Commuting Strategies for Product-line Reliability Analysis

Thiago Castro – PhD student

Prof. Vander Alves, PhD – Supervisor

Prof. Leopoldo Teixeira, PhD – Co-supervisor

Universidade de Brasília

Universidade Federal de Pernambuco

Universität Passau

Université de Namur

Commuting Strategies for Product-line Reliability Analysis

Type checking

Data-flow analysis

Control-flow analysis

Model checking

Theorem proving

...

Commuting Strategies for Product-line Reliability Analysis

Type checking

Data-flow analysis

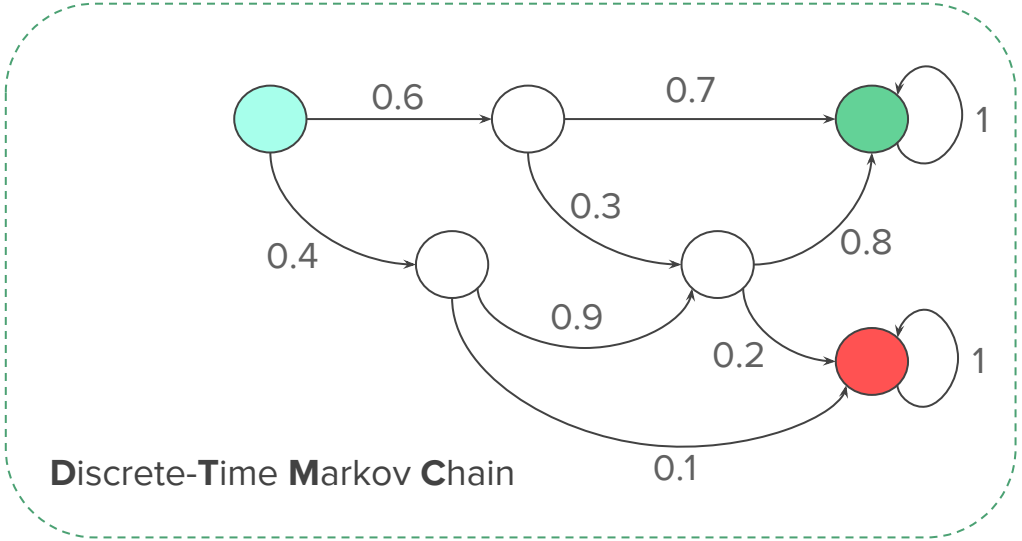
Control-flow analysis

Model checking

Theorem proving

...

Commuting Strategies for Product-line Reliability Analysis

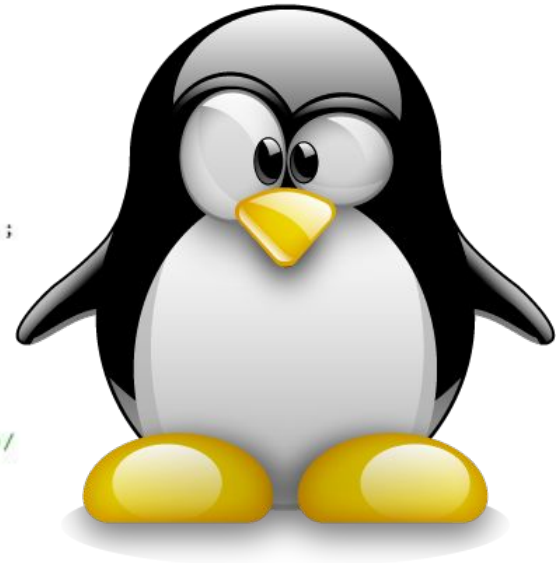


$$P_{=?} [\Diamond \textit{success}]$$

$$(0.6 * 0.7) + (0.6 * 0.3 * 0.8) + (0.4 * 0.9 * 0.8)$$

Commuting Strategies for Product-line Reliability Analysis

```
struct task_struct {
        volatile long state;    /* -1 unrunnable, 0 runnable, >0 stopped */
        void *stack;
        atomic_t usage;
        unsigned int flags;     /* per process flags, defined below */
        unsigned int ptrace;

#ifdef CONFIG_SMP
        struct llist_node wake_entry;
        int on_cpu;
        struct task_struct *last_wakee;
        unsigned long wakee_flips;
        unsigned long wakee_flip_decay_ts;

        int wake_cpu;
#endif
        int on_rq;

        int prio, static_prio, normal_prio;
        unsigned int rt_priority;
        const struct sched_class *sched_class;
        struct sched_entity se;
        struct sched_rt_entity rt;
#ifdef CONFIG_CGROUP_SCHED
        struct task_group *sched_task_group;
#endif

#ifdef CONFIG_PREEMPT_NOTIFIERS
        /* list of struct preempt_notifier: */
        struct hlist_head preempt_notifiers;
#endif

#ifdef CONFIG_BLK_DEV_IO_TRACE
        unsigned int btrace_seq;
#endif
```

