Aspect-Oriented Model-Based Testing with UPPAAL Timed Automata

Jüri Vain, Leonidas Tsiopoulos,
Gert Kanter
Tallinn University of Technology,
Estonia

Model-based testing (MBT)

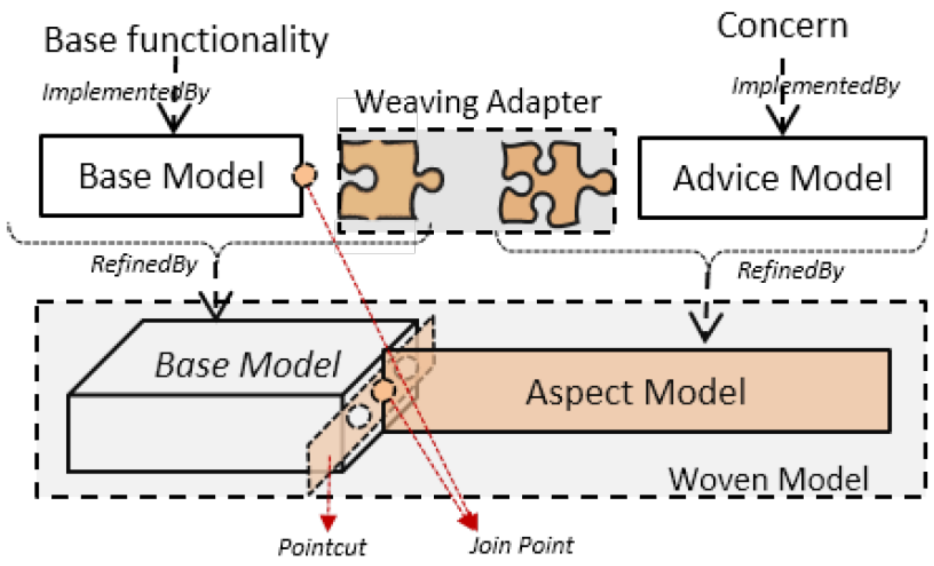
- Model-based conformance testing is a testing approach where
  - models specify the expected behavior of the System Under Test (SUT)
  - test goal can be specified either as set of constraints (on model execution paths/data) or a separate state machine composed with the SUT model.
- Advantages of MBT
  - automatic (online/offline) test generation
  - verification of the test correctness and optimality
  - Easy adjustment when SUT or its requirements change

Why MBT methods need improvement?

- Drawbacks of MBT:
  - Manual model construction is time consuming and error prone process
  - Model construction needs theoretical knowledge and experience
  - Large models are out of human comprehension
  - Unstructured models complicate error tracking
  - Most of model verification and test generation tools have limited scalability
- MBT community has suggested various test modularization approaches to overcome drawbacks: OO, program slicing, design viewpoints, etc.
- Our contribution – introduce modularity to MBT via aspect-oriented modelling (**AOM**)

