Robust and Accurate Period Inference using Regression-Based Techniques

Serban Vadineanu

TU Delft

Mitra Nasri

TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

RTSS 2020

Why inferring period?

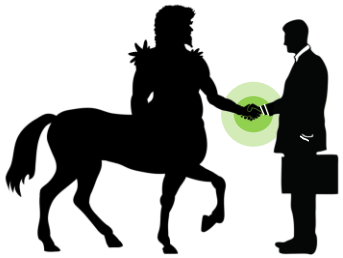
Growth of complexity

Tools and techniques to verify that

the observable behavior of the system

- End-to-end response time
- Actuation instants
- Messages on the CAN bus
- …

meets



Realization of the system

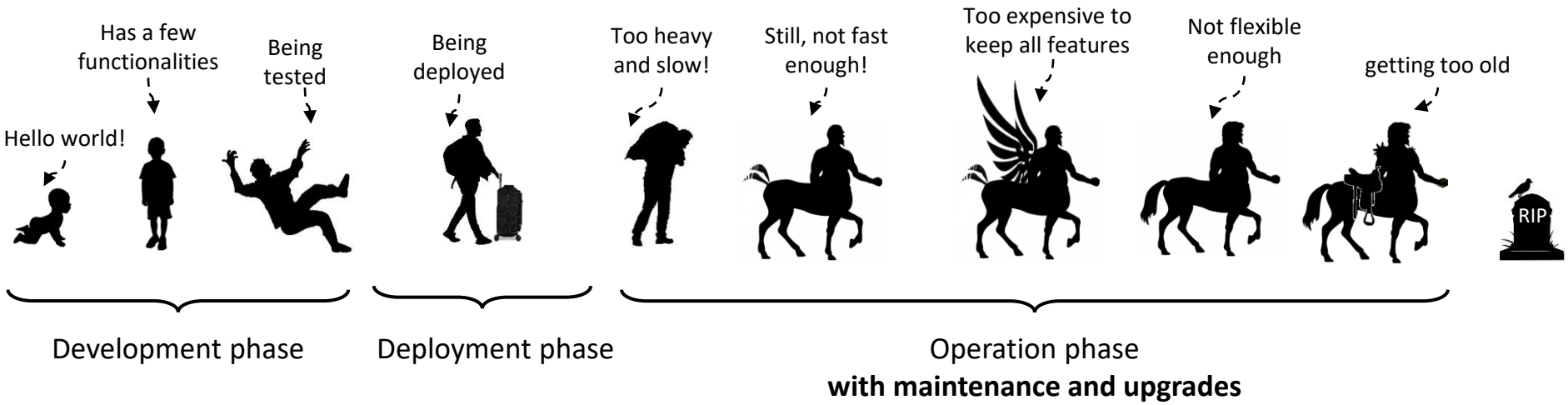
its specifications

- Respecting deadlines
- Performing periodic actuations
- …

A fundamental problem in real-time systems:

> *"Do the activities that are supposed to be **periodic** happen periodically?"*

Why inferring period?



Finding "time bugs"

Deviations from the expected timing behavior

Runtime monitoring

Detecting timing anomalies and security attacks that leave a trace on the observable timing profile

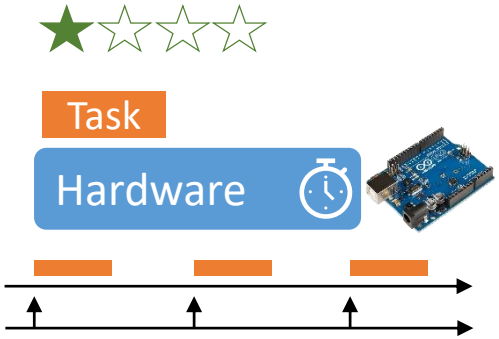
Diagnosing the system after applying a patch or an upgrade

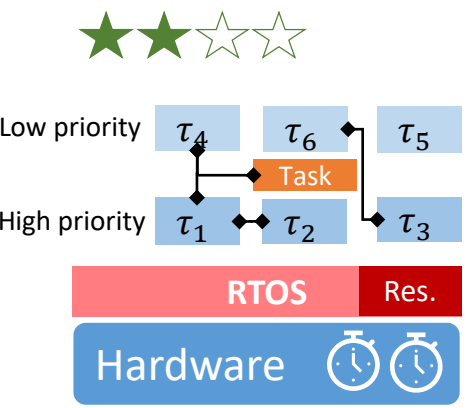
A fundamental problem in real-time systems:

*"Do the activities that are supposed to be **periodic** happen periodically?"*

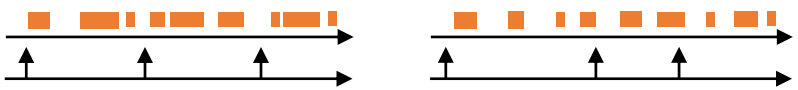
Why inferring period?

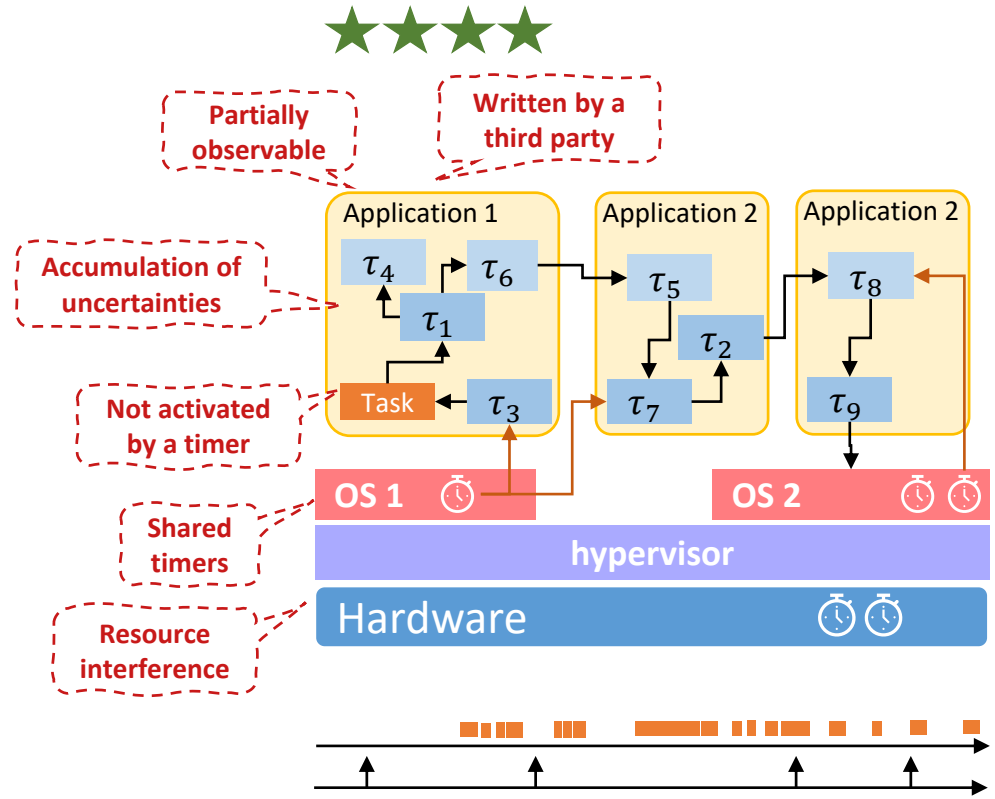
We implement tasks to be periodic!
We know the periods. Don't we?!



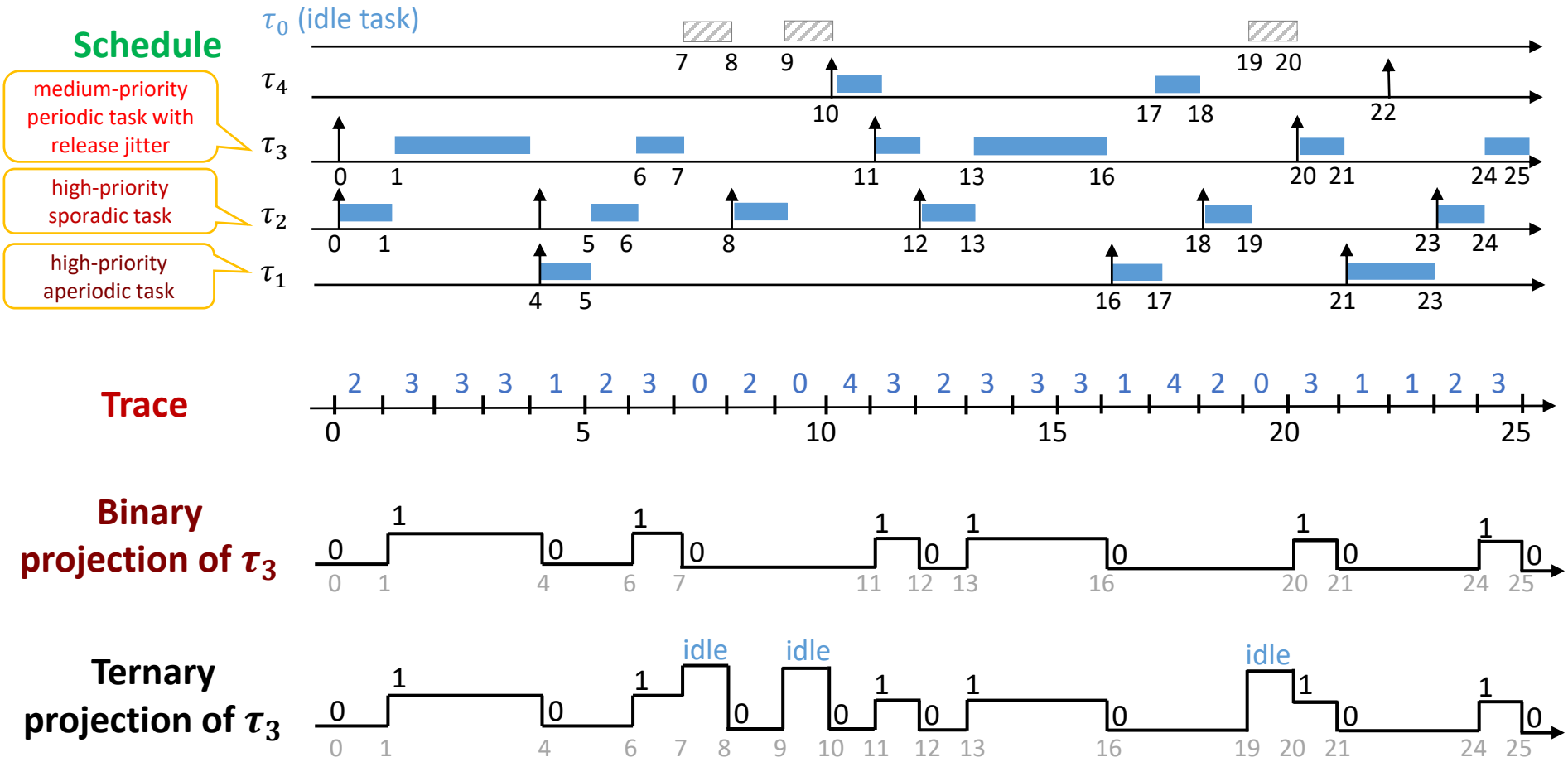


- Data dependency
- Release jitter
- Execution time variation
- Shared resources
- Overheads and delays
- …





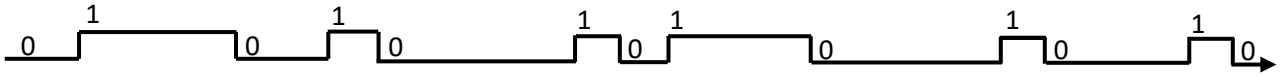
Period inference problem



Problem. Given a binary (or ternary) projection, find the period.