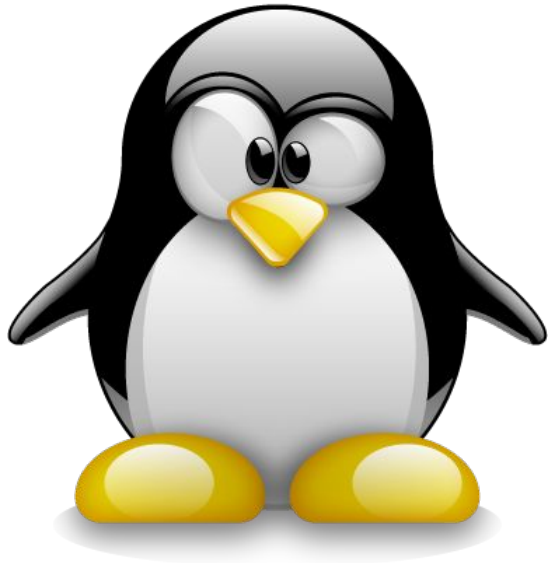


Commuting Strategies for Product-line Reliability Analysis

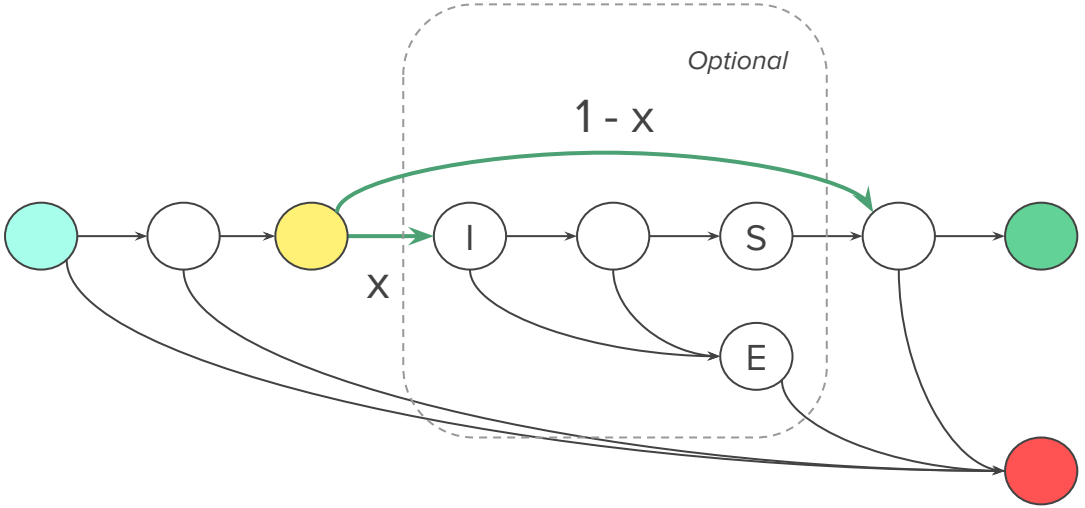
Commuting Strategies for Product-line Reliability Analysis

Products = O(2^{Options})

