

Termination Analysis

1. Consider the following program:

```

update(K,V,[ ],[p(K,V)]) ←.
update(K,V,[p(K,V1)|L],[p(K,V)|L]) ←.
update(K,V,[p(K1,V1)|L],[p(K1,V1)|L1]) ← K ≠ K1, update(K,V,L,L1).

```

Find a level mapping for which the program is recurrent and from which termination follows for the queries $\leftarrow \text{update}(3,[a,b,c],[p(1,a),p(2,f(b))],L)$.

and $\leftarrow \text{update}(3,c,L,[p(1,a),p(2,b),p(3,c),p(4,d)])$.

- (a) Level mapping $||$:

$$\begin{aligned}
 |update(t1,t2,t3,t4)| &= \min(\text{length}(t3), \text{length}(t4)) \text{ where} \\
 \text{length}(t) &= \mathbf{if } t = [X|Y] \mathbf{ then } 1 + \text{length}(Y) \\
 &\quad \mathbf{else } 0 \\
 |t1 \neq t2| &= 0
 \end{aligned}$$

- (b) Recurrency: Only the last clause has a nonempty body and need be checked.

Applying the levelmapping on the head (θ a grounding substitution):

$$\begin{aligned}
 |update(K,V,[p(K1,V1)|L],[p(K1,V1)|L1])\theta| \\
 &= \min(\text{length}([p(K1,V1)|L])\theta, [p(K1,V1)|L1]\theta) \\
 &= \min(1 + \text{length}(L\theta), 1 + \text{length}(L1\theta)) \\
 &= 1 + \min(\text{length}(L\theta), \text{length}(L1\theta)) \quad (1)
 \end{aligned}$$

Applying the level mapping on the body atoms:

$$\begin{aligned}
 |(K \neq K1)\theta| &= 0 \quad (2) \\
 |update(K,V,L,L1)\theta| &= \min(\text{length}(L\theta), \text{length}(L1\theta)) \quad (3)
 \end{aligned}$$

(1) > (2) and (1) > (3) hence update is recurrent wrt. the chosen level mapping.

- (c) $|update(3,[a,b,c],[p(1,a),p(2,f(b))],L)\theta|$
 $= \min(\text{length}([p(1,a),p(2,f(b))]), \text{length}(L\theta))$
 $= \min(2, \text{length}(L\theta)) \leq 2$

hence the query is bounded and thus strongly terminating.

$$\begin{aligned}
 |update(3,c,L,[p(1,a),p(2,b),p(3,c),p(4,d)])\theta| \\
 &= \min(\text{length}(L\theta), \text{length}([p(1,a),p(2,b),p(3,c),p(4,d)])) \\
 &= \min(\text{length}(L\theta), 4) \leq 4
 \end{aligned}$$

hence the query is bounded and thus strongly terminating.

2. Consider the following program:

1. $qs([],[]) \leftarrow$.
2. $qs([X|L],S) \leftarrow \text{part}(X,L,L1,L2),$
 $\quad qs(L1,S1), qs(L2,S2),$
 $\quad \text{append}(S1,[X|S2],S)$.
3. $\text{part}(X,[],[],[]) \leftarrow$.
4. $\text{part}(X,[Y|L],[Y|L1],L2) \leftarrow X \geq Y, \text{part}(X,L,L1,L2)$.
5. $\text{part}(X,[Y|L],L1,[Y|L2]) \leftarrow X < Y, \text{part}(X,L,L1,L2)$.
6. $\text{append}([],L,L) \leftarrow$.
7. $\text{append}([X|U],V,[X|W]) \leftarrow \text{append}(U,V,W)$.

Show that the program is left terminating for all queries $qs(t1,t2)$ where $t1$ is a list with a bounded length.

(a) Level mapping:

$$|qs(t1, t2)| = length(t1)$$

$$|part(t1, t2, t3, t4)| = length(t2)$$

$$|append(t1, t2, t3)| = length(t1)$$

$$|t1 \geq t2| = 0$$

$$|t1 < t2| = 0$$

where $length(t) = \mathbf{if } t = [t1|t2] \mathbf{ then } 1 + length(t2)$
 $\mathbf{ else } 0$

(b) **append/3** is recurrent wrt. level mapping: for all grounding substitutions θ :

$$|append([X|U], V, [X|W])\theta| = 1 + length(U\theta)$$

which is strictly larger than

$$|append(U, V, W)\theta| = length(U\theta).$$

(c) **part/4** is recurrent wrt. level mapping: for all grounding substitutions θ :