Period inference problem

Binary projection



Can be obtained using simple monitoring tools like *top* command (in Linux) or by observing a CAN network

We do not know

- The number of tasks, periods, offsets, execution times, etc.
- Scheduling policy
- Presence of preemptions or self suspensions
- Existence of timing uncertainties (e.g., release jitter, execution time variation)
- Existence of aperiodic tasks
- Presence of deadline misses (for other task or the one we analyze)
- Presence of overloads
- …

Other requirements

High accuracy

Robustness against uncertainties

Generic (works for any periodic system)

Low overhead and memory consumption

If it is a learning-based solution, then it must have **high accuracy even for systems that are different from what it has seen before**

State of the art

Inferring period of an activity in real-time systems

1. **PeTaMi** [Iegorov 2017] inferring period of tasks based on interval analysis

✓ Does **not require** additional learning

⊗ **Poor performance** for non-harmonic periods

⊗ **Not robust** against uncertainties

⊗ **Slow** for longer traces

2. Inferring period of messages on a CAN network using Fourier transformation [Young 2019]

⊗ **Poor performance** for preemptive tasks

State of the art

Inferring period of an activity in real-time systems	Learning-based solutions to infer other properties in real-time systems
No learning-based solution for **period inference**	**K-nearest neighbors** were used to distinguish tasks from each other based on their power trace [Lamichhane 2018]
Lack of robustness	
Limited scope	

State of the art

Inferring period of an activity in real-time systems

No learning-based solution for period inference

Lack of robustness

Limited scope

Learning-based solutions to infer other properties in real-time systems

Not applicable on period inference problem

Period inference using signal-processing techniques

- **Periodogram** [Schuster 1998, Vlachos 2005, Li 2012]
- **Autocorrelation** [Gubner 2006, Malode 2015, Puech 2019]
- Are based on Fourier transformation

Fourier-based techniques

Perform **very poorly** if the signal has noise

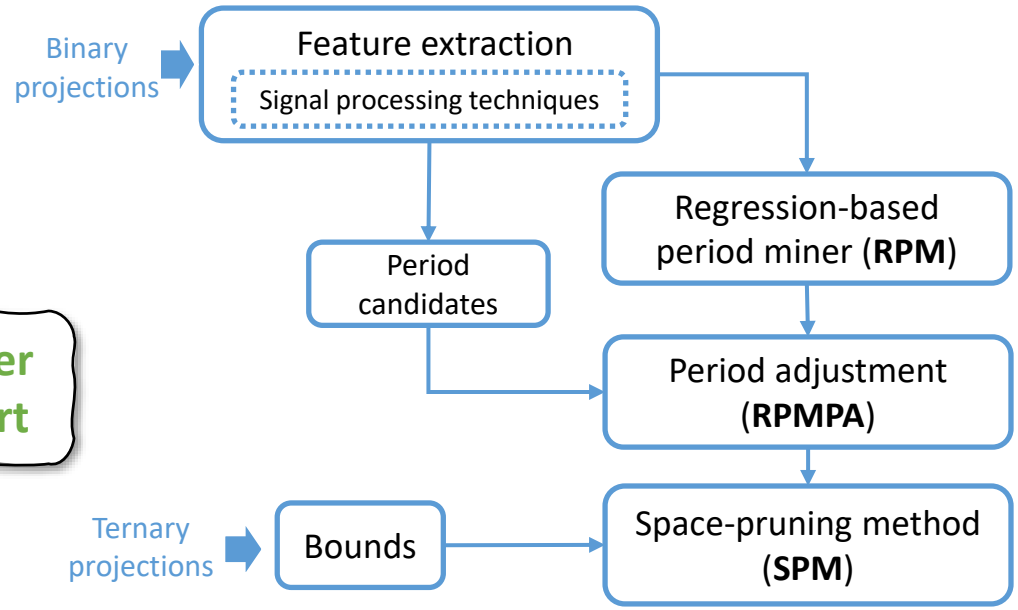
- **Preemptions**, release **jitters**, **execution time variations**

There is a need for **accurate** and **robust** solutions for **period inference**

Contributions

An open-source
period inference framework
based on regression-based techniques

2 to 3 <u>orders of magnitude</u> higher accuracy than the state of the art

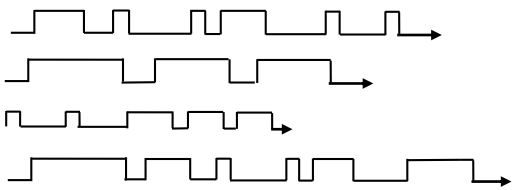


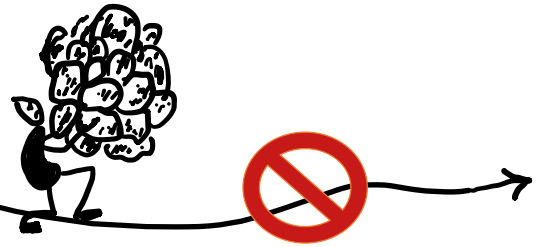
A thorough investigation of regression-based methods' performance

- **Accuracy**
- **Robustness** (in the presence of uncertainties)
- **Generalizability** (robustness w.r.t. new datasets)
- **Runtime cost** (overheads and memory)

Why extracting features?

Binary projections





Regression-based machine learning → **Period**

High dimensional data

low efficiency and large runtime during model learning

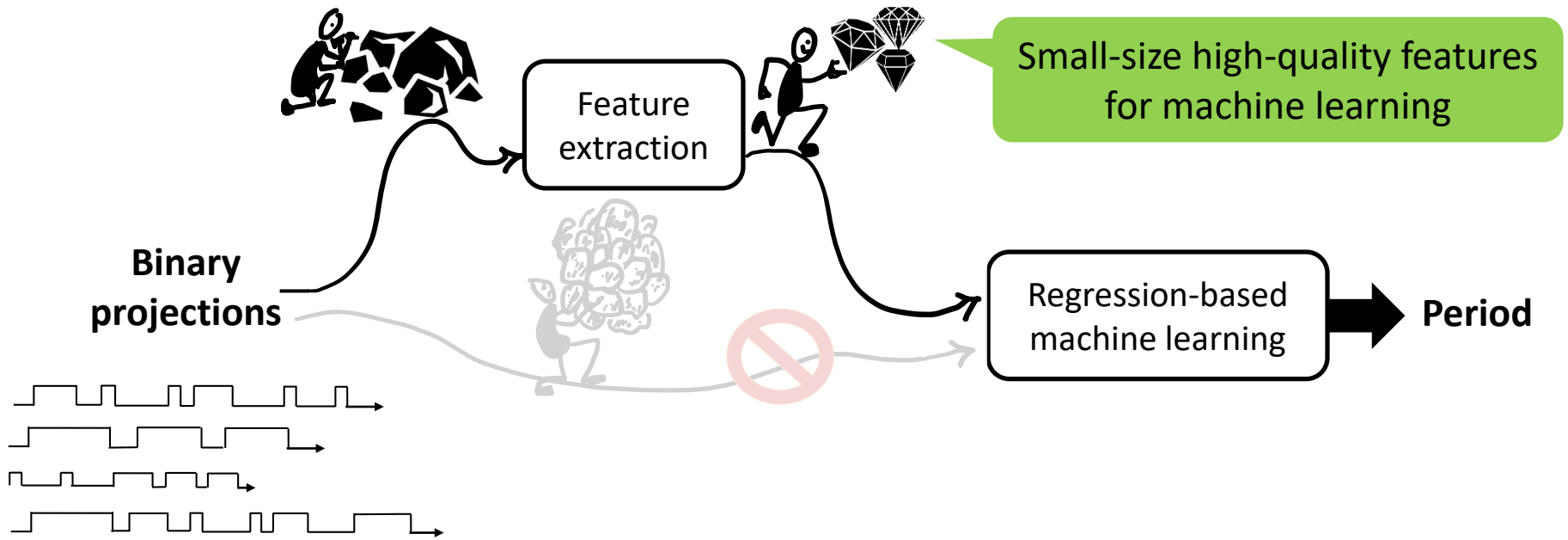
Variable-length input

Most machine-learning methods require a fixed input size

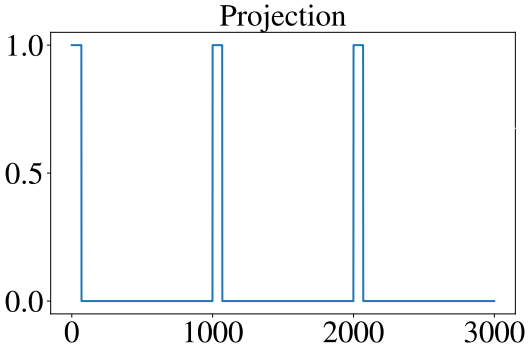
Should projections be cut to a fixed length?

- too short → low accuracy
- too long → large runtime for model learning
- Doesn't solve dimensionality problem

Why extracting features?

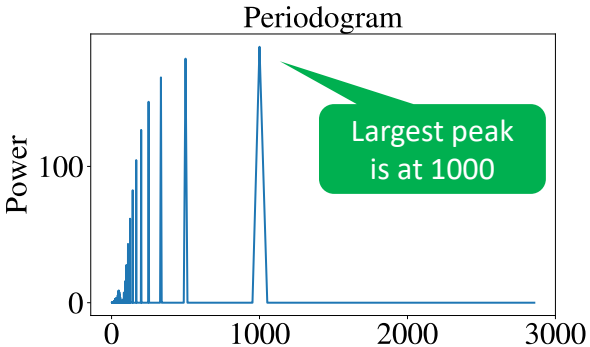


Feature extraction



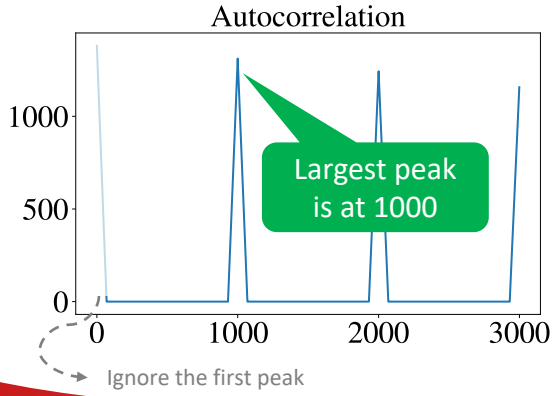
Projection of a high-priority task with period 1000

If these techniques are so good, why do we even need a regression-based solution?