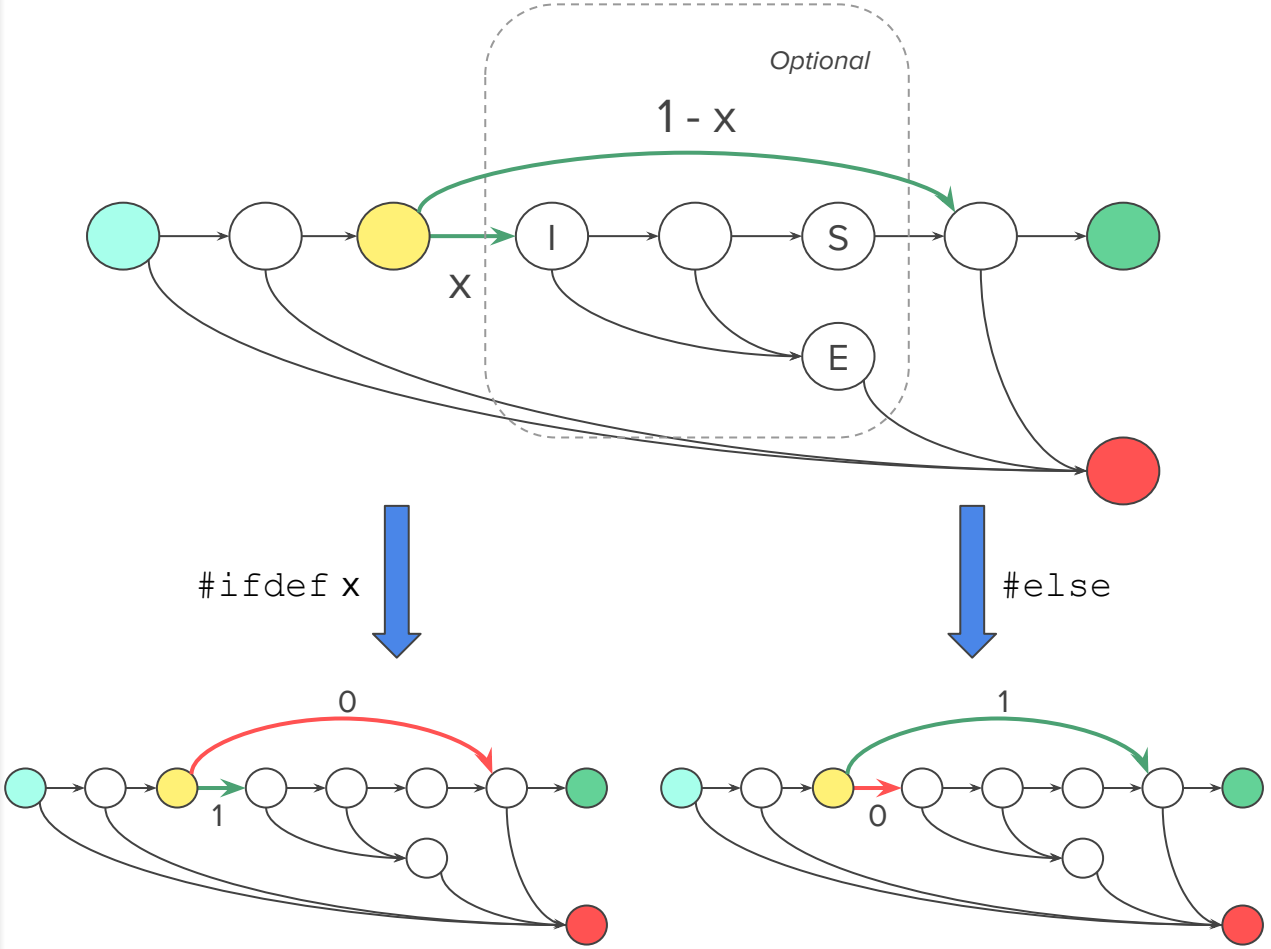
> 6.000 options



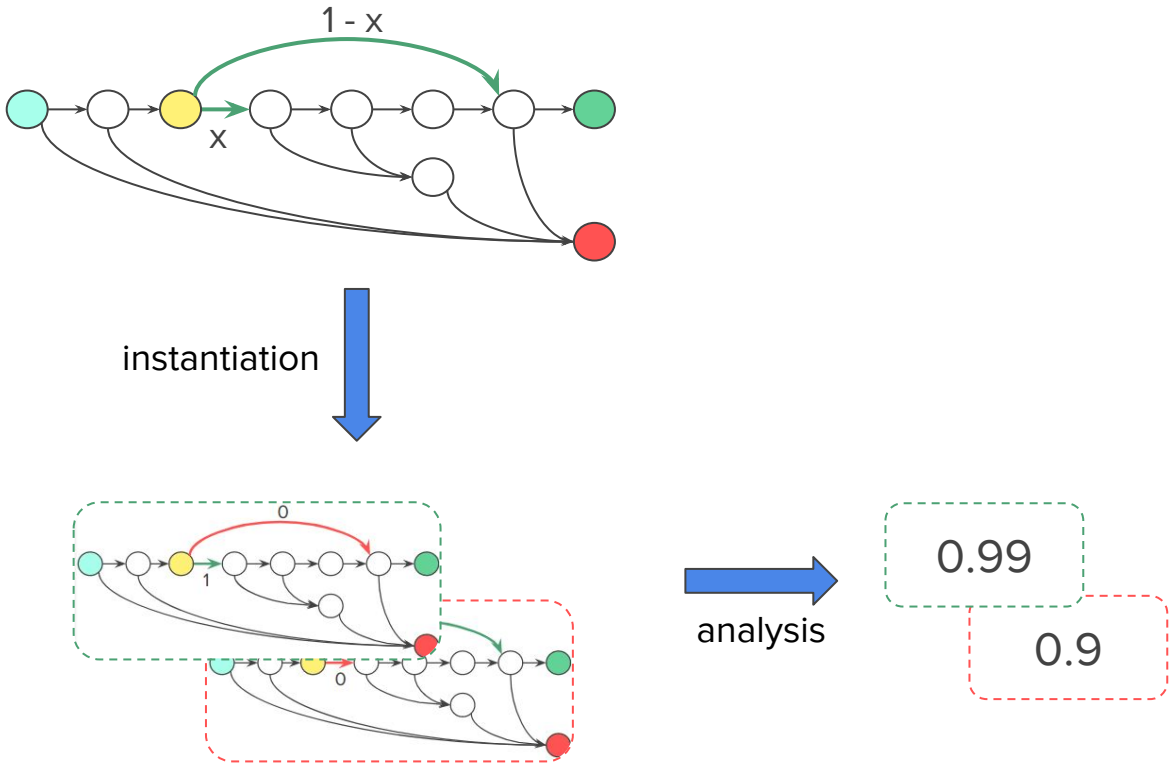
Commuting Strategies for Product-line Reliability Analysis



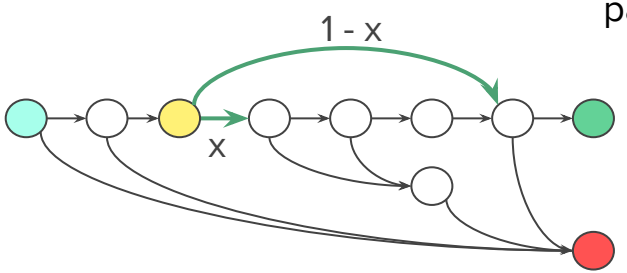
Commuting Strategies for Product-line Reliability Analysis



Commuting Strategies for Product-line Reliability Analysis



Commuting Strategies for Product-line Reliability Analysis



parametric analysis

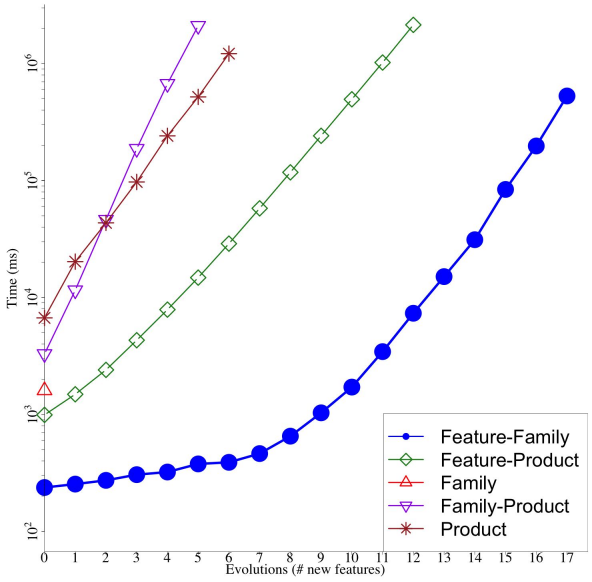
0.99*x + 0.9*(1-x)

