ACTOR: Accelerating Fault Injection Campaigns using Timeout Detection based on Autocorrelation

Tim-Marek Thomas[1], Christian Dietrich[2], Oskar Pusz[1], Daniel Lohmann[1]

[1]Leibniz Universität Hannover, [2]Technische Universität Hamburg

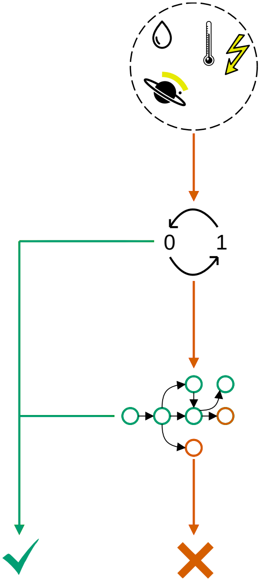
08.09.2022

Motivation – FI Campaigns

Why?

- Transient hardware faults
- Test reliability
- Be as precise and complete as possible

→ (systematic) Fault Injection Campaign

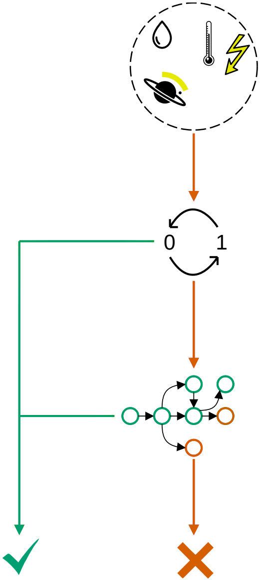


Motivation - FI Campaigns

Why?

- Transient hardware faults
- Test reliability
- Be as precise and complete as possible

→ (systematic) Fault Injection Campaign



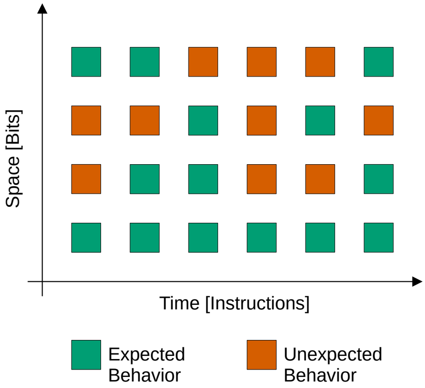
Motivation - FI Campaigns

Problem: time

- Covering whole fault space is expensive
- Depending on layer even more (here: ISA)

Solutions:

- Pruning fault space e.g. def-use based
- Accelerate every single experiment



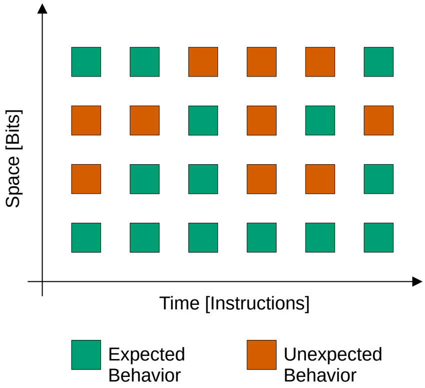
Motivation – FI Campaigns

Problem: time

- Covering whole fault space is expensive
- Depending on layer even more (here: ISA)

Solutions:

- Pruning fault space e.g. def-use based
- Accelerate every single experiment



Motivation - Timeouts

- Different kinds of results per experiment:
  - SDC, Benign, Traps, ..., Timeouts
- Timeouts:
  - Low amount of experiments: up to 27.05 %
  - High amount of simulated Instructions: up to 61.67 %
- Goal:
  - Minimize effect on campaign quality → low FP-rate
  - Maximize campaign time savings → high TP-rate

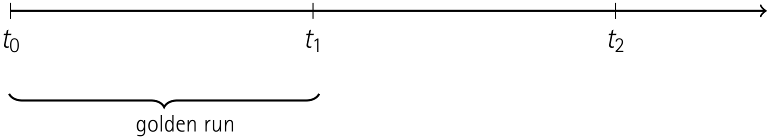
→ Lets take a look!