Clause 4: $|part(X, [Y|L], [Y|L1], L2)\theta| = 1 + length(L\theta)$ which is strictly larger than:

$$|(X \geq Y)\theta| = 0 \text{ and}$$

$$|part(X, L, L1, L2)\theta| = length(L\theta).$$

Clause 5: $|part(X, [Y|L], L1, [Y|L2])\theta| = 1 + length(L\theta)$ which is strictly larger than:

$$|(X < Y)\theta| = 0 \text{ and}$$

$$|part(X, L, L1, L2)\theta| = length(L\theta).$$

(d) **qs/2** is not recurrent. To show that it is acceptable, we need a model that is sufficiently precise for the predicates **part/4**, **qs/2**, and **append/3**: $I = \{t1 < t2 | t1 < t2 \in B_P\} \cup \{t1 \geq t2 | t1 \geq t2 \in B_P\} \cup \{append(t1, t2, t3) | append(t1, t2, t3) \in B_P \wedge length(t1) + length(t2) = length(t3)\} \cup \{part(t1, t2, t3, t4) | part(t1, t2, t3, t4) \in B_P \wedge length(t2) = length(t3) + length(t4)\} \cup \{qs(t1, t2) | qs(t1, t2) \in B_P \wedge length(t1) = length(t2)\}$.

- **I** is a Herbrand model

(θ a grounding substitution)

Clause 6. $append([], L, L)\theta \in I$ because $length([]) + length(L\theta) = 0 + length(L\theta) = length(L\theta)$, thus the clause is true.

Clause 7. If $append(U, V, W)\theta \in I$ then $length(U\theta) + length(V\theta) = length(W\theta)$

hence $length([X|U]\theta) + length(V\theta) = length([X|W]\theta)$ thus $append([X|U], V, [X|W])\theta \in I$ and the clause is true.

Clause 3. $part(X, [], [], []) \in I$ thus the clause is true.

Clause 4. If $part(X, L, L1, L2)\theta \in I$ then $length(L\theta) = length(L1\theta) + length(L2\theta)$

hence $length([Y|L]\theta) = length([Y|L1]\theta) + length(L2\theta)$ thus $part(X, [Y|L], [Y|L1], L2)\theta \in I$ and the clause is true.

Clause 5. Similar to clause 4.

Clause 1. $qs([], []) \in I$ hence **I** is model

Clause 2. If $part(X, L, L1, L2)\theta, qs(L1, S1)\theta, qs(L2, S2)\theta, append(S1, [X|S2], S)\theta$

is true in **I** then it holds that:

$$length(L\theta) = length(L1\theta) + length(L2\theta)$$

$$length(L1\theta) = length(S1\theta) \text{ and}$$

$$length(L2\theta) = length(S2\theta) \text{ and}$$

$$length(S1\theta) + 1 + length(S2\theta) = length(S\theta) \text{ hence}$$

$$length(L\theta) = length(S\theta) - 1, \text{ thus } length([X|L]\theta) = length(S\theta) \text{ so}$$

$qs([X|L], S)\theta \in I$ and the clause is true.
 (clause instances are trivially true in I if their body is false)

- $qs/2$ is acceptable wrt. the level mapping

With θ a grounding substitution, we have that

$$|qs([X|L], S)\theta| = 1 + length(L\theta) \quad (1)$$

$$|part(X, L, L1, L2)\theta| = length(L\theta) \text{ which is strictly smaller than (1)}$$

If $part(X, L, L1, L2)\theta \in I$ then

$$length(L\theta) = length(L1\theta) + length(L2\theta) \text{ hence}$$

$$length(L1\theta) \leq length(L\theta) \text{ and}$$

$$length(L2\theta) \leq length(L\theta) \text{ so}$$

$$|qs(L1, S1)\theta| = length(L1\theta) \text{ is strictly less than (1) and also}$$

$$|qs(L2, S2)\theta| = length(L2\theta) \text{ is strictly less than (1).}$$

If $part(X, L, L1, L2)\theta, qs(L1, S1)\theta, qs(L2, S2)\theta$ true in I then

$$length(L\theta) = length(L1\theta) + length(L2\theta),$$

$$length(L1\theta) = length(S1\theta), \text{ and}$$

$$length(L2\theta) = length(S2\theta) \text{ hence}$$

$$length(L\theta) \geq length(S1\theta) \text{ thus}$$

$$|append(S1, [X|S2], S)\theta| = length(S1\theta) \leq length(L\theta) \text{ is strictly smaller than (1).}$$

Hence $qs/2$ is acceptable.