Periodogram

- An estimate of the spectral density of the signal
- **How?**
  Provided by the squared length of each Fourier coefficient of the signal

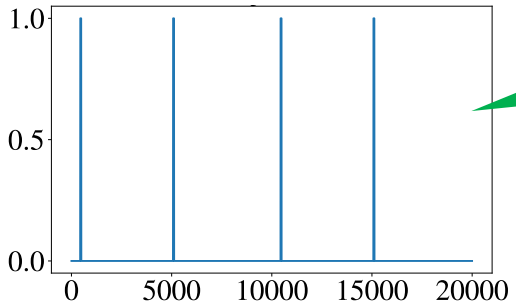


Circular autocorrelation

- Examines the similarity of a sequence to its previous values at different time lags
- **How?**
  Inverse Fourier transform of dot product between the signal and its conjugation

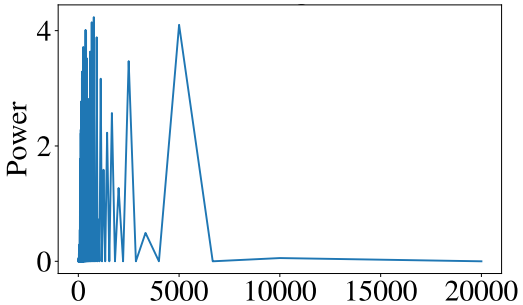
Challenges of feature extraction

Projection

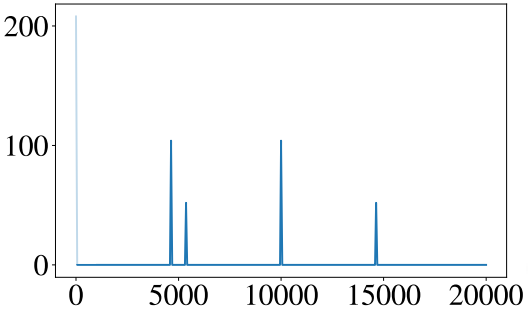


Projection of a low-priority task with period 5000 (no preemption)

Periodogram

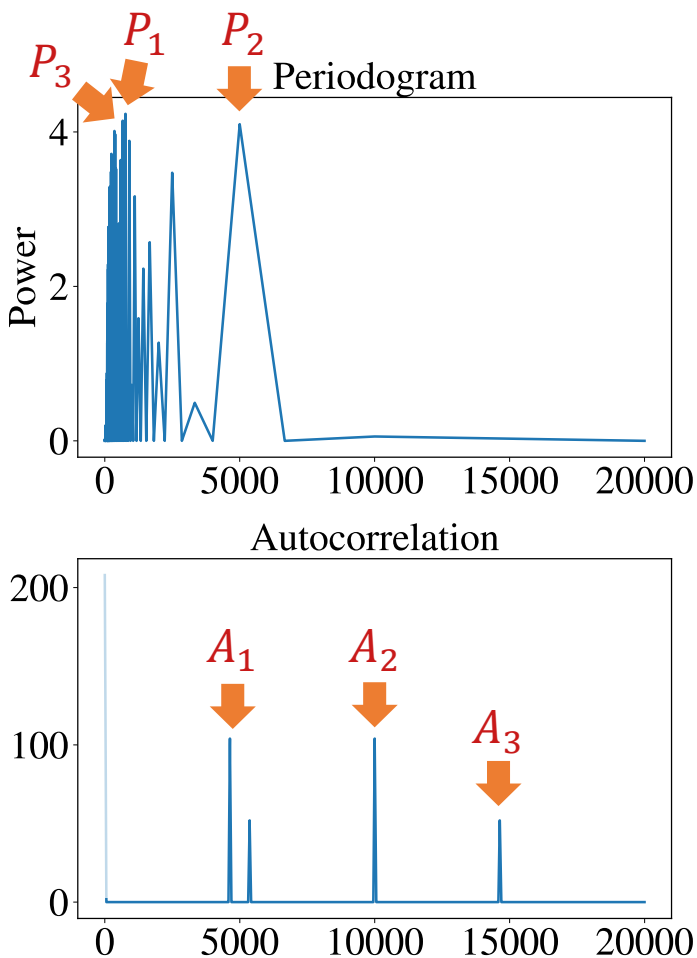


Autocorrelation



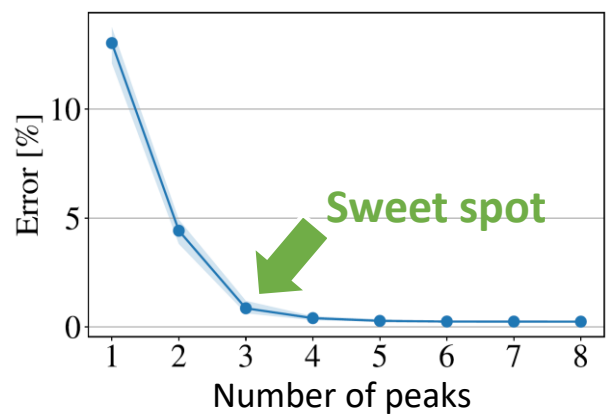
1. **Have multiple peaks**
2. **The highest peak is not necessarily the period**
3. **May not even have a peak at the true period**
   (e.g., in autocorrelation)

Features

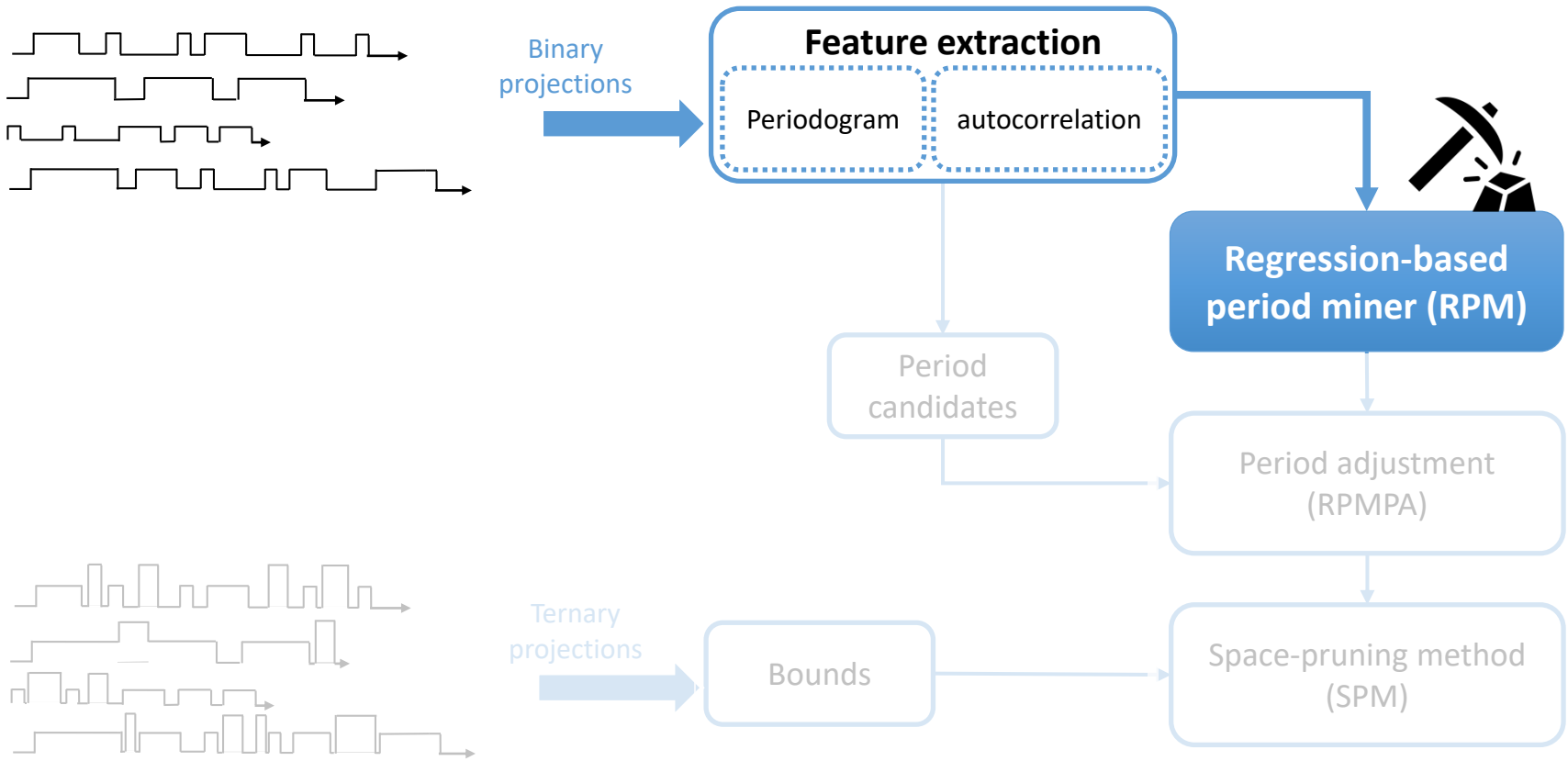


We select the **3** highest peaks from **Periodogram** and **3** highest peaks from **autocorrelation**

- Features for an input projection: $\{P_1, P_2, P_3, A_1, A_2, A_3\}$
- ✓ Small set
- ✓ Fixed size
- ✓ Independent of task's period, hyperperiod, projection length, etc.



Our solution in a nutshell



Regression-based period miner (RPM)

Features
- Top 3 periods from periodogram
- Top 3 periods from autocorrelation

→ Regression-based machine learning → **Period**

Which regression algorithm would result in a better accuracy for inferring periods?

M. Fernández-Delgado, M. S. Sirsat, E. Cernadas, S. Alawadi, S. Barro, and M. Febrero-Bande. "**An extensive experimental survey of regression methods**", Neural Networks, pp. 11-34, **2019**.

6 best families of regression techniques

Algorithm	Nickname	Category
Cubist Regression [24]–[26]	*cubist*	Rule-based
Generalized Boosting Regression [27]	*gbm*	Boosting
Averaged Neural Network [28]	*avNNet*	Neural Networks
Extremely Randomized Regression Trees [29]	*extraTrees*	Random Forests
Bayesian Additive Regression Tree [30]	*bartMachine*	Bayesian Models
Support Vector Regression [31]	*svr*	Support Vector Machines

Which regression algorithms?

Cubist regression

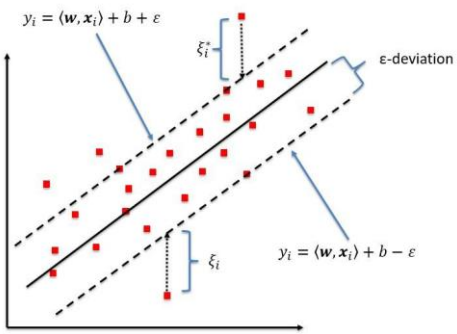
extremely randomized regression trees (extraTrees)

gradient boosting method (gbm)

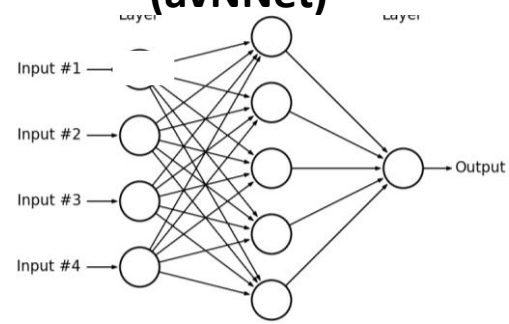
Bayesian additive regression trees (bartMachine)

Tree-based algorithms

support vector regression (svr)



averaged neural networks (avNNet)