Motivation - Timeouts

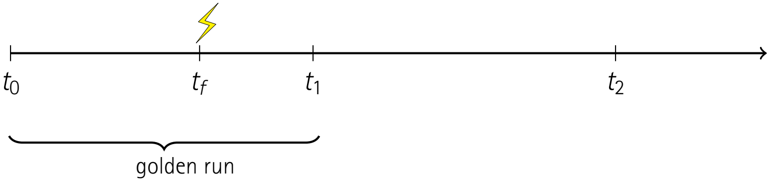
- Different kinds of results per experiment:
  - SDC, Benign, Traps, ..., Timeouts
- Timeouts:
  - Low amount of experiments: up to 27.05 %
  - High amount of simulated Instructions: up to 61.67 %
- Goal:
  - Minimize effect on campaign quality $\rightarrow$ low FP-rate
  - Maximize campaign time savings $\rightarrow$ high TP-rate

$\rightarrow$ Lets take a look!

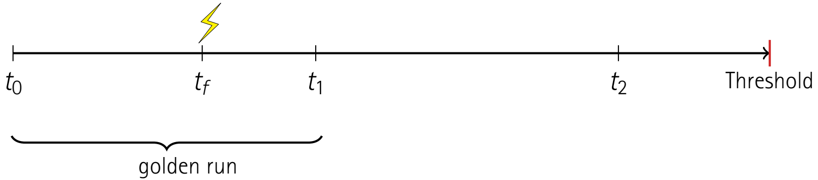
General Musings



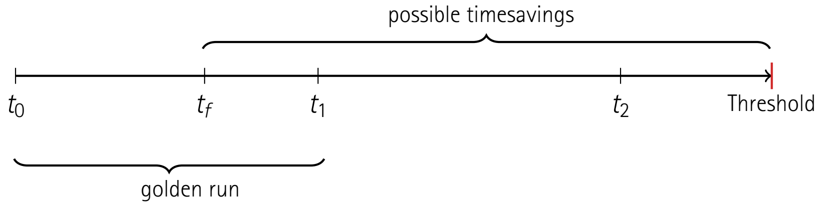
General Musings



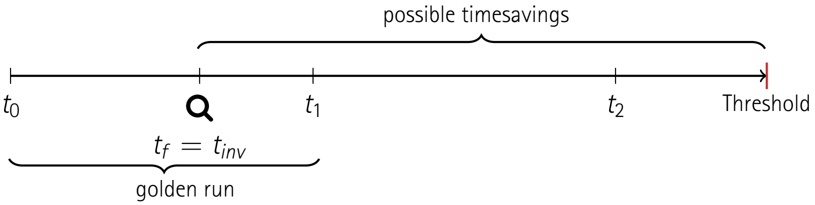
General Musings



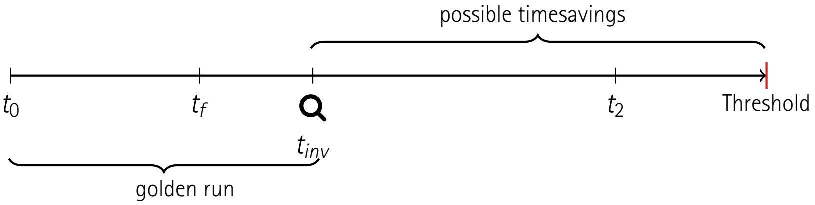
General Musings



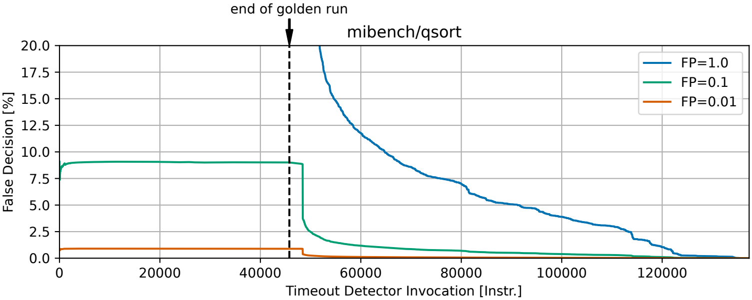
General Musings



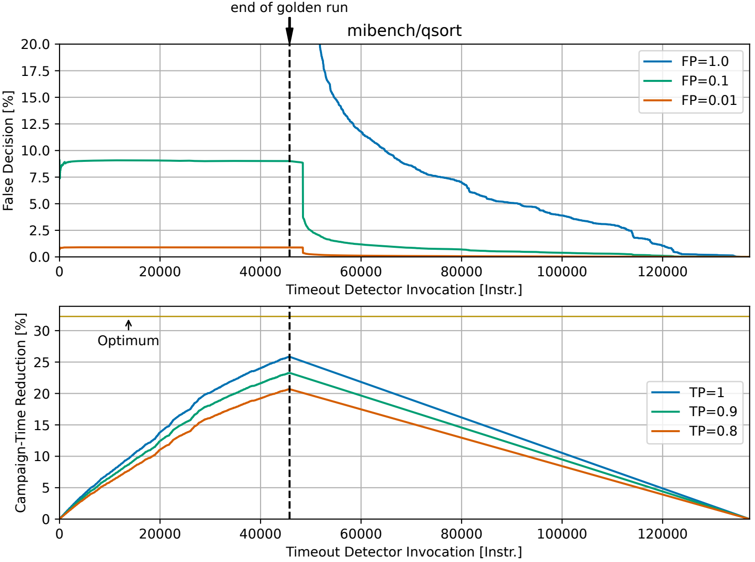
