Learning to Reuse

Adaptive Model Learning for Evolving Systems

Carlos Diego N. Damasceno
damascenodiego@usp.br
University of Sao Paulo, BR and
University of Leicester, UK

Mohammad Reza Mousavi
mm789@leicester.ac.uk
University of Leicester
Leicester, UK

Adenilso Simao
adenilso@icmc.usp.br
University of Sao Paulo
São Carlos, BR

December 5th 2019

Problem Statement

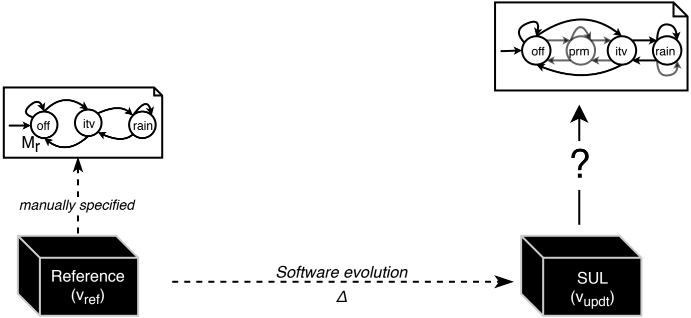


Figure: How can we efficiently build **behavioral models** from evolving systems?

Contribution

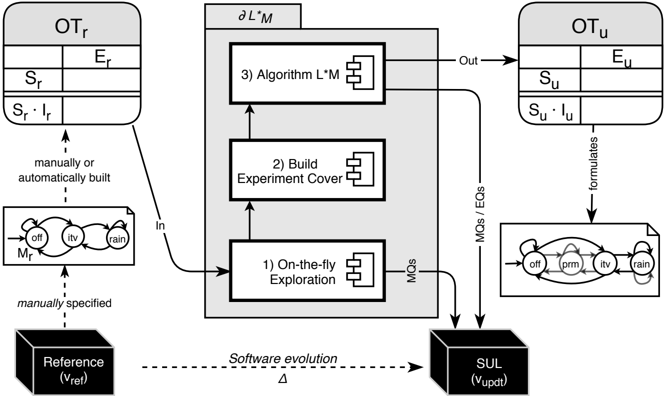


Figure: Introduce an adaptive algorithm that is **more efficient than the state-of-the-art** for **learning** behavioral models from evolving systems **by reuse**

Model Learning

Model Learning

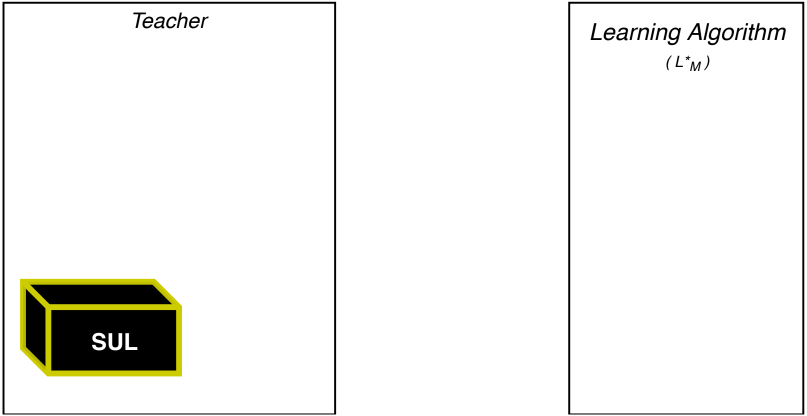


Figure: Active Model Learning (Angluin, 1987)

Model Learning

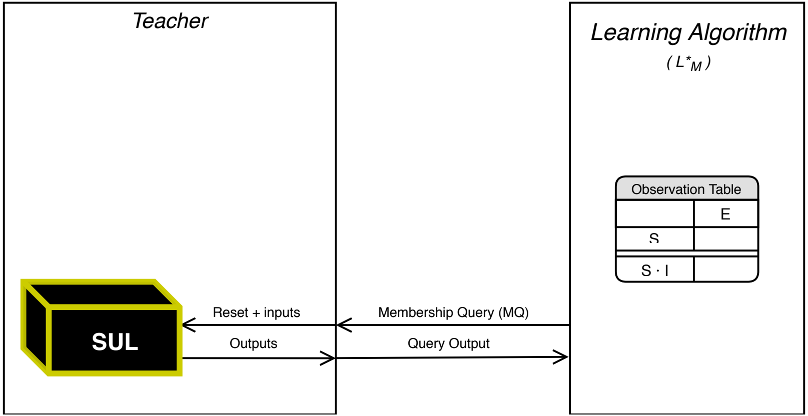


Figure: Active Model Learning (Angluin, 1987)