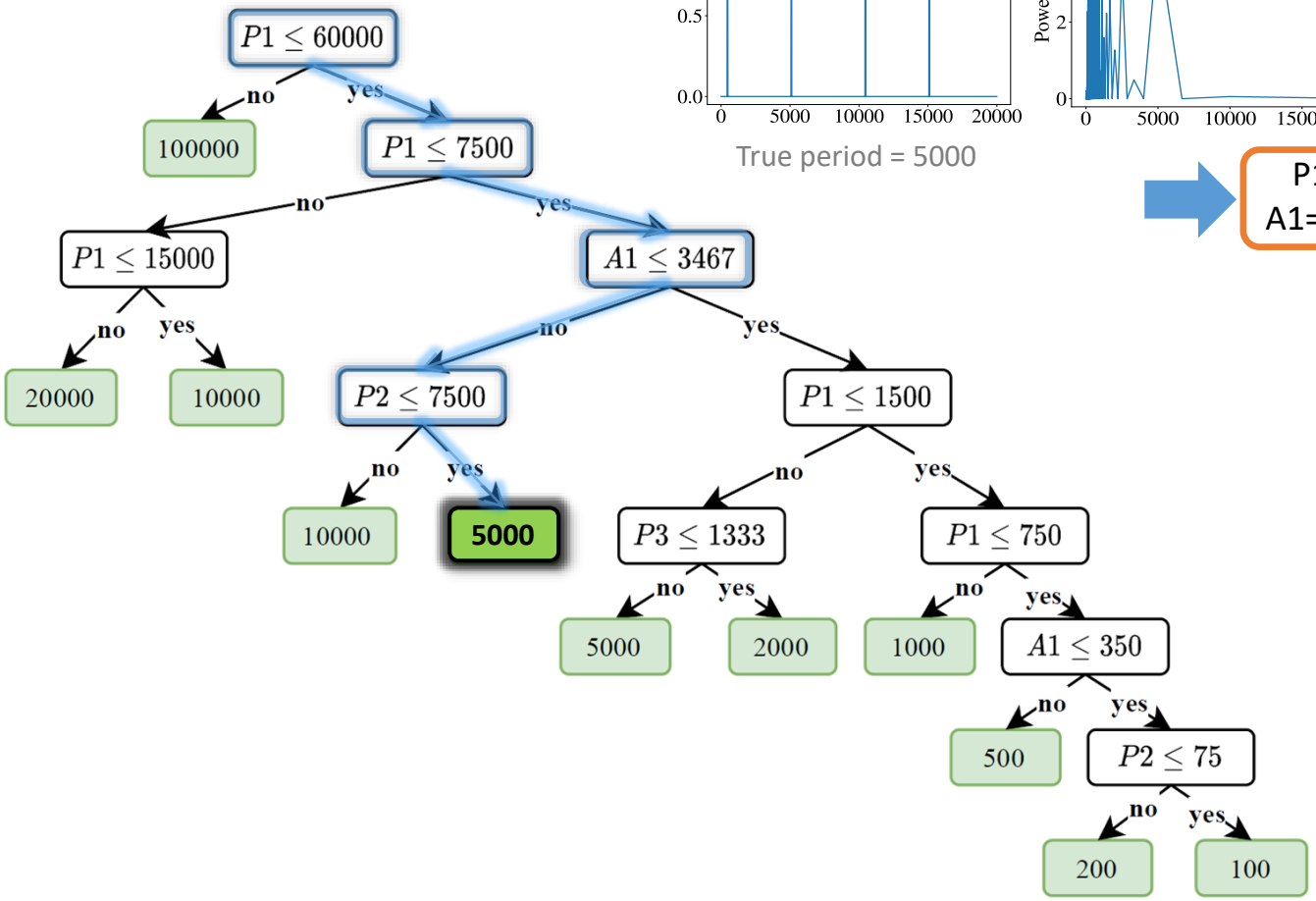


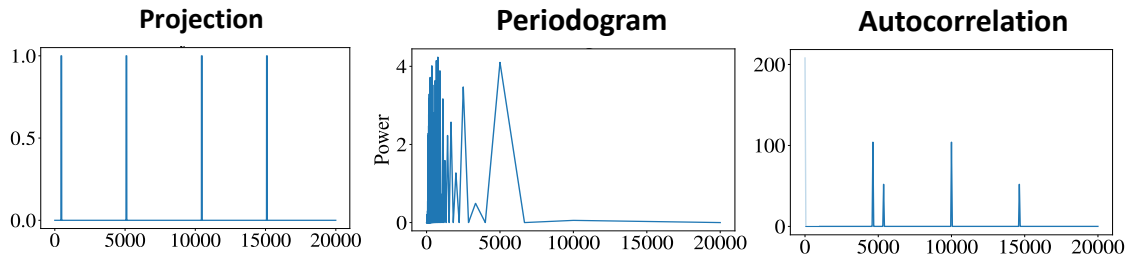
Cubist [Quinlan 1992, 1993, 2014]
ExtraTrees [Geurts 2006]
gbm [Friedman 2002]

bartMachine [Chipman 2010]
avNNet [Ripley 2007]
Svr [Cortes 1995]

What is a regression tree?

Simple





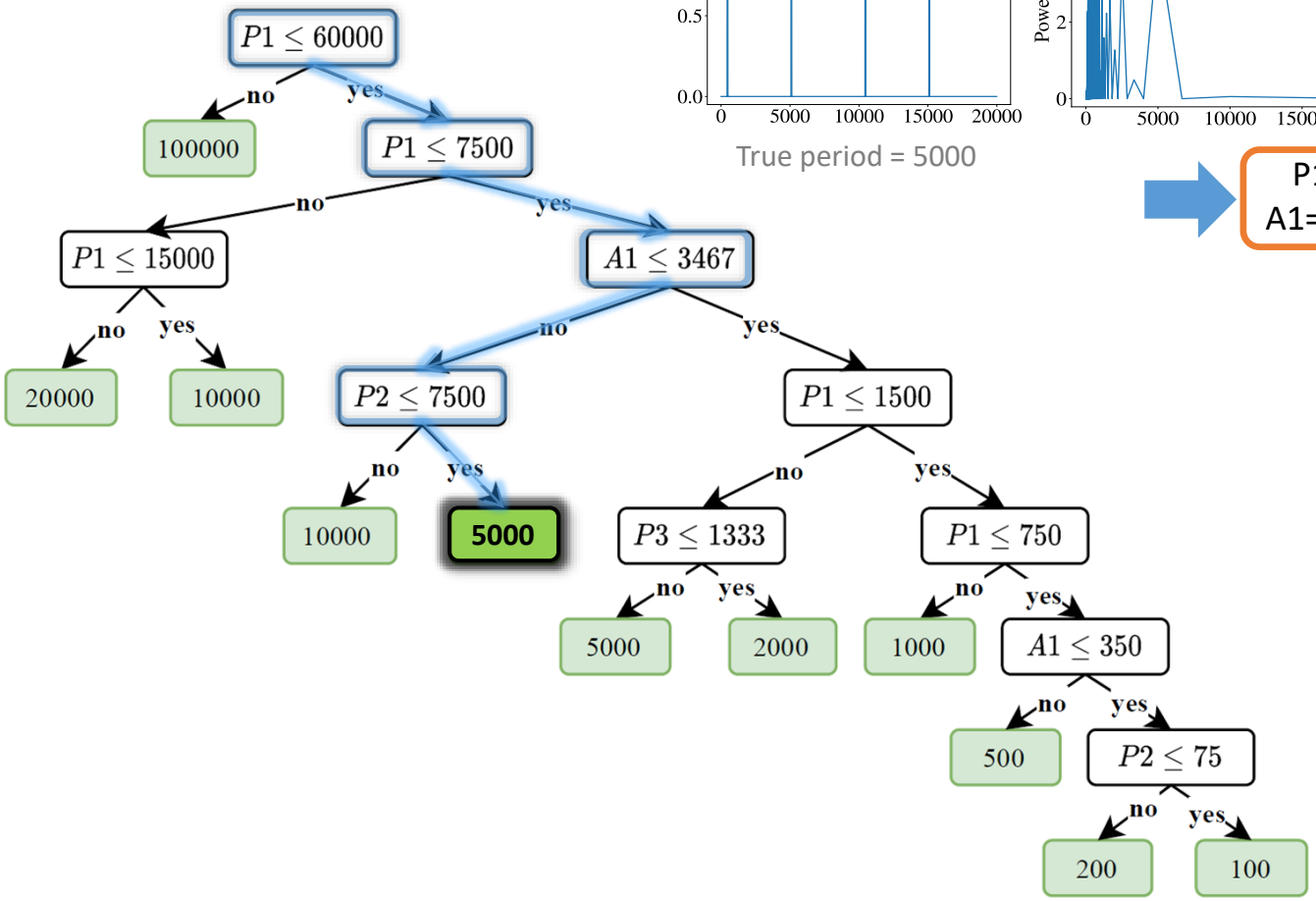
True period = 5000

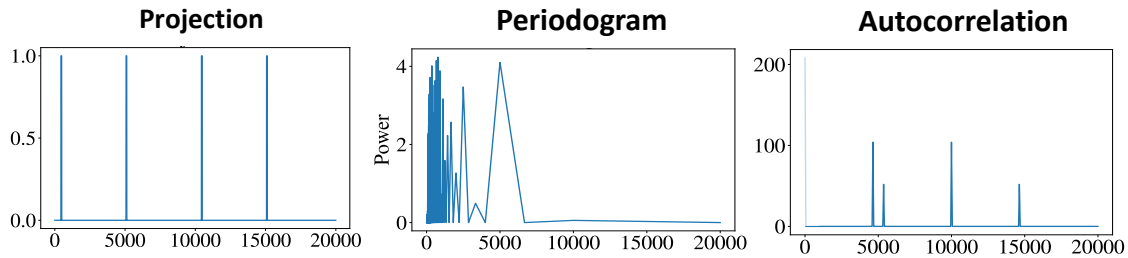
P1=769, P2=666, P3=5000,
A1=4635, A2=10000, A3=5365

P1, P2, P3 – top 3 periods from periodogram
A1, A2, A3 – top 3 periods from autocorrelation

What is a regression tree?

Simple





True period = 5000

P1=769, P2=666, P3=5000,
A1=4635, A2=10000, A3=5365

P1, P2, P3 – top 3 periods from periodogram
A1, A2, A3 – top 3 periods from autocorrelation

Regression-tree based algorithms

Cubist regression

*Collapses the tree into a set of **rules***

extremely randomized regression trees

Trains multiple trees, averages the output

gradient boosting method (gbm)

Creates a tree to minimize the error of previous trees

Bayesian additive regression trees

Like gbm, reduces the error of previous trees using a probability model for the likelihood of leaf values

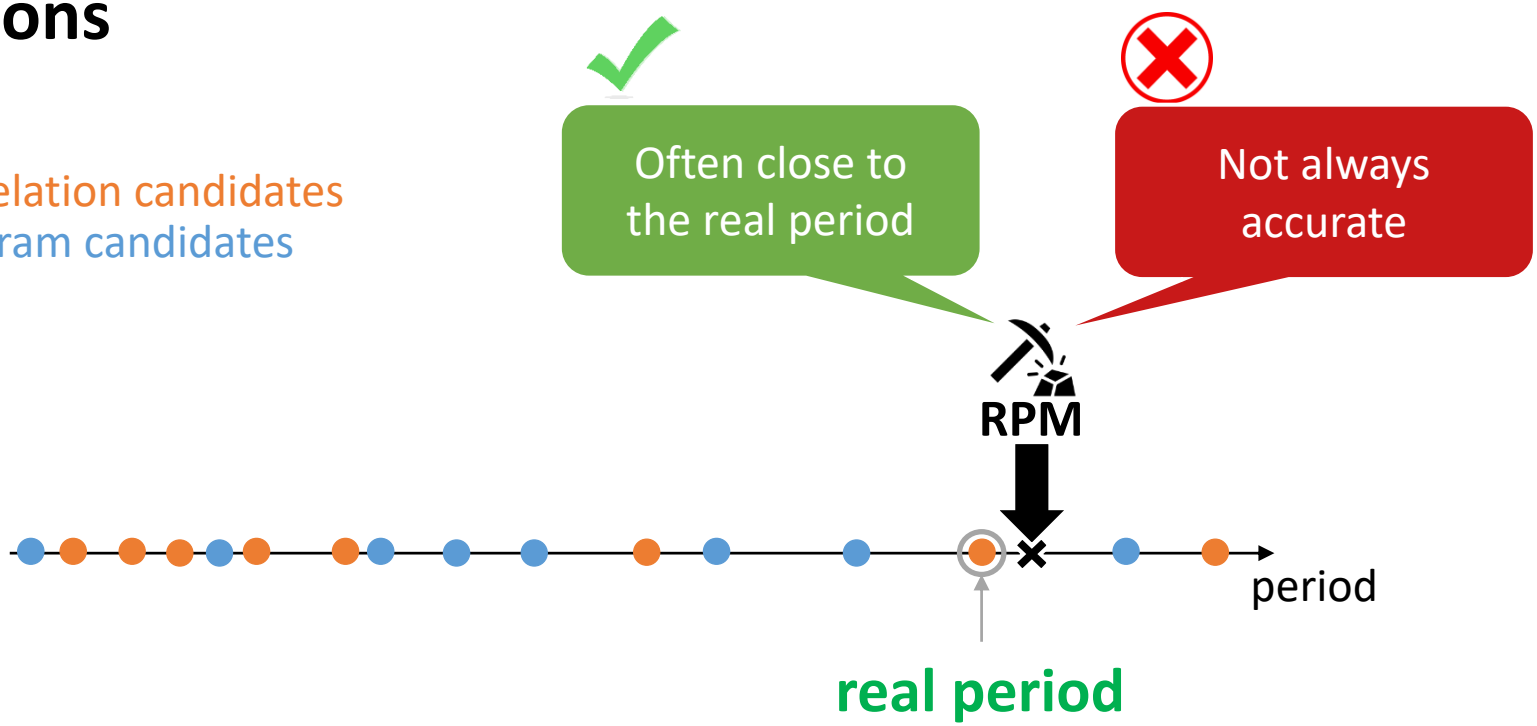
See the paper to learn more about them

Cubist [Quinlan 1992, 1993, 2014]
ExtraTrees [Geurts 2006]
gbm [Friedman 2002]

bartMachine [Chipman 2010]
avNNet [Ripley 2007]
Svr [Cortes 1995]