evaluation

0.99

0.9

Commuting
Strategies
for
Product-line
Reliability
Analysis



Trade-offs

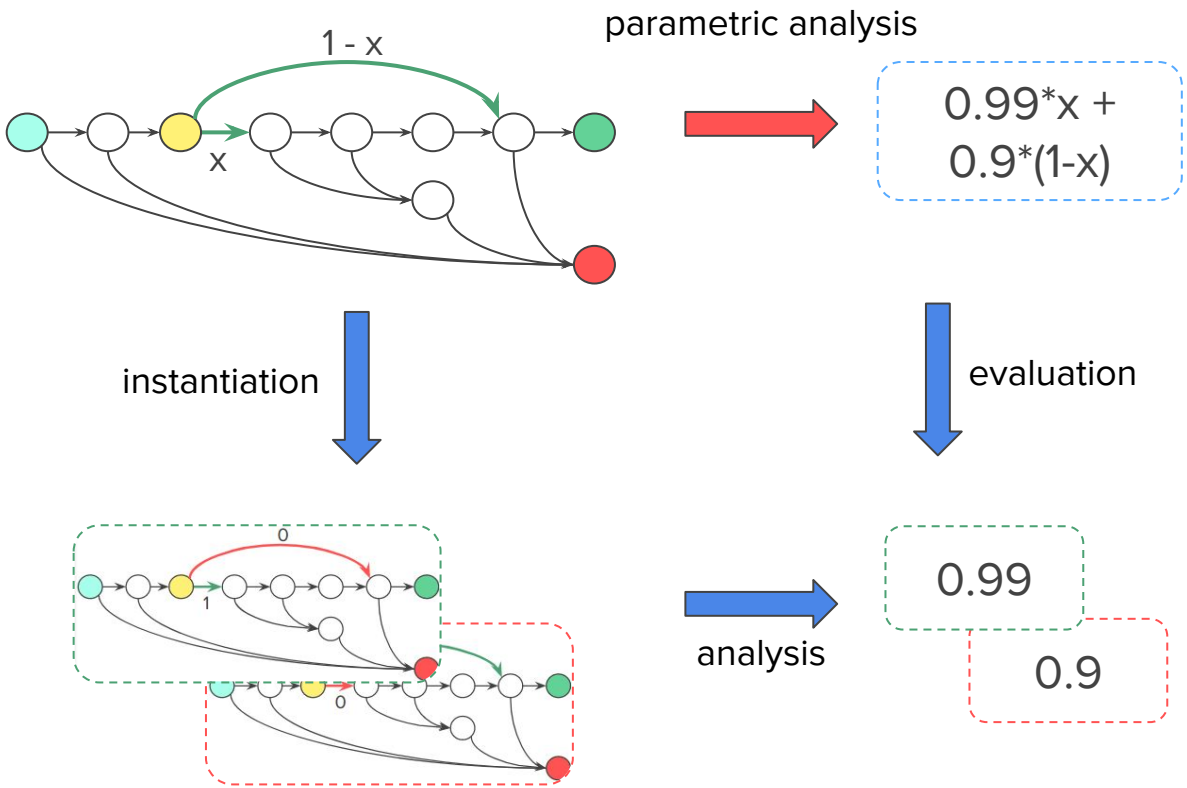
Space

Time

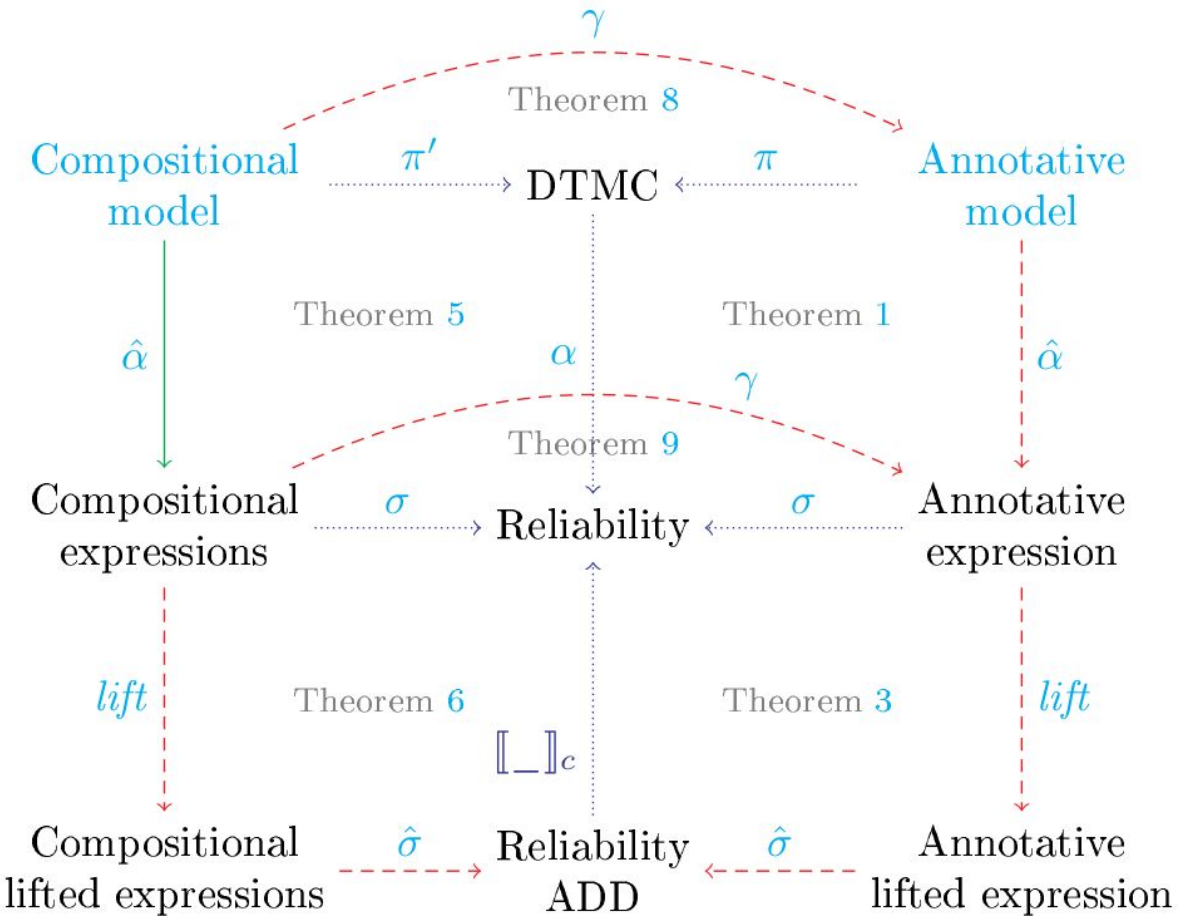
Simplicity

...

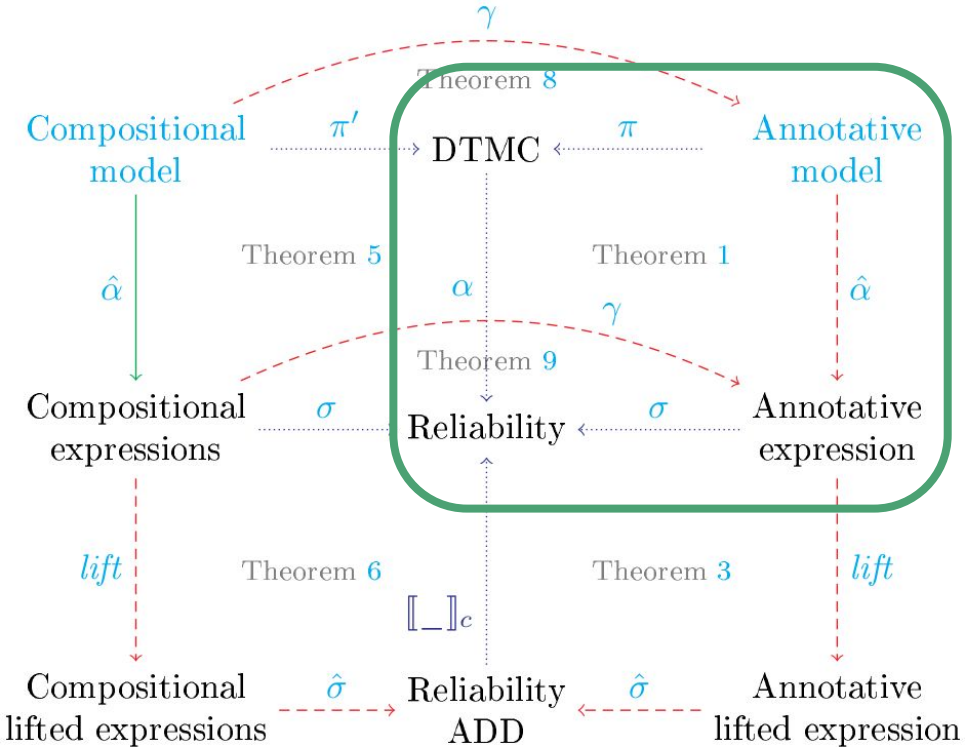
Commuting Strategies for Product-line Reliability Analysis



