) \cdot z) \approx y \cdot w, (x \cdot (y \cdot z)) \cdot z \approx y \cdot z\}
\]

with some more details on the “deduce” steps: for the first one, the critical pairs are

\[
\begin{align*}
(x \cdot (y \cdot z)) \cdot ((y \cdot z) \cdot (z \cdot w)) & \to (x \cdot (y \cdot z)) \cdot ((y \cdot w) \cdot z) \\
(x \cdot (y \cdot z)) \cdot z & \to y \cdot z \\
y \cdot ((y \cdot w) \cdot z) & \to y \cdot w
\end{align*}
\]

\[
(5)
\]
and for the second one,

\[(x \cdot y) \cdot (y \cdot z) \cdot z \quad y \cdot ((y \cdot w) \cdot (w \cdot z))\]

\[\begin{array}{c}
\begin{array}{c}
\quad y \cdot z \\
\quad y \cdot w \end{array} \\
\quad y \cdot z \\
\quad y \cdot w
\end{array}\]

There is no rule to apply on the last system.

(b)

\[f \ a \ b \approx g \ (g \ a)\]

\[b \approx g \ (g \ b)\]

\[a \approx g \ b\]

\[f \ b \ a \approx f \ a \ (g \ a)\]

**Solution:**

\[
\begin{array}{c}
\{f \ b \ a \approx f(a(g \ a))\}, \\
\{f \ a \ b \approx f(a(g \ a))\}, \\
\{f \ (g \ b) \ b \approx g \ (g \ b)\}
\end{array}
\]

Orient

\[
\begin{array}{c}
\begin{array}{c}
a \rightarrow g \ b \\
g (g \ b) \rightarrow b \\
f \ a \ b \rightarrow g \ (g \ a)
\end{array} \\
\begin{array}{c}
a \rightarrow g \ b \\
g (g \ b) \rightarrow b \\
f \ a \ b \rightarrow g \ (g \ a)
\end{array}
\end{array}\]

Deduce

Simplify-identity

\[
\begin{array}{c}
\begin{array}{c}
f \ b \ (g \ b) \approx f \ (g \ b) \ (g \ (g \ b)) \\
f \ (g \ b) \ b \approx g \ (g \ (g \ b))
\end{array} \\
\begin{array}{c}
a \rightarrow g \ b \\
g (g \ b) \rightarrow b \\
f \ a \ b \rightarrow g \ (g \ a)
\end{array}
\end{array}\]

L-Simplify-rule

Simplify-identity

\[
\begin{array}{c}
\begin{array}{c}
f \ b \ (g \ b) \approx f \ (g \ b) \\
f \ (g \ b) \ b \approx g \ b
\end{array} \\
\begin{array}{c}
a \rightarrow g \ b \\
g (g \ b) \rightarrow b \\
f \ a \ b \rightarrow g \ b
\end{array}
\end{array}\]

Orient

Simplify-identity

\[
\begin{array}{c}
\begin{array}{c}
f \ b \ (g \ b) \approx f \ (g \ b) \\
f \ (g \ b) \ b \approx g \ b
\end{array} \\
\begin{array}{c}
a \rightarrow g \ b \\
g (g \ b) \rightarrow b \\
f \ a \ b \rightarrow g \ b
\end{array}
\end{array}\]

Delete

\[
\begin{array}{c}
\begin{array}{c}
a \rightarrow g \ b \\
g (g \ b) \rightarrow b \\
f \ a \ b \rightarrow g \ b
\end{array}
\end{array}\]

Orient

\[
\begin{array}{c}
\begin{array}{c}
a \rightarrow g \ b \\
g (g \ b) \rightarrow b \\
f \ a \ b \rightarrow g \ b
\end{array}
\end{array}\]

\[
\emptyset,
\begin{array}{c}
\begin{array}{c}
a \rightarrow g \ b \\
g (g \ b) \rightarrow b \\
f \ a \ b \rightarrow g \ b
\end{array}
\end{array}\]

with, for the “L-Simplify-rule”, \(f \ a \ b \rightarrow g \ (g \ a)\) simplified to \(f \ (g \ b) \ b \rightarrow g \ (g \ a)\), with \(a \rightarrow g \ b\) we have indeed \(a \sqsubseteq f \ a \ b\).

