MODULE *SendSeqUndoP*

This module adds a prophecy variable $p$ to specification *SpecU* of module *SendSeqUndo* to obtain the specification *SpecUP*. It then asserts that *SpecUP* implements specification *Spec* of module *SendSeq* under a suitable refinement mapping, which implies that *SpecU* implements $\exists\, y : Spec$.

EXTENDS *SendSeqUndo*

Our definitions make use of the operators defined in module *Prophecy*. You should read that module to understand the meanings of those operators. We begin by defining the set *Pi* of possible individual predictions and the domain *Dom* of $p$, where $p[d]$ makes a prediction associated with $d$. In this case, $d$ is the domain of $y$ (which equals $1 \,..\, Len(y)$), and $p[d]$ predicts whether element number $d$ of $y$ is either sent or undone (removed from $y$ by an *Undo* step).

$Pi \triangleq \{\text{"send"}, \text{"undo"}\}$
$Dom \triangleq \text{DOMAIN } y$

INSTANCE *Prophecy* WITH $Pi \leftarrow \{\text{"send"}, \text{"undo"}\}, DomPrime \leftarrow Dom'$

Adding the prophecy variable requires replacing each subaction $A$ of a disjunctive representation with an action $Ap$. We use the disjunctive representation with subactions *Choose*, *Send*, *Rcv*, and $Undo(i)$. Each action $Ap$ is defined by defining:

- An operator *PredA*, where $PredA(p)$ is the prediction that the value of $p$ is making about action $A$ .

- *PredDomA*, the subset of *Dom* consisting of the elements $d$ for which $p[d]$ is used in the prediction $PredA(p)$.

- *DomInjA*, an injection from a subset of *Dom* to $Dom'$ describing the correspondence between predictions made by $p$ and those made by $p'$. For the prophecy variable we are defining, *DomInjA* specifies the obvious correspondence between the elements of the sequences $y$ and $y'$.

These definitions for each subaction $A$ follow. For example, *PredDomChoose* is *PredDomA* for $A$ the *Choose* action.

$PredDomChoose \triangleq \{\}$
$DomInjChoose \triangleq [i \in Dom \mapsto i]$
$PredChoose(p) \triangleq \text{TRUE}$

$PredDomSend \triangleq \{1\}$
$DomInjSend \triangleq [i \in 2 \,..\, Len(y) \mapsto i - 1]$
$PredSend(p) \triangleq p[1] = \text{"send"}$

$PredDomRcv \triangleq \{\}$
$DomInjRcv \triangleq [d \in Dom \mapsto d]$
$PredRcv(p) \triangleq \text{TRUE}$

$PredDomUndo(i) \triangleq \{i\}$
$DomInjUndo(i) \triangleq [j \in 1 \,..\, Len(y) \setminus \{i\} \mapsto \text{IF } j < i \text{ THEN } j \text{ ELSE } j - 1]$
$PredUndo(p, i) \triangleq p[i] = \text{"undo"}$

The following theorem asserts the action requirements described in Section 4.5 of "Auxiliary Variables in TLA+", which must be satisfied to ensure that $\exists\, p : SpecUP$ is equivalent to *SpecU*.

$Condition \triangleq \land\, ProphCondition(Choose, DomInjChoose, PredDomChoose,$

$$\begin{array}{l}
\qquad\qquad\qquad\qquad\quad PredChoose) \\
\land\ ProphCondition(Send,\ DomInjSend,\ PredDomSend,\ PredSend) \\
\land\ ProphCondition(Rcv,\ DomInjRcv,\ PredDomRcv,\ PredRcv) \\
\land\ \forall\, i \in Dom : \\
\quad ProphCondition(Undo(i),\ DomInjUndo(i),\ PredDomUndo(i), \\
\qquad\qquad\qquad\quad \text{LAMBDA}\ p : PredUndo(p,\ i))
\end{array}$$

THEOREM $SpecU \Rightarrow \Box[Condition]_{vars}$

Temporarily end the module here to use *TLC* to check the theorem.

---

VARIABLE $p$

$varsP \triangleq \langle vars,\ p\rangle$

$TypeOKP \triangleq TypeOK \land (p \in [Dom \to Pi])$

$InitUP \triangleq Init \land (p \in [Dom \to Pi])$

The actions *Ap* are defined using the *ProphAction* operator defined in the *Prophecy* module.

$$\begin{array}{l}
ChooseP \triangleq ProphAction(Choose,\ p,\ p', \\
\qquad\qquad\qquad\qquad DomInjChoose,\ PredDomChoose,\ PredChoose)
\end{array}$$

$SendP \triangleq ProphAction(Send,\ p,\ p',\ DomInjSend,\ PredDomSend,\ PredSend)$

$RcvP \triangleq ProphAction(Rcv,\ p,\ p',\ DomInjRcv,\ PredDomRcv,\ PredRcv)$

$$\begin{array}{l}
UndoP(i) \triangleq ProphAction(Undo(i),\ p,\ p',\ DomInjUndo(i),\ PredDomUndo(i), \\
\qquad\qquad\qquad\qquad \text{LAMBDA}\ j : PredUndo(j,\ i))
\end{array}$$

$NextUP \triangleq ChooseP \lor SendP \lor RcvP \lor (\exists\, i \in 1\,..\,Len(y) : UndoP(i))$

$SpecUP \triangleq InitUP \land \Box[NextUP]_{varsP}$

---

The theorem below asserts that *SpecUP* implements specification *Spec* of module *SendSeq* under the refinement mapping $y \leftarrow yBar$, where *yBar* is defined here to be the subsequence of $y$ consisting of those elements for which $p$ predicts "send".

$$\begin{array}{l}
yBar \triangleq \text{LET RECURSIVE}\ R(_,\ _) \\
\qquad\qquad R(yseq,\ pseq) \triangleq \\
\qquad\qquad\quad \text{IF}\ yseq = \langle\rangle \\
\qquad\qquad\qquad \text{THEN}\ yseq \\
\qquad\qquad\qquad \text{ELSE IF}\ Head(pseq) = \text{"send"} \\
\qquad\qquad\qquad\qquad \text{THEN}\ \langle Head(yseq)\rangle \circ R(Tail(yseq),\ Tail(pseq)) \\
\qquad\qquad\qquad\qquad \text{ELSE}\ \ R(Tail(yseq),\ Tail(pseq)) \\
\qquad\quad \text{IN}\quad R(y,\ p)
\end{array}$$

$SS \triangleq$ INSTANCE *SendSeq* WITH $y \leftarrow yBar$

THEOREM $SpecUP \Rightarrow SS!Spec$

* Modification History
* Last modified Sat *Dec* 31 16:10:30 *PST* 2016 by *lamport*
* Created *Thu Sep* 15 02:52:00 *PDT* 2016 by *lamport*