Commuting Strategies for Product-line Reliability Analysis



Commuting Strategies for Product-line Reliability Analysis

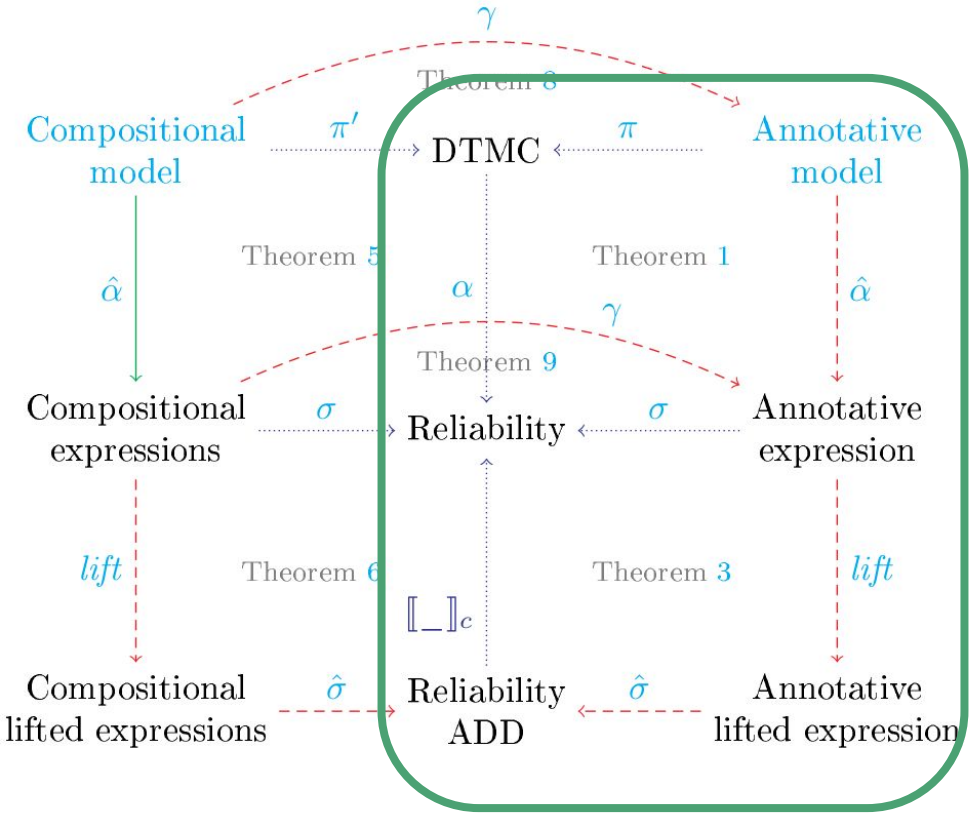


PVS

VS

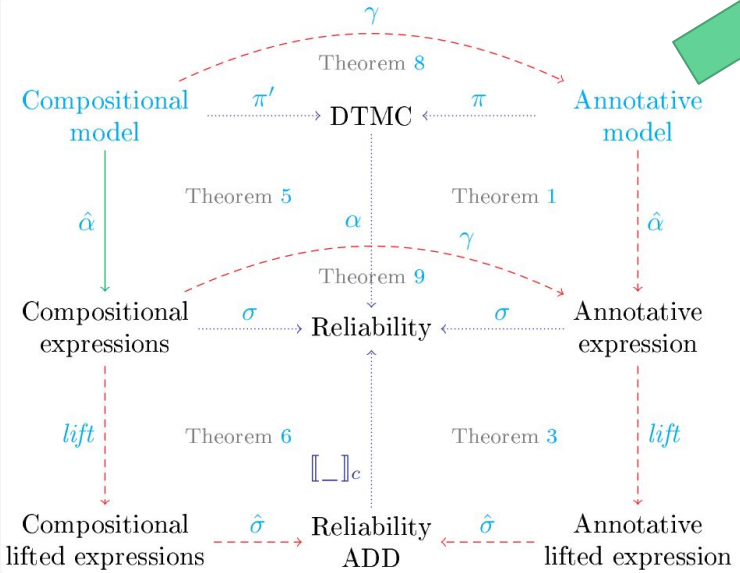
Coq

Commuting Strategies for Product-line Reliability Analysis



PVS

Commuting Strategies for Product-line ~~Reliability~~ Analysis



Type checking

Data-flow analysis

Control-flow analysis

Model checking *(other properties)*

Theorem proving

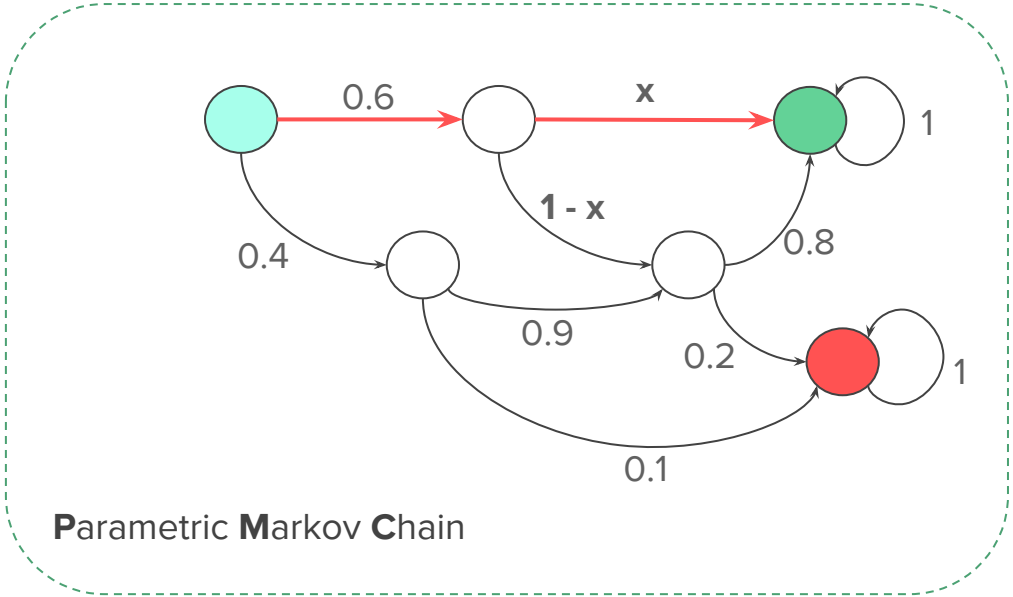
...

Thank you!

Backup Slides

Reliability Analysis of PMCs

