WoS: an Example of Backward Trace Slicing for Rewriting Logic Theories

1 WoS: Problem description

Let us illustrate our backward trace slicing technique for rewrite theories by means of a nontrivial producer/consumer example.

The War of Souls (WoS) is a game where an angel and a daemon fight to conquer the souls of human beings. Basically, when a human being passes away, his/her soul is sent to heaven or to hell depending on his/her faith as well as the strength of the angel and the daemon in play. Human beings with a higher level of faith are more likely to go to heaven.

This game is specified as a rewrite theory whose implementation, which is written in Maude, is shown in Fig. 1. The angel (resp. the daemon) is modeled by using the constructor term A(strength) (resp. D(strength)), where A (resp. D) is a constructor function symbol (i.e., a function symbol which doesn’t appear as root symbol of the left-hand side of any equation or rule) and strength is a natural number specifying the strength of goodness of the angel (resp. the strength of evilness of the daemon). A human being is formalized by means of the constructor term H(faith), where H is a constructor function symbol and faith is a natural number in the range 0..100 modeling his/her faith. Basically, the dynamic of the game is specified by means of two rules — namely, the creation rule and the death rule. The creation rule generates a new human being with a random faith, provided that there are at least two living human beings. The auxiliary equation newFaith allows us to consider a random value for the faith. The death rule selects a human being H and judges whether H should be sent to heaven or to hell. This decision is taken by using the auxiliary equation judgment that weights the faith of H w.r.t. both the goodness of the angel and evilness of the daemon involved.

2 The Backward Trace Slicing Technique in Action

Let us consider the execution trace \( \mathcal{T} : t_0 \to \ldots \to t_9 \) given in Fig. 2, which reaches the state \( t_9 = \langle \text{HEAVEN}, \text{D(30)}, \text{A(40)}, \text{H(70)}, \text{H(80)}, \text{H(90)} \rangle \) from the initial state \( t_0 = \langle \text{A(40)}, \text{D(30)}, \text{H(70)}, \text{H(80)}, \text{H(90)} \rangle \). This is done by running the \texttt{flw} command of Maude, which ensures both local fairness and rule fairness. The bracketed number between the command and the term to be rewritten (see Fig. 2), provides an upper bound for the number of rule applications that are allowed.

In Fig. 2, the application of every rule, equation, and axiom is displayed, showing the corresponding substitution, the current state, and the subterm where the next axiom is applied, before and after its application. Note that, since the equational simplification

---

\(^1\) Operator declarations in Maude may include equational attributes, such as \texttt{assoc}, \texttt{comm}, and \texttt{id}, stating, for example, that the operator is associative and commutative and has a certain identity element. Such attributes are used to represent certain kinds of equational axioms in a way that allows Maude to use these equations efficiently in a built-in way. The operator attribute \texttt{ctor} declares that the operator is a constructor, as opposed to a function defined by means of rules or equations.
mod war-of-souls is inc

INT + RANDOM + COUNTER .

sorts Human Angel Daemon Soul .

subsorts Human Angel Daemon < Soul .

vars faith good bad f1 f2 : Int .

var a : Angel . var d : Daemon .

var h h1 h2 : Human .


op D : Nat -> Daemon [ctor] .

--- constants

ops HEAVEN HELL : -> Soul [ctor] .

--- equations

op MAX-FAITH : -> Int .

eq MAX-FAITH = 100 .

op . : Soul Soul -> Soul [assoc comm] .

eq .(s1, s2) = .(s2, s1) .

op newFaith : -> Int .

eq newFaith = random(counter) rem MAX-FAITH .

op judgment : Angel Daemon Human -> Soul .

eq judgment (A(good), D(bad), H(faith)) =

if ( (good * faith) >=

(bad * (MAX-FAITH - faith)) )

then HEAVEN

else HELL

fi .

