Robust and Accurate Period Inference using Regression-Based Techniques

Serban Vadineanu

Mitra Nasri

TU Delft

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

its specifications
- Respecting deadlines
- Performing periodic actuations
- ...

Realization of the system

A fundamental problem in real-time systems:

“Do the activities that are supposed to be periodic happen periodically?”
Robust and Accurate Period Inference using Regression-Based Techniques

Why inferring period?

Has a few functionalities Being tested Being deployed Too heavy and slow! Still, not fast enough! Too expensive to keep all features Not flexible enough getting too old

Development phase Deployment phase Operation phase with maintenance and upgrades

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
- ...

Partially observable
Written by a third party

Accumulation of uncertainties
Not activated by a timer
Shared timers
Resource interference

Task Hardware

Low priority
\(\tau_4\) \(\tau_6\) \(\tau_5\)
High priority
\(\tau_1\) \(\tau_2\) \(\tau_3\)

OS 1\; \text{OS 2}\; \text{hypervisor}

Written by a third party

Robust and Accurate Period Inference using Regression-Based Techniques
Period inference problem

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

**Schedule**

- medium-priority periodic task with release jitter
- high-priority sporadic task
- high-priority aperiodic task

**Trace**

**Binary projection of τ₃**

**Ternary projection of τ₃**

Robust and Accurate Period Inference using Regression-Based Techniques
Period inference problem

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
- ...

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

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
- Lack of robustness
- Limited scope

**K-nearest neighbors** were used to distinguish tasks from each other based on their power trace [Lamichhane 2018]
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

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

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

A thorough investigation of regression-based methods’ performance
Why extracting features?

Binary projections

Regression-based machine learning

Period

Should projections be cut to a fixed length?

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

High dimensional data

Variable-length input

low efficiency and large runtime during model learning

Most machine-learning methods require a fixed input size
Why extracting features?

Binary projections → Feature extraction → Small-size high-quality features for machine learning → Period

Regression-based machine learning

Features for machine learning:
- Small-size
- High-quality
**Feature extraction**

**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

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

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)
Robust and Accurate Period Inference using Regression-Based Techniques

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.

**Sweet spot**
Robust and Accurate Period Inference using Regression-Based Techniques

Our solution in a nutshell

- Feature extraction
  - Periodogram
  - Autocorrelation

- Regression-based period miner (RPM)
  - Period candidates

- Period adjustment (RPMPA)

- Bounds

- Space-pruning method (SPM)

Binary projections

Ternary projections
Robust and Accurate Period Inference using Regression-Based Techniques

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?


6 best families of regression techniques

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Nickname</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cubist Regression [24]–[26]</td>
<td>cubist</td>
<td>Rule-based</td>
</tr>
<tr>
<td>Generalized Boosting Regression [27]</td>
<td>gbm</td>
<td>Boosting</td>
</tr>
<tr>
<td>Averaged Neural Network [28]</td>
<td>avNNet</td>
<td>Neural Networks</td>
</tr>
<tr>
<td>Extremely Randomized Regression Trees [29]</td>
<td>extraTrees</td>
<td>Random Forests</td>
</tr>
<tr>
<td>Bayesian Additive Regression Tree [30]</td>
<td>bartMachine</td>
<td>Bayesian Models</td>
</tr>
<tr>
<td>Support Vector Regression [31]</td>
<td>svr</td>
<td>Support Vector Machines</td>
</tr>
</tbody>
</table>
Which regression algorithms?

**Tree-based algorithms**

- Cubist regression
- Extremely randomized regression trees (extraTrees)
- Gradient boosting method (gbm)
- Bayesian additive regression trees (bartMachine)

**Support vector regression (svr)**
- ExtraTrees [Geurts 2006]
- gbm [Friedman 2002]

**Averaged neural networks (avNNNet)**
- bartMachine [Chipman 2010]
- avNNNet [Ripley 2007]
- Svr [Cortes 1995]
What is a regression tree?

P1, P2, P3 – top 3 periods from periodogram
A1, A2, A3 – top 3 periods from autocorrelation
What is a regression tree?

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

Robust and Accurate Period Inference using Regression-Based Techniques
**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

---

ExtraTrees [Geurts 2006]
gbm [Friedman 2002]
bartMachine [Chipman 2010]
avNNNet [Ripley 2007]
Svr [Cortes 1995]
Robust and Accurate Period Inference using Regression-Based Techniques

**Observations**

- autocorrelation candidates
- periodogram candidates

Often close to the real period

Not always accurate

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.