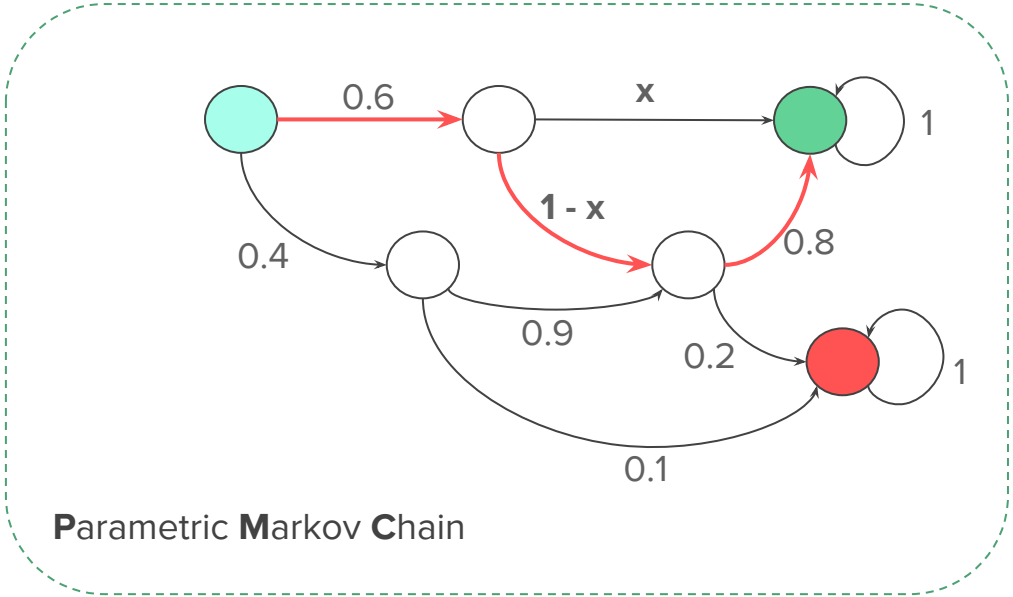
R = 0.6 * x
+ 0.6 * (1-x) * 0.8
+ 0.4 * 0.9 * 0.8



Parametric **M**arkov **C**hain

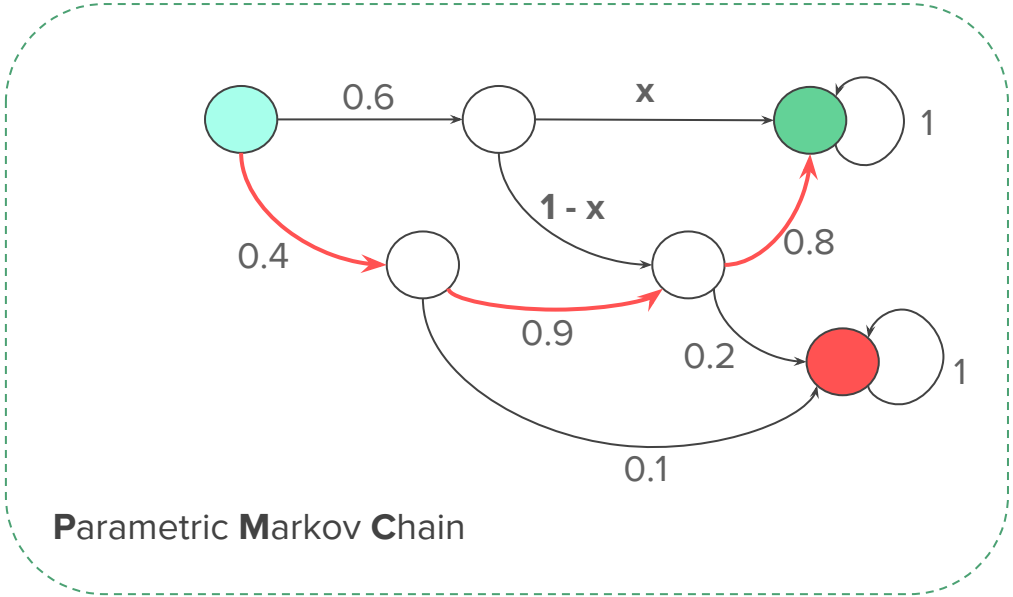
Reliability Analysis of PMCs

R = 0.6 * x
+ 0.6 * (1-x) * 0.8
+ 0.4 * 0.9 * 0.8



Reliability Analysis of PMCs

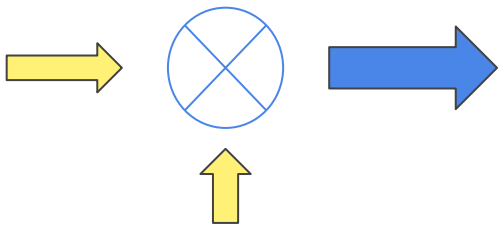
R = 0.6 * x
+ 0.6 * (1-x) * 0.8
+ 0.4 * 0.9 * 0.8



Parametric **M**arkov **C**hain

Expression Evaluation

0.98*x + 0.9*y + 0.01



Configuration	x	y
{ ..., F, ... }	0.8	0.01
{ ..., F, G, G1, ... }	0.5	0.5
{ ..., F, G, G2, ... }	0.01	1
...	...	...