For the deduce,

\[
\begin{array}{c}
f \ a \ b
\end{array}\]

\[
\begin{array}{c}
f \ (g \ b) \ b \\
g \ (g \ a)
\end{array}\]
Exercise 4:
The goal of this exercise is to evince the necessity of the encompassment condition of the “L-Simplify-rule”. Consider the set (where all function symbols are unary, and application is right-associative, à la string-rewriting system) \( E_0 = \{ f \text{ g f } x \approx f \text{ g } x, g \text{ g } x \approx g \text{ x} \} \) with the lexicographic path order with precedence \( f > g \) and assume that the side condition of “L-Simplify-rule” is replaced by \( s \rightarrow R \ u \).

For an infinite run, we note \( R_\omega \) the set of identities that belong to some \( R_i \) and are never removed in later inference steps (more formally, \( R_\omega := \bigcup_{i \geq 0} \bigcap_{j \geq i} R_j \)).

Show that this procedure can generate a set of rules \( R_\omega \) that is not equivalent to the input set of identities. For this, we can show that the procedure can generate infinitely many rules \( \ell \rightarrow r_n \) with left-hand side \( \ell = f g f \text{ x} \) and right-hand sides \( r_n = f g^{2^n} \text{ x} \).

**Solution:**
We can start the following tree

\[
\begin{array}{c}
\{ f g f \approx f g \} \quad \text{Orient} \\
\emptyset, \{ f g f \rightarrow f g, g g \rightarrow g \} \quad \text{Deduce} \\
\{ g g \approx g g \}, R_1 \quad \text{Delete} \\
\emptyset, R_1 \\
\{ f g f \approx f g f \}, R_1 \quad \text{Deduce} \\
\{ f g f \approx f g f \}, R_1 \\
\{ f g \rightarrow f g \}, R_0 \quad \text{Simplify-Identity (twice)} \\
\emptyset, \{ f g \rightarrow f g \} \\
\{ f g \approx f g \}, R_0 \quad \text{Simplify-Identity} \\
\emptyset, R_0 \\
\{ f g f \approx f g f \}, R_0 \quad \text{L-Simplify-rule} \\
\emptyset, R_0 \\
\{ f g f \approx f g f \}, R_0 \quad \text{Simplify-Identity} \\
\emptyset, R_0 \\
\{ f g f \approx f g f \}, R_0 \quad \text{Simplify-Identity} \\
\emptyset, R_0 \\
\{ f g f \approx f g f \}, R_0 \quad \text{Orient} \\
\emptyset, R_0 \\
\end{array}
\]

Now assume we have generated \( \emptyset, \{ f g f \rightarrow f g^{2^n}, g g \rightarrow g \} \),

\[
\begin{array}{c}
\emptyset, \{ f g f \rightarrow f g^{2^n}, g g \rightarrow g \} \quad \text{Deduce} \\
\{ f g^{2^n} \approx f g \}, R_0 \quad \text{Simplify-Identity} \\
\{ f g^{2^n} \approx f g^{2^n+1} \}, R_0 \quad \text{Simplify-Identity} \\
\{ f g f \approx f g^{2^n+1} \}, R_0 \quad \text{Orient} \\
\emptyset, R_0 \\
\end{array}
\]

and we can once more eliminate the rule \( f g f \rightarrow f g^{2^n} \) as previously. Hence, we have \( R_\omega = \{ g g \approx g \} \) which is not equivalent to \( E_0 \).
Algorithm 1 Huet’s completion procedure

Require: A finite set $E$ of identities and a reduction order $>$

Ensure: A finite convergent (terminating and confluent) rewrite system $R$ equivalent to $E$ if the procedure terminates successfully, FAIL if the procedure terminates unsuccessfully

