

PROARTIS

(Probabilistically Analysable Real-Time Systems)

Introduction to PROARTIS FP7 Project: Vision and Approach

Francisco J. Cazorla (BSC)

**ARPA Workshop. 8th HiPEAC.
Berlin, January 22, 2013**

This project and the research leading to these results has received funding from the European Community's Seventh Framework Programme [FP7/2007-2013] under grant agreement n° 249100.



www.proartis-project.eu

PROARTIS consortium

- **PROARTIS FP7 Project (2010-2013)**

- Coordinator: Francisco J. Cazorla (BSC)
- Web: www.proartis-project.eu
- Partners/contributors:
 - **BSC:** Francisco J. Cazorla, Eduardo Quiñones, Leonidas Kosmidis, Jaume Abella, Gina Alioto, Emery Berger, Charlie Curtsinger
 - **U. of Padua:** Tullio Vardanega, Elisa Turrini, Enrico Mezzetti, Andrea Baldovin
 - **Rapita systems:** Guillem Bernat, Mike Houston, Ian Broster
 - **INRIA:** Liliana Cucu, Adriana Gogonel, Dorin Maxim, Code Lo
 - **Airbus:** Benoît Triquet, Franck Wartel



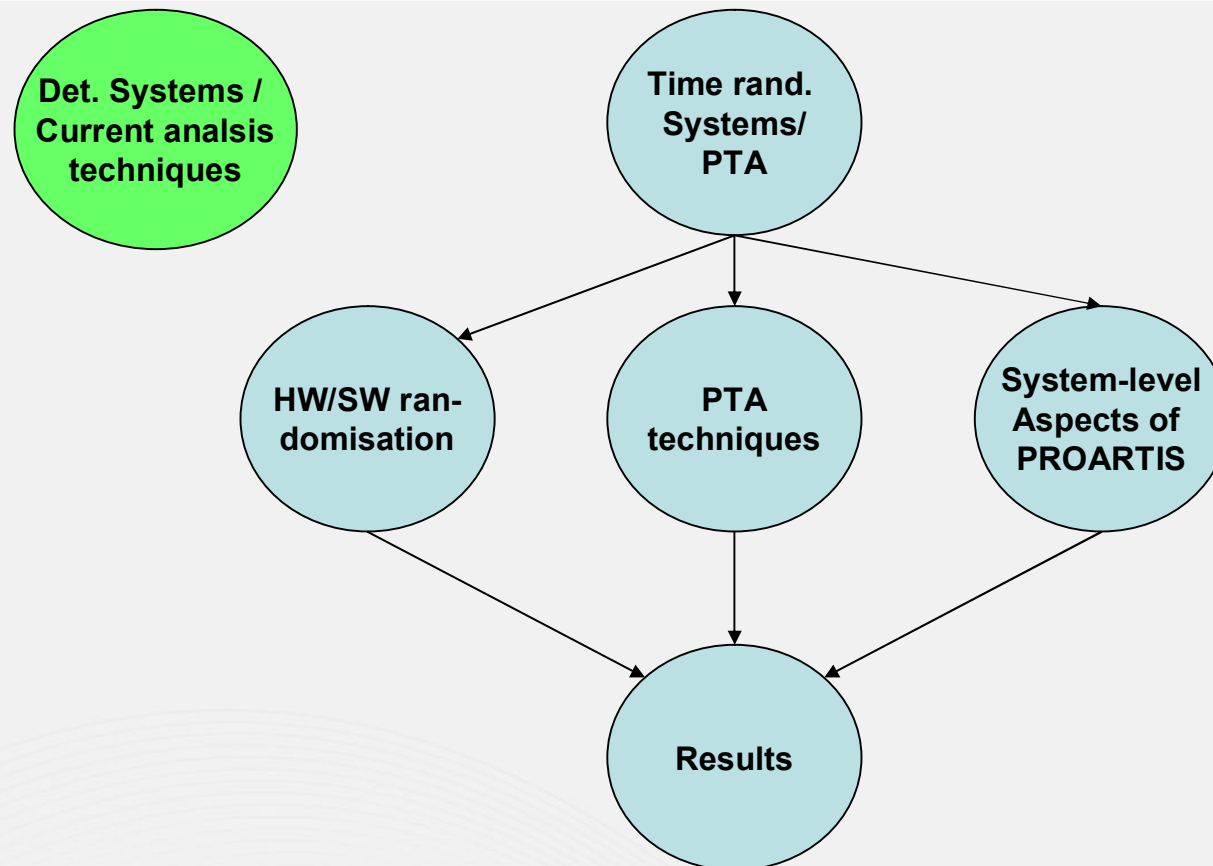
- *This work is the outcome of the effort of all the partners in the project*

Motivation

- *Critical Real-Time Embedded (CRTE) systems*
 - Used in Space, Aerospace, Transportation,... industries
- *CRTES requirements*
 - Recurring: Reduced development and production costs
 - Emerging: Increased functional value
- *More functional value → more computational power*
 - More complex SW
 - More complex HW: multicore and caches
- *Complex HW and SW affect time analysability*
- *PROARTIS view:*
 - Simple processors hard to analyse with current analysis tech.
 - Current analysis techniques cannot keep pace with novel HW

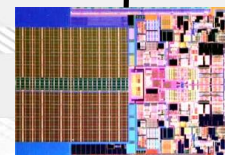
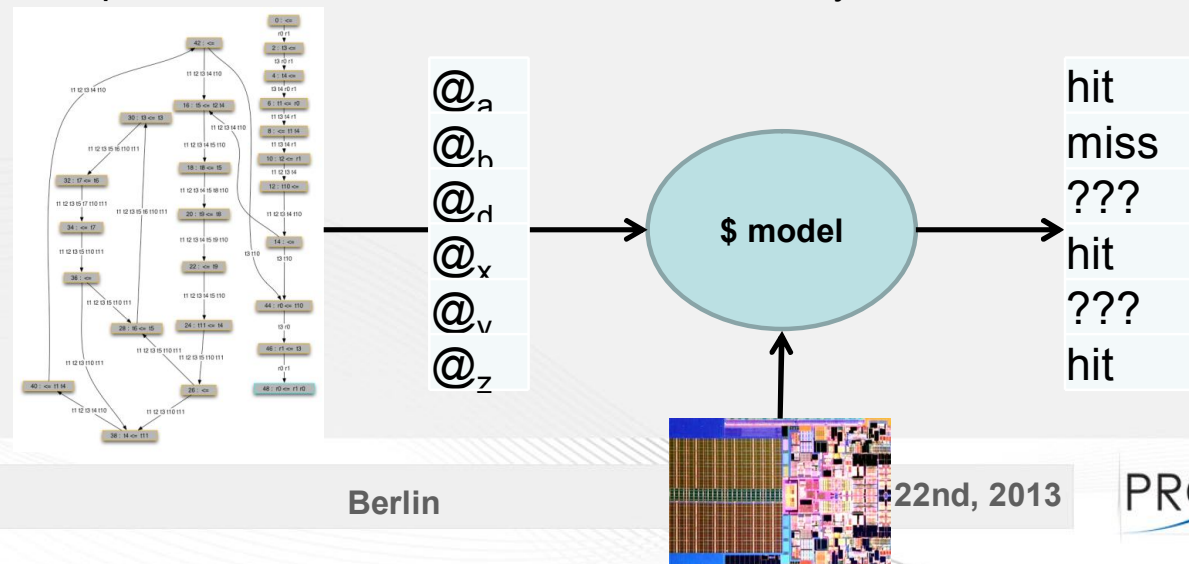
Time composability

- The timing behaviour of an individual component does not change in the face of composition with other components.
- The WCET computed for a given program unit does not change regardless of the co-runners
- Reduces time V&V
- Simplifies integration of the system
- Simplifies upgrading the system