RPM – regression-based period miner
RPMPA – regression-based period miner with period adjustment
What can go wrong?

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

May choose a **wrong** candidate

---

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 $\infty$. 
Lower bound derivation

\[ \Delta: \text{the longest interval of zeros in the binary projection} \]

\[ \text{period} > \frac{\Delta}{2} \]

\[ \text{period} = 8 \]
\[ \text{WCET} = 2 \]
\[ \text{Implicit deadline} \]
Two jobs of the task must have been released during [7, 8] and [15, 18]

Period ≤ B

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

Upper bound derivation
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 period adjustment
UB – upper bound
LB – lower bound
Evaluations
Evaluation questions

Did our solutions improve accuracy?

How do our solutions compare in terms of accuracy?

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

**Synthetic task sets**
- Random periods chosen from [10, 1000] with log-uniform distribution [Emerson 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

<table>
<thead>
<tr>
<th>Accuracy</th>
<th>Robustness against uncertainties</th>
<th>Generalizability of learning (a.k.a. learning robustness)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Period types</td>
<td>Presence of high-priority aperiodic tasks</td>
<td>Impact of projection length on testing’s accuracy</td>
</tr>
<tr>
<td>Number of tasks</td>
<td>Tasks may drop jobs</td>
<td>Impact of training and testing on different task set types</td>
</tr>
<tr>
<td>System utilization</td>
<td></td>
<td>Impact of training and testing on different task set sizes</td>
</tr>
<tr>
<td>Release jitter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Execution time variation</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Average estimation error

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

- $\hat{T}_i$: Estimated period
- $T_i$: Actual periods
- $N$: Number of tasks
Impact of system utilization

120x increase in error for PeTaMi and autocorrelation

Impact of system utilization

Robust and Accurate Period Inference using Regression-Based Techniques

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

Impact of system utilization

Tree-based solutions have high accuracy

Robustness to variable execution time and release jitter

Robust and Accurate Period Inference using Regression-Based Techniques

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

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 medium priority

RPMPA reduces the error for all algorithms

More than 1000 times better accuracy than PeTaMi

**up to 3.5x better accuracy**

RPM – regression-based period miner

RPMPA – regression-based period miner with period adjustment
Memory comparison

-loguniform, n = 12-

- 229.64 KB for cubist
- 984.05 KB for extraTrees
- 1809.05 KB for gbm
- 1159.48 KB for bartMachine
- 1120.52 KB for svr
- 4246.52 KB for avNNet

4x less memory consumption than extraTrees

-loguniform, n = 12-

- Number of stored rules vs. Utilization
- Confidence interval

Robust and Accurate Period Inference using Regression-Based Techniques
What is the performance on real data?

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

<table>
<thead>
<tr>
<th>Data set</th>
<th>Algorithm</th>
<th>RPM [%]</th>
<th>RPMPA [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Car hacking</td>
<td>cubist</td>
<td>3.2461</td>
<td>0.9703</td>
</tr>
<tr>
<td></td>
<td>extraTrees</td>
<td>28.2568</td>
<td>1.6179</td>
</tr>
<tr>
<td></td>
<td>gbm</td>
<td>15.8224</td>
<td>1.3919</td>
</tr>
<tr>
<td></td>
<td>bartMachine</td>
<td>20.8588</td>
<td>14.0103</td>
</tr>
<tr>
<td></td>
<td>PeTaMi</td>
<td>222.9416</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Periodogram</td>
<td>5.3368</td>
<td>-</td>
</tr>
<tr>
<td>CAN intrusion</td>
<td>cubist</td>
<td>5.5005</td>
<td>2.8963</td>
</tr>
<tr>
<td></td>
<td>extraTrees</td>
<td>13.0256</td>
<td>1.7039</td>
</tr>
<tr>
<td></td>
<td>gbm</td>
<td>19.8341</td>
<td>9.8881</td>
</tr>
<tr>
<td></td>
<td>bartMachine</td>
<td>14.956</td>
<td>5.0264</td>
</tr>
<tr>
<td></td>
<td>PeTaMi</td>
<td>177.6681</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Periodogram</td>
<td>9.5448</td>
<td>-</td>
</tr>
</tbody>
</table>

-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)

- **extraTrees**
- **cubist**
- **gbm**
- **bartMachine**

**Varying # of tasks for training**
(test sets have 16 tasks)

**Varying # of tasks for testing**
(training sets have 16 tasks)

- **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.

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.
Robust and Accurate Period Inference using Regression-Based Techniques

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

Thank you