1: $R_0 \leftarrow \varnothing$; $E_0 \leftarrow E$; $i \leftarrow 0$; all identities of $E$ are unmarked
2: while $E_i \neq \varnothing$ or there is an unmarked rule in $R_i$ do
3: \hspace{1em} while $E_i \neq \varnothing$ do
4: \hspace{2em} Choose an identity $(s, t) \in E$ and reduce $s$ and $t$ to some $R_i$-normal forms $\tilde{s}$ and $\tilde{t}$
5: \hspace{2em} if $\tilde{s} = \tilde{t}$ then
6: \hspace{3em} $R_{i+1} \leftarrow R_i$; $E_{i+1} \leftarrow E_i \setminus \{(s, t)\}$; $i \leftarrow i + 1$
7: \hspace{2em} else if $\tilde{s} \neq \tilde{t}$ \& $\tilde{t} \neq \tilde{s}$ then
8: \hspace{3em} terminates with output FAIL
9: \hspace{2em} else
10: \hspace{3em} let $l$ and $r$ such that $\{l, r\} = \{\tilde{s}, \tilde{t}\}$ and $l > r$
11: \hspace{3em} $R_{i+1} \leftarrow \{(g, d) | (g, d) \in R_i \wedge g$ cannot be reduced with $l \rightarrow r \wedge \tilde{d}$ is a $R_i \cup \{(l, r)\}$-normal form of $d\} \cup \{(l, r)\}$
12: \hspace{3em} $(l, r)$ is not marked and $(g, d)$ is marked in $R_{i+1}$ iff $(g, d)$ is marked in $R_i$
13: \hspace{3em} $E_{i+1} \leftarrow (E_i \setminus \{(s, t)\}) \cup \{(g', d) | (g, d) \in R_i \wedge g$ can be reduced to $g'$ with $l \rightarrow r\}$
14: \hspace{3em} $i \leftarrow i + 1$
15: \hspace{2em} end if
16: \hspace{2em} end while
17: \hspace{1em} if there is an unmarked rule in $R_i$ then
18: \hspace{2em} let $(l, r)$ be such a rule
19: \hspace{3em} $R_{i+1} \leftarrow R_i$
20: \hspace{3em} $E_{i+1} \leftarrow \{(s, t) | (s, t)$ is a critical pair of $(l, r)$ with itself or with a marked rule in $R_i\}$
21: \hspace{3em} Mark $(l, r)$; $i \leftarrow i + 1$
22: \hspace{1em} end if
23: \hspace{2em} end while
24: \hspace{1em} return $R_i$

Exercise 5:

Apply the Huet’s completion procedure on the following set of identities, with the suitable reduction order:

\begin{align*}
(\{x \ast (y + z), (x \ast y) + (x \ast z), ((u + v) \ast w, (u \ast w) + (v \ast w))\} \text{ and } > \text{ the LPO with } \ast > +).
\end{align*}

Solution:

We only consider non trivial critical pairs.

From the LPO it holds that $x \ast (y + z) \triangleright_{lpo} (x \ast y) + (x \ast z)$ and $(u + v) \ast w \triangleright_{lpo} (u \ast w) + (v \ast w)$.

$E_0 = \{(x \ast (y + z), (x \ast y) + (x \ast z)), ((u + v) \ast w, (u \ast w) + (v \ast w))\}$ and $R_0 = \emptyset$. Select the first identity, that is already in normal form w.r.t. $R_0$. The procedure will compute $E_1 = \{((u + v) \ast w, (u \ast w) + (v \ast w))\}$ and $R_1 = \{(x \ast (y + z), (x \ast y) + (x \ast z))\}$. Select now the only identity in $E_1$, which is already in normal form w.r.t. $R_1$. $x \ast (y + z)$ cannot be reduced using $(u + v) \ast w$ (line 13) and therefore $R_2 = \{(x \ast (y + z), (x \ast y) + (x \ast z)), ((u + v) \ast w, (u \ast w) + (v \ast w))\}$ and $E_2 = \emptyset$. Both elements in $R_2$ are not marked and therefore the procedure will continue by selecting one of these two identities (line 21). Since each identity does not have critical pairs with itself, and $R_2$ does not have any marked rules, it will holds (for example if the first identity is selected) $R_3 = R_2$, $E_3 = E_2$ and the first identity is marked. Now the procedure will select the second one (line 21) and compute the critical pair between that identity and the first rule. The only critical pair between $x \ast (y + z)$ and $(u + v) \ast w$ is determined by the substitution $\sigma = [x/u + v, w/y + z]$ which leads to the following diagram.