General Musings



General Musings - QSort



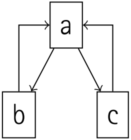
General Musings - QSort



Autocorrelation as Timeout Detector

Idea from Ibing et.al.[2]:

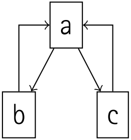
- Interpret branch history as discrete signal
- Autocorrelation as indicator for periodicity
- High value → High probability of infinite loop

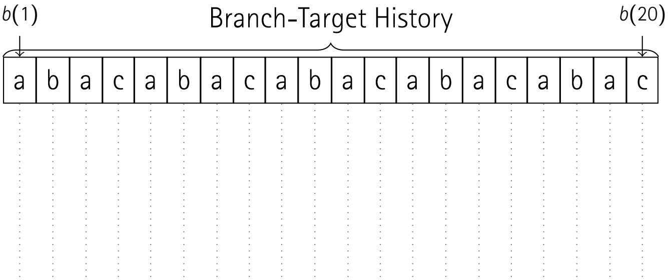


Autocorrelation as Timeout Detector

Idea from Ibing et.al.[2]:

- Interpret branch history as discrete signal
- Autocorrelation as indicator for periodicity
- High value → High probability of infinite loop

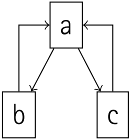


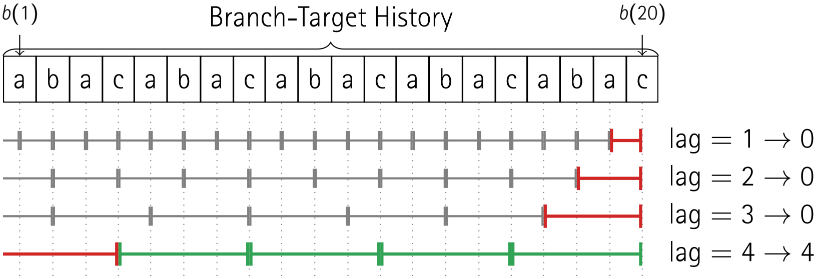


Autocorrelation as Timeout Detector

Idea from Ibing et.al.[2]:

- Interpret branch history as discrete signal
- Autocorrelation as indicator for periodicity
- High value → High probability of infinite loop





Autocorrelation as Timeout Detector

Adaption to FI-Campaigns:

1. Runtime is problematic (slowdown factors of 100x to 225x)
   - Collection of branches
   - Amount of invocations

2. Choose parameters:
   - Invocation time point
   - History length m
   - Maximum lag l_{max}
   - Threshold T

Autocorrelation as Timeout Detector

Adaption to FI-Campaigns:

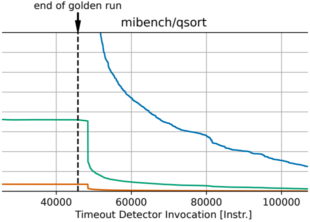
1. Runtime is problematic (slowdown factors of 100x to 225x)
   - Collection of branches
   - Amount of invocations

2. Choose parameters:
   - Invocation time point
   - History length m
   - Maximum lag l_{max}
   - Threshold T

Autocorrelation as Timeout Detector

Derive parameters from the golden run:

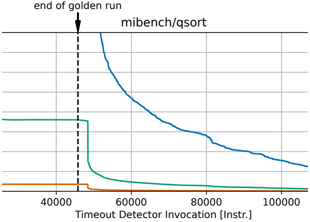
- Ivocation roughly at $1.2 t_1 \rightarrow$ when m is filled



Autocorrelation as Timeout Detector

Derive parameters from the golden run:

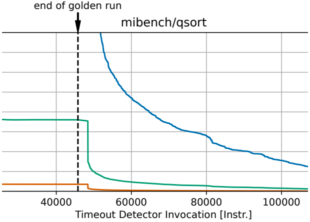
- Ivocation roughly at $1.2 t_1 \rightarrow$ when m is filled
- Branch density: $m = \frac{\#\text{Instructions}}{\#\text{Branches}} * 0.2$



Autocorrelation as Timeout Detector

Derive parameters from the golden run:

- Ivocation roughly at $1.2t_1 \rightarrow$ when m is filled
- Branch density: $m = \frac{\#\text{Instructions}}{\#\text{Branches}} * 0.2$
- $l_{max} = 16$, for detecting tight loops

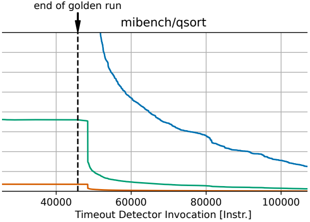


Autocorrelation as Timeout Detector

Derive parameters from the golden run:

- Ivocation roughly at $1.2t_1 \rightarrow$ when m is filled
- Branch density: $m = \frac{\#\text{Instructions}}{\#\text{Branches}} * 0.2$
- $l_{max} = 16$, for detecting tight loops
- Lag specific threshold T_l:

$$T_l = 1 + \max_{s \in [0,(|H|-m)]} R_{bb}(l, H[s, s+m])$$

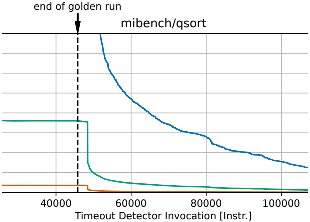


Autocorrelation as Timeout Detector

Derive parameters from the golden run:

- Ivocation roughly at $1.2 t_1 \rightarrow$ when m is filled
- Branch density: $m = \frac{\#\text{Instructions}}{\#\text{Branches}} * 0.2$
- $l_{max} = 16$, for detecting tight loops
- Lag specific threshold T_l:

$$T_l = 1 + \max_{s \in [0, (|H|-m)]} R_{bb}(l, H[s, s+m])$$



...	a	b	a	c	a	b	a	c	a	b	a	c	b	a	a	c	a	a	c	c	...