Observations

- autocorrelation candidates
- periodogram candidates

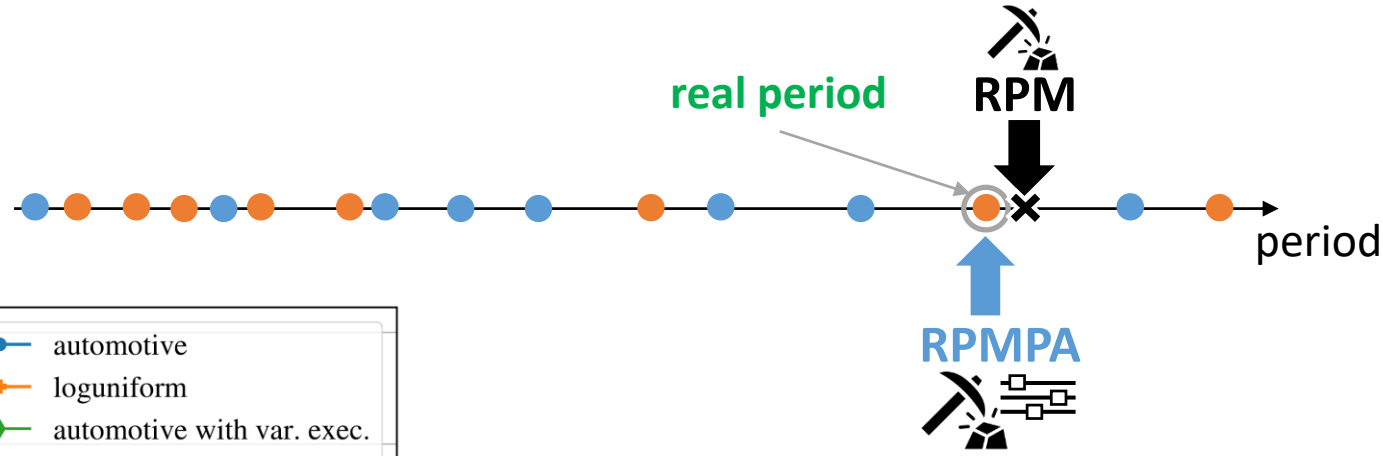


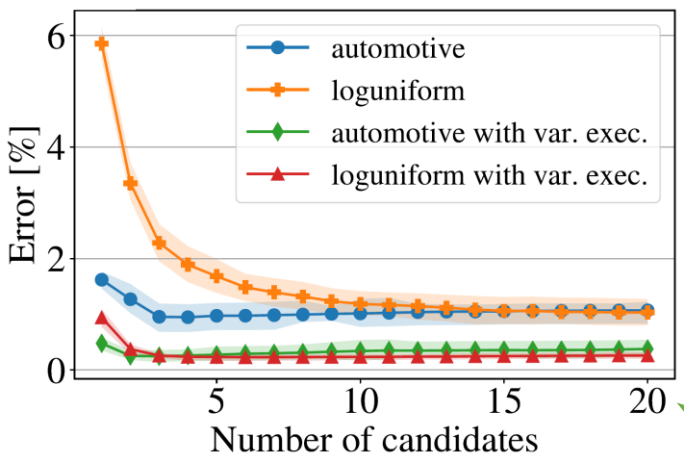
Next, we try to improve the accuracy by our period adjustment technique

RPM – our regression-based period miner

Period adjustment (RPMPA)

1. Create a candidate list from the top 20 peaks of **periodogram** & **autocorrelation**
2. Choose the candidate closest to what RPM estimates



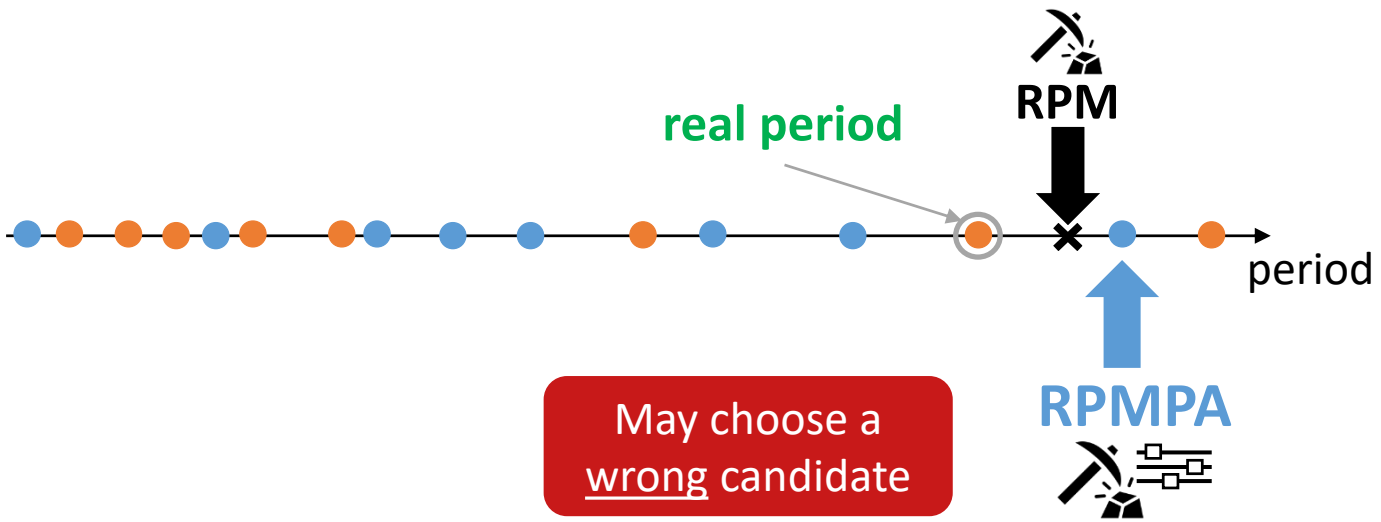


Good for various system types

RPM – regression-based period miner
RPMPA – regression-based period miner with <u>period adjustment</u>

What can go wrong?

How to derive upper and lower bounds for period candidates if we know that the scheduler is work-conserving?



RPM – regression-based period miner
RPMPA – regression-based period miner with period adjustment

Bounds on the period

1. **Lower bound** – based on the longest non-running interval from a projection.

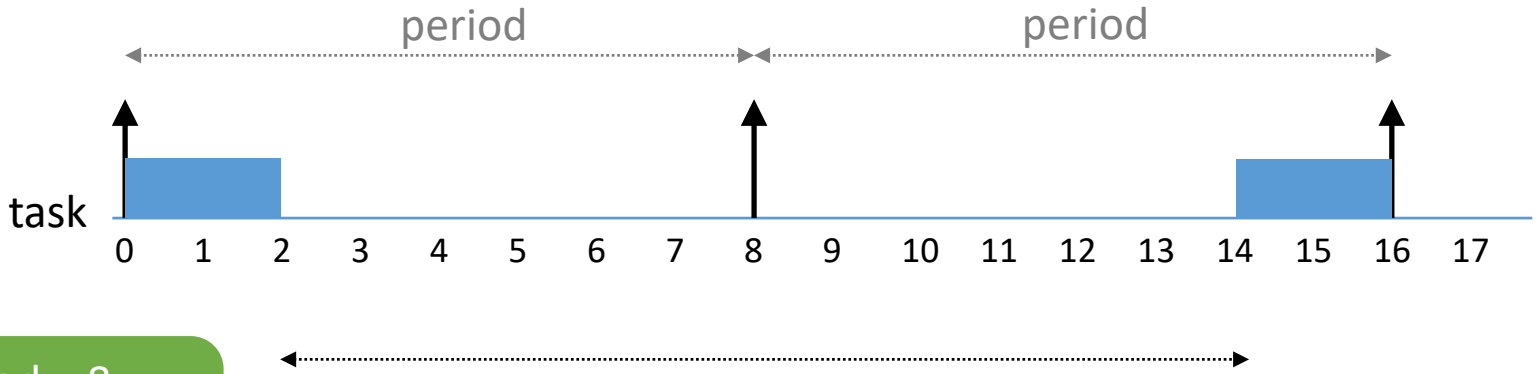
   Required assumption: The task under analysis does not miss a deadline or skip a job in the projection.

2. **Upper bound** – based on the knowledge about the *idle times* in ternary projections.

   Required assumption: The scheduling policy is work conserving, the system has one processor, and we have access to ternary projections.

If the assumptions cannot be verified, then the lower bound is 0 and upper bound is ∞.