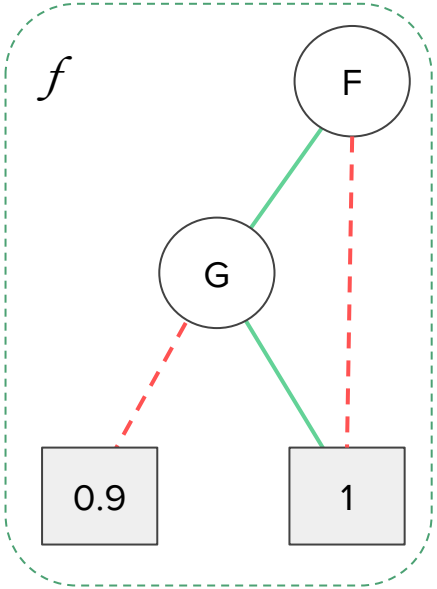
Configuration	R
{ ..., F, ... }	0.884
{ ..., F, G, G1, ... }	0.95
{ ..., F, G, G2, ... }	0.92
...	...

Algebraic **D**ecision **D**iagrams

$$f(F, G) = \begin{cases} 0.9 & \text{if } F \wedge \neg G \\ 1 & \text{otherwise} \end{cases}$$

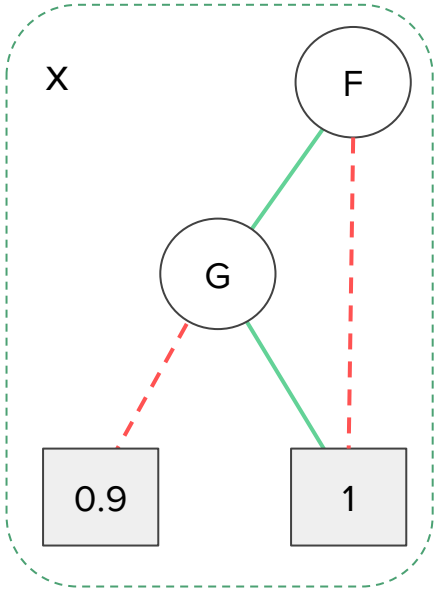
Features

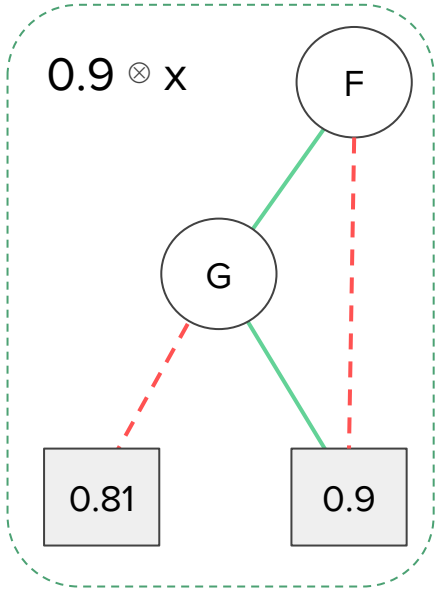
Presence condition



Expression Evaluation with ADDs

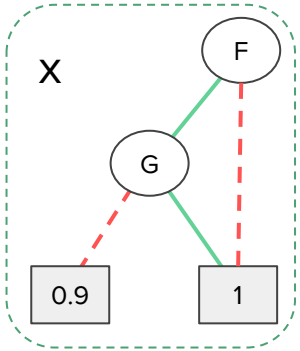
F	G	x
0	0	1
0	1	1
1	0	0.9
1	1	1

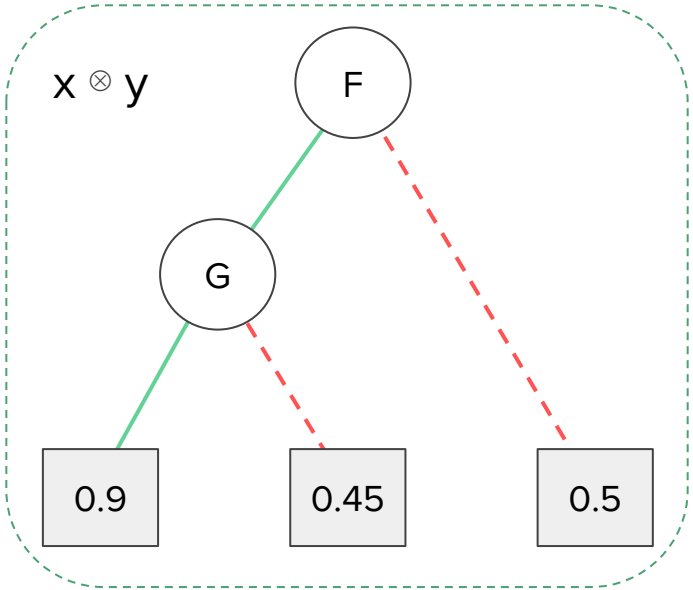




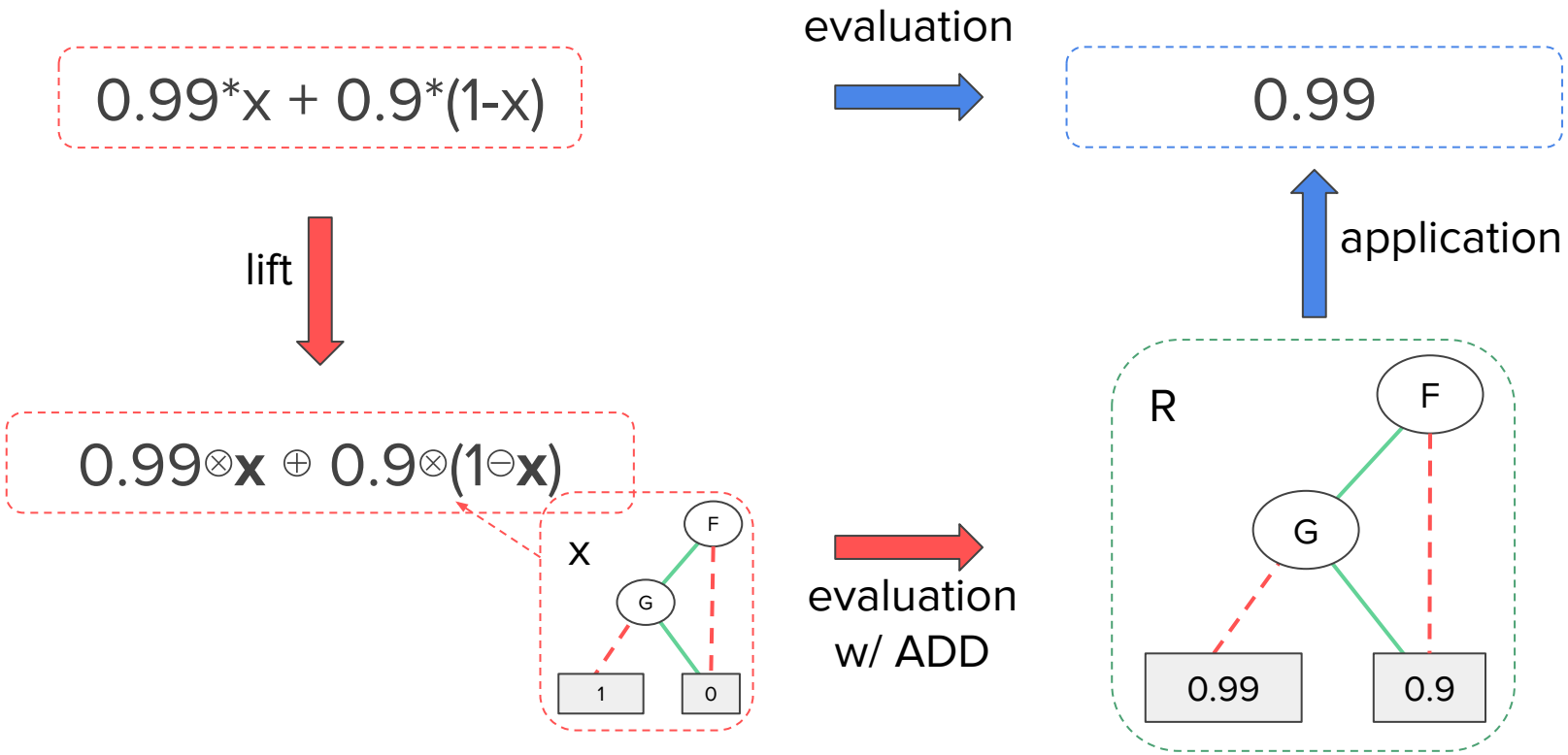
Expression Evaluation with ADDs

F	G	x	y	x⊗y
0	0	1	0.5	0.5
0	1	1	0.5	0.5
1	0	0.9	0.5	0.45
1	1	1	0.9	0.9

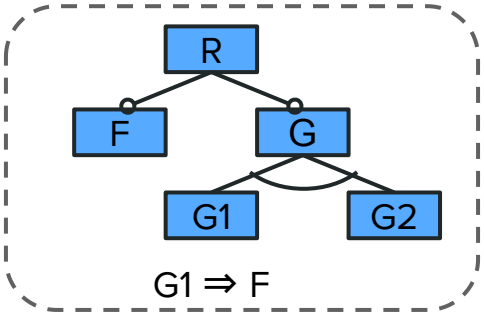




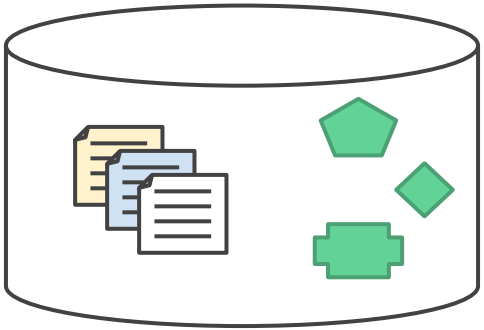
Alternative Strategies



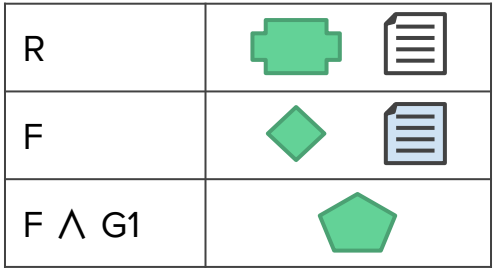
Commuting Strategies for Product-line Reliability Analysis



Valid configurations



Reusable parts



Assembly instructions

References

Thüm, T. et al. (2014). **A Classification and Survey of Analysis Strategies for Software Product Lines.** ACM Computing Surveys , 47(1): 145.

von Rhein, A. et al. (2013). **The PLA model.** In *Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems* - VaMoS '13 , page 1, New York, New York, USA. ACM Press.

Ghezzi, C. and Molzam Sharifloo, A. (2013). **Model-based verification of quantitative non-functional properties for software product lines.** Information and Software Technology , 55(3):508524.

Apel, S. et al. (2013). **Strategies for product-line verification: Case studies and experiments.** In *Proceedings - International Conference on Software Engineering*, ICSE '13, pages 482-491, Piscataway, NJ, USA. IEEE Press.

Related Work

Midtgaard, J. et al. (2015). **Systematic derivation of correct variability-aware program analyses.** *Science of Computer Programming* 105.

Walkingshaw, E. et al. (2014). **Variational Data Structures: Exploring Tradeoffs in Computing with Variability.** In *Proceedings of the 2014 ACM International Symposium on New Ideas, New Paradigms, and Reflections on Programming & Software* – Onward! '14.

Chen, S., and Erwig, M. (2014). **Type-based parametric analysis of program families.** In *Proceedings of the 19th ACM SIGPLAN international Conference on Functional Programming*.

Erwig, M. and Walkingshaw, E. (2011). **The choice calculus: A representation for software variation.** *ACM Transactions on Software Engineering and Methodology (TOSEM)*.