Evaluation - Setup

Setup:

- Integrated into FAIL*[3] - ISA level FI tool
- Static $3t_1$ detector as ground truth
- Seven benchmarks from MiBench [1]
  - Automotive and security branch
  - One with added TMR

Fault Model:

- Uniformly distributed single-bit flips
- In registers and memory
- Benign, SDC, trap and timeout (TO)

Evaluation - Setup

Setup:

- Integrated into FAIL*[3] - ISA level FI tool
- Static $3t_1$ detector as ground truth
- Seven benchmarks from MiBench [1]
  - Automotive and security branch
  - One with added TMR

Fault Model:

- Uniformly distributed single-bit flips
- In registers and memory
- Benign, SDC, trap and timeout (TO)

Evaluation - Quality

Benchmark	Classification Error Δ %				$1.2t_1$ Detector	ACTOR Detector		
	Ben.	SDC	Trap	TO	TO [Δ %]	Inv.	TPR	FPR
BitCount	+0.00	−0.02	+0.00	+0.02				
BitCount-TMR								
QSort								
SHA								
Blowfish (enc)								
Blowfish (dec)								
AES (enc)								
AES (dec)								

Evaluation - Quality

Benchmark	Classification Error Δ %				$1.2t_1$ Detector	ACTOR Detector		
	Ben.	SDC	Trap	TO	TO [Δ %]	Inv.	TPR	FPR
BitCount	+0.00	−0.02	+0.00	+0.02				
BitCount-TMR	−0.21	−0.01	+0.00	+0.22				
QSort	−0.05	−0.35	−0.03	+0.43				
SHA	+0.00	−0.16	−0.11	+0.27				
Blowfish (enc)	−0.06	−0.12	−0.01	+0.19				
Blowfish (dec)	−0.07	−0.13	+0.00	+0.20				
AES (enc)	−0.03	−0.26	−0.11	+0.40				
AES (dec)	−0.07	−0.08	−0.03	+0.19				

Evaluation - Quality

Benchmark	Classification Error Δ %				$1.2t_1$ Detector	ACTOR Detector		
	Ben.	SDC	Trap	TO	TO [Δ %]	Inv.	TPR	FPR
BitCount	+0.00	−0.02	+0.00	+0.02	+0.04			
BitCount-TMR	−0.21	−0.01	+0.00	+0.22	+38.14			
QSort	−0.05	−0.35	−0.03	+0.43	+1.41			
SHA	+0.00	−0.16	−0.11	+0.27	+13.75			
Blowfish (enc)	−0.06	−0.12	−0.01	+0.19	+0.31			
Blowfish (dec)	−0.07	−0.13	+0.00	+0.20	+0.28			
AES (enc)	−0.03	−0.26	−0.11	+0.40	+0.46			
AES (dec)	−0.07	−0.08	−0.03	+0.19	+1.49			

Evaluation - Quality

Benchmark	Classification Error Δ %				$\boxed{1.2t_1 \text{ Detector}}$	ACTOR Detector		
	Ben.	SDC	Trap	TO	TO [Δ %]	Inv.	TPR	FPR
BitCount	+0.00	−0.02	+0.00	+0.02	+0.04			
BitCount-TMR	−0.21	−0.01	+0.00	+0.22	+38.14			
QSort	−0.05	−0.35	−0.03	+0.43	+1.41			
SHA	+0.00	−0.16	−0.11	+0.27	+13.75			
Blowfish (enc)	−0.06	−0.12	−0.01	+0.19	+0.31			
Blowfish (dec)	−0.07	−0.13	+0.00	+0.20	+0.28			
AES (enc)	−0.03	−0.26	−0.11	+0.40	+0.46			
AES (dec)	−0.07	−0.08	−0.03	+0.19	+1.49			