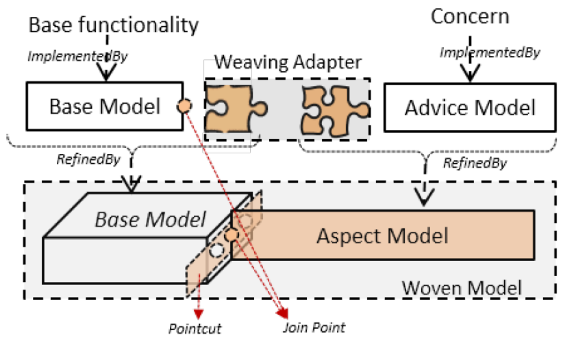
Aspect-oriented modelling

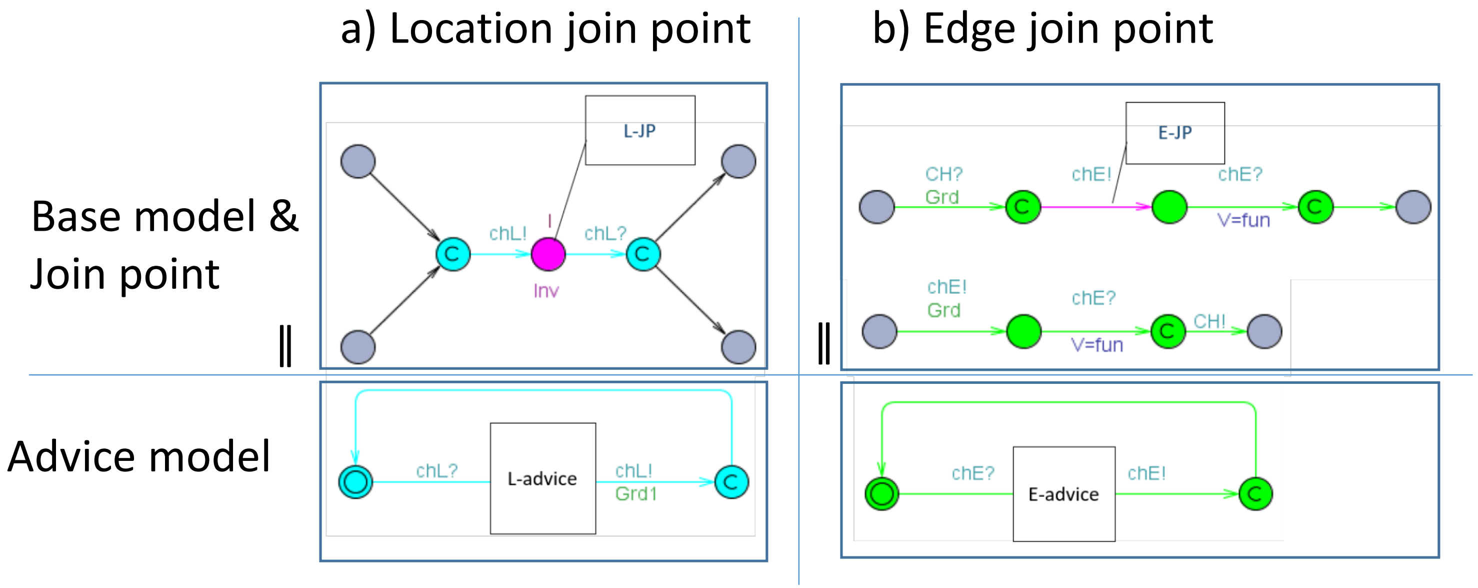
- ***Base model*** - represents the core functionality of the system
- ***Advice** model* represent a crosscutting concern
- ***Weaving*** is composing a base model with the advice model via weaving adapter (weaver).
- ***Join points*** are model fragments in the base model to which an aspect can be woven.
- ***Pointcut*** is the set of join points and conditions under which an advice can be woven.
- ***Woven model*** is composition of base and (possibly several) advice model(s).



Aspect-oriented modelling with Uppaal TA

- Uppaal TA (UTA) is a closed network of extended timed automata (*processes*) composed by CCS type synchronous parallel composition.
- Join points and weaving adapters in UTA





AO test coverage criteria

- The AO test coverage categories are:
  - aspect coverage,
  - join point coverage,
  - advice path coverage,
  - advice element coverage.
- All categories have strong and weak forms.
- Example:
  - *strong join point coverage* (SJPC): given an aspect A_i all of its join points must be covered by test runs
  - SJPC coverage expressions in TCTL:

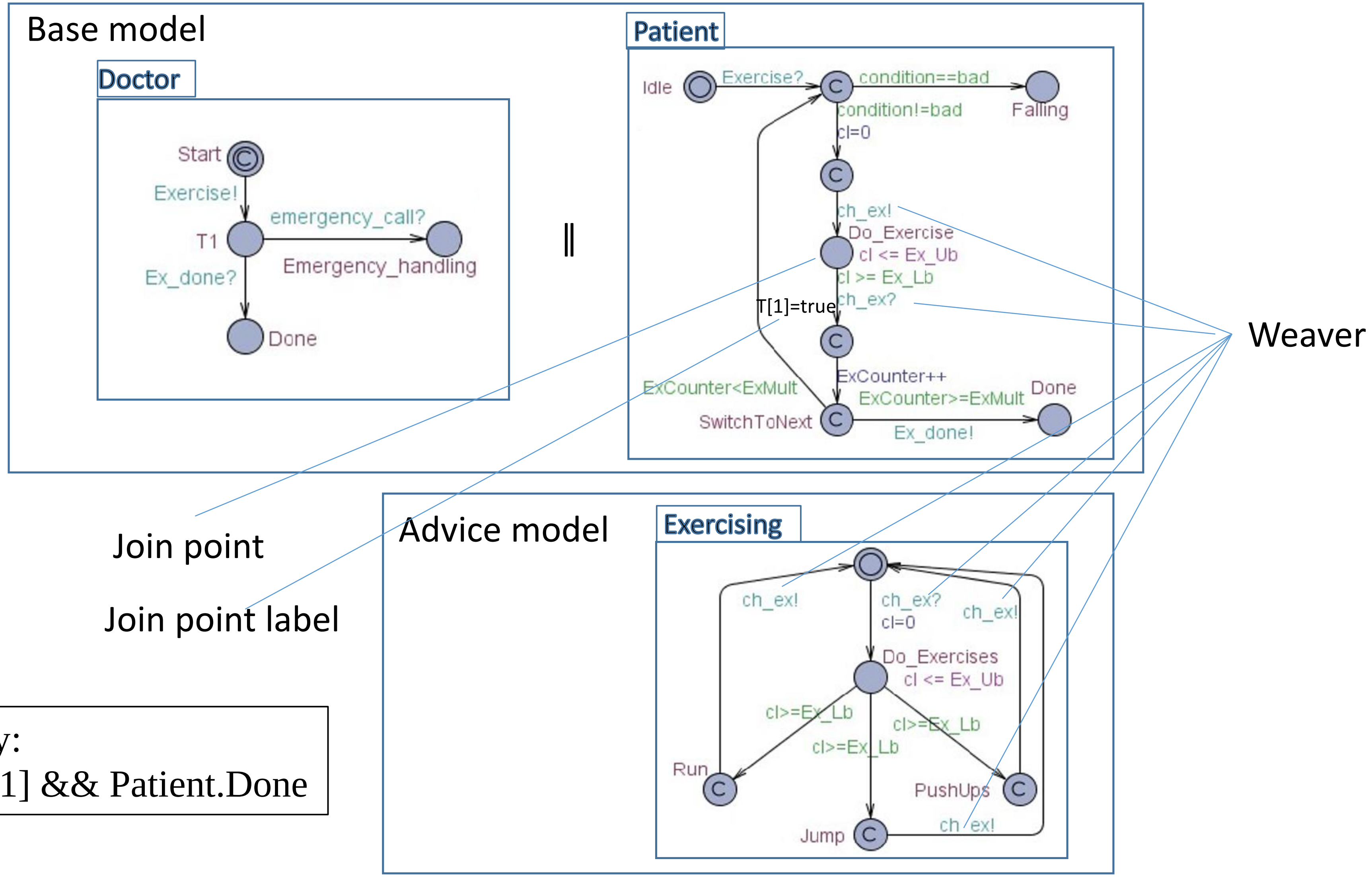
$$\mathrm{E}\Diamond\ \mathit{forall}\ (j : \mathrm{int}[1, m])\ R(i, j),$$

where

$\mathrm{E}\Diamond$ – TCTL temporal operator “*eventually*“

$R(i)$ – predicate that evaluates *true* in the model when j-th join point of aspect i is traversed

Example: Home Rehabilitation System (HRS)



Strong JP coverage query:
E◊ *forall* (*j* : int[1, 1]) *T*[1] && Patient.Done

Analytical validation: test generation effort

- Off-line test generation complexity = witness trace generation complexity by MC
- Time complexity of model checking TCTL formula ϕ over TA M, with clocks x in ϕ and the number of aspects m in M:

$$\mathcal{O}(m, |\phi|) = m(|\frac{\phi}{m}| \times ((\frac{n}{m})! \times 2^{\frac{n}{m}} \times C \times |\frac{L}{m}|^2)$$