Model Learning

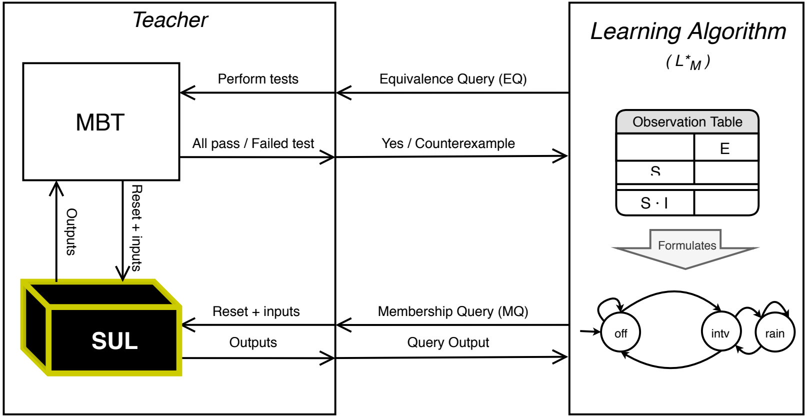


Figure: Active Model Learning (Angluin, 1987)

Model Learning (Example)

Figure: Windscreen wiper supporting intervaled and fast wiping

Model Learning (Example)

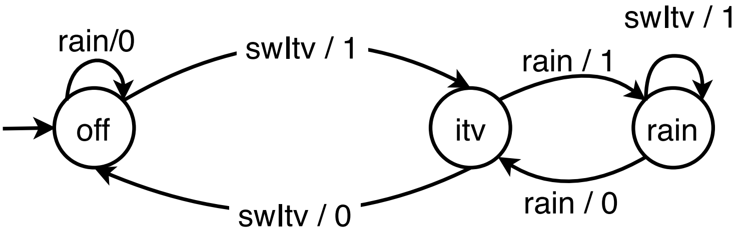


Figure: Windscreen wiper supporting intervaled and fast wiping

Model Learning (Example)

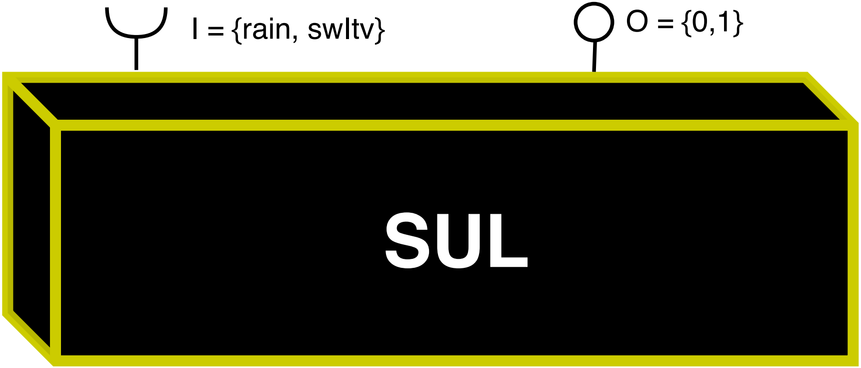


Figure: Windscreen wiper supporting intervaled and fast wiping

Model Learning (Learning Mealy Machines)

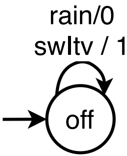


Figure: Initial Hypothesis

		rain	swItv
S	ϵ	0	1
$S \cdot I$	rain	0	1
	swItv	1	0

Table: Initial observation table (OT)

Model Learning (Learning Mealy Machines)

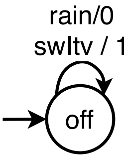


Figure: First Hypothesis

		`rain`	`swItv`
S	ϵ	0	1
$S \cdot I$	`rain`	0	1
	`swItv`	1	0

Table: First observation table

Model Learning (Learning Mealy Machines)

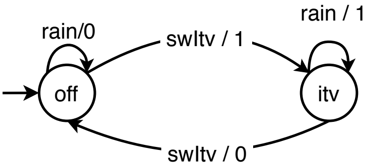


Figure: Second Hypothesis

		rain	swItv
S	ϵ	0	1
	swItv	1	0
$S \cdot I$	rain	0	1
	swItv · rain	0	1
	swItv · swItv	0	1

