Hiding Variables in TLA

"*Spec* with variable b hidden" is the formula that is satisfied by a behavior iff there is some way of assigning values to b in the states of the behavior that makes *Spec* true. This formula says nothing about the actual values of b in a behavior.

Such hiding is expressed mathematically by existential quantification. The formula $\exists\, x \,:\, P(x)$ means that there is some value that can be assigned to x that makes $P(x)$ true. It says nothing about the actual value of x; the formulas $\exists\, x \,:\, P(x)$ and $\exists\, y \,:\, P(y)$ are equivalent.

The formula "*Spec* with variable b hidden" is written in the TLA logic as $\exists\!\!\exists\, b \,:\, Spec$. The operator $\exists\!\!\exists$ is temporal existential quantification. It differs from the operator $\exists$ of ordinary (non-temporal) logic because $\exists\!\!\exists\, b \,:\, Spec$ asserts not that there is a single value of b that makes *Spec* true, but rather a sequence of values—one for each state of the behavior.

While the informal meaning of $\exists\!\!\exists\, b \,:\, Spec$ seems clear, it's not a legal TLA$^+$ formula. The $\exists\!\!\exists\, b$ introduces b as a new, bound variable, which is illegal in TLA$^+$ because b has already been declared in defining *Spec*. The way mathematicians normally define the meaning of such a formula, $\exists\!\!\exists\, b \,:\, Spec$ is equivalent to *Spec*. In TLA$^+$, to write "*Spec* with b hidden", we need to define *Spec* in a different module that gets imported with the INSTANCE statement introduced later.

Variable hiding is of mainly philosophical interest because model checking a specification that uses it is very expensive. TLC does not handle any property that contains the $\exists\!\!\exists$ operator.

CLOSE