Prof. Dr. A. Podelski,
Dr. B. Westphal

Summer term 2021

Softwaretechnik/Software Engineering

http://swt.informatik.uni-freiburg.de/teaching/SS2021/swtvl

Exercise Sheet 5

Early submission: Monday, 2021-07-05, 13:00 Regular submission: Tuesday, 2021-07-06, 13:00

Exercise 1 – Evaluating OCL Formulae (5/20 Points)

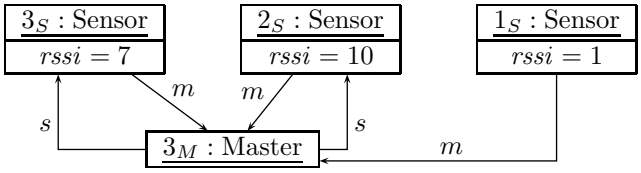

Figure 1: Complete object diagram.

Consider the system state σ_1 given by the complete Object Diagram in Figure 1.

(i) To which value does the Proto-OCL formula

$$F := \forall\, self \in allInstances_{\text{Master}} \bullet \forall\, n \in s(self) \bullet rssi(n) > 1$$

evaluate for σ_1? Prove your claim, i.e., compute $\mathcal{I}[\![F]\!](\sigma_1, \emptyset)$. (3)

(ii) Provide system states σ_2 and σ_3 such that for each truth value (*true*, *false*, $\perp$), there is an $i \in \{1, 2, 3\}$ such that $\mathcal{I}[\![F]\!](\sigma_i, \emptyset)$ evaluates to this truth value. Argue your claim. (2)

Hint: As we refer to the same OCL formula as in Task (i), you may use the detailed proof that you provided for Task (i) to guide your argument here.

Exercise 2 – Computation Graph / Transition Graph (6/20 Points)

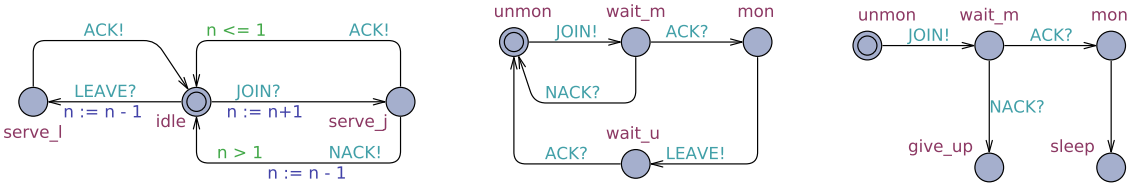

Figure 2: Network of Communicating Finite Automata

Provide the reachable part of the transition graph of the CFA model shown in Figure 2. (6)

Hint: Make sure to clearly indicate the initial configuration(s). And you may want to introduce abbreviations for location names if this increases readability of your computation graph.

Exercise 3 – Basic Behavioural Model Analysis (7/20 Points)

The file `sensormaster.xml` includes a behavioural model of some aspects of the sensor/master system from Exercise Sheet 4. Note that the model in `sensormaster.xml` has similarities to the one shown in Figure 2 but is not identical in different aspects.

(i) Use the UPPAAL[1] simulator to create one computation path which shows that the model is

[1] See Appendix A for instructions.

able to exhibit the simple scenario that one sensor joins a master and then leaves again. (1)

Hint: Include a screenshot showing the final state of the computation path and the sequence diagram into your submitted document; and submit the `.xtr` *file of the simulation as well so that your tutor can reproduce your result.*

(ii) a) Recall the requirement that masters only need to monitor a bounded number of sensors. Use the verifier to check whether it is possible for some master to monitor (as indicated by the value of its variable `n`) the allowed maximum number of sensors (which is given by the global constant (or parameter) `max_sns`).

What is the result? How should this result be interpreted? (2)

Hint: That is, what can we conclude from the result of the above check about the modelled design idea?

b) Use the verifier to check whether it holds that for each master, the value of `n` never exceeds the assumed maximum number of sensors. Some queries have been proposed for the check and are included in `sensormaster.xml`.

What are the results? How should these results be interpreted? (2)

Hint: Also submit a copy of `sensormaster.xml` *with your 1.(ii).a) query filled in; make sure that your tutor understands which of the many files you submit is to be considered for which task.*

(iii) The author of the model has expressed doubts about the model's correctness. One invariant that needs to hold for the model to be correct (according to the requirements department) is that if the value of variable `m` in any sensor is a proper master's id, then the value of `n` of this master must not be 0.