Table: Second observation table

Model Learning (Learning Mealy Machines)

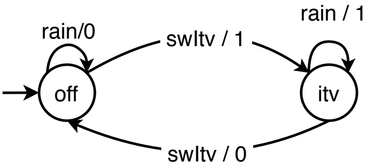


Figure: Second Hypothesis

		rain	swItv
S	ϵ	0	1
	swItv	1	0
$S \cdot I$	rain	0	1
	swItv · rain	0	1
	swItv · swItv	0	1

Table: Second observation table ($\mathcal{H} \neq$ SUL)

$$\texttt{EQ} = \texttt{swItv} \cdot \texttt{rain} \cdot \boxed{rain \cdot rain}$$

$$1 \cdot 1 \cdot \boxed{1 \cdot 1} \neq 1 \cdot 1 \cdot \boxed{0 \cdot 1}$$

Model Learning (Learning Mealy Machines)

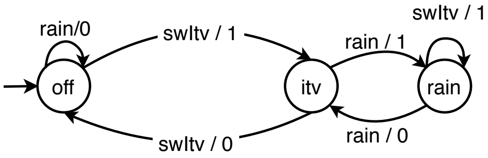


Figure: Final Hypothesis

		rain	swItv	rain · rain
S	ϵ	0	1	0 · 0
	swItv	1	0	1 · 0
	swItv · rain	0	1	0 · 1
$S \cdot I$	rain	0	1	0 · 0
	swItv · swItv	0	1	0 · 0
	swItv · rain · rain	1	0	1 · 0
	swItv · rain · swItv	0	1	0 · 1

Table: Final OT

$$\mathtt{EQ} = \mathtt{Yes}$$

What if our SUL evolves?

Learning models from evolving systems

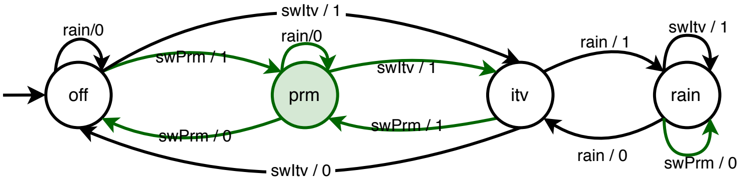


Figure: Windscreen wiper supporting intervaled and fast wiping **+ permanent movement**

Adaptive model learning for evolving systems

- Variant of model learning that attempts to speed up learning
- Reuse transfer and/or separating sequences from pre-existing models
  - Reduce the time for model checking (Groce et al., 2002; Chaki et al., 2008)
  - Find states maintained in newer versions (Windmüller et al., 2013)

Adaptive model learning for evolving systems

- Variant of model learning that attempts to speed up learning
- Reuse transfer and/or separating sequences from pre-existing models
  - Reduce the time for model checking (Groce et al., 2002; Chaki et al., 2008)
  - Find states maintained in newer versions (Windmüller et al., 2013)

Adaptive model learning for evolving systems

- Variant of model learning that attempts to speed up learning
- Reuse transfer and/or separating sequences from pre-existing models
  - Reduce the time for model checking (Groce et al., 2002; Chaki et al., 2008)
  - Find states maintained in newer versions (Windmüller et al., 2013)

Adaptive model learning for evolving systems

- Variant of model learning that attempts to speed up learning
- Reuse transfer and/or separating sequences from pre-existing models
  - Reduce the time for model checking (Groce et al., 2002; Chaki et al., 2008)
  - Find states maintained in newer versions (Windmüller et al., 2013)
- **Research Gaps:**
  - Reuse low quality sequences → Irrelevant `MQs` (Huistra et al., 2018)
  - How can we calculate good-quality subsets of sequences?

Adaptive model learning for evolving systems

- Variant of model learning that attempts to speed up learning
- Reuse transfer and/or separating sequences from pre-existing models
  - Reduce the time for model checking (Groce et al., 2002; Chaki et al., 2008)
  - Find states maintained in newer versions (Windmüller et al., 2013)
- **Research Gaps:**
  - Reuse low quality sequences → Irrelevant `MQs` (Huistra et al., 2018)
  - How can we calculate good-quality subsets of sequences?