Evaluation - Quality

Benchmark	Classification Error Δ %				$1.2t_1$ Detector	ACTOR Detector		
	Ben.	SDC	Trap	TO	TO [Δ %]	Inv.	TPR	FPR
BitCount	+0.00	−0.02	+0.00	+0.02	+0.04	4.69 %		
BitCount-TMR	−0.21	−0.01	+0.00	+0.22	+38.14	43.86 %		
QSort	−0.05	−0.35	−0.03	+0.43	+1.41	9.01 %		
SHA	+0.00	−0.16	−0.11	+0.27	+13.75	27.50 %		
Blowfish (enc)	−0.06	−0.12	−0.01	+0.19	+0.31	8.37 %		
Blowfish (dec)	−0.07	−0.13	+0.00	+0.20	+0.28	8.21 %		
AES (enc)	−0.03	−0.26	−0.11	+0.40	+0.46	5.63 %		
AES (dec)	−0.07	−0.08	−0.03	+0.19	+1.49	8.51 %		

Evaluation - Quality

Benchmark	Classification Error Δ %				$1.2t_1$ Detector	ACTOR Detector		
	Ben.	SDC	Trap	TO	TO [Δ %]	Inv.	TPR	FPR
BitCount	+0.00	−0.02	+0.00	+0.02	+0.04	4.69 %	98.51 %	
BitCount-TMR	−0.21	−0.01	+0.00	+0.22	+38.14	43.86 %	86.12 %	
QSort	−0.05	−0.35	−0.03	+0.43	+1.41	9.01 %	88.55 %	
SHA	+0.00	−0.16	−0.11	+0.27	+13.75	27.50 %	99.75 %	
Blowfish (enc)	−0.06	−0.12	−0.01	+0.19	+0.31	8.37 %	98.27 %	
Blowfish (dec)	−0.07	−0.13	+0.00	+0.20	+0.28	8.21 %	96.76 %	
AES (enc)	−0.03	−0.26	−0.11	+0.40	+0.46	5.63 %	99.92 %	
AES (dec)	−0.07	−0.08	−0.03	+0.19	+1.49	8.51 %	85.65 %	

Evaluation - Quality

Benchmark	Classification Error Δ %				$1.2t_1$ Detector	ACTOR Detector		
	Ben.	SDC	Trap	TO	TO [Δ %]	Inv.	TPR	FPR
BitCount	+0.00	−0.02	+0.00	+0.02	+0.04	4.69 %	98.51 %	93.40 %
BitCount-TMR	−0.21	−0.01	+0.00	+0.22	+38.14	43.86 %	86.12 %	0.59 %
QSort	−0.05	−0.35	−0.03	+0.43	+1.41	9.01 %	88.55 %	37.72 %
SHA	+0.00	−0.16	−0.11	+0.27	+13.75	27.50 %	99.75 %	42.29 %
Blowfish (enc)	−0.06	−0.12	−0.01	+0.19	+0.31	8.37 %	98.27 %	82.40 %
Blowfish (dec)	−0.07	−0.13	+0.00	+0.20	+0.28	8.21 %	96.76 %	84.12 %
AES (enc)	−0.03	−0.26	−0.11	+0.40	+0.46	5.63 %	99.92 %	56.39 %
AES (dec)	−0.07	−0.08	−0.03	+0.19	+1.49	8.51 %	85.65 %	5.27 %