- (e) Hence the program is acceptable wrt. the chosen level mappings; moreover queries $qs(t1, t2)$ with $t1$ a list of bounded length are bounded wrt. the level mapping, hence are left terminating.

3. Consider the following program:

$even(0) \leftarrow.$

$even(X) \leftarrow pred(X, Y), not(even(Y)).$

$pred(s(X), X) \leftarrow.$

- (a) Is this an acyclic program? Explain.

With t a ground term, the clause $even(t) \leftarrow pred(t, t), not(even(t))$ is an instance of the program clause. Obviously, for this instance, the body atom $not(even(t))$ has always the same level mapping as the head atom, hence the program is not acyclic.

- (b) Find a level mapping and a model such that the program is acceptable.

First we will define a function $f : U_L \rightarrow N$ from the Herbrand universe to natural numbers:

$$f(0) = 0$$

$$f(s(t)) = 1 + f(t)$$

Then a level mapping $||$:

$$|even(t)| = f(t)$$

$$|pred(t1, t2)| = f(t1) - 1$$

And interpretation $I = \{even(t) | f(t) = 2n, n \in N\} \cup \{pred(s(t), t) | t \in U_L\}$

We have to show that I is a model of the completion of $P_{even}/1$ which is equal to P .

- $T_P(I) \subseteq I$
 - From definition of I : $pred(t_1, t_2) \in I$ iff $t_1 = s(t_2)$
From fact $pred(s(X), X) \leftarrow: pred(t_1, t_2) \in T_P(I)$ iff $t_1 = s(t_2)$
Hence $pred(t_1, t_2) \in I$ iff $pred(t_1, t_2) \in T_P(I)$
 - Assume $even(t_1) \in T_P(I)$
Either $t_1 = 0$ hence $f(t_1) = 0$ and thus $even(t_1) \in I$
Or we have that $I \models pred(t_1, t_2), not(even(t_2))$
which means that $t_1 = s(t_2)$, hence $f(t_1) = 1 + f(t_2)$
and $even(t_2) \notin I$, i.e. $f(t_2) \neq 2n$ thus $f(t_2) = 2n + 1$
 $f(t_1) = f(t_2) + 1 = 2n + 1 + 1 = 2(n + 1)$ thus $even(t_1) \in I$
- $I \subseteq T_P(I)$
 - $pred(t_1, t_2) \in I$ iff $pred(t_1, t_2) \in T_P(I)$ (see above)
 - Assume $even(t_1) \in I$ thus $f(t_1) = 2n, \forall n \in \mathbb{N}$
Either $n = 0$ hence $t_1 = 0$ and $even(0) \in T_P(I)$
Or $n > 0$ hence $f(t_1) = 2m + 2, t_1 = s(t_2)$ and $f(t_2) = 2m + 1$ thus
 $pred(t_1, t_2) \in I$ and $even(t_2) \notin I$ hence $even(t_1) \in T_P(I)$

The program is acceptable.

Consider the ground clause $even(t_1) \leftarrow pred(t_1, t_2), not(even(t_2))$

- $|even(t_1)| = f(t_1)$ (1)
 $|pred(t_1, t_2)| = f(t_1) - 1$ (2)
(1) > (2) hence ok.
- Let $I \models pred(t_1, t_2)$. This means that $f(t_1) = f(t_2) + 1$. Then
 $|not(even(t_2))| = f(t_2) = f(t_1) - 1$ (3)
(1) > (3) hence ok

(c) What follows for the termination of the query $\leftarrow even(s(s(s(s(0)))))$. ?

The query $\leftarrow even(s(s(s(s(0)))))$ is bounded wrt $||$ thus a LD-derivation for it will terminate.

(d) And for termination of the query $\leftarrow even(s(s(X)))$. ?

The query $\leftarrow even(s(s(X)))$ is not bounded and thus termination is not guaranteed by the above analysis. (Note that it will terminate due to floundering.)

4. Consider the following program:

```

qs([],[]) ←.
qs([X|L],S) ← part(X,L,L1,L2),
               qs(L1,S1), qs(L2,S2),
               append(S1,[X|S2],S).
part(X,[],[],[]) ←.
part(X,[Y|L],[Y|L1],L2) ← X ≥ Y, part(X,L,L1,L2).
part(X,[Y|L],L1,[Y|L2]) ← X < Y, part(X,L,L1,L2).
append([],L,L) ←.
append([X|U],V,[X|W]) ← append(U,V,W).

