Supporting Materials for Unrealizable Cores for Reactive Systems Specifications

Abstract—**This document provides supporting materials for the ICSE'21 paper submission titled "Unrealizable Cores for Reactive Systems Specifications".**

I. A REMARK ABOUT USING MEMOIZATION CORRECTLY

Recall the `QuickCore` algorithm (Alg. 1 below).

Algorithm 1 QuickCore Given an unrealizable specification find a locally minimal subset of guarantees that keeps it unrealizable

```
Require: An unrealizable specification $S = \langle V_e, V_s, D, M_e, M_s \rangle$, where
    $M_e = \langle I_e, T_e, J_e \rangle$ and $M_s = \langle I_s, T_s, J_s \rangle$
Ensure: An unrealizability core of $S$
            ▷ Begin by minimizing only justice guarantees
 1: if Realizable($\langle V_e, V_s, D, M_e, \langle I_s, T_s, \emptyset \rangle \rangle$) then
 2:     $J_c \leftarrow$ MinWBase($DDMin, B_{M_s}, I_s \cup T_s, J_s, \neg$Realizable($S$))
 3: else
 4:     $J_c \leftarrow \emptyset$
 5:     $M_e \leftarrow \langle I_e, T_e, \emptyset \rangle$
 6: end if
            ▷ Continue with minimizing only safety guarantees
 7: $T_c \leftarrow$ MinWBase($DDMin, B_{M_s}, I_s \cup J_c, T_s, \neg$Realizable($S$))
            ▷ End with minimizing only initial guarantees
 8: $w \leftarrow$ ComputeWinRegion($\langle V_e, V_s, D, M_e, \langle I_s, T_c, J_c \rangle \rangle$)
 9: $I_c \leftarrow I_s$
10: $envIni \leftarrow \wedge_{d \in I_e} d$
11: for $i \in I_c$ do        ▷ Use linear minimization for initial guarantees
12:     $I_c \leftarrow I_c \setminus \{i\}$
13:     if SysWin($envIni, \wedge_{d \in I_c} d, w$) then
14:         $I_c \leftarrow I_c \cup \{i\}$
15:     end if
16: end for
17: return $I_c \cup T_c \cup J_c$
```

Remark 1. *Note that realizability results in* `QuickCore` *may correspond to specifications with an altered set of assumptions, because from line 7 of* `QuickCore` *on, if no justice guarantees are required for a core, all environment justices are removed. These realizability check results are stored and used by the memoization of* `QuickCore` *and* `PQC`*. However, considerations detailed in the correctness of* `QuickCore` *ensure that in this case realizability checking results are the same for specifications where the justice assumptions are kept.*

II. A REMARK WITH ADDIONAL RESULTS

Some additional results concerining `AM+GN` which we use in the paper only for scaling, because they are automatically generated.

Remark 2. `AM+GN` *specifications are parametric and are automatically generated. Therefore the nature of the cores of these specifications and finding all their cores, do not seem to be of practical interest. Accordingly, We did not report results for them in Tables 3 and 4.We did observe, however, that multiple cores abound for these specifications and and that some of them have over 100 cores. Moreover* `QuickCore` *finds slightly smaller cores than DDMin for* `AMBA` *and* `GENBUF` *specifications, and both algorithms find cores which are about 15 times smaller than the total number of all guarantees on average.*