Lower bound derivation

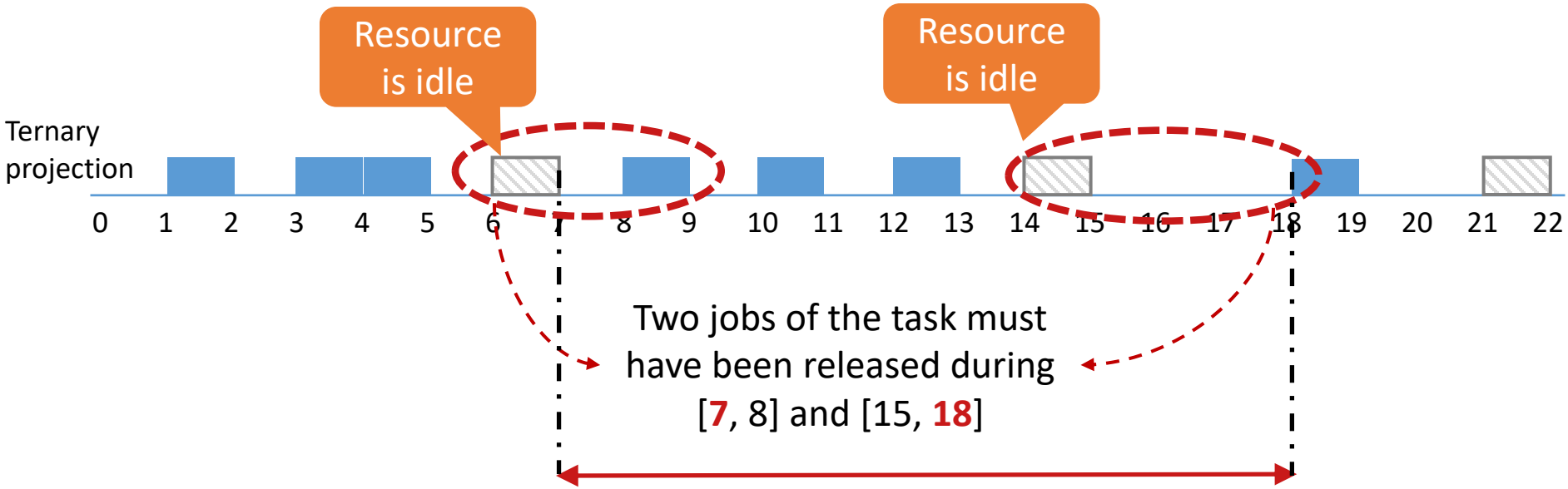


period = 8
WCET = 2
Implicit deadline

Δ: the longest interval of zeros in the binary projection < **2 x period**

$$period > \frac{\Delta}{2}$$

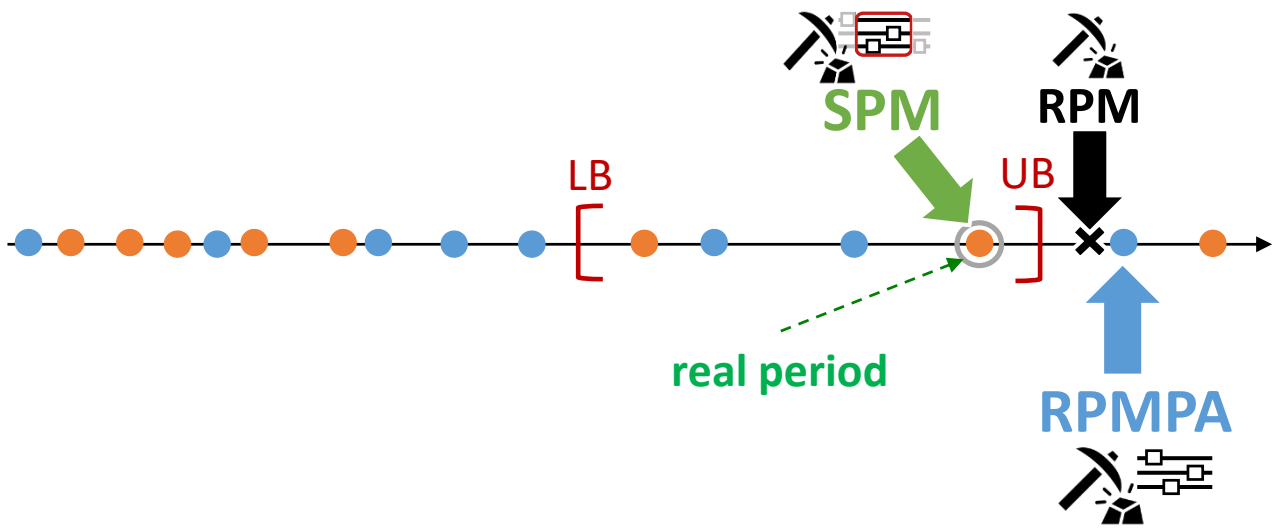
Upper bound derivation



- Scan the projection
- Derive an upper bound per such occasion.
- Use the tightest upper bound

Space-pruning method (SPM)

1. Create candidate list from top 20 values of periodogram & autocorrelation
2. **Filter the candidates based on the bounds**
3. Choose the candidate closest to what RPM estimates



RPM – regression-based period miner
RPMPA – regression-based period miner with <u>period adjustment</u>
UB – upper bound
LB – lower bound

Evaluations

Evaluation questions

Did our solutions improve accuracy?

How do our solutions compare in terms of accuracy?

RPM RPMPA SPM

How do the six families of regression methods compare when applied on the period inference problem?

In terms of

- Accuracy, runtime, memory consumption, robustness against uncertainties, and learning robustness

RPM – regression-based period miner
RPMPA – regression-based period miner with period adjustment
SPM – space-pruning method

Evaluations: datasets

automotive traces

Automotive benchmark application

- Task sets used in automotive domain [Krammer 2015]
- Periods from {1, 2, 5, 10, 20, 50, 100, 200, 1000}ms

log-uniform traces

Synthetic task sets

- Random periods chosen from [10, 1000] with log-uniform distribution [Emberson 2010]

Traces were generated by **Simso** simulator [Chéramy 2014].

Case study

Two datasets from message traces of the CAN bus of actual vehicles

Evaluations: metric and parameters

Accuracy

- Period types
- Number of tasks
- System utilization
- Release jitter
- Execution time variation

Robustness against uncertainties

- Presence of high-priority aperiodic tasks
- Tasks may drop jobs

Generalizability of learning (a.k.a. learning robustness)

- Impact of projection length on testing's accuracy
- Impact of training and testing on different task set types
- Impact of training and testing on different task set sizes

Average estimation error

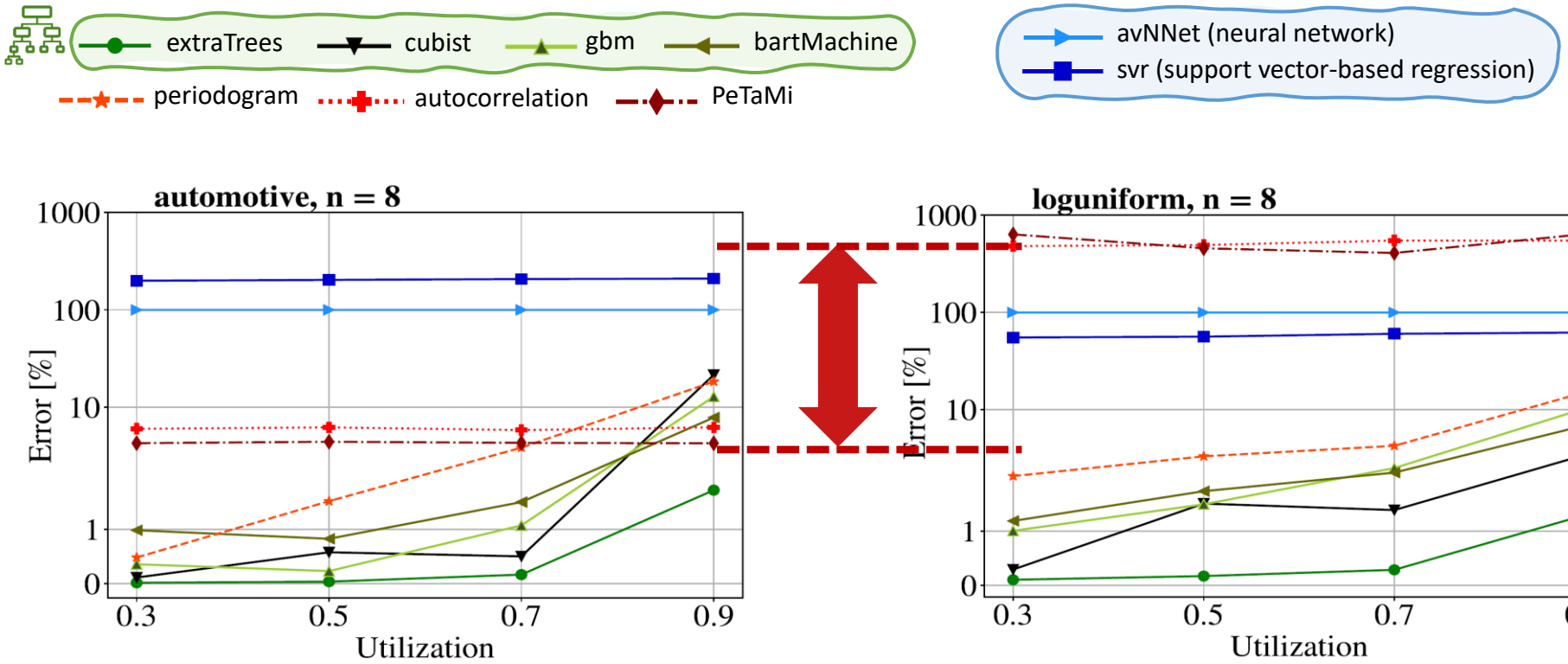
$$e = \frac{\sum_{i=1}^{N} \frac{\widehat{T}_i - T_i}{T_i}}{N}$$

$\widehat{T}_i$: Estimated period

T_i: Actual periods

N: Number of tasks

Impact of system utilization

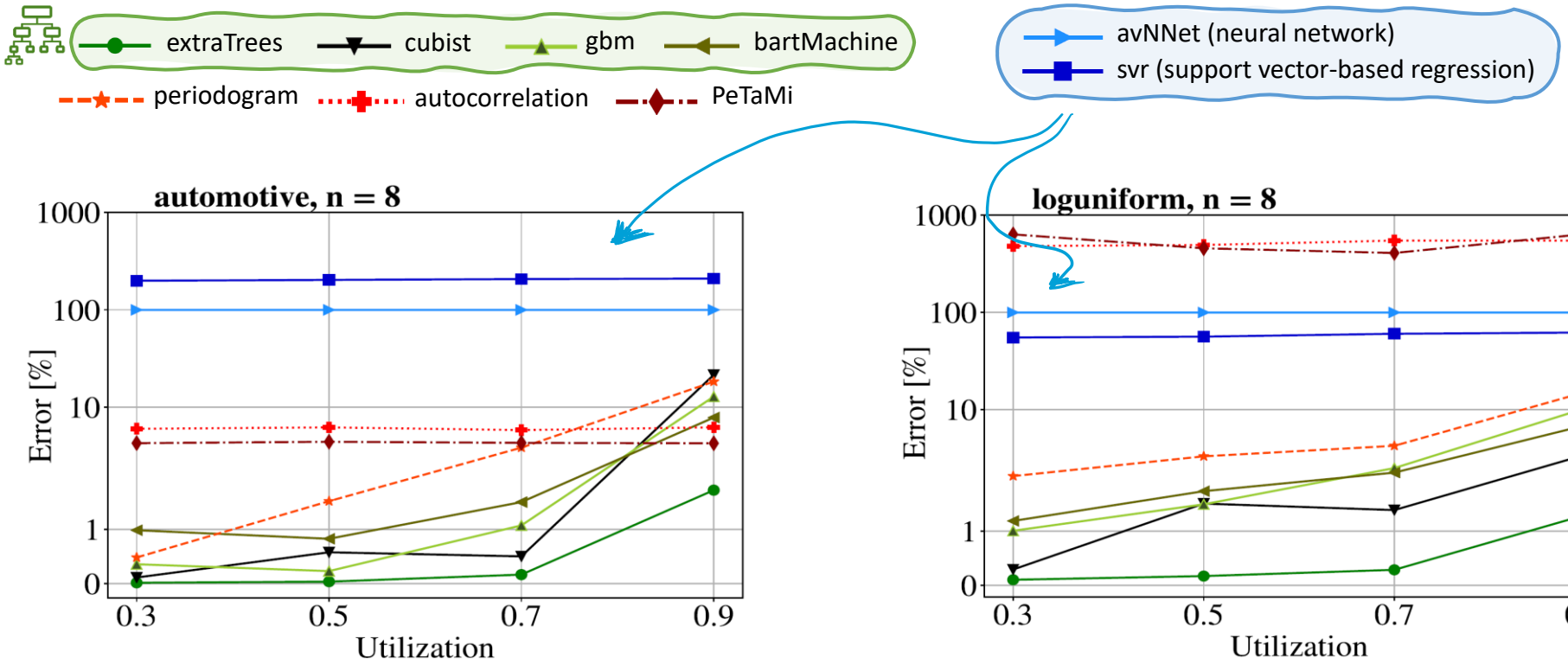


120x increase in error for PeTaMi and autocorrelation

PeTaMi: O. Iegorov, R. Torres, and S. Fischmeister. "Periodic task mining in embedded system traces," RTAS, 2017.