Adaptive model learning for evolving systems

- Variant of model learning that attempts to speed up learning
- Reuse transfer and/or separating sequences from pre-existing models
  - Reduce the time for model checking (Groce et al., 2002; Chaki et al., 2008)
  - Find states maintained in newer versions (Windmüller et al., 2013)
- **Research Gaps:**
  - Reuse low quality sequences $\rightarrow$ Irrelevant `MQs` (Huistra et al., 2018)
  - How can we calculate good-quality subsets of sequences?

partial-Dynamic L^*_M

The partial-Dynamic L^*_M algorithm

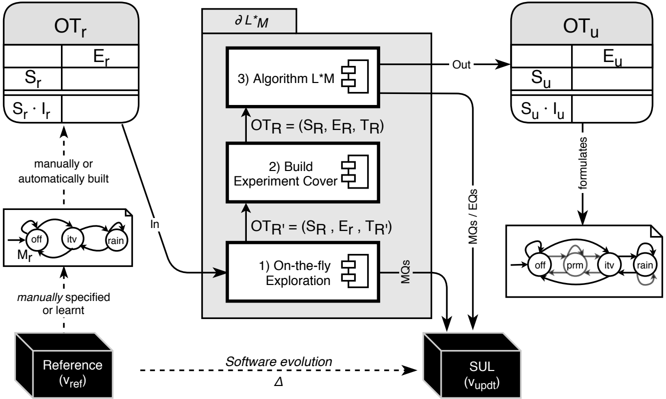


Figure: Schematic representation of the partial-Dynamic L^*_M[1]

[1]We have implemented our approach on top of the LearnLib framework (LearnLib, 2017)

Step 1: On-the-fly exploration of the reused OT

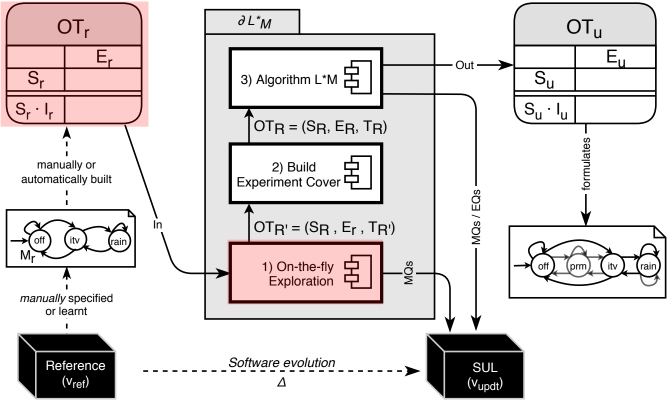
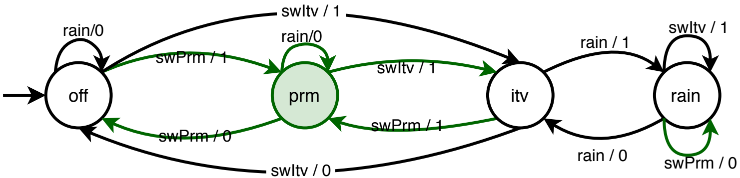


Figure: The `partial-Dynamic` $\mathtt{L}^*_\mathtt{M}$ algorithm starts by exploring reused `OT`s on-the-fly to discard *redundant* transfer sequences [2]

---

[2]**Improvement #1:** We designed this step inspired by Chaki et al. (2008)

Step 1: On-the-fly exploration of the reused OT



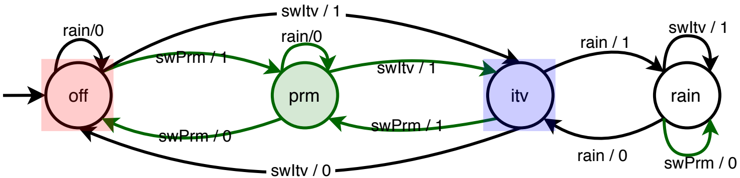
Let the sets of reused prefixes and suffixes be

$$S_r = \{ \epsilon, swItv, swItv \cdot rain, swItv \cdot rain \cdot rain, swItv \cdot rain \cdot rain \cdot swItv, rain \}$$

$$E_r = \{rain, swItv, swPrm, rain \cdot rain\}$$

Goal: Find a $S_R \subseteq S_r$ with the same state coverage capability but less prefixes

Step 1: On-the-fly exploration of the reused OT



Let the sets of reused prefixes and suffixes be

$$S_r = \{ \epsilon, \; swItv, \; swItv \cdot rain, \; swItv \cdot rain \cdot rain, \; swItv \cdot rain \cdot rain \cdot swItv, \; rain \}$$

$$E_r = \{rain, swItv, swPrm, rain \cdot rain\}$$

Goal: Find a $S_R \subseteq S_r$ with the same state coverage capability but less prefixes