\[
\begin{align*}
(u + v) \ast (y + z) &\quad \downarrow \quad (u \ast (y + z)) + (v \ast (y + z)) \\
((u + v) \ast y) + ((u + v) \ast z) &\quad (u \ast y) + (u \ast z) + (v \ast y) + (v \ast z) \\
((u \ast y) + (v \ast y)) + ((u \ast z) + (v \ast z)) &\quad ((u \ast y) + (u \ast z)) + ((v \ast y) + (v \ast z))
\end{align*}
\]
where the two terms \(((u+v)\cdot y)+(u+v)\cdot z\) and \((u\cdot (y+z))+(v\cdot (y+z))\) from a critical pair whereas the two terms \(((u+y)\cdot (v+y))+(u\cdot z)+(v\cdot z)\) and \(((u+y)\cdot (v+y))+(v\cdot y)+(v\cdot z)\) are their normal form. So \(R_4 = \{(x\cdot (y+z), (x\cdot y)+(x\cdot z)), ((u+v)\cdot w, (u\cdot w)+(v\cdot w))\}\) whereas \(E_4 = \{((u+v)\cdot y)+(u+v)\cdot z), (u\cdot (y+z))+(v\cdot (y+z))\}\), with both identities of \(R_4\) marked. The procedure will then evaluate the loop starting at line 3 by selecting \(((u+v)\cdot y)+(u+v)\cdot z), (u\cdot (y+z))+(v\cdot (y+z))\) and reducing it to its normal form \((\bar{s}, \bar{l})\). It holds that \(\bar{s} \neq \bar{l}\), \(s \neq \text{lpo} \bar{l}\) and \(\bar{l} \neq \text{lpo} \bar{s}\) and the condition at line 9 will be verified and the procedure will terminate with output \text{FAIL}.

(b) \(\{(f(g(f(x))), x)\}\) and the LPO with \(f > g\).

**Solution:**
Trivially \(R_1 = \{(f(g(f(x))), x)\}\), \(E_1 = \emptyset\) and \(f(g(f(x))) \triangleright_{\text{LPO}} x\) w.r.t. the LPO \(\triangleright_{\text{LPO}}\). The only critical pair of \(R_1\) is determined by the substitution \(\sigma = [x/g(f(z))]\) obtained considering \(f(g(f(x)))\) with its renaming \(f(g(f(z)))\). \(\sigma\) leads to the following diagram

\[
\begin{array}{c}
g(f(z)) \\
\downarrow \\
f(g(f(z))) \\
\end{array}
\begin{array}{c}
f(g(z)) \\
\end{array}
\]

Where \((g(f(z)), f(g(z)))\) form a critical pair. The two term of these pair are already in normal form and it holds \(f(g(z)) \triangleright_{\text{LPO}} g(f(z))\). Therefore \(R_2 = \{(f(g(f(x))), x)\}\) whereas \(E_2 = \{(f(g(z)), f(g(z)))\}\). The identity \((f(g(z)), g(f(z)))\) will then be selected (line 4). The procedure will enter the else branch in line 12 where \((f(g(f(x))), x)\) will be reduced by \((f(g(z)), g(f(z)))\) to \((f(g(x))), x)\). It will therefore hold \(E_3 = \{(f(g(f(x))), x)\}\) and \(R_3 = \{(f(g(z)), g(f(z)))\}\). The procedure will then select the only element in \(E_3\). It holds \(g(f(g(x))) \triangleright_{\text{LPO}} x\) and this identity cannot be used to reduce \((f(g(z)), g(f(z)))\). Therefore \(R_4 = R_3 \cup E_3\), \(E_4 = \emptyset\) and both rules in \(R_4\) are not marked. The procedure will then mark one of the two rules in \(R_4 = \{(f(g(f(x))), x), (f(g(z)), g(f(z)))\}\) and then check the critical pairs between the unmarked rule and the marked one. There are two critical pairs determined from the substitutions \(\sigma' = [z/f(x)]\) and \(\sigma'' = [x/g(z)]\). From these substitutions we get the following diagrams:

