Solutions to Event-B Exercises 6

Refinement

The complete Event-B solutions are to be found in sol6.zip. This contains some explanations for those solutions.

1. Q1 is concerned with refinement of events in which there is significant nondeterminism. The rules for such refinement can be found in the semantics lecture notes.

Refined Events

Refined Events have the following form

\[
\begin{array}{c}
\text{ANY} \\
\text{WHEN} \\
\text{WITH} \\
\text{THEN}
\end{array}
\begin{array}{c}
x_r \\
G_r \\
w : W \\
\text{Action}_r
\end{array}
\]

Proof Obligations for Refined Events

\text{guard refinement}
\forall S, C, S_r, C_r, V, V_r, x, x_r \cdot A_r \land I_r \land I \Rightarrow (G_r \Rightarrow G)

\text{witness}
\forall S, C, S_r, C_r, V, V_r, x, x_r \cdot A_r \land I_r \land I \Rightarrow (\exists w \cdot W)

\text{Simulation}
\forall S, C, S_r, C_r, V, V_r, x, x_r \cdot A_r \land I_r \land I \land W \land [\text{Action}_r]I_r \Rightarrow [\text{Action}]I

where $A_r$ denotes the refinement axioms.

Notice that the rule for \textit{guard refinement} allows the guards to be strengthened and with that the scope of the actions can be reduced. This would seem in principle to allow the guards to be refined to $\bot$ (false), in which case the refined events effectively have no actions.

This is clearly not consistent with the refined event unless some events overlap. Since the firing of events is nondeterministic, events with overlapping guards can be refined in such a way that the union of the guards is not strengthened. If this is done then some actions can be refined to \textit{skip}.

Now consider Q1 and its refinement Q1R.
MACHINE Q1
VARIABLES
x
n
INVARIANTS
inv1: x ∈ \mathbb{N}
inv2: n ∈ \mathbb{N}

EVENTS
Initialisation
begin act1: x ∈ \mathbb{N}
act2: n ∈ \mathbb{N}
end

Event E1 ≡
when grd1: x = 1
then act1: n := 1
end

Event E2 ≡
when grd1: x < 5
then act1: n := 2
end

Event E3 ≡
when grd1: x > 3
then act1: n := 3
end

Event E4 ≡
when grd1: x ≥ 4
then act1: n := 4
end

END

MACHINE Q1R
REFINES Q1
VARIABLES
x
n
EVENTS
Initialisation
begin act1: x ∈ \mathbb{N}
act2: n ∈ \mathbb{N}
end

Event E1 ≡
refines E1
when grd1: ⊥
then act1: n := 1
end

Event E2 ≡
refines E2
when grd1: x ≤ 3
then act1: n := 2
end

Event E3 ≡
refines E3
when grd1: x = 4
then act1: n := 3
end

Event E4 ≡
refines E4
when grd1: x ≥ 5
then act1: n := 4
end

END
Note:

(a) This solution Q1R is only one of many.
(b) Any of the guards could be strengthened to false, but that would be strengthening them too far. However Rodin appears to accept it!

What we have done in Q1R is

(a) in general guards are strengthened without strengthening the union of the guards. That means than whenever at least one event in Q1 was enabled, at least one —and in this solution only one— event of Q1R is enabled.
(b) all nondeterminacy has been eliminated. That is, the guards have been strengthened to the point where it is only possible for one event to be enabled for each value of $x$ in the “domain” of the events. Where there was choice, there is no longer any choice.

The justification is that Q1 expresses satisfaction with the choice, therefore the refinement can satisfy that choice any way that is consistent with Q1.

2. (a)
(b) For Q2 and Q2R see the archive.
(c) In the case $x \notin \text{ran}(s)$ both of the events in the refinement are blocked: the guards of both will become false. However, the invariant is not broken.

3. This is an invitation to invent the binary search algorithm. Note carefully how ordering is specified: $i \leq i \Rightarrow s(i) \leq s(j)$.

4. The techniques used above cannot work for Q4 as the domain of the function $f$ is not numeric and nor does it have any ordering that we are aware of. We construct our own arbitrary ordering by creating the bijection

$$sd \in 1..\text{card(dom}(s)) \mapsto \text{dom}(s)$$

This allows us to sequence through the domain of $s$ and hence adapt the strategy used in Q2R for a solution.

There are some important points to note about Q4 and its refinement Q4R shown over the page.

(a) The specification of Search in Q4 does not guarantee that $y$ exists such that $s(y) = x$. If there is no such $y$ then $\text{grd3}$ will never be true and Search will never be enabled.
(b) If there is no $y$ such that $s(y) = x$ then eventually all of the guards of Search1 will be false and so will the guards of Search. Thus the events of Q4R will deadlock and no solution will be produced.

Notice that Q4 is the norm for many/most searches required in practice and many strategies, such as search trees and hashing, are used to provide a mechanism to enable a search.
MACHINE Q4

Another search problem, but this time with a domain that isn’t numeric

SEES Q4_ctx

VARIABLES

pos
found

INvariants

inv1: pos ∈ Y
inv2: found ∈ BOOL
inv3: found = TRUE ⇒ s(pos) = x

EVENTS

Initialisation

begin act1: pos ∈ Y
act2: found := FALSE
end

Event Search ≜

any y
when grd1: found = FALSE
grd2: y ∈ dom(s)
grd3: s(y) = x
then act1: pos := y
act2: found := TRUE
end

END
MACHINE Q4R

We have to develop a concrete search with an opaque set $s$

REFINES Q4

SEES Q4_ctx

VARIABLES

pos
found
sd
$p$

INVARINTANTS

inv1: $sd \in 1..\text{card}(s) \mapsto \text{dom}(s)$

sd gives us a numeric mapping of the domain of $s$

inv2: $p \in \text{dom}(sd)$

inv3: $\text{found} = \text{FALSE} \Rightarrow (\forall i \in 1..p-1 \Rightarrow s(sd(i)) \neq x)$

thm1: $\text{card}(sd) = \text{card}(s)$

EVENTS

 Initialisation

\text{begin act1: } pos \in Y \\
\text{act2: } found := \text{FALSE} \\
\text{act3: } sd \in 1..\text{card}(s) \mapsto \text{dom}(s) \\
\text{act4: } p := 1 \\
\text{end}

Event $\text{Search} \cong$

refines $\text{Search}$

when $\text{grd1: } found = \text{FALSE}$

$\text{grd2: } s(sd(p)) = x$

with $y: y = sd(p)$

then $\text{act1: } pos := sd(p)$

$\text{act2: } found := \text{TRUE}$

end

Event $\text{Search1} \cong$

COMP2111 13 May 2010

Refinement
\textbf{Status} convergent

\textbf{when} \textit{grd1}: \textit{found} = \textit{FALSE} \\
\textit{grd2}: \textit{s(sd(p))} \neq x \\
\textit{grd3}: \textit{p} \neq \text{card}(s) \\
\textbf{then} \textit{act1}: \textit{p} := \textit{p} + 1 \\
\textbf{end}

\textbf{VARIANT}

\text{card}(s) - p

\textbf{END}