Execution history (EH)

- *Current architectures exploit knowledge on EH*
 - To improve average-case execution time (ACET)
 - Caches: Temporal and spatial locality
- ***Current analysis techniques require info about EH***
 - The ACET and WCET of programs heavily **depend on EH**
 - E.g: Cache analysis: To determine whether a given access is a hit:
 - History of all access to the cache
 - Representation of the cache state after every access



WCET and dependence on EH

- ***Limitations & increasing effort in acquiring EH info***
 - Complex processor architectures (IP protected)
 - Incomplete and/or inaccurate documentation
 - Program information may be unknown at analysis time
- *Reduction of available knowledge about HW/SW → pessimistic assumptions → degradation of the tightness of the WCET*

WCET and dependence on EH

- ***Limitations & increasing effort in acquiring EH info***

- Complex processor architectures (IP protected)
- Incomplete and/or inaccurate documentation
- Program information may be unknown at analysis time

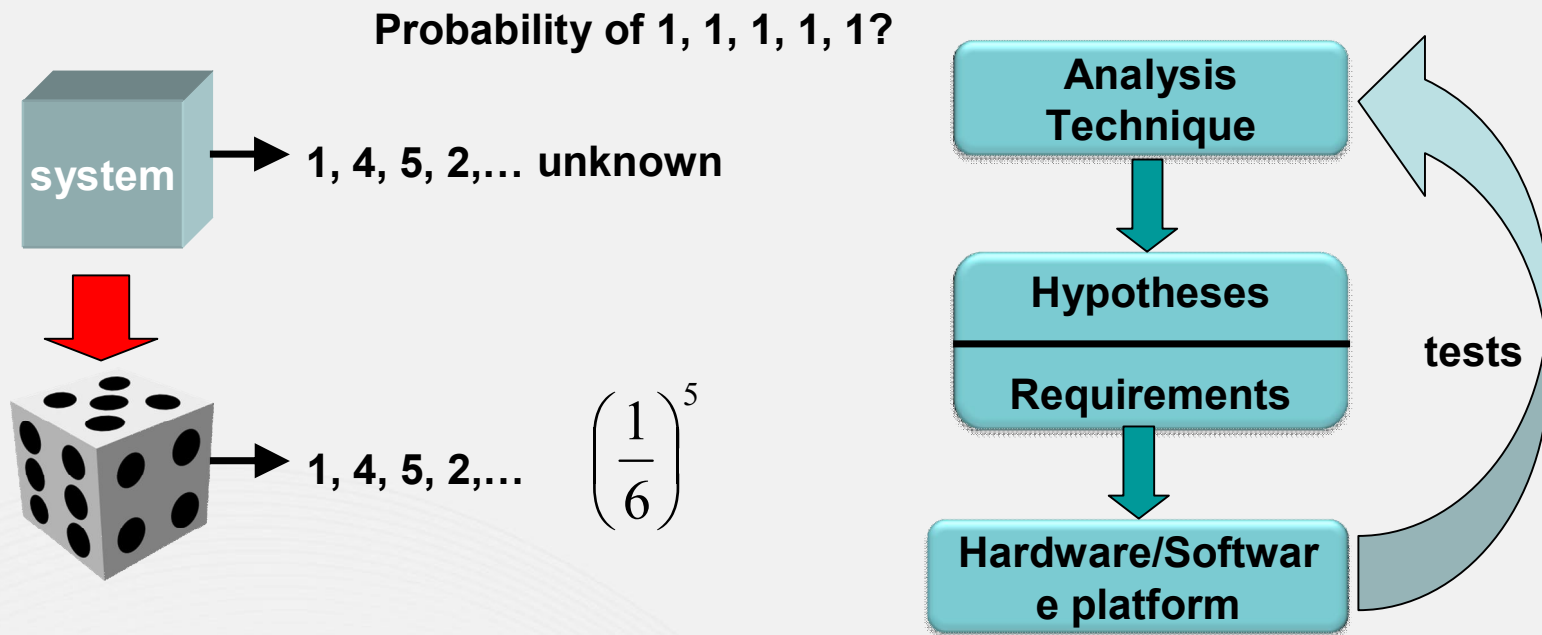
- **Design HW and SW whose execution time behaviour does not depend on execution history**

- Without removing performance-improving hw!
- The functional behaviour is left unchanged

- **Provide new timing analysis techniques**

Probabilistically Analysable Real-Time Systems

- Probability of appearance of an event *is not equal to its* frequency of appearance within a finite time interval



Probabilistic nature of the system

- pWCET estimations for high-integrity systems
 - Counter intuitive
- Probabilistic modeling is in close match with actual nature of the system
 - Mechanical parts → designed with a probability of failure in mind
 - Hardware → affected by radiation, temperature, ...
Probability of failure
 - Failure rate for airborne applications → 10^{-9} per hour of operation
 - ASIL levles in automotive: < 1 failure per 10^X hours of operation
- The system as a whole has a distinct **prob. of failure**
- PTA: upperbounds the execution time of programs with an attached probability of exceedance
 - Time Failures can be considered just another type of failure

Probabilistic nature of the system

- pWCET estimations for high-integrity systems
 - Counter intuitive
- Probabilistic modeling is in close match with actual nature

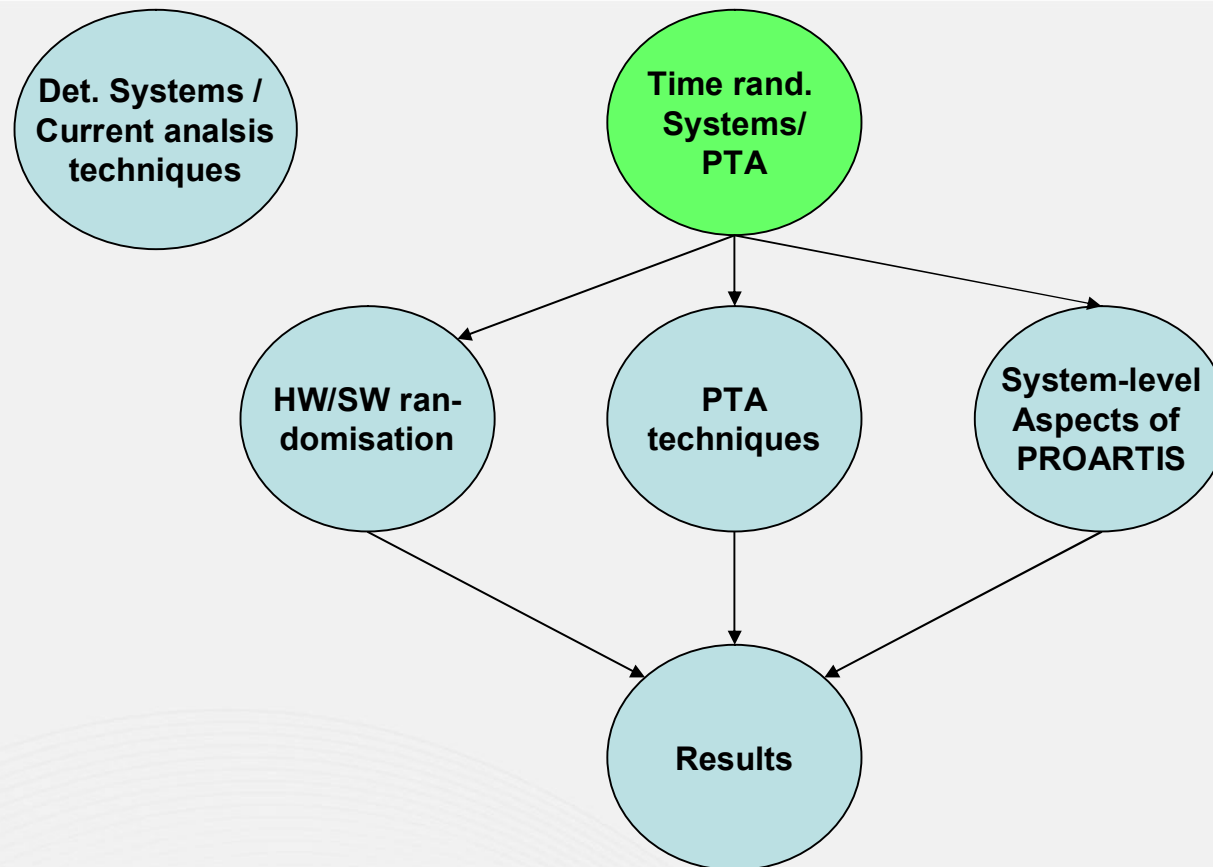
F. J. Cazorla, E. Quinones, T. Vardanega, L. Cucu, B. Triquet, G. Bernat, E. Berger, J. Abella, F. Wartel, M. Houston, L. Santinelli, L. Kosmidis, C. Lo, D. Maxim.

