Title:

TLA+ Truffle: Composition Capers

Abstract:

In the spirit of Programming Pearls and Graphics Gems, TLA+ Truffles are instructive examples of TLA+ specification or proof idioms, neither too trivial nor too complex. They need not report original research results or on an industry case-study, but are meant to present useful, reusable specification idioms that possibly inspire new ways of thinking about problems. To quote Richard Bird, editor of the Functional Pearls column in Journal of Functional programming, they should to be “polished, elegant, instructive, entertaining.”

TLA+ is a formal logic for the specification of discrete or hybrid dynamic systems and proofs of propositions about them, designed by Leslie Lamport and combining his Temporal Logic of Actions with a formal ZFC set theory. While composition in TLA+ is elegant and represented by simple logical conjunction, when specifying engineered systems, Lamport recommends a specification in the form of a single state-machine, expressed as the “canonical” or “normal” TLA+ form, Init ∧ □[Next]_vars ∧ Fairness, rather than as a composition (conjunction) of general temporal formulas. Normal forms are also the only TLA+ formulas that can be checked by TLC, a model-checker for a subset of TLA+ included in the TLA+ Toolbox. The canonical state-machine form is, indeed, appropriate for most specifications. Nevertheless, this talk will present two uses for the composition of canonical forms and show how to model-check them in TLC.

The first shows the use of composition for specifications following David Harel’s Behavioral Programming style. This technique allows the gradual specification of a system through a series of conjoined rules that are added piecemeal, allowing model checking at each stage.

The second uses composition to harness TLC for model-based trace-checking, a technique that could perhaps be used to help establish (unsoundly) that a real-world system implements a TLA+ specification by checking that execution traces obtained from system logs represent valid behaviors of a specification, by treating them as a special case of refinement. The conjunction technique can be used, in some cases, to exploit TLC's capabilities to automatically establish refinement without writing a complex refinement mapping or adding auxiliary variables even if the log records just a subset of the specification variables or does not include all internal transitions.