Electrum

Lightweight specification of behavioral models with rich configurations

Julien Brunel¹, David Chemouil¹, Alcino Cunha², Nuno Macedo² et al.
Workshop on the Future of Alloy, April 30 & May 1, 2018, MIT.

¹ONERA/DTIS & Université de Toulouse
²INESC-TEC & Universidade do Minho
Observations

Many Alloy models feature both structural and behavioral aspects, but:

• Behavior modeling requires systematic “boilerplate”
  • explicit modeling of state (local/global state idiom)
  • every mutable construct must be indexed by state/time
  • specification of a linear model of time (most of the time (!))
  • specific handling of the last state of a trace
• Essentially to model check safety properties, indeed:
  • spurious counterexamples to liveness properties may happen, unless traces are enriched with lassos [Cunha 14, Biere et al. 99]
  • even then, limited to bounded model-checking (BMC)

(Safety properties rule out unwanted behaviors, liveness properties characterize expected behaviors)
Linear temporal logic (LTL):

• is more expressive than propositional logic
• is decidable
• relies on a *simple* & *uniform* model of time: *infinite traces* of states
• benefits from dedicated, *complete* model-checking procedures
Introducing Electrum

Mark mutable fields or signatures as such (using a new `var` keyword).
Add $LTL$ + primed variables (as, e.g., in TLA+).

Dedicated analyses:

- BMC by reduction to Alloy + traces with lassos
- Unbounded MC (UMC) by reduction to NuSMV or nuXmv
Example: Chord

sig Node {
    var fst : lone Node,
    var snd : lone Node,
    var prdc : lone Node,
    var todo : Status→Node }

var sig members in Node {}

var sig ringMembers in members {}

fact {
    always members =
        { n: Node | some n.fst and
        some n.snd and
        some n.prdc }
    always ringMembers =
        { m : members | m in m.^succ }

    fun succ : Node → lone Node { ... }
}

pred join [new : Node] { // an event
    new not in members
    some m : members {
        between[m, new, m.fst]
        fst' = fst ++ new→m.fst
        snd' = snd ++ new→m.snd
        prdc' = prdc ++ new→m
        todo' = todo }
}

fact strongFairness {
    all n, m : Node {
        (always eventually rectifyEnabled[n,m])
        ⇒ (always eventually rectify[n,m])
        ... }
}

assert correctness {
    (eventually always not (join or fail)
        implies eventually always ideal ) }

/5/7
Assessment

Fits well most Alloy models with *behavior*.

Often leaner than plain Alloy (not always: *e.g.* counting events).

BMC efficiency on par with classic Alloy.

UMC with nuXmv comparable to TLA+’s TLC (room for improvement) (note: nuXmv is *not* free software; other, non-evaluated, tools exist).

Modeling [Zave 2017]’s version of Chord raised various corner cases: analyzing “abstract” liveness properties if useful (even with BMC).
Prospect

Enhance modeling of the “system” (automaton) part, e.g.: actions (guard + post-condition), frame rules, fairness constraints...

Most models may then rely on LTL for assertions only.

So add branching time (CTL) too?

No more a conservative extension of Alloy, though.