PROARTIS: Probabilistically Analysable Real-Time Systems.

In ACM Transactions on Embedded Computing Systems. Special issue on Probabilistic Computing.

WWW.PROARTIS-PROJECT.EU

- PTA. upperbounds the execution time of programs with an attached probability of exceedance
 - Time Failures can be considered just another type of failure

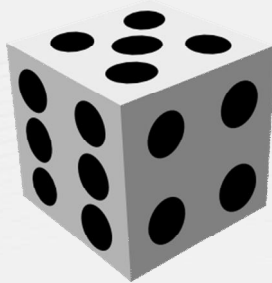


PROARTIS Approach

- *Reduce dependence on execution history*
 - HW and SW whose execution time behaviour does not depend on execution history
 - While benefiting performance-improving hardware!
- *PROARTIS*
 - Introduces **randomisation** into the **timing behaviour** of the hardware and software
 - The functional behaviour is left unchanged
 - Provides new **probabilistic timing analysis** techniques

Example

- *Let's assume a processor with a single level of cache.*
 - Core operations take a fix latency (2)
 - Cache ops. take either a hit latency (3) or a miss latency (100)
 - Hit probability of 0.8
- *Let's assume a program with the following sequence of instructions*
 - What is the probabilistic ET of this program?
 - What is the probability of the WCET?



1	Add
2	Load
3	Add
4	Load
5	Store
6	Add
7	Load
8	Add
9	Add
10	Store
11	Load
12	store

Example: Execution Time Profiles

- *ETP Core* {2}{1.0}
- *ETP Mem* {3,100} {0.8, 0.2}

Add	(2)(1.0)	Add	(2)(1.0)	(2)	(1.0)
Load	(3,100)(0.8, 0.2)	Add	(2)(1.0)		
Add	(2)(1.0)	Add	(2)(1.0)		
Load	(3,100)(0.8, 0.2)	Add	(2)(1.0)		
Store	(3,100)(0.8, 0.2)	Add	(2)(1.0)		
Add	(2)(1.0)	Load	(3,100)(0.8, 0.2)		
Load	(3,100)(0.8, 0.2)	Load	(3,100)(0.8, 0.2)		
Add	(2)(1.0)	Store	(3,100)(0.8, 0.2)		
Add	(2)(1.0)	Load	(3,100)(0.8, 0.2)		
Store	(3,100)(0.8, 0.2)	Store	(3,100)(0.8, 0.2)		
Load	(3,100)(0.8, 0.2)	Load	(3,100)(0.8, 0.2)		
store	(3,100)(0.8, 0.2)	store	(3,100)(0.8, 0.2)		

Convolution

$$\{(10)(1)\} \Theta \{(3,100)(0.8, 0.2)\} = \{ (10+3, 10+100) (1*0.8, 1*0.2) \}$$

$$\{(t1,t2)(p1,p2)\} \Theta \{(t3,t4)(p3,p4)\} = \{ (t1+t2, t1+t3, t2+t3,t2+t4) (p1xp2, p1xp3, p2xp3,p2xp4) \}$$

Example: Execution Time Profiles

- *ETP Core {2}{1.0}*
- *ETP Mem {3,100} {0.8, 0.2}*

Add	(2)(1.0)	Add	(2)(1.0)	(2)	(1.0)
Load	(3,100)(0.8, 0.2)	Add	(2)(1.0)	(4)	(1.0)
Add	(2)(1.0)	Add	(2)(1.0)	(6)	(1.0)
Load	(3,100)(0.8, 0.2)	Add	(2)(1.0)	(8)	(1.0)
Store	(3,100)(0.8, 0.2)	Add	(2)(1.0)	(10)	(1.0)
Add	(2)(1.0)	Load	(3,100)(0.8, 0.2)	(13, 110)	(0.8, 0.2)
Load	(3,100)(0.8, 0.2)	Load	(3,100)(0.8, 0.2)	(16, 113, 210)	(0.64, 0.32, 0.04)
Add	(2)(1.0)	Store	(3,100)(0.8, 0.2)	(19, 116, 213, 310)	(0.512, 0.384, 0.096, 0.08)
Add	(2)(1.0)	Load	(3,100)(0.8, 0.2)		
Store	(3,100)(0.8, 0.2)	Store	(3,100)(0.8, 0.2)		
Load	(3,100)(0.8, 0.2)	Load	(3,100)(0.8, 0.2)		
store	(3,100)(0.8, 0.2)	store	(3,100)(0.8, 0.2)		

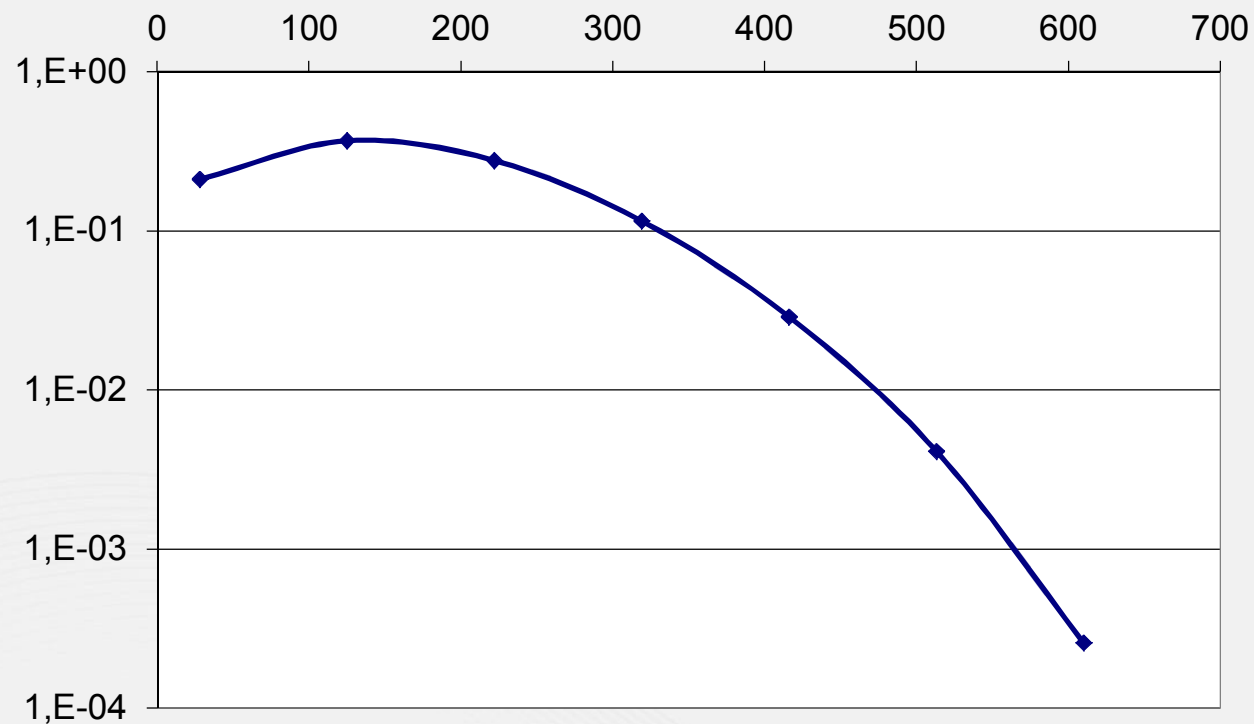
Convolution

$$\{(10)(1)\} \Theta \{(3,100)(0.8, 0.2)\} = \{ (10+3, 10+100) (1*0.8, 1*0.2) \}$$

$$\{(t_1, t_2)(p_1, p_2)\} \Theta \{(t_3, t_4)(p_3, p_4)\} = \{ (t_1+t_2, t_1+t_3, t_2+t_3, t_2+t_4) (p_1 \times p_2, p_1 \times p_3, p_2 \times p_3, p_2 \times p_4) \}$$