Step 1: On-the-fly exploration of the reused `OT`

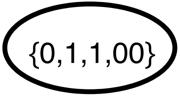


Figure: On-the-fly exploration using **depth-first search**

Step 1: On-the-fly exploration of the reused `OT`

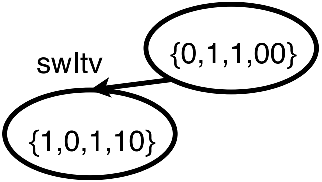


Figure: On-the-fly exploration using **depth-first search**

Step 1: On-the-fly exploration of the reused `OT`

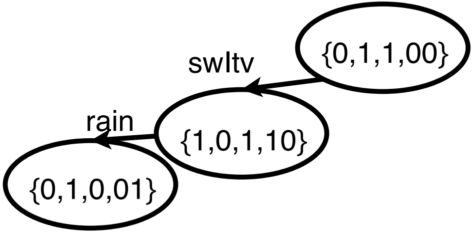


Figure: On-the-fly exploration using **depth-first search**

Step 1: On-the-fly exploration of the reused `OT`

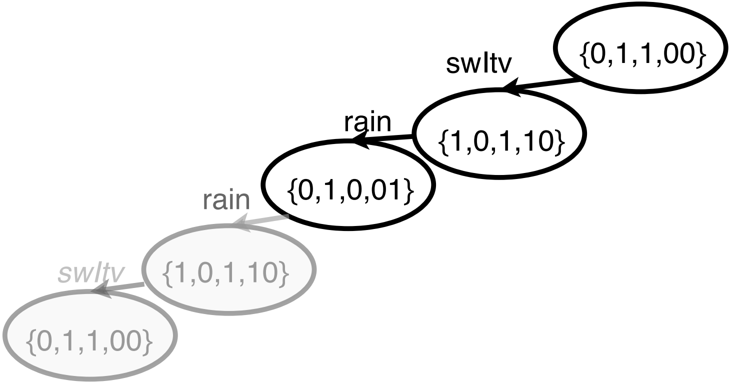


Figure: On-the-fly exploration using **depth-first search**

Step 1: On-the-fly exploration of the reused `OT`

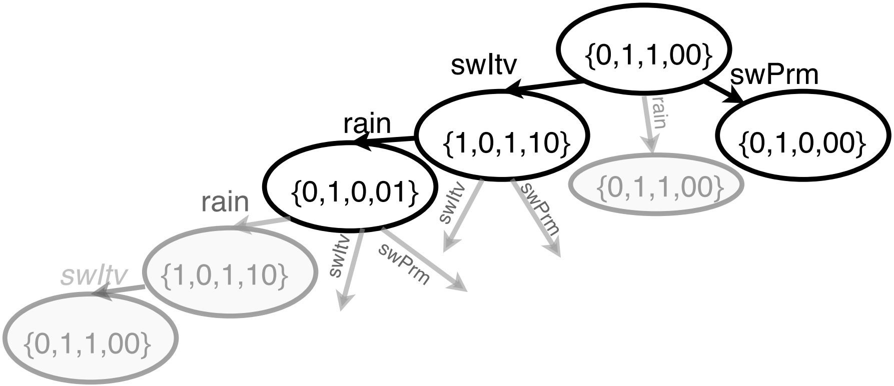


Figure: On-the-fly exploration using **depth-first search**

40 `MQs` vs. 76 `MQs`

Step 2: Building an experiment cover tree

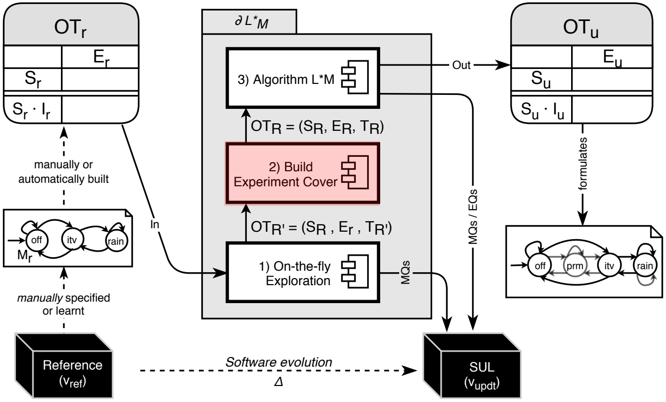
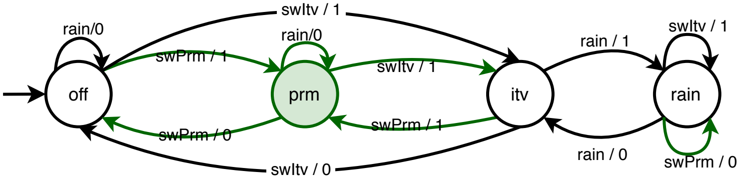