Impact of system utilization

RPM

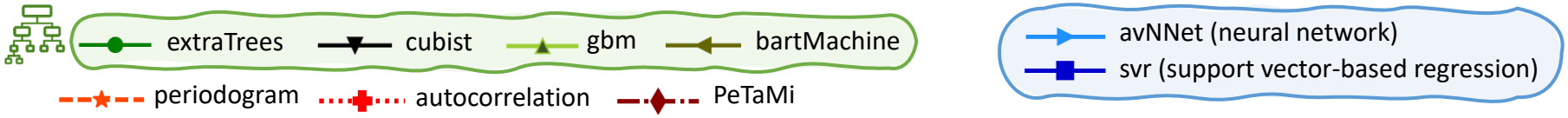


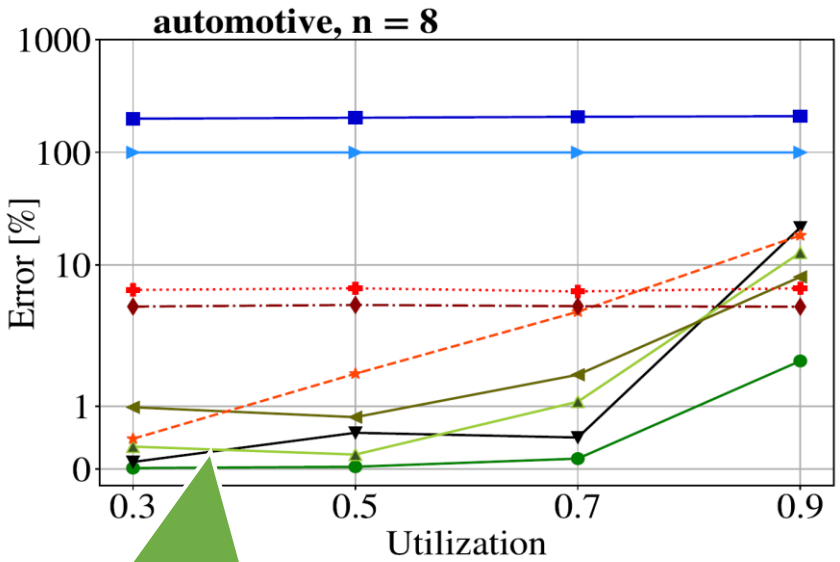
avNNet and svr perform poorly
(unable to learn non-linear mapping between the features and the period)

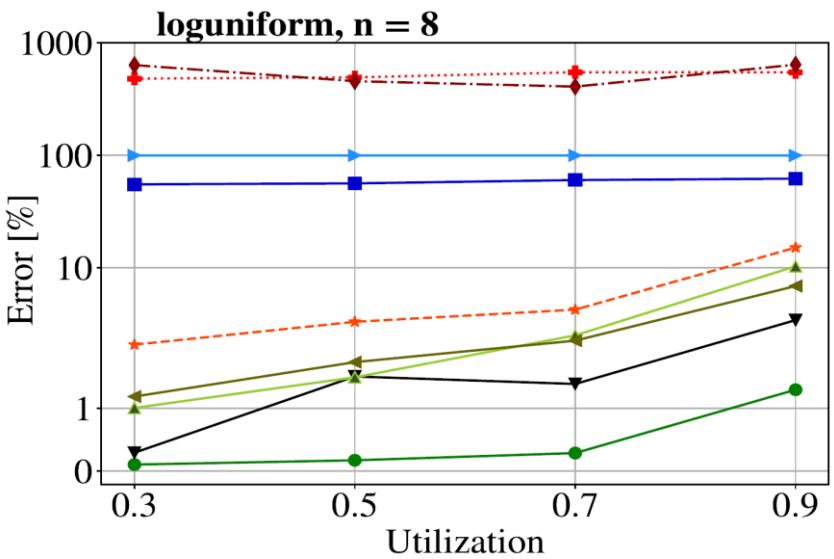
PeTaMi: O. Iegorov, R. Torres, and S. Fischmeister. "Periodic task mining in embedded system traces," RTAS, 2017.

Impact of system utilization

RPM







Tree-based solutions have high accuracy

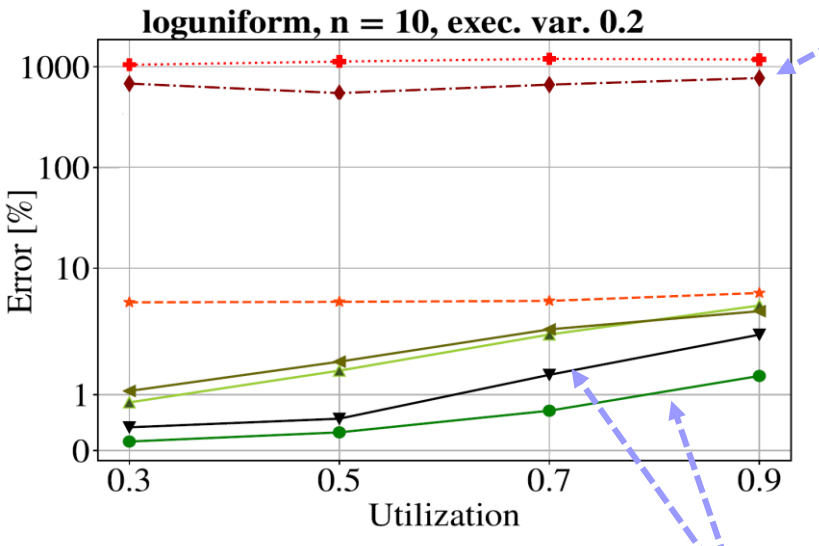
PeTaMi: O. Iegorov, R. Torres, and S. Fischmeister. “Periodic task mining in embedded system traces,” RTAS, 2017.

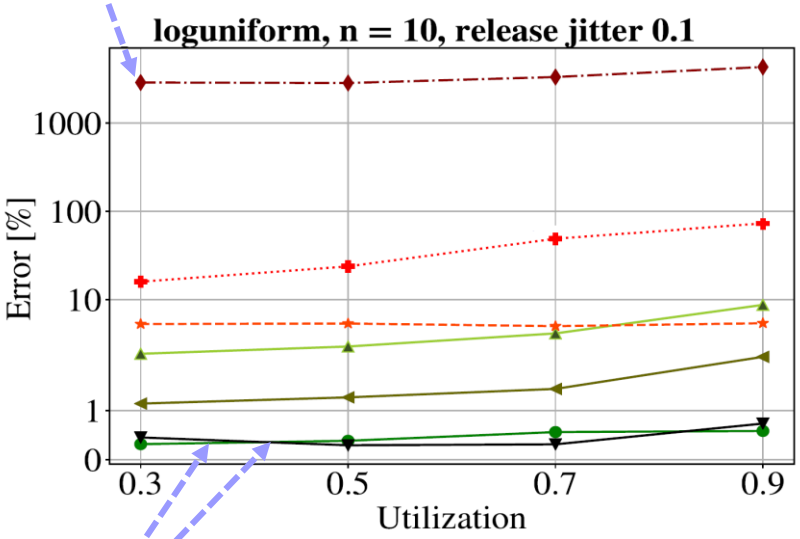
Robustness to variable execution time and release jitter

RPM

extraTrees | cubist | gbm | bartMachine | periodogram | autocorrelation | PeTaMi

PeTaMi is heavily affected by **execution time variation** and **release jitter**





cubist and extraTrees are the most robust and most accurate solutions

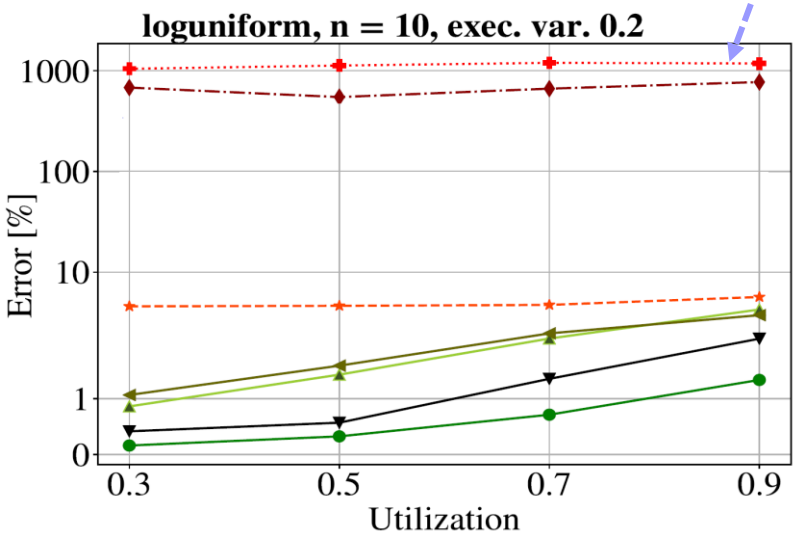
Execution time ∈ [(1 - exec. var.) × WCET, WCET]

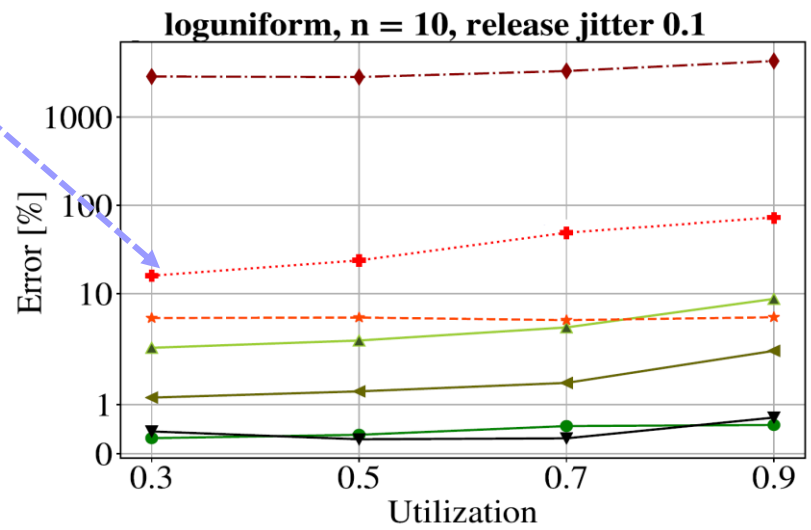
Robustness to variable execution time and release jitter

RPM

extraTrees | cubist | gbm | bartMachine | periodogram | autocorrelation | PeTaMi

Execution-time variation has a more negative impact on autocorrelation than **release jitter**