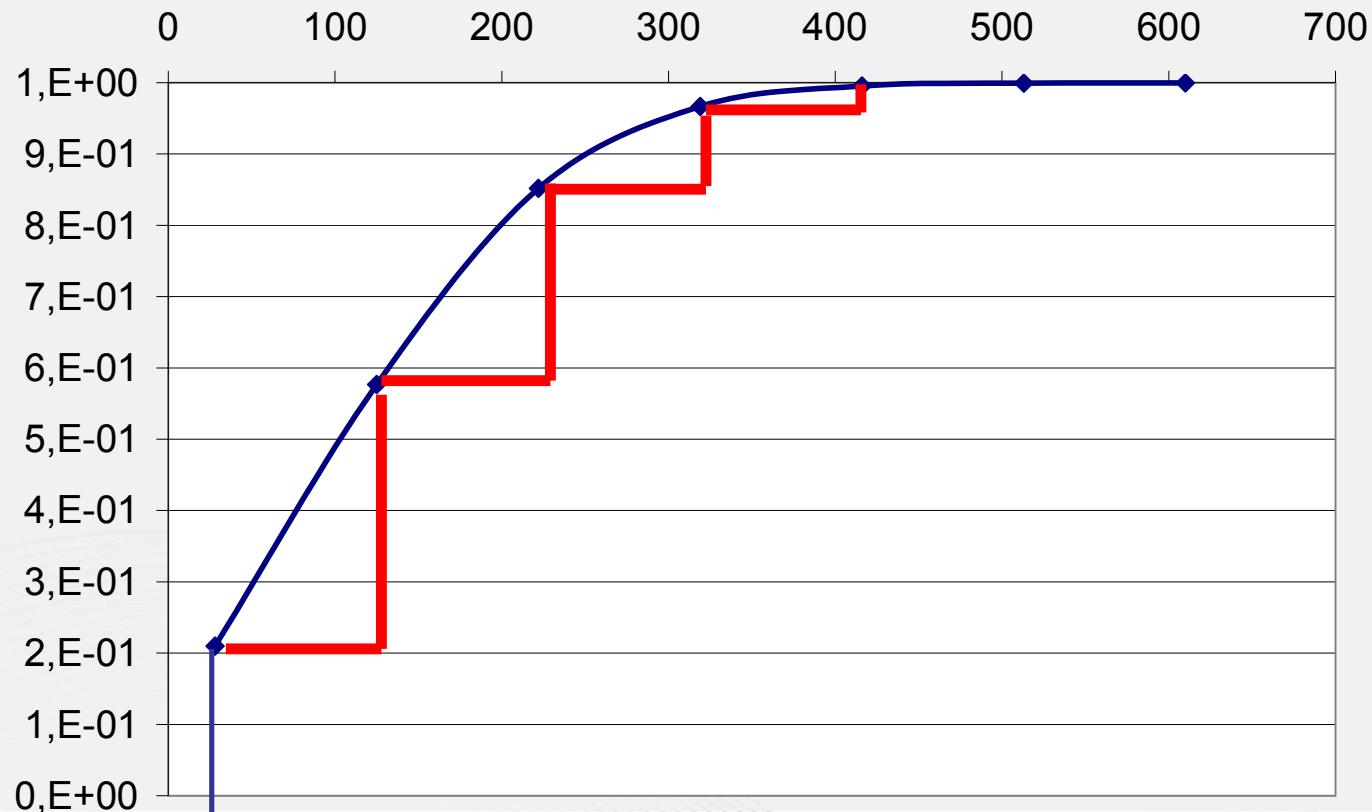
Probability of each execution time

Probability of the WCET = probability all mem
operations missing = $2,6^{-4}$



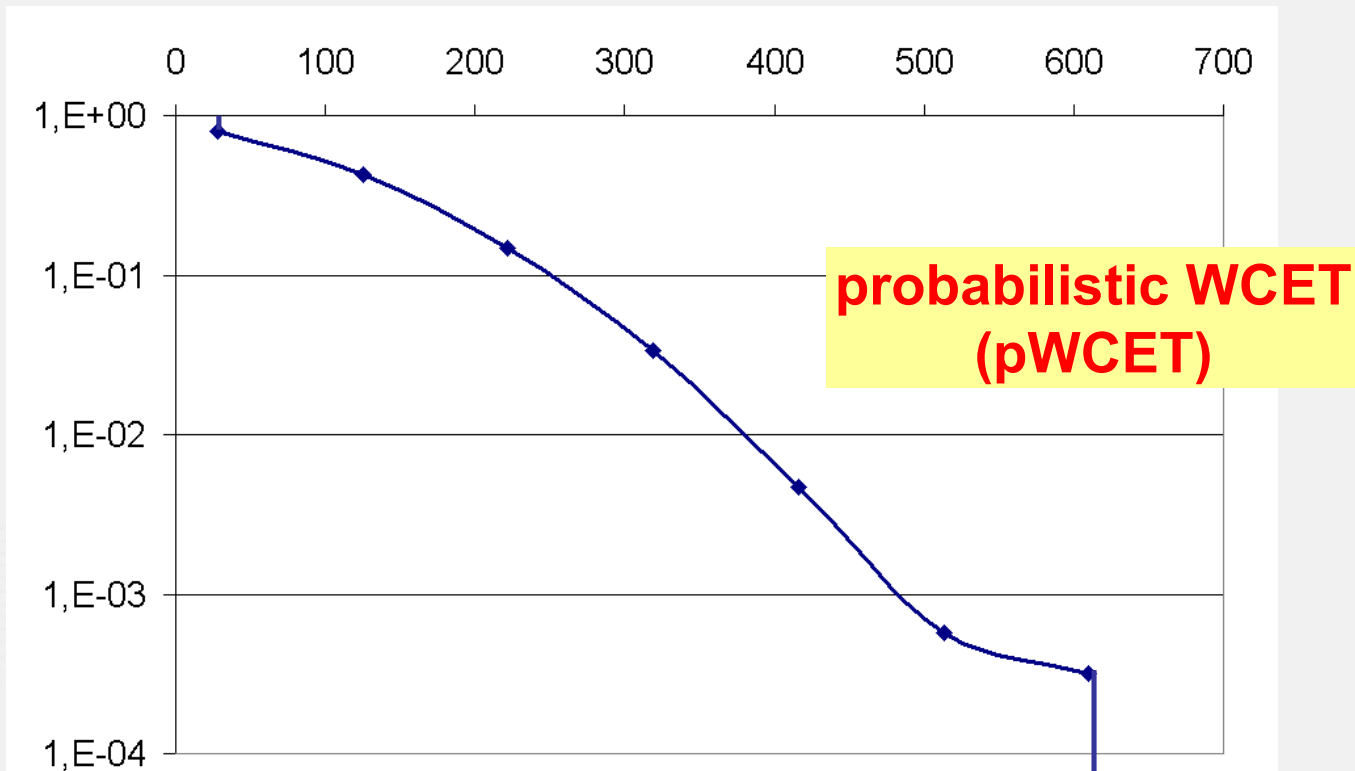
Cumulative probability

- *Probability that the program takes less than (or equal to) a given execution time*