Figure: The partial-Dynamic L_M^* algorithm searches for *deprecated* separating sequences [3]

[3]**Improvement #2:** We used breadth-first search to minimize the set of separating sequences

Step 2: Building an experiment cover tree



Let the sets of prefixes and suffixes be

$$S_R = \{ \epsilon, swItv, swItv \cdot rain, swPrm \}$$

$$E_r = \{rain, swItv, swPrm, rain \cdot rain\}$$

Goal: Find a smaller subset $E_R \subseteq E_r$ of representative separating sequences.

Step 2: Building an experiment cover tree

Figure: Finding an optimal subset of representative *separating sequences* using **breadth-first search** to group transfer sequences into **equivalence classes**

Step 2: Building an experiment cover tree

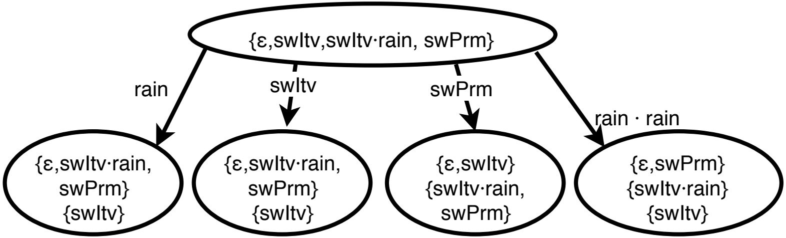


Figure: Finding an optimal subset of representative *separating sequences* using **breadth-first search** to group transfer sequences into **equivalence classes**

Step 2: Building an experiment cover tree

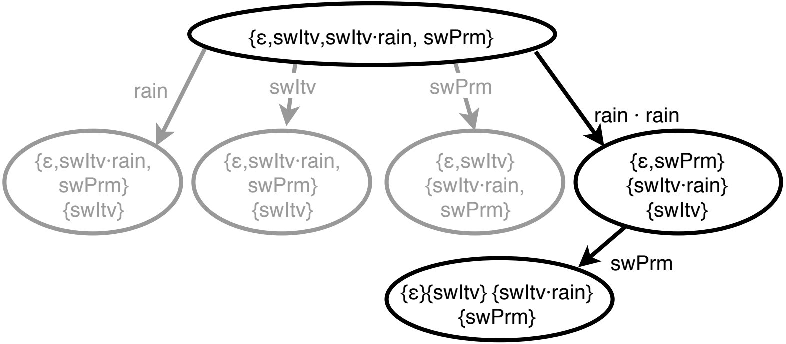


Figure: Finding an optimal subset of representative *separating sequences* using **breadth-first search** to group transfer sequences into **equivalence classes**

2 sequences vs. 4 sequences

Step 3: Running L_M^* using the outcomes of partial-Dynamic L_M^*

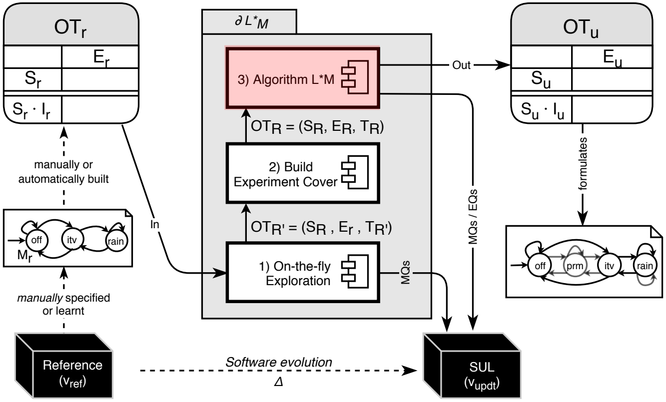


Figure: The L_M^* algorithm **starts from reused transfer and separating sequences** to reach and distinguish more states than **in the traditional setup** (i.e., initial state only) [4]

[4] **Improvement #3:** We use the subsets of reused sequences as the initial setup for model learning

Empirical evaluation

Empirical evaluation

(RQ1) Is our technique more efficient than the state-of-the-art of adaptive learning? (i.e., `MQs` and `EQs`)

(RQ2) Is the effectiveness of adaptive learning strongly affected by the temporal distance between versions?