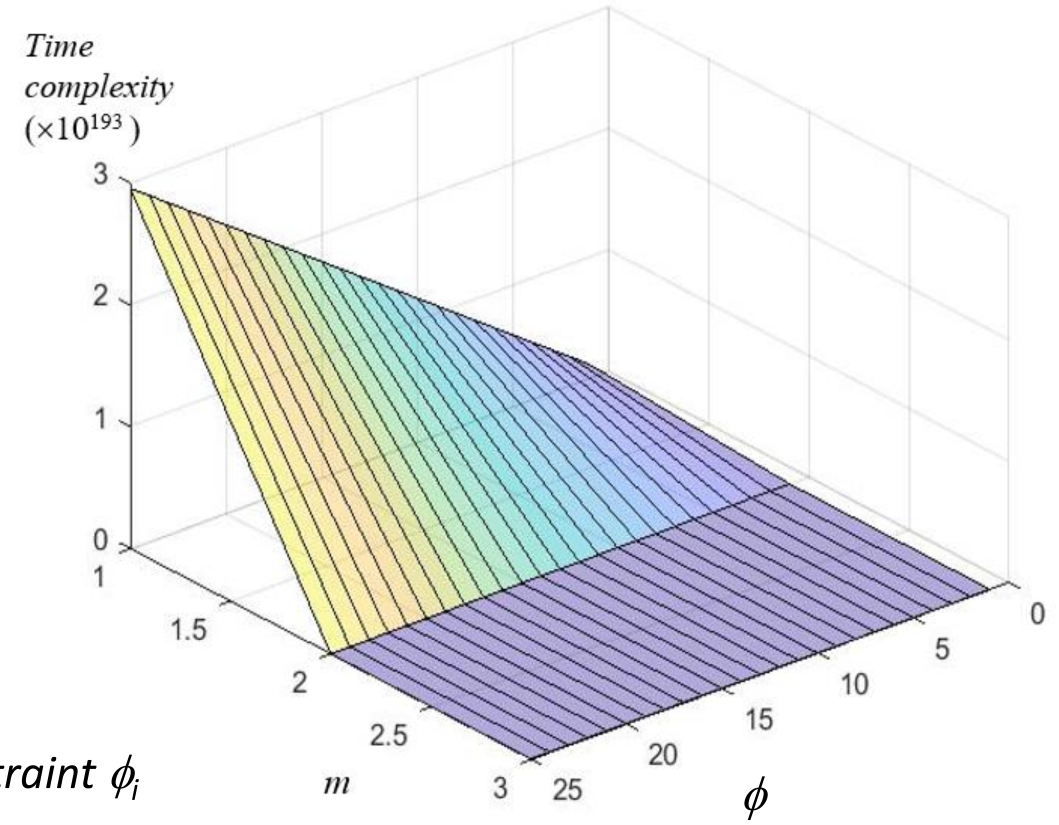
where

- $C = \prod_{x \in \phi} c_x$
- c_x – max time bound of clock x in TCTL formula ϕ
- n – number of clock regions
- L – set of symbolic states of M

Analytical validation: test generation effort

Observation

Any reduction of the model symbolic state space provides exponential reduction in model checking complexity aka <u>*test generation complexity.*</u>



Corollary:

$$M^B \oplus M^{A_j} \models \phi_i \wedge [\![M^B \oplus M^{A_j}]\!] \subseteq [\![M]\!] \Rightarrow$$
$$\mathbf{E}(M^B \oplus M^{A_j}, T^{\phi_i}) \leq \mathbf{E}(M, T^{\phi_i}).$$

where

$\mathbf{E}(\ .\ , T^{\phi_i})$ - effort of generating test T *that satisfies constraint* ϕ_i

$[\![M^B \oplus M^{Aj}]\!]$ - operational semantics of the base model M^B augmented with advice M^{Aj}