--- rewrite rules ----

rl [creation] : .(H(f1), H(f2)) =>

.(H(newFaith), H(f1), H(f2)) [label creation] .

f1 --> 70;

f2 --> 80

.(D(30), A(40), H(70), H(80), H(90))

t2| ================= intermediate term: .(.(D(30), A(40), H(90)), .(H(70), H(80)))

| --->

t3| .(.(D(30), A(40), H(90)), .(H(newFaith), H(70), H(80)))

| --->

| empty substitution

| newFaith

| --->

t4| ..

| 44

t5| ..

| 44

t6| ..

| 44

t7| ..

| 44

t8| ..

| 44

t9| result Soul: .(HEAVEN,D(30),A(40),H(70),H(80),H(90))

Fig. 1. Maude specification of the WoS game.

Fig. 2. Trace given by the frew command of Maude.

hides some basic steps, the original trace delivered by Maude has been extended with the sequence of intermediate terms that allows one to explicitly handle the $B$-equivalence by
means of flat/unflat transformations as well as the equational computations. Such extension has been implemented by augmenting the original specification with suitable rewrite rules. For example, in Fig. 2 the term $t_4$ stems from the term $t_3$ where the operator $\text{newFaith}$ has been replaced with its outcome $44$.

We can write the trace of Fig. 2 simply as follows:

$$T : t_0 \xrightarrow{\text{creation},\sigma_1} t_1 \xrightarrow{\text{unflat}} t_2 \xrightarrow{\text{newFaith},\sigma_2} t_3 \xrightarrow{\text{flat}} t_4 \xrightarrow{\text{unflat}} t_5 \xrightarrow{\text{flat}} t_6$$

where $\sigma_1 = \{h_1/H(70), h_2/H(80)\}$, $\sigma_2 = 0$, $\sigma_3 = \{h/H(44), a/A(40), d/D(30)\}$, and $\sigma_4 = \{\text{good}/40, \text{bad}/30, \text{faith}/44\}$.

In the following, we apply our slicing technique to the trace $T$ w.r.t. the rewrite theory $\mathcal{R}$ of Fig. 1. Specifically, we employ backward trace slicing to observe how the symbol $\text{HEAVEN}$ of $t_9$ is produced.

Step 1: Rule and equation labeling. First of all, we label the rules and the equations according to the proposed labeling procedure. For the sake of readability, the equation $\text{judgment}$ containing an if-then-else construct has been simplified by splitting it into two different labeled equations representing the two possible branches. Furthermore, in the equation $\text{newFaith}$, we consider the same labeling for any randomly created number.

$$\begin{align*}
\text{creation}^L &= \cdot(H^b(f1), H^c(f2)) \rightarrow \cdot(a,b,c)(\text{newFaith}^{a,b,c}, H^b(f1), H^c(f2)) \\
\text{death}^L &= \cdot(h,a,d) \rightarrow \cdot(a,d,\text{judgment}^{a,d,h}) \\
\text{judgment}^L &= \text{judgment}^{a,b}(\text{good}, D^c(bad), H^d(faith)) \rightarrow \text{HEAVEN}^{a,b,c,d} \\
\text{newFaith}^L &= \text{newFaith}^a \rightarrow [\text{a random number}]^a
\end{align*}$$

Step 2: Slicing criterion. Assume we want to discover which pieces of information in $T$ contribute to the generation of the value $\text{HEAVEN}$ in $t_9$. Therefore, since the symbol $\text{HEAVEN}$ occurs at position 1, we feed our slicing procedure with the slicing criterion $\{1\}$.

Step 3: Rewrite step labeling and sequence of relevant position sets. Figure 3 shows the labeling inferred for every rewrite step along with the sequence of relevant position sets $[P_0, \ldots, P_9]$ which have been computed w.r.t. the slicing criterion $\{1\}$.

Step 4: Trace slice. Finally, the trace slice $T^*$ is obtained by computing the term slices

$$t_0^* = \text{slice}(t_0, P_0), \ldots, t_9^* = \text{slice}(t_9, P_9)$$

w.r.t the sequence of relevant position sets $[P_0, \ldots, P_9]$. The outcome delivered by our tool is the trace slice $T^* = t_0^* \rightarrow^* t_9^*$ that is shown in Fig. 4 together with the corresponding original trace $T$. 
Fig. 3. Labeling and sequence of relevant position sets. In order to facilitate the understanding, the terms involved in each step are underlined.