Subject systems

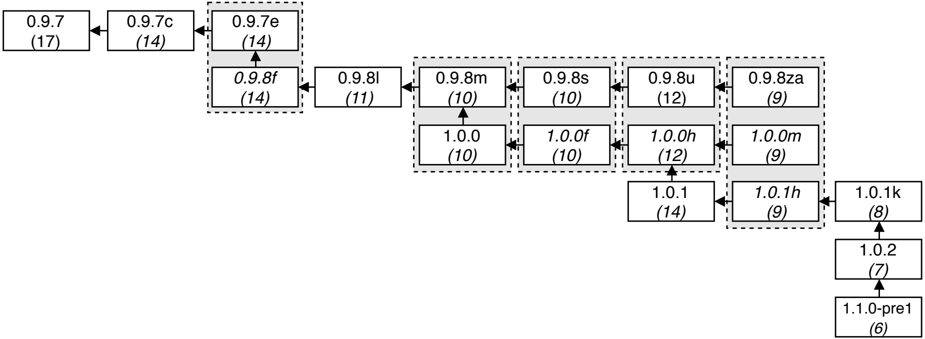


Figure: OpenSSL toolkit: 18 FSMs versions used as SUL (de Ruiter, 2016)

Experiment design

- We learnt models for all pairs of versions and precedents (given their release dates)
- We calculated the temporal distance (in years) for all pairs of versions
- We measured the numbers of `MQ`s and `EQ`s for all learning experiments
- Four adaptive learning algorithms (Huistra et al., 2018; Chaki et al., 2008)

Analysis of Results (Average number of MQs)

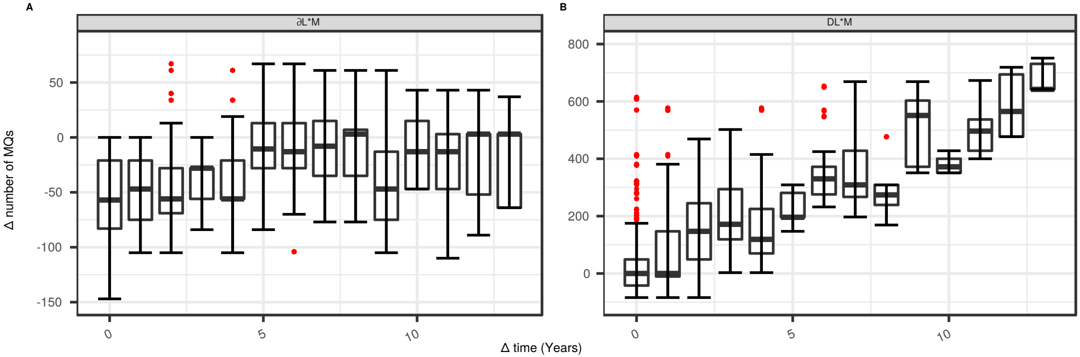


Figure: Our technique required less MQs

Analysis of Results (Average number of MQs)

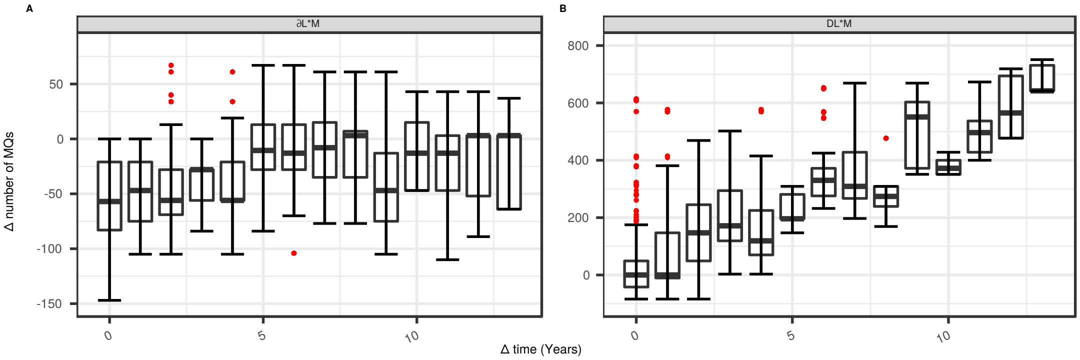


Figure: Our technique was not influenced by the temporal distance between versions

Analysis of Results (Average number of EQs)