Experimental validation

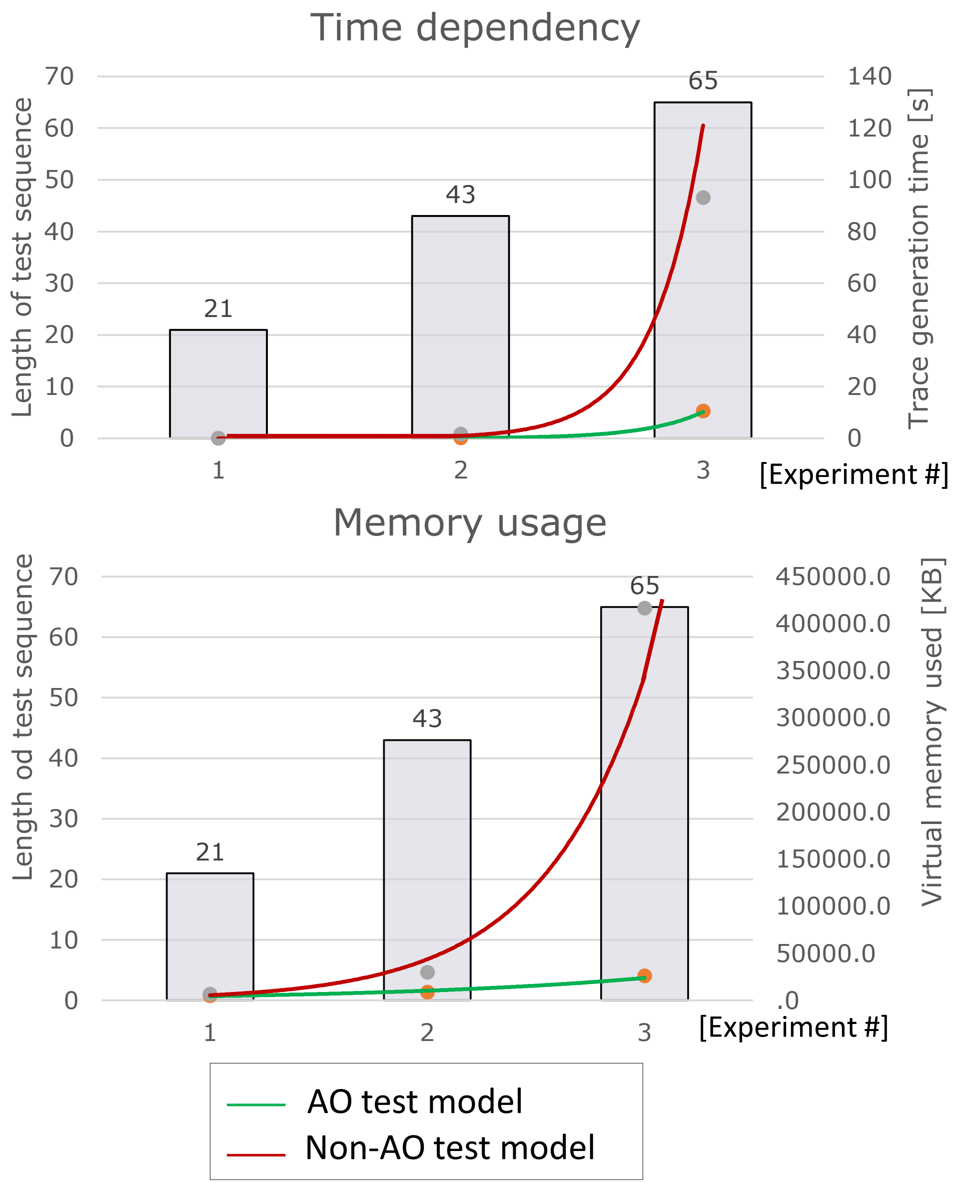
Generating tests that cover selected paths in the advice models

- Test path selection condition:

$$E\Diamond\ forall(k\!:K)\ forall(i\!:I)\ T[k][i]\ \&\&\ Model.stop$$

where
T is a Boolean array where its element $T[k][i]$ is updated to *true* when the i-th path of k-th advice is traversed in the model.

Note.
The comparison is made with weakly bisimilar non-AO models.



Conclusions

- We gave interpretation of generic AOM concepts in terms of UTA formalism
- Provided taxonomy of AOT coverage criteria that improve the traceability of bugs
- Defined AOM correctness properties in terms of TCTL model checking query templates.
- Defined AO test purpose feasibility conditions that can be verified by Uppaal model checker
- Demonstrated both analythically and experimentally that
  - AOM simplifies test purpose specification and model construction effort
  - reduces the model-based offline test generation complexity exponentially.

- Thank you!