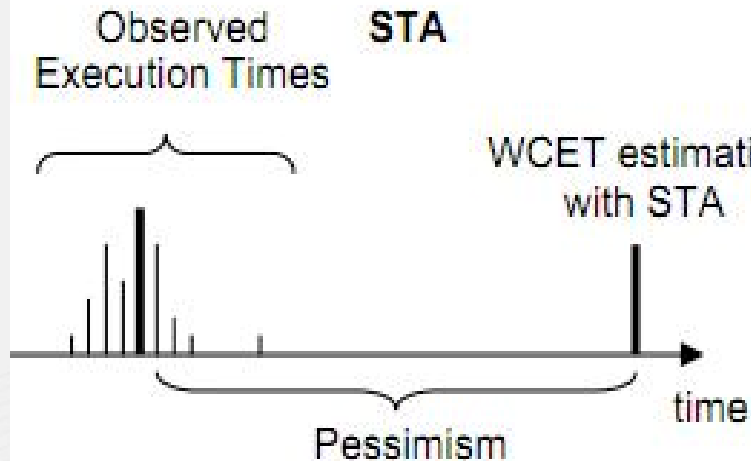
Exceedance probability

- *Probability that your program takes longer than a given threshold (execution time)*

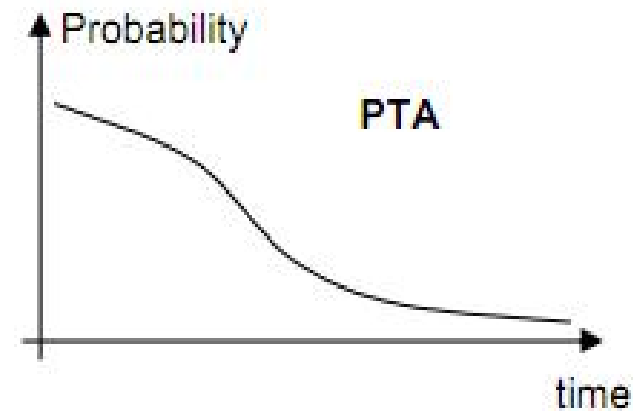


Probabilistic Timing Analysis

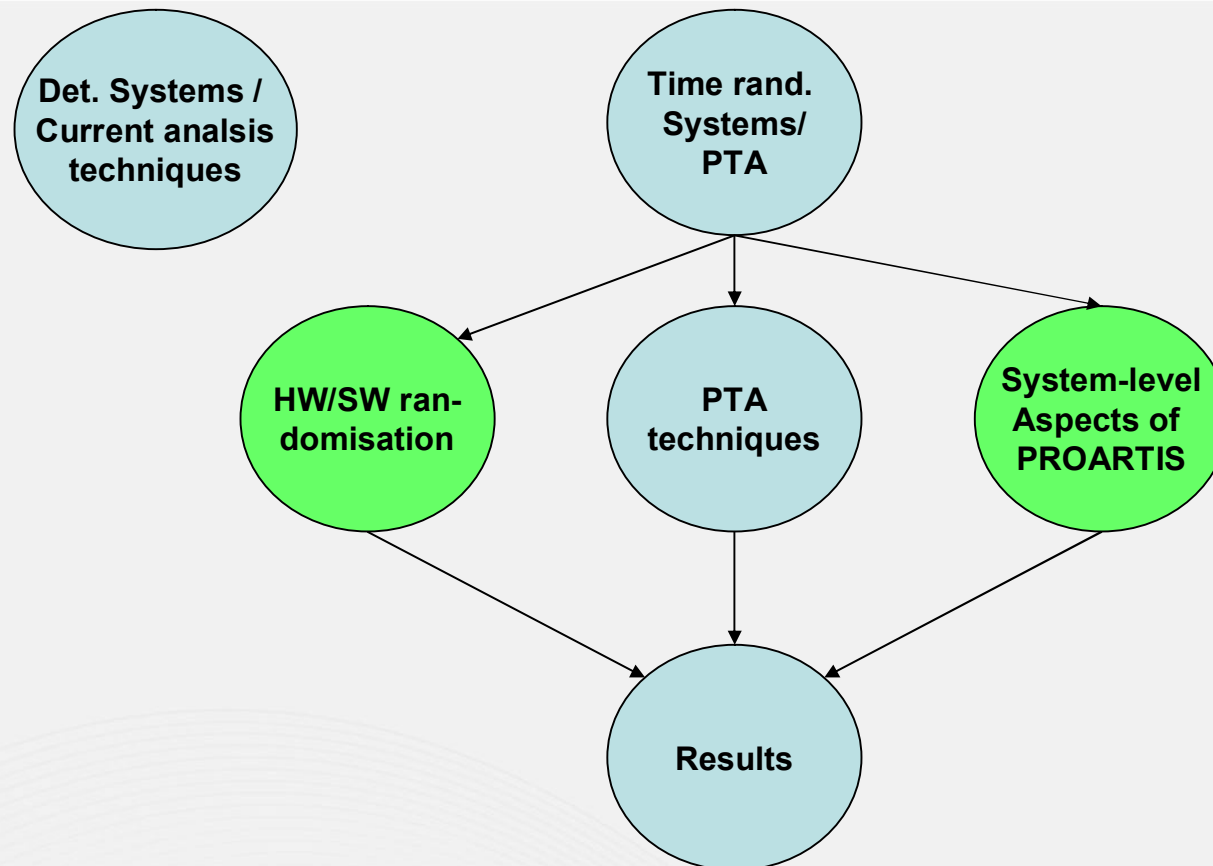
- *PTA allows cutting the WCET bound tail at the level of probability suited for the system (e.g. 10^{-16} per hour of operation)*
 - Prob. Failure per hour = probability of failure of the program x execution rate per hour



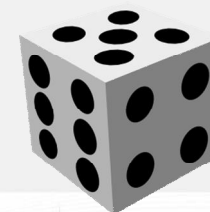
(a)



(b)



- *How to architect HW/SW to behave as...*



PROARTIS

Design objectives

- *DO1) Timing behavior at the granularity of instruction by construction must have*
 - Either no dependence on execution history at all
 - Or probabilistically characterisable dependence
- *DO1 achieved by dividing processor hardware resources into three groups based on their jitter*
 - No jitter
 - Response time jitter can be upper bounded without incurring excessive pessimism
 - Response time jitter is randomised so it can be characterised probabilistically

(jitter = variability in the response time)

Design objectives

- *DO2) Time behavior of software at the granularity of program units (e.g, procedures) must be time composable*
 - The bound determined for each procedure must stay valid in the face of composition
 - Allows the program units at application level to be submitted to PTA incrementally and in isolation

Program unit: Granularity at which we apply PTA.
Input for schedulability analysability

Design objectives

- *Fulfilling DO2*
 - DO1 → processor state retains no dependence on EH
 - The variability of the timing behaviour of the application software can only residually stem from either
 - (Part1) application logic, the execution paths that are traversed, or
 - (Part2) the perturbation effects caused by the Operating System
- *Part 1 of the problem addressed by PTA itself*
- *Part 2 solved by designing a time-composable OS¹*
 - The OS services have a tightly upperbounded response time
 - Their execution does not cause perturbation on the processor state on return from the call

¹ Baldovin, A., E. Mezzetti, And T. Vardanega

A Time-composable Operating System

12th International Workshop on Worst-Case Execution-Time Analysis, Pisa, Italy, 07/2012