\[
\begin{array}{c}
f(g(f(f(x)))) \\
\downarrow \\
g(f(f(x))) \\
\end{array}
\begin{array}{c}
g(f(z)) \\
\end{array}
\]

\[
\begin{array}{c}
f(g(f(x))) \\
\downarrow \\
f(x) \\
\end{array}
\begin{array}{c}
g(f(g(z))) \\
\end{array}
\]

The two critical pairs \((g(f(f(f(x)))), f(x))\) and \((g(f(g(f(x)))), g(z))\) will then be added to \(E_5\), whereas \(R_5 = R_4 = \{(g(f(f(x))), x), (f(g(z)), g(f(z)))\}\). The procedure will then select the identities in \(E_5\) but, since the two diagrams shown are confluent, it holds that the normal form \((\bar{s}, \bar{l})\) of each identity is such that \(\bar{s} = \bar{l}\). Therefore (line 5) the procedure will remove the elements from \(E\) without any update on \(R\). Lastly, since all elements of \(R_5\) were already marked, the procedure will successfully return \{\((g(f(f(x))), x), (f(g(z)), g(f(z)))\)\}.

(c) \(\{(x + 0, x), (z + s(y), s(z + y))\}\) and > the KBO with \(s > +\) and weight 1 for all variables and symbols. Consider then the KBO with \(+ > s\) and weight 1 for all variables and symbols.

**Solution:**
From the KBO (herein \(\triangleright_{\text{kbo}}\)) it holds that \(s(z + y) >_{\text{kbo}} z + s(y)\) and \(x + 0 >_{\text{kbo}} x\). Skipping some easy steps, it will holds

\[R_2 = \{(s(z + y), z + s(y)), (x + 0, x)\}\] \(\text{and } E_2 = \emptyset\)

The only critical pair of \(R_2\) is determined by the substitution \(\sigma = [y/0, z/x]\) which leads to the following diagram
Let \( R \) be a left and right linear TRS. For every, \((l, r) \in R\), define \( t \xrightarrow{(l,r)} s \) iff \( t \rightarrow_R s \) using the rule \((l, r)\). Assume given a well founded order \(>\) on \( R \). It holds that if every critical peak of \((\rightarrow_R, R, >)\) has a decreasing diagram, then \( \rightarrow_R \) is confluent. This principle is called the

**Exercise 6:**

Let \( R \) be a left and right linear TRS. For every, \((l, r) \in R\), define \( t \xrightarrow{(l,r)} s \) iff \( t \rightarrow_R s \) using the rule \((l, r)\). Assume given a well founded order \(>\) on \( R \). It holds that if every critical peak of \((\rightarrow_R, R, >)\) has a decreasing diagram, then \( \rightarrow_R \) is confluent. This principle is called the

**Solution:**

If \( R \) is left and right linear, then for each local peak \((a, b)\) that is not a critical peak will hold that one of \((a, b, b, a), (a, b, b, a)\) or \((a, b, a, a)\) is a decreasing diagram. Indeed, all three tuples satisfy \(|b| \geq_{\text{mut}} |b'|/|a|\) and \(|a| \geq_{\text{mut}} |a'|/|b|\), so it’s only sufficient to show that the labelled rewrite relation \((\rightarrow_R, R, >)\) leads to one of those diagrams. This can be easily shown by considering the positions where the rewrite rules are applied. For example, consider \( s \xrightarrow{(l,r)} t \) and \( s \xrightarrow{(l',r')} t' \) where \((l, r)\) and \((l', r')\) are applied in incomparable positions, i.e. there exists two positions \( p, p' \) and two substitutions \( \sigma, \sigma' \) such that \( p \not< p', p' \not< p, s|_p = l\sigma, s|_{p'} = l'\sigma' \) and \( t = s[\sigma|_p \sigma']_{p'} \) and \( t' = s[r\sigma'|_p \sigma']_{p'} \). Then from \( p \not< p' \) and \( p' \not< p \) it holds that \( t'[\sigma'|_p \sigma']_{p'} = t'[\sigma]_{p'} = s[\sigma|_p \sigma']_{p'} \). Therefore \(((l, r), (l', r'), (l', r'), (l, r))\) is a decreasing diagram for \(((l, r), (l', r'))\). A similar analysis can be done for non-critical local peak resulting from two rewrite rules applied in two positions \( p, p' \) where \( p < p' \) (in this case is important to use the hypothesis of left and right linearity).