Explain how the outcome of checking the corresponding query in `sensormaster.xml` (unfortunately) confirms these doubts. (1)

(iv) Explain the reason for the incorrectness of the model according to Task (iii) and suggest an as small-as-possible change that preserves the outcomes of the Tasks (i) and (ii), but changes the outcome of the previous check. (1)

Hint: Also submit the changed model in a separate file. If you do not spot any reasons for the incorrectness, ask your tutor for clues (together with a clear description of the state of your investigation and your hypotheses so far).

Exercise 4 – Advanced Model Analysis (2/20 Points + 5 Bonus)

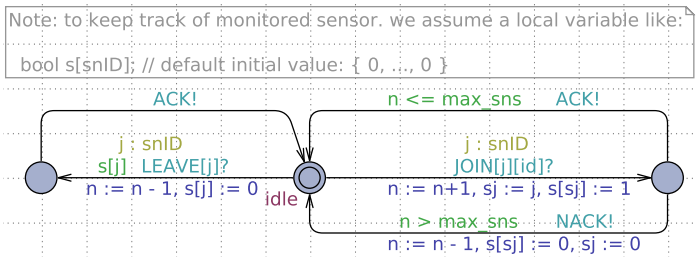

Figure 3: Sketch of design idea.

After *counting* the number of sensors in the masters has been solved, the model should be extended such that the masters keep track of *which* sensors they are monitoring.
A proposed solution is sketched in Figure 3.

(i) Extend the model from Task (iv) of the previous exercise such that the extended model implements the sketch[2] given by Figure 3. (1)

Hint: Submit the resulting file; make sure that the model includes comments that point out where you needed to 'fill in' things missing in the sketch (and that these comments are clearly recognisable by your tutor).

(ii) There are two important correctness properties of the sensor/master system, for which the requirements department came up with the following (untypically precise) phrases:

a) For each master j and sensor i, it holds that, if the i-th entry of the value of j's `s` variable is not false (or: not 0), and if j is idle, this implies that the value of `m` in i is j.

b) For each sensor i and master j, it holds that, if i is monitored and its `m` has value j, this implies that the i-th entry of j's `s` value is not false.

Formalise these requirements in UPPAAL query language and check these queries on your model from Task (i), once with 1 master (as in the given model), and once with 2 masters.[3]

What are the results? How should these results be interpreted? (1)

(iii) Some experienced designer had early doubts on the correctness of the proposed solution, yet the team did not want to rely on 'feelings' but wanted to be convinced and insisted on checking the model.

Your results from the previous task should prove the experienced designer right.[4]

Provide a comprehensive[5] description of what goes wrong, both technically and intuitively. Outline a solution (and model and check it, if you like). (5 Bonus)

A Tool Availability

UPPAAL is available on the Linux machines in the computer pool.
To run UPPAAL, use the following command:

```
/usr/local/ufrb/uppaal/uppaal-4.1.19/uppaal
```

Note: The (otherwise remarkably) stable UPPAAL tool faced some regressions regarding file formats lately. Please use exactly version 4.1.19 as mentioned above so that your tutor can easily check your submissions. If you (for whatever reason) must use a different version, please make sure that version 4.1.19 is able to read the model and trace files that you submit.

[2] Here, 'sketch' is used in the sense that the design department has just quickly drawn Figure 3 and (for good reason) assumes that you, as an (at least half) CFA-expert, will know how it defines a well-formed UPPAAL model. That is, only changing the CFA of template 'Master' may not be sufficient: The declaration of `s` needs to be introduced at the right place, and other smaller issues like this one may need to be resolved.

[3] The number of masters and sensors that are considered for simulation and verification are determined by the values of the the global constants/parameters `num_ms` and `num_sn` in the models that we use here.

[4] If this is not the case, immediately contact your tutor. `;-)`

[5] Comprehensive in the sense that you have very high confidence that at least 90 % of your fellow students who did not work on this task would be convinced of your analysis and consider the analysed design idea proven incorrect. In other words: as high-quality you would write if this task were your assignment of the day in an industry job; given to you by the boss of the experienced designer, since the 'big boss' wants to get an unbiased analysis.