Design objectives

- *Fulfilling DO2*

- DO1 → processor state retains no dependence on EH
- The variability of the timing behaviour of the application software can only residually stem from either
 - (Part1) application logic, the execution paths that are traversed, or

Sources of program execution time variability →
bounded from above or randomised

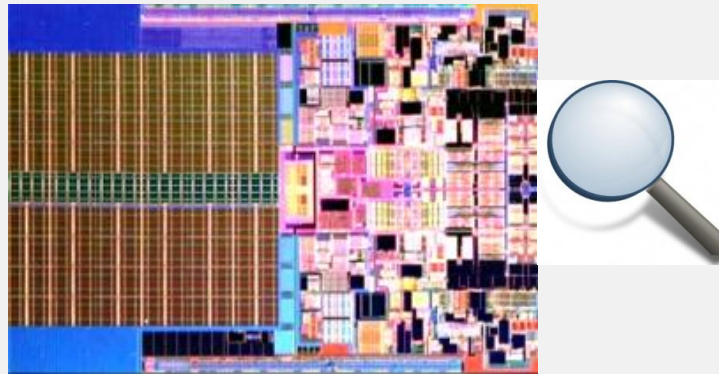
- The OS services have a tightly upperbounded response time
- Their execution does not cause perturbation on the processor state on return from the call

¹ Baldovin, A., E. Mezzetti, And T. Vardanega

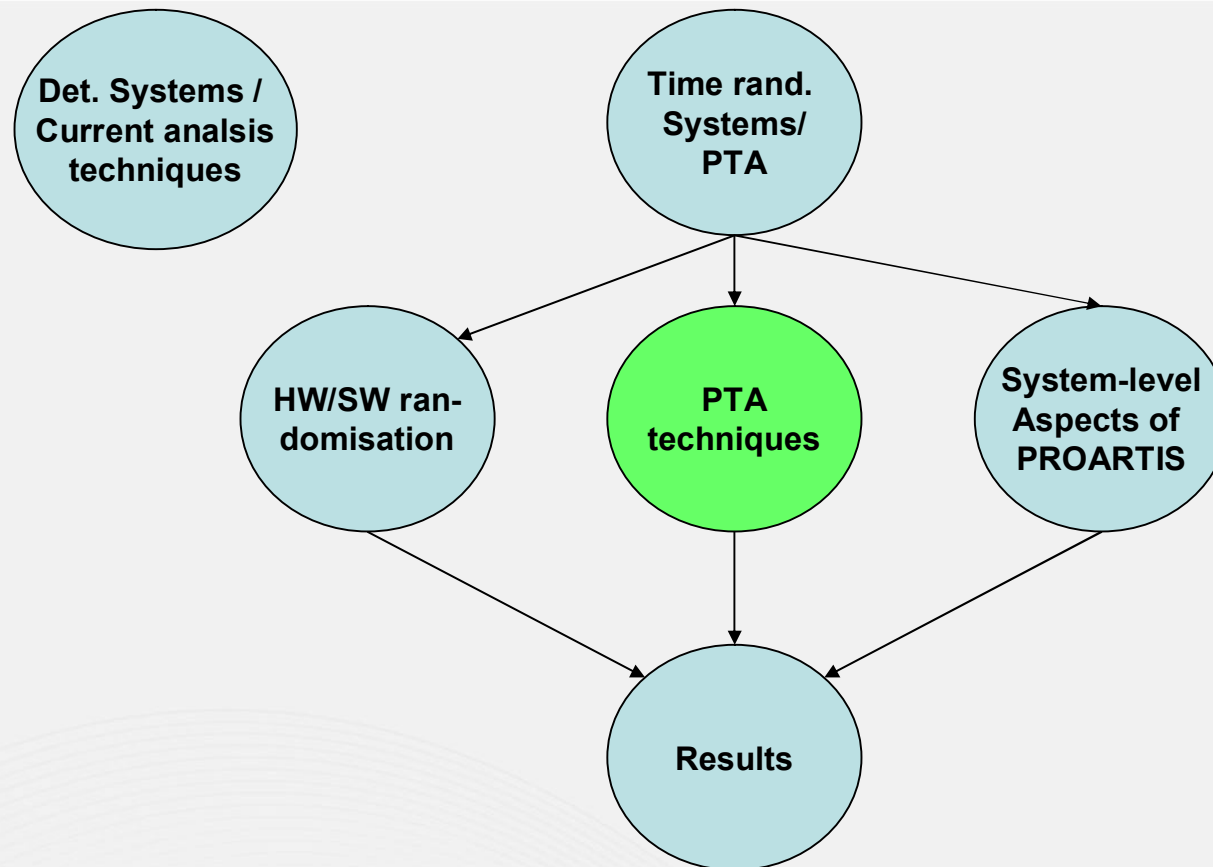
A Time-composable Operating System

12th International Workshop on Worst-Case Execution-Time Analysis, Pisa, Italy, 07/2012

Deterministic → Probabilistic



IF ALL DESIGN OBJECTIVES ACHIEVED:
Prob. analysis can ignore the processor internals!
It is sufficient that chip manufactures ensure certain properties hold instead of exposing internal operation information

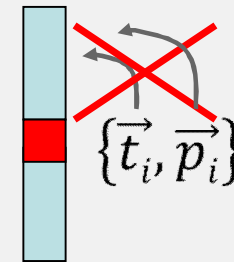


PTA Requeriments /1

- *PTA allows modeling ET of components of interest at a given level of execution granularity (e.g. basic block).*
 - The response time of each component at that granularity must have a distinct probability of occurrence
 - $ETP(C_i) = \{\vec{t}_i, \vec{p}_i\} = \left\{ \{t_i^1, t_i^2, \dots, t_i^{N_i}\}, \{p_i^1, p_i^2, \dots, p_i^{N_i}\} \right\}$
 - $t_i \rightarrow$ all possible execution times of the component C_i
 - $p_i \rightarrow$ corresponding probabilities of occurrence with $\sum_{j=1}^{N_i} p_j = 1$

PTA Requirements/2

- *Level of granularity of the timing events of interest is different for SPTA and MBPTA*
 - Static Probabilistic Timing Analysis
 - Measurement-Based Probabilistic Timing Analysis
- *SPTA/MBPTA require timing events to be modeled with i.i.d random variables*
 - SPTA → instruction level
 - MBPTA → execution times of end-to-end paths
- *Timing events that ETP describes are i.i.d if*
 - t_i does not vary with the history of previous execution
 - p_i does not vary with the history of previous execution



SPTA

- Determines **execution time distributions** for individual operations with their associated **probabilities**
- **Convolution** is used to generate the execution time distribution of the program
- Requires i.i.d. to hold for all levels in which ETP are to be built

