MODULE *DieHarder*

We now generalize the problem from Die Hard into one with an arbitrary number of jugs, each holding some specified amount of water.

EXTENDS *Naturals*

We now declare two constant parameters.

CONSTANT	Jug,	The set of all jugs.
	$Capacity$,	A function, where $Capacity[j]$ is the capacity of jug j.
	$Goal$	The quantity of water our heros must measure.

We make an assumption about these constants–namely, that *Capacity* is a function from jugs to positive integers, and *Goal* is a natural number.

$$\text{ASSUME } \land\ Capacity \in [Jug \rightarrow \{n \in Nat : n > 0\}]$$
$$\land\ Goal \in Nat$$

We are going to need the *Min* operator again, so let's define it here. (I prefer defining constant operators like this in the part of the spec where constants are declared.

$$Min(m, n) \triangleq \text{IF } m < n \text{ THEN } m \text{ ELSE } n$$

We declare the specification's single variable and define its type invariant and initial predicate.

VARIABLE $contents$ — $contents[j]$ is the amount of water in jug j

$$TypeOK \triangleq contents \in [Jug \rightarrow Nat]$$

$$Init \triangleq contents = [j \in Jug \mapsto 0]$$

Now we define the actions that can be performed. They are the obvious generalizations of the ones from the simple *DieHard* spec. First come the actions of filling and emptying jug j, then the action of pouring water from jug j to jug k.

Note: The definitions use the TLA+ notation

$$[f \text{ EXCEPT } ![c] = e]$$

which is the function g that is the same as f except $g[c] = e$. In the expression e, the symbol @ stands for $f[c]$. This has the more general form

$$[f \text{ EXCEPT } ![c1] = e1, \ldots, ![ck] = ek]$$

that has the expected meaning.

$$FillJug(j) \triangleq contents' = [contents \text{ EXCEPT } ![j] = Capacity[j]]$$

$$EmptyJug(j) \triangleq contents' = [contents \text{ EXCEPT } ![j] = 0]$$

$$\begin{aligned} &JugToJug(j, k) \triangleq \\ &\quad \text{LET } amountPoured \triangleq Min(contents[j], Capacity[k] - contents[k]) \\ &\quad \text{IN } contents' = [contents \text{ EXCEPT } ![j] = @ - amountPoured, \\ &\qquad\qquad\qquad\qquad\qquad\qquad ![k] = @ + amountPoured] \end{aligned}$$

As usual, the next-state relation *Next* is the disjunction of all possible actions, where existential quantification is a general form of disjunction.

$$Next \triangleq \exists j \in Jug : \lor FillJug(j)$$

$\lor$ $EmptyJug(j)$
$\lor$ $\exists\, k \in Jug \setminus \{j\} : JugToJug(j, k)$

We define the formula *Spec* to be the complete specification, asserting of a behavior that it begins in a state satisfying *Init*, and that every step either satisfies *Next* or else leaves contents unchanged.

$$Spec \triangleq Init \land \Box[Next]_{contents}$$

We define *NotSolved* to be true of a state iff no jug contains *Goal* gallons of water.

$$NotSolved \triangleq \forall\, j \in Jug : contents[j] \neq Goal$$

We find a solution by having *TLC* check if *NotSolved* is an invariant, which will cause it to print out an "error trace" consisting of a behavior ending in a states where *NotSolved* is false. Such a behavior is the desired solution.