UNIVERSITY of York

Efficient Formal Verification with Confidence Intervals

Authors: Naif Alasmari & Radu Calinescu

{nnma500, radu.calinescu}@york.ac.uk

Department of Computer Science

University of York

York, UK

About the Presenter – Naif Alasmari

- I am a Ph.D. student in the Department of Computer Science at the University of York, UK.
- My research interests are in the quantitative verification of component-based systems, verification under uncertainty, the correct-by-construction synthesis of discrete-event controllers, and performance models for cloud-based big data applications.
- I have a bachelor's degree in information systems from King Khalid University, Abha, Saudi Arabia and a master's degree in computer information systems from St. Mary's University, San Antonio, USA.
- I have worked as a system analyst's assistant at Umm Al Qura University, Makkah, Saudi Arabia and a lecturer in the College of Arts and Sciences at King Khalid University, where I was also head of the Department of Information Systems.

Outline

- Overview
- Introduction – FACT and ePMC
- eFACT
- Evaluation
- Conclusion & Future Work
- Bibliography

Overview

- FACT: **f**ormal verific**a**tion with **c**onfidence in**t**ervals [1]
- ePMC: **e**fficient **p**arametric **m**odel **c**hecking [2]
- eFACT: **e**fficient **f**ormal verific**a**tion with **c**onfidence in**t**ervals

Overview

- eFACT integrates FACT with ePMC.
- The purpose of eFACT is to:
  - Compute confidence intervals for non-trivial parametric Markov model properties when observations of the unknown transition probabilities of these models are available
  - Utilise an efficient binary search technique to further speed up the determination of the highest confidence level at which a non-functional requirement can be deemed violated or satisfied.

Introduction: FACT

- FACT is a model checking tool that is used to calculate confidence intervals for parametric discrete time Markov chains (pDTMCs) that have at least one transition probability with observation.
- FACT employs PRISM [3] to obtain an algebraic expression for the evaluated properties of pDTMCs models.
- FACT fails to produce confidence intervals for non-trivial pDTMCs models when:
  - The produced algebraic expression is too large to be computed
  - PRISM fails to produce an algebraic expression.

Introduction: ePMC

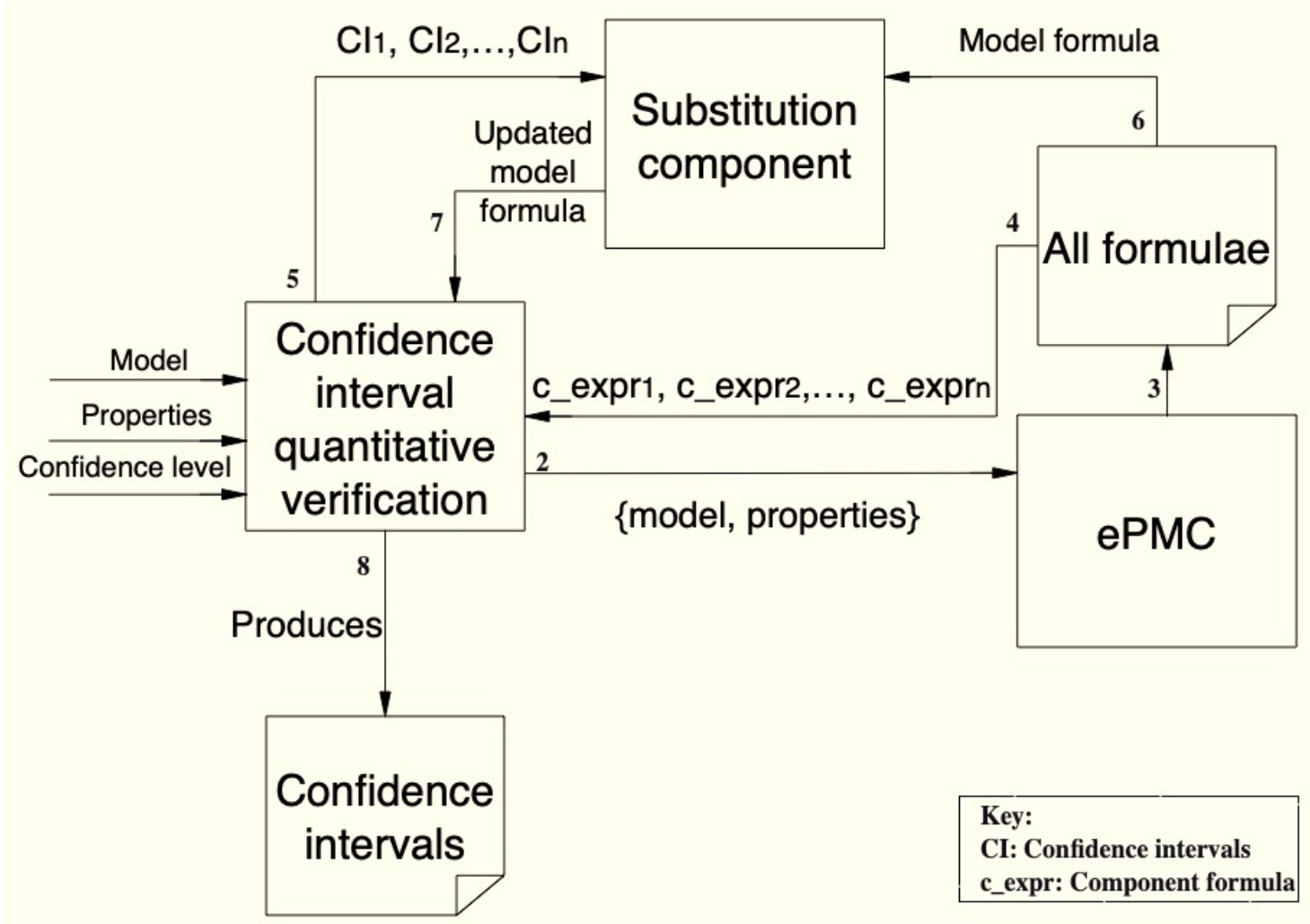
- ePMC uses domain-specific modelling patterns to produce sets of closed-form subexpressions for the analysed properties of pDTMCs.