```

Consider the set of calls $C = \{qs(X,Y) | X \text{ is a list of integers and } Y \text{ is free}\}$.

(a) Is the program strongly terminating wrt. C?

Taking an atom from S with a non-empty list in the first argument, one can easily construct an infinite derivation: using the recursive clause creates a resolvent with among others, calls $qs(L1, S1)$ and $qs(L2, S2)$ with both arguments free variables. Now, one can forever select a $qs/2$ atom with both arguments free variables. Indeed, using the recursive clause introduces two new such atoms.

(b) Follow the framework for automated termination analysis to show that the program is left-terminating wrt. C.

Compute $lfp(T_P^C)$:

$$\begin{aligned}
T_P^C \uparrow 0 &= \{qs(X, Y) | X \text{ is a list of integers and } Y \text{ is free}\} \\
T_P^C \uparrow 1 &= \{qs(X, Y) | X \text{ is a list of integers and } Y \text{ is free}\} \cup \\
&\quad \{part(X, L, L1, L2) | X \text{ is integer, } L \text{ is a list of integers and } L1, L2 \text{ are free}\} \cup \\
&\quad \{append(L1, L2, L3) | L1 \text{ and } L2 \text{ are lists of integers and } L3 \text{ is free}\} \\
T_P^C \uparrow 2 &= T_P^C \uparrow 1 = T_P^C \uparrow \omega \text{ (fix point)}
\end{aligned}$$

Level mapping $||$:

$$\begin{aligned}
|qs([U_1, \dots, U_n], L2)| &= length([U_1, \dots, U_n]) = n \\
|append([U_1, \dots, U_n], L2, L3)| &= length([U_1, \dots, U_n]) = n \\
|part(X, [U_1, \dots, U_n], L1, L2)| &= length([U_1, \dots, U_n]) = n
\end{aligned}$$

Acceptability:

- Consider a call $append([U_1, \dots, U_n], L2, L3)$
Its level mapping: $|append([U_1, \dots, U_n], L2, L3)| = n$
Unification with head of the recursive clause:
 $\theta = \{X/U_1, U/[U_2, \dots, U_n], V/L2, L3/[X|W]\}$
Recursive call: $append([U_2, \dots, U_n], L2, W)$
Its level mapping: $|append([U_2, \dots, U_n], L2, W)| = n - 1$ is strictly smaller than the level mapping of the original call.
Hence these calls are recurrent.
- Consider a call $part(Z, [U_1, \dots, U_n], V1, V2)$
Its level mapping: $|part(Z, [U_1, \dots, U_n], [U_1|L1], L2)| = n$
Unification with head of the first recursive clause:
 $\theta = \{X/Z, Y/U_1, L/[U_2, \dots, U_n], V1/[U_1|L1], V2/L2\}$
Recursive call: $part(Z, [U_2, \dots, U_n], L1, L2)$
Its level mapping $|part(Z, [U_2, \dots, U_n], L1, L2)| = n - 1$ is strictly smaller than the level mapping of the original call.
The same holds for the second recursive clause of $part/4$.
Hence these calls are recurrent.
- Consider a call $qs([U_1, \dots, U_n], V)$
Its level mapping: $|qs([U_1, \dots, U_n], V)| = n$
Unification with head of the recursive clause: $\theta = \{X/U_1, L/[U_2, \dots, U_n], S/V\}$
The resultant is $qs([U_1, \dots, U_n], V) \leftarrow part(U_1, [U_2, \dots, U_n], L1, L2), qs(L1, S1), qs(L2, S2) \dots$
By solving the call to $part/4$ we obtain a *cas* σ
We know for this *cas* that $n - 1 = length(L1\sigma) + length(L2\sigma)$ hence that $n - 1 \geq length(L1\sigma)$ and $n - 1 \geq length(L2\sigma)$
So the first left recursive resultant is of the form

$qs([U_1, \dots, U_n], V)\sigma \leftarrow qs(L1, S1)\sigma, qs(L2, S2)\sigma \dots$

We have $|qs(L1, S1)\sigma| = length(L1\sigma) \leq n-1$ hence strictly smaller than the level mapping of the call.

Resolving $qs(L1, S1)\sigma$ gives another left recursive resultant of the form

$qs([U_1, \dots, U_n], V)\sigma\theta \leftarrow qs(L2, S2)\sigma\theta \dots$

We have $|qs(L2, S2)\sigma\theta| = length(L2\sigma\theta) = length(L2\sigma) \leq n-1$ hence strictly smaller than the level mapping of the call.

Hence these calls are acceptable.