3 Trace Slice Analysis

Let us briefly comment on our results. First of all, we note that the trace slice \( T^* \) is much simpler than the original execution trace \( T \), since \( T^* \) only keeps track of those symbols that influence the generation of the value HEAVEN. Also, we can observe that HEAVEN refers to the judgment of the human being H(44) (see term slice \( t^a_4 \)), who has been created by the pair \((H(70), H(80))\) (through the sliced rewrite step \( t^a_2 \rightarrow t^a_3 \)).

On one hand, it can be observed that the strength values good and bad (40 and 30, respectively) appear to influence the observed symbol HEAVEN. This is correct, because these values are used in the operation judgment to define the value HEAVEN. Finally, the generation of the observed outcome implies the existence of two human beings, whereas the values of their initial faith are irrelevant for the result, as shown in the term slice \( t^a_0 \). This is also correct, because these values are not used to compute the faith for \( H(44) \), which simply depends on a random value.

Regarding the size reduction of the execution trace, we can conclude the following facts. We denote by \( |T| \) the size of the trace \( T \), namely the sum of the number of
<table>
<thead>
<tr>
<th>Original Execution Trace $T$</th>
<th>Trace slice $T^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$: $(A(40), D(30), H(70), H(80), H(90))$</td>
<td>$t^*_1$: $(A(40), D(30), H(\bullet), H(\bullet), \bullet)$</td>
</tr>
<tr>
<td>$t_2$: $(D(30), A(40), H(70), H(80), H(90))$</td>
<td>$t^*_2$: $(D(30), A(40), H(\bullet), H(\bullet), \bullet)$</td>
</tr>
<tr>
<td>$t_3$: $(D(30), A(40), H(90))$</td>
<td>$t^*_3$: $(D(30), A(40), \bullet, (H(\bullet), H(\bullet)))$</td>
</tr>
<tr>
<td>$t_4$: $(D(30), A(40), H(90))$</td>
<td>$t^*_4$: $(D(30), A(40), \bullet, (H(\bullet), H(\bullet)))$</td>
</tr>
<tr>
<td>$t_5$: $(D(30), A(40), H(90))$</td>
<td>$t^*_5$: $(D(30), A(40), \bullet, (H(\bullet), H(\bullet)))$</td>
</tr>
<tr>
<td>$t_6$: $(D(30), A(40), H(90))$</td>
<td>$t^*_6$: $(D(30), A(40), \bullet, (H(\bullet), H(\bullet)))$</td>
</tr>
<tr>
<td>$t_7$: $(D(30), A(40), H(90))$</td>
<td>$t^*_7$: $(D(30), A(40), \bullet, (H(\bullet), H(\bullet)))$</td>
</tr>
<tr>
<td>$t_8$: $(D(30), A(40), H(90))$</td>
<td>$t^*_8$: $(D(30), A(40), \bullet, (H(\bullet), H(\bullet)))$</td>
</tr>
<tr>
<td>$t_9$: $(D(30), A(40), H(90))$</td>
<td>$t^*_9$: $(D(30), A(40), \bullet, (H(\bullet), H(\bullet)))$</td>
</tr>
<tr>
<td>$t_{10}$: $(D(30), A(40), H(90))$</td>
<td>$t^*_{10}$: $(D(30), A(40), \bullet, (H(\bullet), H(\bullet)))$</td>
</tr>
</tbody>
</table>

Fig. 4. The execution trace $T$ and its corresponding trace slice $T^*$.

symbols of all trace states. Then, the reduction given by the slicing technique is: $\frac{|T^*|}{|T|} = \frac{75}{139} = 0.54$ (i.e., a reduction of 46%) for this simple example. It is worth noting that, when we fix the slicing criterion \{2.1, 3.1\}, which allows us to observe the daemon $D(30)$ and angel $A(30)$ in play, the reduction achieved is about 71%. In this case, a relevant part of the original execution trace is discarded, since the behaviors of $D(30)$ and $A(30)$ are not affected by any other operator across the whole trace. This example along with other test cases can be found at the URL http://www.dsic.upv.es/~dromero/slicing.html.