- ePMC uses these subexpressions as terms in the main formula to analyse an entire pDTMCs model.

eFACT: Computing Confidence Intervals

- To overcome the limitation of FACT, eFACT exploits ePMC to compute the confidence intervals for non-trivial pDTMCs models.
- It invokes ePMC instead of PRISM to obtain algebraic expressions (subexpressions and main formula) and evaluate a property.
- It first computes confidence intervals for subexpressions.
- Then, it substitutes the results in the main formula to obtain the confidence intervals of a property.

eFACT: Steps for Computing Confidence Intervals

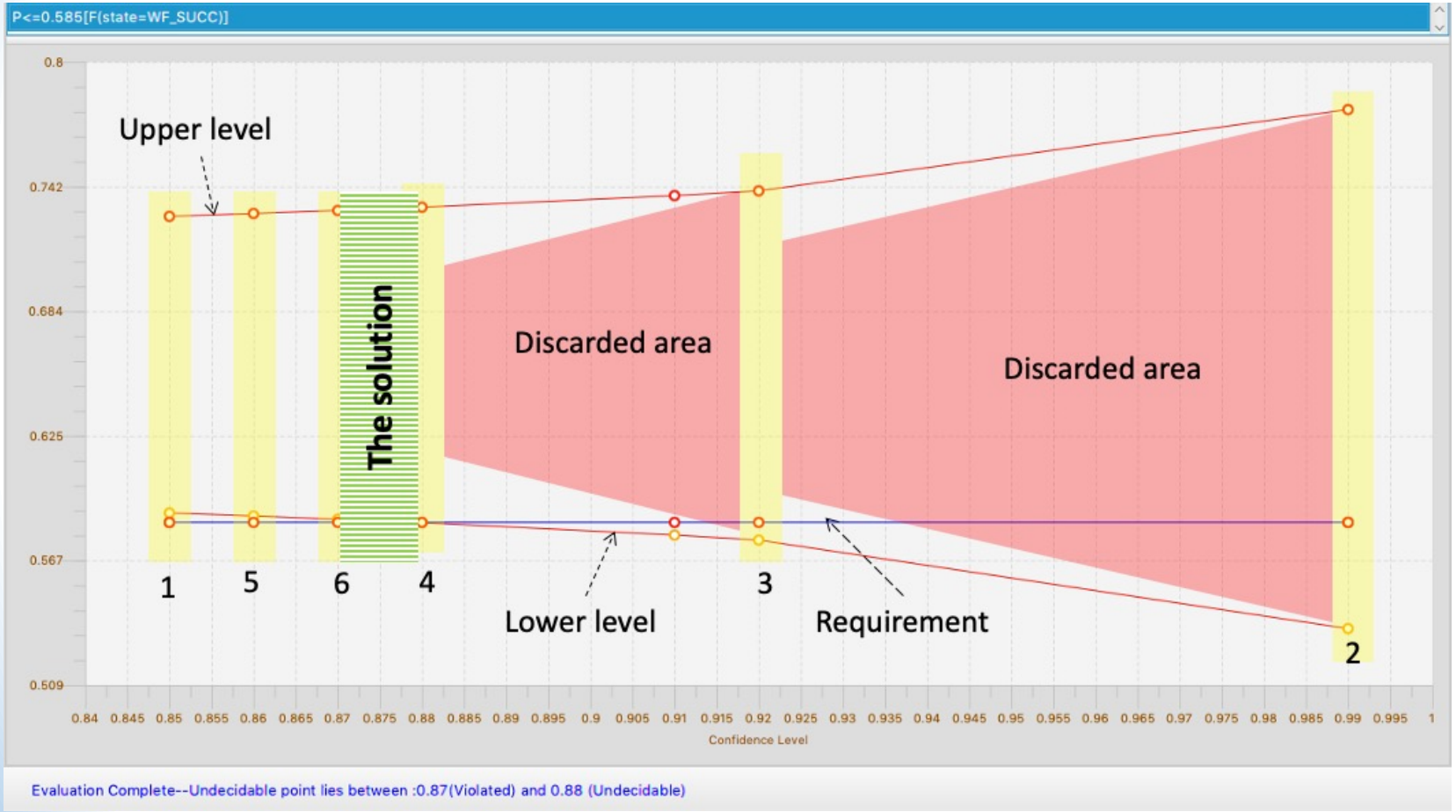


eFACT: Using Binary Search

Goal: to obtain the confidence level at which a non-functional requirement can be deemed violated or satisfied for an inserted range of confidence levels by following these steps:

1. The process begins by verifying the first inserted confidence level, computing its confidence intervals (CIs), and obtaining the CIs for the last level.
2. There are two confidence levels with their intervals, enabling eFACT to check whether the analysed property bound is located inside those intervals.
3. If the bound is located inside all the intervals, the process will terminate with a message stating that the requirement is undecidable for the given range of confidence levels (because the width of intervals is increased when the confidence level becomes greater).
4. Then, verify the middle confidence level and find its CIs.
5. The area between the middle level and another level (either the first or last) that has similar results is ignored.
6. The process moves to the area between the middle level and the other level whose result is different and verifies a new middle level for this area to obtain its CIs.
7. Repeat steps 5 and 6 until you find the solution.

eFACT: Using Binary Search



- Input levels: 0.85 – 0.99
- Solution: between 0.87 and 0.88
- Number of steps: 6 instead of 15 if we check all levels.

Evaluation: Case Studies

- Two different case studies to evaluate their properties.
  - A Foreign Exchange System [4]
    - It aims to assist traders and offers them two operational modes: expert or normal.
    - It comprises internal system components and possible independent third-party components implemented as services.
    - There are different ways in which services can conduct operations similar to those of service-based systems but with different probabilities in their execution time (t_1,...,t_n), costs (c_1,...,c_n) and successes (p_1,...,p_n).
  - A three-tier server [5]
    - It provides three services: web, database and application services.
    - The services are hosted on four different physical servers (A, B, C and D) and operate on different virtual machines (VMs).
    - The system can be scaled up to include more servers, VMs and service instances.