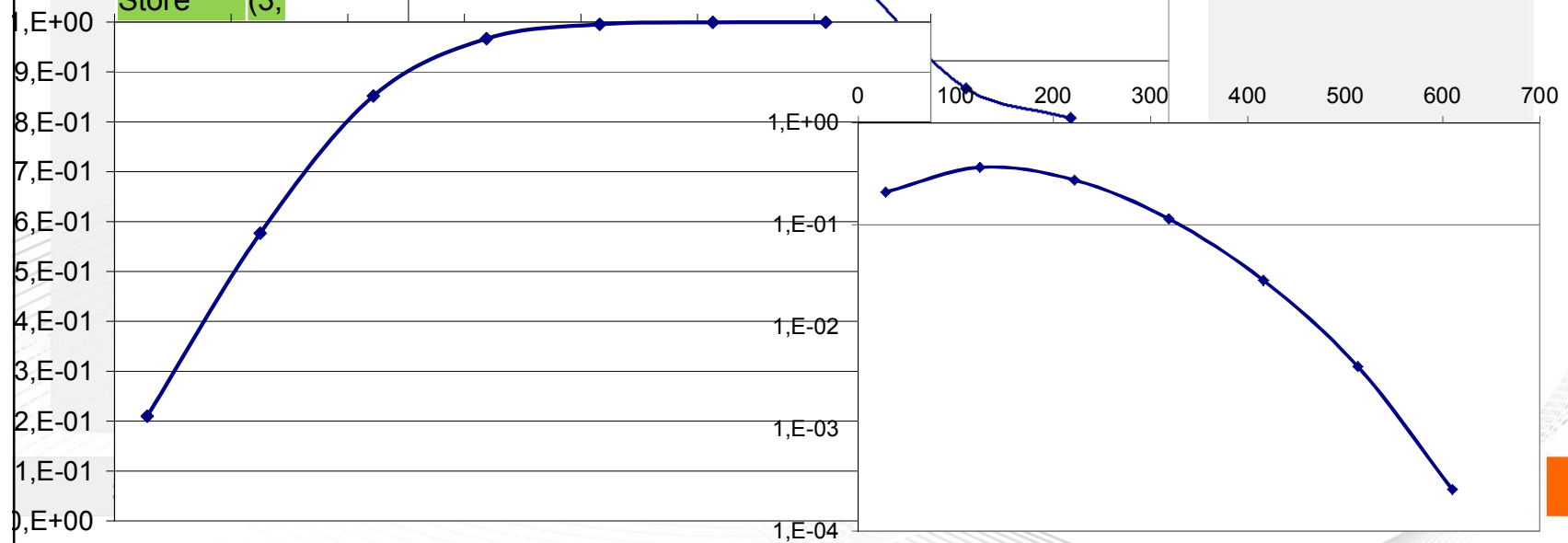
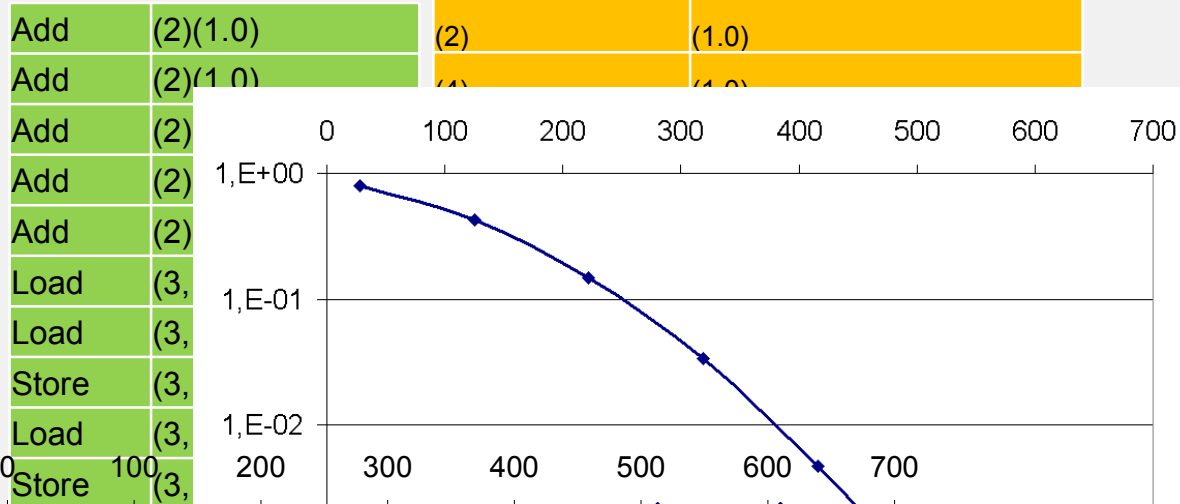


F.J. Cazorla, E. Quinones, T. Vardanega, L. Cucu, B. Triquet, G. Bernat, E. Berger, J. Abella, F. Wartel, M. Houston, L. Santinelli, L. Kosmidis, C. Lo, D. Maxim.

PROARTIS: Probabilistically Analysable Real-Time Systems.

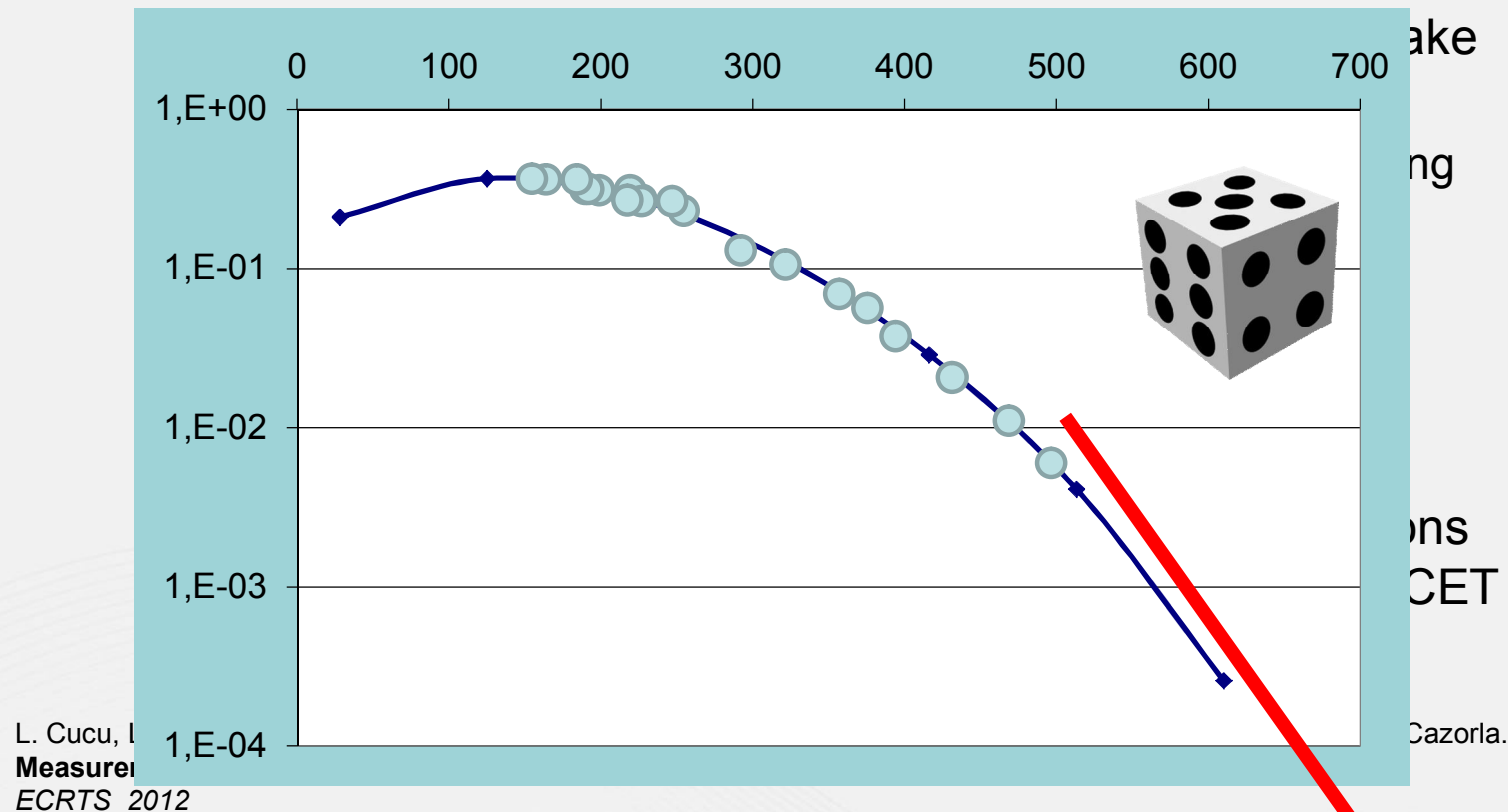
In *ACM Transactions on Embedded Computing Systems. Special issue on Probabilistic Computing. Summer 2012.*