ISE SRA

Evaluation - Quality

Benchmark	Classification Error Δ %				$1.2 t_1$ Detector	ACTOR Detector		
	Ben.	SDC	Trap	TO	TO [Δ %]	Inv.	TPR	FPR
BitCount	+0.00	−0.02	+0.00	+0.02	+0.04	4.69 %	98.51 %	93.40 %
BitCount-TMR	−0.21	−0.01	+0.00	+0.22	+38.14	43.86 %	86.12 %	0.59 %
QSort	−0.05	−0.35	−0.03	+0.43	+1.41	9.01 %	88.55 %	37.72 %
SHA	+0.00	−0.16	−0.11	+0.27	+13.75	27.50 %	99.75 %	42.29 %
Blowfish (enc)	−0.06	−0.12	−0.01	+0.19	+0.31	8.37 %	98.27 %	82.40 %
Blowfish (dec)	−0.07	−0.13	+0.00	+0.20	+0.28	8.21 %	96.76 %	84.12 %
AES (enc)	−0.03	−0.26	−0.11	+0.40	+0.46	5.63 %	99.92 %	56.39 %
AES (dec)	−0.07	−0.08	−0.03	+0.19	+1.49	8.51 %	85.65 %	5.27 %

Evaluation - Time Savings

Benchmark	Sim. Post-Inj. Instr. [%]			E2E [%]
	ACTOR	$1.2t_1$	OPT_{t_f}	
BitCount	-13.1	-17.3	-25.2	-12.66

Evaluation - Time Savings

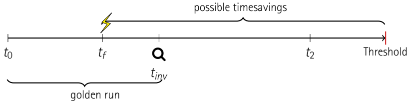
Benchmark	Sim. Post-Inj. Instr. [%]			E2E [%]
	ACTOR	$1.2t_1$	OPT_{t_f}	
BitCount	−13.1	−17.3	−25.2	−12.66
BitCount-TMR	−9.9	−14.6	−21.1	−7.39
QSort	−19.5	−23.3	−32.3	−16.07
SHA	−30.6	−44.4	−61.7	−27.64
Blowfish (enc)	−16.1	−20.7	−28.7	−15.91
Blowfish (dec)	−15.8	−20.4	−28.4	−15.72
AES (enc)	−17.2	−16.0	−22.2	−17.59
AES (dec)	−13.1	−14.5	−20.1	−12.24

Evaluation - Time Savings

Benchmark	Sim. Post-Inj. Instr. [%]			E2E [%]
	ACTOR	$1.2t_1$	OPT_{t_f}	
BitCount	-13.1	-17.3	-25.2	-12.66
BitCount-TMR	-9.9	-14.6	-21.1	-7.39
QSort	-19.5	-23.3	-32.3	-16.07
SHA	-30.6	-44.4	-61.7	-27.64
Blowfish (enc)	-16.1	-20.7	-28.7	-15.91
Blowfish (dec)	-15.8	-20.4	-28.4	-15.72
AES (enc)	-17.2	-16.0	-22.2	-17.59
AES (dec)	-13.1	-14.5	-20.1	-12.24

ACTOR

- Autocorrelation-based dynamic timeout detector
- Low classification error $< 0.5\%$
- High TP-Rate: 85% up to 99.9%
- Up to 27.6 % end-to-end savings



Takeaways:

- Timeouts take (over-proportional) time!
- Execute detectors shortly after golden program run-time

Bibliography

1. Guthaus, M.R., Ringenberg, J.S., Ernst, D., Austin, T.M., Mudge, T., and Brown, R.B.: MiBench: A free, commercially representative embedded benchmark suite. In: Fourth Annual IEEE Intl. Workshop on Workload Characterization. WWC-4 (2001). DOI: `10.1109/WWC.2001.990739`
2. Ibing, A., Kirsch, J., and Panny, L.: Autocorrelation-Based Detection of Infinite Loops at Runtime. In: IEEE Int. Conf. Dependable, Autonomic and Secure Computing (2016). DOI: `10.1109/DASC-PICom-DataCom-CyberSciTec.2016.78`
3. Schirmeier, H., Hoffmann, M., Dietrich, C., Lenz, M., Lohmann, D., and Spinczyk, O.: FAIL*: An Open and Versatile Fault-Injection Framework for the Assessment of Software-Implemented Hardware Fault Tolerance. In: Sens, P. (ed.) 11th European Dependable Computing Conference (EDCC '15) (2015). DOI: `10.1109/EDCC.2015.28`