Since any non-critical local peak \((a, b)\) has a decreasing diagram, if also every critical peak of \((\rightarrow_R, R, >)\) has a decreasing diagram, then by Theorem it follows that \( \rightarrow_R \) is confluent.

*rule-labelling heuristic.*
(a) Consider the following TRS:

\[
\begin{align*}
\text{nat} & \rightarrow 0 : \text{inc(nat)} \\
\text{inc}(x : y) & \rightarrow s(x) : \text{inc}(y) \\
\text{tl}(x : y) & \rightarrow y \\
\text{inc}(\text{tl}(\text{nat})) & \rightarrow \text{tl}(\text{inc}(\text{nat}))
\end{align*}
\]

Prove its confluence using the rule-labelling heuristic.

**Solution:**
The TRS is left and right linear. We refer to its rules as follows

1 : nat → 0 : inc(nat)
2 : inc(x : y) → s(x) : inc(y)
3 : tl(x : y) → y
4 : inc(tl(nat)) → tl(inc(nat))

The only critical pair arises from the term tl(inc(nat)) and leads to the critical peak (4,1). The diagram (4, 1, 123, 3) is decreasing by considering any ordering where 4 is greater than 1, 2 and 3. For example lets consider 4 > 3 > 2 > 1. It holds that \[\| 3 \| = |3| \geq_{mul} |123|/|4| = 0 \text{ and } \| 4 \| = |4| \geq_{mul} |3|/|1| = \| 3 \|\].

(b) Consider the following (non-confluent) TRS \(R\):

\[
\begin{align*}
a & \rightarrow b \\
b & \rightarrow a \\
a & \rightarrow 0 \\
b & \rightarrow 1
\end{align*}
\]

Show that the rule-labelling heuristic cannot hold by proving that for any well-founded order \(>\) on \(R\) there exists a critical peak with no decreasing diagrams.

**Solution:**
We will refer to the rules of the TRS as: 1 (a → b), 2 (b → a), 3 (a → 0), 4 (b → 1). The TRS is therefore represented by the following diagram.

\[
\begin{array}{c}
0 \quad 3 \\
\downarrow & \quad \swarrow \\
a & \quad 1 \\
\downarrow & \quad \searrow \\
b & \quad 4 \\
\end{array}
\]

The TRS has two critical peaks (1,3) and (2,4). Let’s consider the first one: after doing a 1 transition we can reach the irreducible state 0 only with paths of the form 2(12)*3. We conclude that the diagrams for the critical peak (1,3) can be characterized by \{\{(1,3,2(12)^n3,ε) | n ∈ N\}. It must therefore hold that 1 > 2, otherwise for all n it will not hold that |3| \geq_{mul} |2(12)^n3|/|1|. Similarly, by considering the second critical peak, we conclude that after doing a transition 2 we can reach the irreducible state 1 only with paths of the form 1(21)*4. Therefore, the diagrams of the critical peak (2, 4) can be characterized by \{\{(2, 4, 1(21)^n4, ε) | n ∈ N\}. As such, it must hold that 2 > 1, otherwise for all n \in N it will not hold that |4| \geq_{mul} |1(21)^n4|/|2|. We conclude that it does not exists a well founded order > on the TRS such that every of its critical peaks have a decreasing diagram for it.