Evaluation: Foreign Exchange System – Results

Pattern	Services	eFACT			FACT		
		P1	P2	P3	P1	P2	P3
SEQ	1	141.405	175.357	188.243	98.817	105.328	100.098
	2	147.276	194.089	194.503	T	T	T
	3	188.993	225.028	227.659	-	-	-
	4	286.416	354.003	353.514	-	-	-
SEQ-R	2	197.213	284.967	282.978	T	T	T
	3	253.189	348.103	355.679	-	-	-
	4	1507.257	1314.996	1285.241	-	-	-
SEQ-R1	2	187.587	270.305	272.994	T*	T	T
	3	222.844	322.971	322.634	-	-	-
	4	423.71	501.281	510.492	-	-	-
PAR	2	141.299	189.637	185.0	T	T	T
	3	186.318	269.309	221.306	-	-	-
	4	290.874	384.679	296.634	-	-	-
PAR-R	2	199.079	299.229	280.785	T	T	T
	3	248.545	380.596	337.698	-	-	-
	4	1487.141	1787.865	1352.024	-	-	-
PROB	2	138.287	183.473	182.499	182.595	1130.434	1025.849
	3	143.724	187.631	192.325	-	-	-
	4	148.537	197.401	200.723	-	-	-
PROB-R	2	197.09	262.869	263.637	T	T	T
	3	220.979	294.346	295.803	-	-	-
	4	238.253	339.933	334.826	-	-	-
PROB-R1	2	186.974	262.863	260.842	T	T	T
	3	216.312	293.574	294.891	-	-	-
	4	230.583	337.127	345.044	-	-	-

- (T) denotes the time out (1,800 seconds).
- (T*) means the tool has failed to produce an algebraic expression.
- (-) indicates that we skipped this experiment, as the previous model is smaller than the current one.
- FACT is better in the first row, where the model has a single service (SEQ pattern with one service) and the expression is small. However, eFACT requires more time to compute confidence intervals for the component expressions (more than one expression) before substituting their results into the model formula.
- eFACT's execution time is better than FACT in most of the cases.

Evaluation: The Three-Tier Server – Results

- FACT takes less execution time to analyse the model of deployment D1, which is found in the first row. The model is simple and produces a small expression that FACT can handle.
- In deployment D2, the model has some complexity (loop), and the expressions for both properties are too large. Therefore, FACT fails to analyse them before the time out.
- The third row is for deployment D3, which is a loop-free model. We note that FACT can analyse this model, but its analysis time is longer than eFACT.
- The last row shows the superiority of eFACT, as FACT cannot analyse the properties of this model since the algebraic expression is not produced in 1,800 seconds.

Deployment	Number of instances	Server type				eFACT		FACT	
		Server A	Server B	Server C	Server D	P1	P2	P1	P
D1	6	V	V	B	B	203.266	214.387	94.088	86.317
D2	6	VM	VM	B	B	331.342	359.419	T	T
D3	10	V	V	V	V	337.281	365.966	687.554	730.999
D4	10	VM	VM	VM	VM	824.153	873.638	T*	T*

Conclusion & Future Work

- eFACT exploits ePMC to calculate confidence intervals for non-trivial pDTMCs models with at least one unknown transition probability with observations.
- eFACT utilises a binary search to find the confidence level at which a non-functional requirement can be violated or satisfied.
- The results show that eFACT can be useful in two cases:
  - When FACT cannot compute confidence intervals due to the complexity of an algebraic expression
  - When PRISM (used by FACT to obtain an algebraic expression) cannot produce an algebraic expression.
- For future work:
  - Examine the scalability of eFACT.
  - Increase the efficiency of eFACT by analysing the component expressions generated by ePMC in parallel.

Bibliography

[1] R. Calinescu, K. Johnson, and C. Paterson, "FACT: A probabilistic model checker for formal verification with confidence intervals," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pp. 540–546, Springer, 2016.

[2] R. Calinescu, C. Paterson, and K. Johnson, "Efficient parametric model checking using domain knowledge," *IEEE Transactions on Software Engineering*, vol. 47, no. 6, pp. 1114–1133, 2019.

[3] M. Kwiatkowska, G. Norman, and D. Parker, "Prism: Probabilistic symbolic model checker," in *International Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, pp. 200–204, Springer, 2002.

[4] S. Gerasimou, G. Tamburrelli, and R. Calinescu, "Search-based synthesis of probabilistic models for quality-of-service software engineering," in *2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pp. 319–330, IEEE, 2015.

[5] R. Calinescu, S. Kikuchi, and K. Johnson, "Compositional reverification of probabilistic safety properties for large-scale complex IT systems," in *Monterey Workshop*, pp. 303–329, Springer, 2012.

THANK YOU

If you have any questions, please send me an email:
nnma500@york.ac.uk