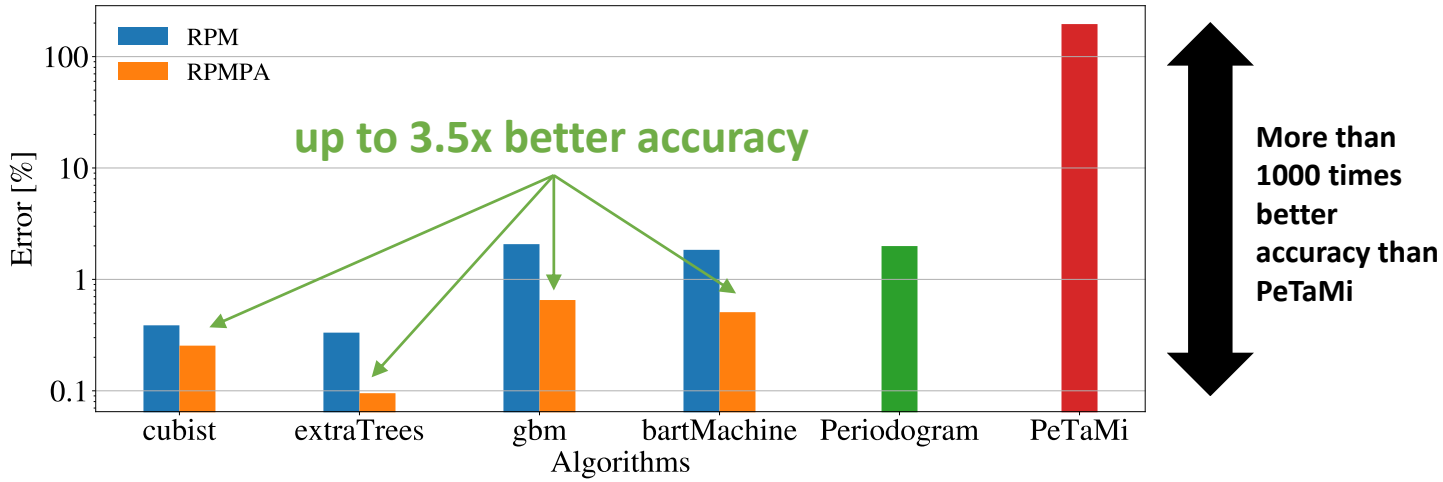


Execution time ∈ [(1 - exec. var.) × WCET, WCET]

Robustness in the presence of high-priority aperiodic tasks

- 12 automotive tasks
  - 6 periodic
  - 6 sporadic
- Aperiodic jobs arrive according to a Poisson distribution. They preempt any of the 12 tasks.
- Here, the task under analysis has a <u>medium</u> priority

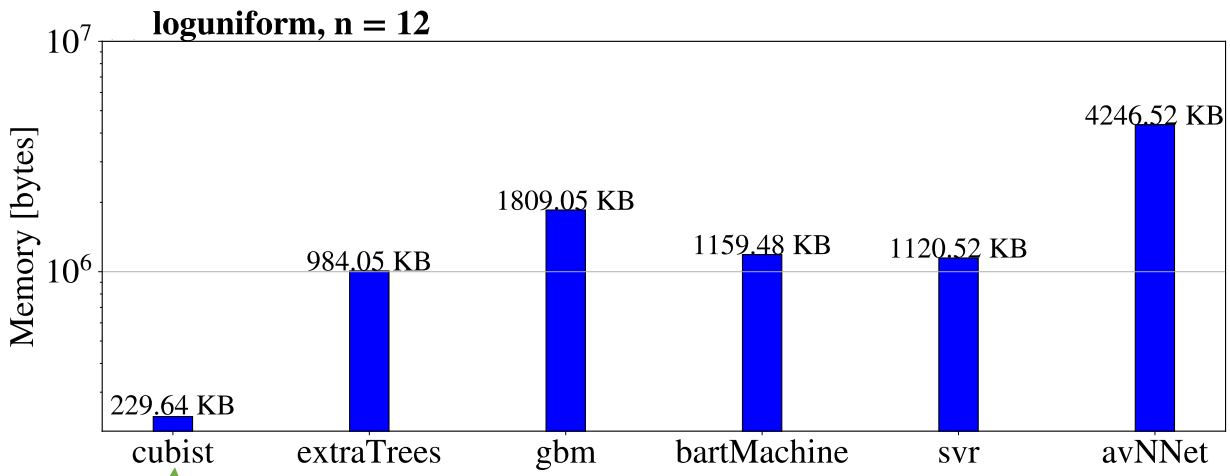
RPMPA reduces the error for all algorithms



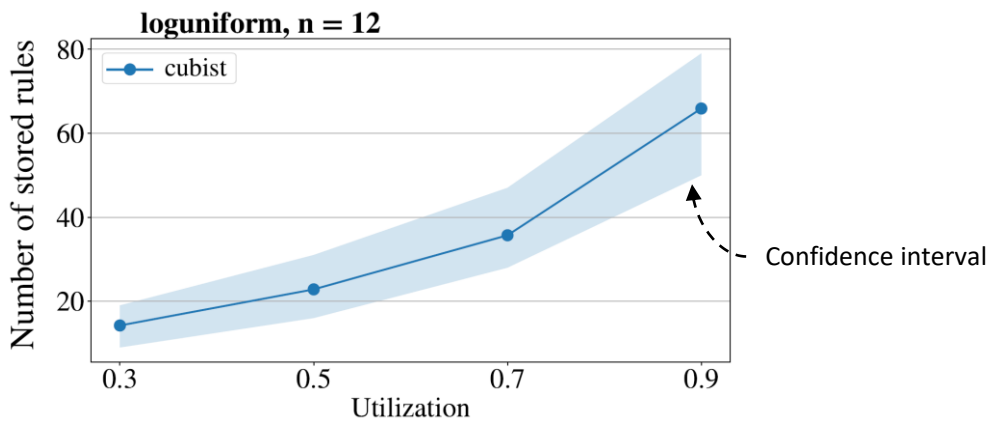
RPM – regression-based period miner

RPMPA – regression-based period miner with <u>period adjustment</u>

Memory comparison



4x less memory consumption than extraTrees



What is the performance on real data?

Two datasets containing controller area network (CAN) messages obtained from vehicles

Data set	Algorithm	RPM [%]	RPMPA [%]
Car hacking	**cubist**	**3.2461**	**0.9703**
	extraTrees	28.2568	1.6179
	gbm	15.8224	1.3919
	bartMachine	20.8588	14.0103
	PeTaMi	222.9416	-
	Periodogram	5.3368	-
CAN intrusion	**cubist**	**5.5005**	2.8963
	extraTrees	13.0256	**1.7039**
	gbm	19.8341	9.8881
	bartMachine	14.956	5.0264
	PeTaMi	177.6681	-
	Periodogram	9.5448	-

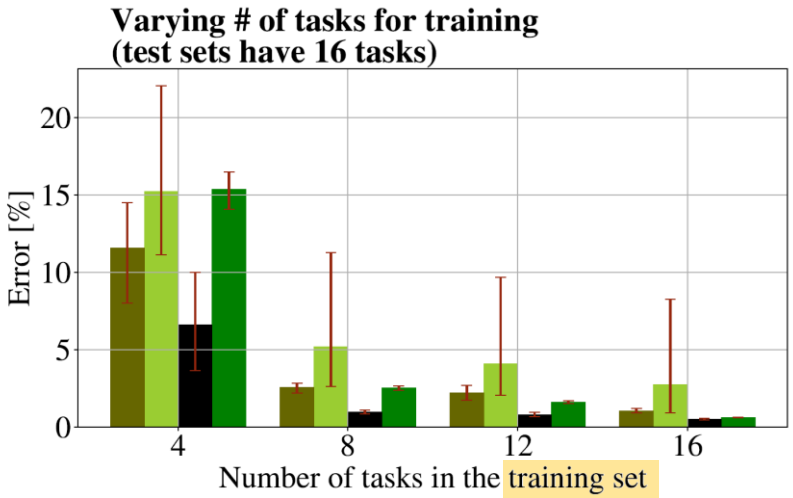
cubist has the **smallest** error

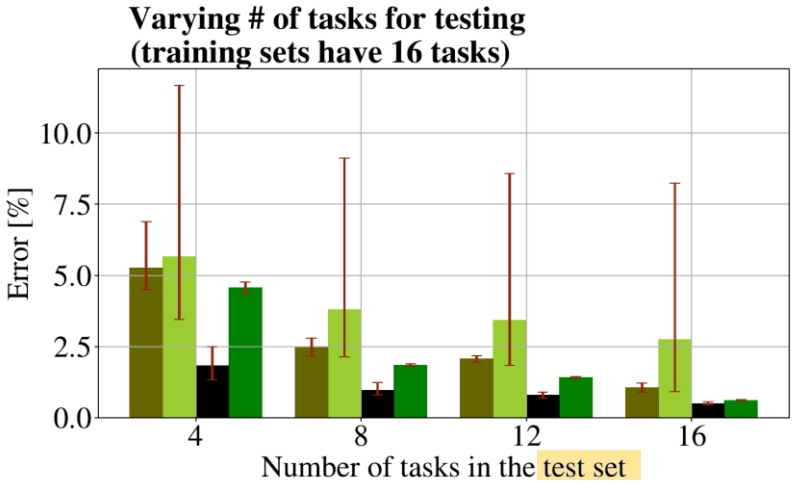
We have a lower error than the state of the art

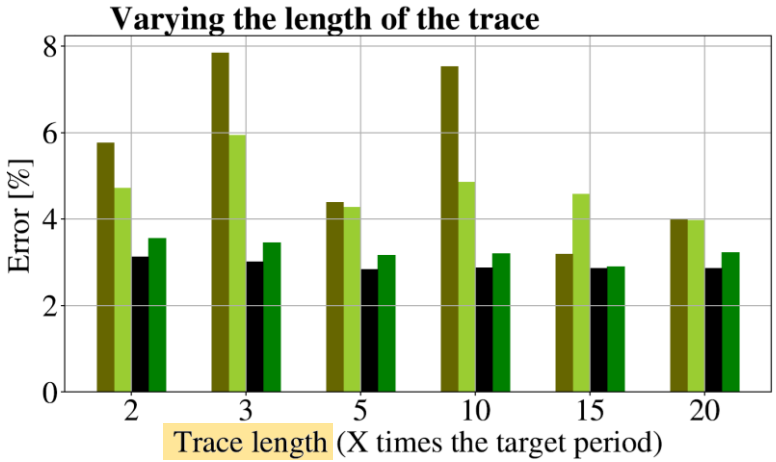
RPMPA significantly reduces the errors

Hacking and Countermeasure Research Lab.
http://ocslab.hksecurity.net/Datasets/

Generalizability (robustness of learning)







Cubist and **extraTrees** maintain their accuracy in all these scenarios

Conclusions

We showed how to use **regression-based machine learning (RBML)** for the problem of period inference

Our framework reduces the **error** of period inference by **2 to 3 orders of magnitude** w.r.t. the state of the art.

And even has a smaller runtime

Our investigation showed that **Cubist regression** has:

- **the lowest** memory requirements;
- **the lowest** runtime;
- **the lowest** error on real traces.
- **It is robust and has a high learning robustness (generalizability)**

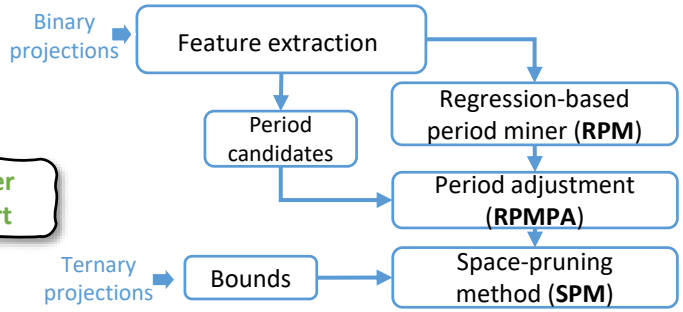
Future work

- Exploring RBML methods to infer **other timing properties** (WCET, precedence constraints, etc.)
- Exploring learning-based methods to **"predict" future events**, e.g., the next time a certain message will appear on a CAN bus.

Thank you

An open-source
period inference framework
based on regression-based techniques

2 to 3 orders of magnitude higher accuracy than the state of the art



A thorough investigation of regression-based methods' performance

- **Accuracy**
- **Robustness** (in the presence of uncertainties)
- **Generalizability** (robustness w.r.t. new datasets)
- **Runtime cost** (overheads and memory)

More information
http://www.es.ele.tue.nl/~m.nasri/

Github
https://github.com/SerbanVadineanu/period_inference