(c) Consider the following TRS \(R\):

\[
\begin{align*}
a & \rightarrow b \\
p(a) & \rightarrow s(p(a)) \\
p(b) & \rightarrow r \\
s(x) & \rightarrow m(x,x) \\
m(x,y) & \rightarrow r
\end{align*}
\]
Why the rule-labelling heuristic cannot be applied? Is it possible to find a set of labels \( I \) and a well-founded order \( > \) on it such that every local peak of \( (\rightarrow_R, I, >) \) has a decreasing diagram?

**Solution:**
The rule-labelling heuristic requires the TRS to be right-linear, which is not the case here. In particular, the rule \( s(x) \rightarrow m(x, x) \) can “self-duplicate”. For example consider the term \( s(s(r)) \). Still, it is easy to find a decreasing labelling noting that the duplicated variable has on the right-hand side less \( s \) symbols above it than on it left-hand side. Therefore, by labelling the steps first by the number of \( s \) symbols above the term and then by the rule, we can prove that any local peak has a decreasing diagram by considering the lexicographic order on these new labels.

**Exercise 7:**
Prove Newman’s lemma using decreasing diagrams techniques.

**Solution:**
Suppose \( \rightarrow \) be a terminating and locally confluent rewrite relation on terms \( T \). Since \( \rightarrow \) is terminating, \( \rightarrow^+ \) is a well-founded order. We consider the labelled rewrite relation \( (\rightarrow, T, \rightarrow^+) \) where each arrow \( s \rightarrow t \) is labelled by \( s \). As such, every local peak will have the form \( (s, s) \) for some \( s \in T \). To prove the Lemma we just need to show that for every of such peaks there exists a decreasing diagram. Let \( (s, s) \) be the local peak corresponding to \( s \rightarrow t' \) and \( s \rightarrow t'' \). Since \( \rightarrow \) is locally confluent, it holds that there exists a term \( t \) such that \( t' \rightarrow^* t \) and \( t'' \rightarrow^* t \). Let \( b' \) and \( a' \) be the concatenation of labels corresponding to \( t' \rightarrow^* t \) and \( t'' \rightarrow^* t \) respectively. It holds that \( s \rightarrow^+ t \) for each term \( t \) in the path \( t' \rightarrow^* t \) and in the path \( t'' \rightarrow^* t \). As such, \([s]|(\rightarrow^*)_{mul}|b'|/[s]| = \emptyset \) and \([s]|(\rightarrow^*)_{mul}|a'|/[s]| = \emptyset \). We conclude that there exists a decreasing diagram from each local peak of \( (\rightarrow, T, \rightarrow^+) \). By applying the Theorem above, \( \rightarrow \) is therefore confluent.

**Exercise 8:**
Consider the TRS \( L \)

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

(a) Is \( \rightarrow \) locally confluent?

**Solution:**
A TRS is locally confluent if and only if all its critical pairs are joinable. Since \( L \) does not have any non-trivial critical pair, it is locally confluent.

(b) Define a term \( \Omega \) such that for all terms \( t \), \( @(@, t) \rightarrow^+ @(@, t) \). Deduce that \( \rightarrow \) does not terminate (and therefore we cannot apply Newman’s Lemma to prove confluence).

**Solution:**
The last rule is the only one that is not right-linear and therefore can be used to make copies of a subterm. Moreover we notice that \( @(@(@, a, z), @(@, a, z)) \rightarrow^* @(@, z, z) \) using the first rule. Define \( \Omega = @(@(@, p, a), a) \). For all terms \( t \) it holds

\[
@(@(@(@, p, a), a), t) \rightarrow @(@(@, a, t), @(@, a, t)) \rightarrow @(@(@, a, t), @(@, a, t)) \rightarrow @(@, a, t) \rightarrow @(@, a, t)
\]

We conclude that \( L \) does not terminate since

\[
@(@, \Omega, \Omega) \rightarrow^+ @(@, \Omega, \Omega)
\]

(c) Is \( L \) orthogonal? Is \( \rightarrow \) confluent?
Solution:
Since any orthogonal TRS is confluent (see Lecture) and L is orthogonal, L is confluent.