SPTA



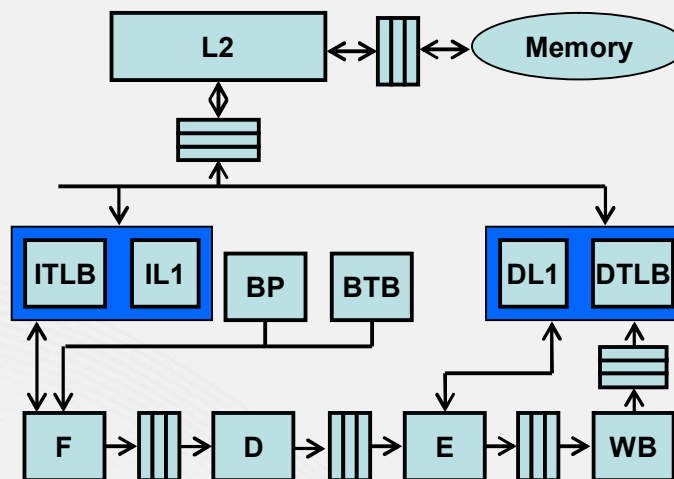
Measurement-based PTA

- *Complete runs of the program are made on the time-randomised target platform*

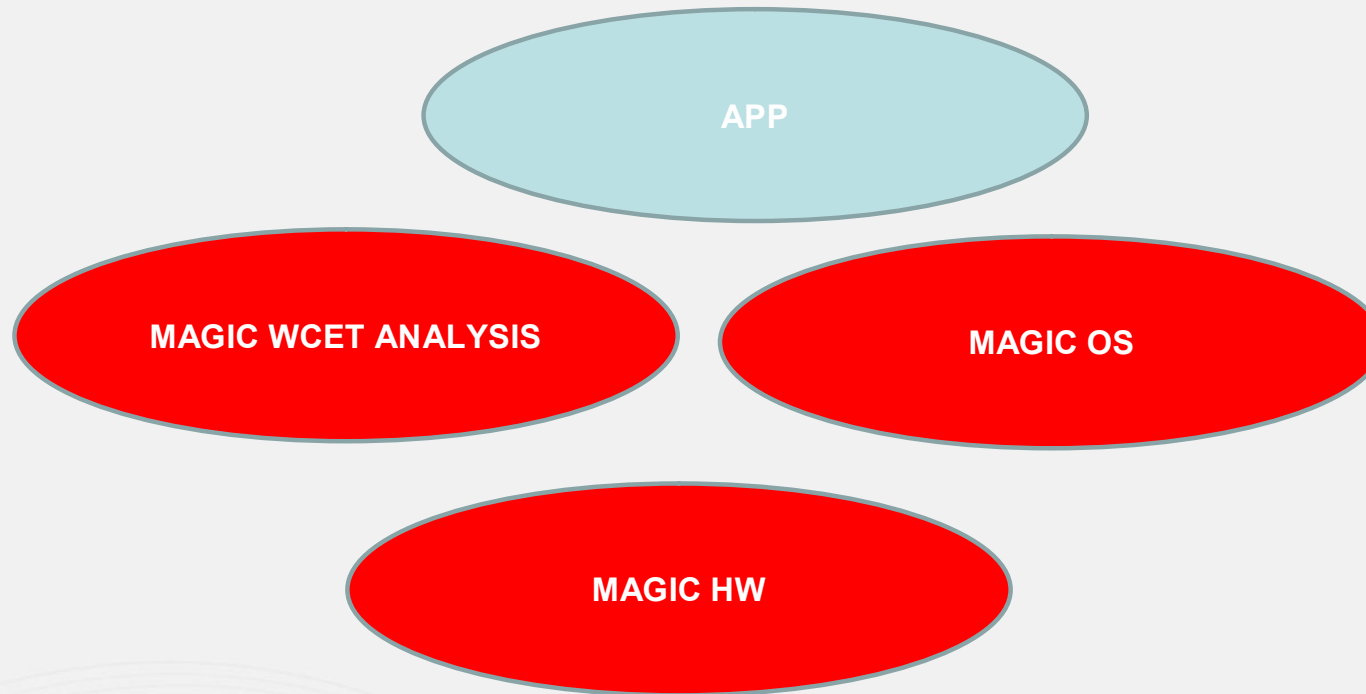


PROARTIS Processor Design

- *Deterministic and probabilistic processor resources can coexist in a PTA-friendly processor*
 - Multiple levels of random cache design, buffers, branch predictors
- *Design below is 90% compatible with Leon3/4 Designs*
 - **Except they do not implement a random placement policy**

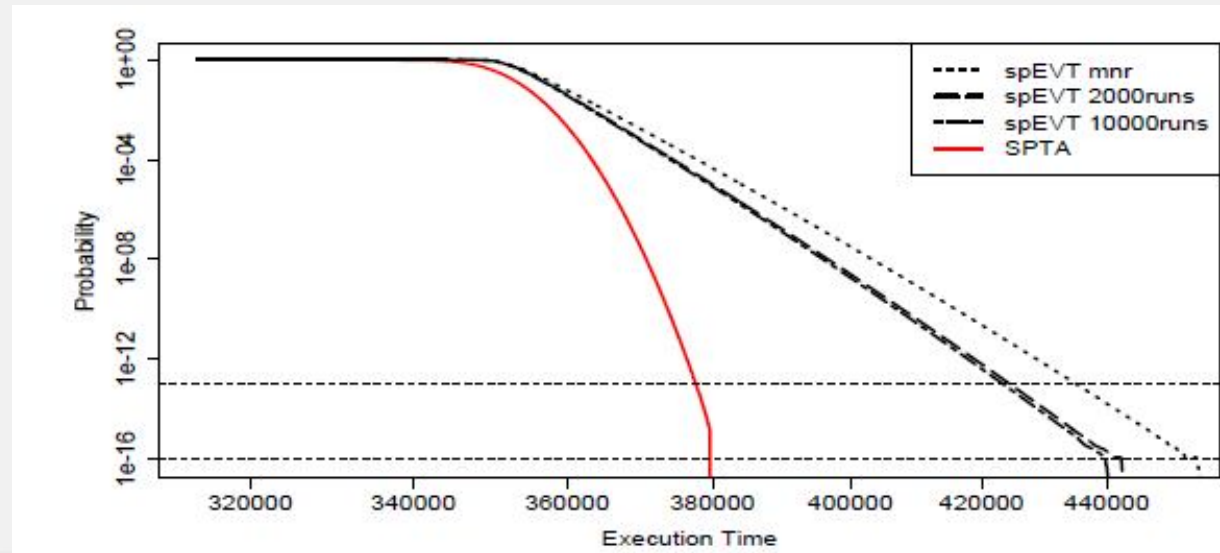


Magic



MBPTA: results on EEMBC

- *MBPTA projections are tight to those provided by SPTA*
 - Recall SPTA is a safe and very tight probabilistic projection



- *Average pWCET increment w.r.t SPTA for all EEMBC:*

probability	AI	AT	CA	CN	PU	RS	TB	TT
10^{-13}	9%	5%	6%	5%	5%	2%	2%	6%
10^{-16}	15%	7%	8%	7%	7%	3%	3%	9%

PROARTIS

(Probabilistically Analysable Real-Time Systems)

Introduction to PROARTIS FP7 Project: Vision and Approach

Francisco J. Cazorla (BSC)

**ARPA Workshop. 8th HiPEAC.
Berlin, January 22, 2013**

This project and the research leading to these results has received funding from the European Community's Seventh Framework Programme [FP7/2007-2013] under grant agreement n° 249100.



www.proartis-project.eu