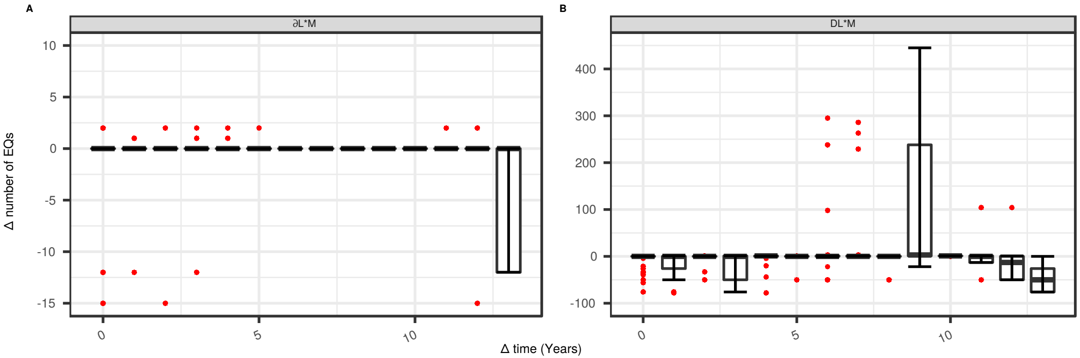


Figure: Boxplots of the μEQs posed by adaptive learning

Analysis of Results (RQs)

Research Question 1:

- Our technique required less `MQs` than the other techniques
- Our technique required a similar number of `EQs` compared to the other techniques

Research Question 2:

- The state-of-the-art of adaptive learning were more sensitive to software evolution
  - *strong* positive correlation (`MQs`)
- Our technique was not influenced by the temporal distance between versions
  - weak positive correlation (`MQs`)
- Temporal distance vs. `EQs` $\rightarrow$ *very weak* positive correlation

Conclusions and Future Work

Conclusions and Future Work

- Software evolution undermines the state-of-the-art of adaptive learning
  - redundant transfer sequences
  - deprecated separating sequences
- We showed that the $\partial L^*_{\mathtt{M}}$ algorithm is:
  - less sensitive to software evolution
  - more efficient than the state-of-the-art in terms of MQs
- Future work:
  - Learning models of software product lines
    - Learn by **merging** product models [5]
    - Learn by **querying** $\rightarrow$ Reuse family models
  - Adaptive learning for *Discrimination tree*-based algorithms
    - TTT (Isberner et al., 2014)

[5] Preliminary findings published at the SPLC 2019 (Damasceno et al., 2019)

Questions?

https://damascenodiego.github.io/DynamicLstarM/

References I

Angluin, D. (1987). Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87–106.

Chaki, S., Clarke, E., Sharygina, N., and Sinha, N. (2008). Verification of evolving software via component substitutability analysis. *Formal Methods in System Design*, 32(3):235–266.

Damasceno, C. D. N., Mousavi, M. R., and da Silva Simao, A. (2019). Learning from difference: An automated approach for learning family models from software product lines. In *Proceeedings of the 23rd International Systems and Software Product Line Conference - Volume 1, SPLC 2019*, Paris, France. ACM Press.

de Ruiter, J. (2016). A tale of the openssl state machine: A large-scale black-box analysis. In *Secure IT Systems - 21st Nordic Conference, NordSec 2016, Oulu, Finland, November 2-4, 2016, Proceedings*, pages 169–184.

Groce, A., Peled, D., and Yannakakis, M. (2002). Adaptive model checking. In *Proceedings of the 8th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, TACAS '02, pages 357–370, London, UK, UK. Springer-Verlag.

References II

Huistra, D., Meijer, J., and Pol, J. (2018). Adaptive learning for learn-based regression testing. In Howar, F. and Barnat, J., editors, *Formal Methods for Industrial Critical Systems*, Lecture Notes in Computer Science, pages 162–177, Switzerland. Springer Publishers.

Isberner, M., Howar, F., and Steffen, B. (2014). The TTT algorithm: A redundancy-free approach to active automata learning. In Bonakdarpour, B. and Smolka, S. A., editors, *Runtime Verification*, pages 307–322. Springer International Publishing.

LearnLib (2017). LearnLib: a framework for automata learning. `https://learnlib.de/`. [Online; accessed 17-Out-2017].

Windmüller, S., Neubauer, J., Steffen, B., Howar, F., and Bauer, O. (2013). Active continuous quality control. In *Proceedings of the 16th International ACM Sigsoft Symposium on Component-based Software Engineering*, CBSE '13, pages 111–120, New York, NY, USA. ACM.