The DEBS 2017 Grand Challenge

Vincenzo Gulisano
Chalmers University of Technology
Hörsalsvägen 11
Gothenburg 41296, Sweden
vincenzo.gulisano@chalmers.se

Zbigniew Jerzak
SAP SE
Münzstraße 15
Berlin 10178, Germany
zbigniew.jerzak@sap.com

Roman Katerinenko
AGT International
Hilpertstrasse 35
Darmstadt 64295, Germany
rkaterinenko@agtinternational.com

Martin Strohbach
AGT International
Hilpertstrasse 35
Darmstadt 64295, Germany
mstrohbach@agtinternational.com

Holger Ziekow
Hochschule Furtwangen
Robert-Gerwig-Platz 1
Furtwangen 78120, Germany
zie@hs-furtwangen.de

ABSTRACT

The ACM DEBS 2017 Grand Challenge is the seventh in a series of challenges which seek to provide a common ground and evaluation criteria for a competition aimed at both research and industrial event-based systems. The focus of the 2017 Grand Challenge is on the analysis of the RDF streaming data generated by digital and analogue sensors embedded within manufacturing equipment. The analysis aims at the detection of anomalies in the behavior of such manufacturing equipment. This paper describes the specifics of the data streams and continuous queries that define the DEBS 2017 Grand Challenge. It also describes the benchmarking platform that supports testing of corresponding solutions.

CCS CONCEPTS

• **General and reference → Performance**; • **Information systems → Data streams**;

KEYWORDS

event processing, streaming, manufacturing

ACM Reference format:
Vincenzo Gulisano, Zbigniew Jerzak, Roman Katerinenko, Martin Strohbach, and Holger Ziekow. 2017. The DEBS 2017 Grand Challenge. In *Proceedings of 11th ACM International Conference on Distributed and Event-Based Systems, Barcelona, Spain, June 19-23 (DEBS '17),* 3 pages.
DOI: 10.1145/3093742.3096342

1 INTRODUCTION

The ACM DEBS 2017 Grand Challenge is the seventh in a series [1–5] of challenges which seek to provide a common ground and evaluation criteria for a competition aimed at both research and industrial event-based systems.

The focus of the 2017 Grand Challenge is on the analysis of the RDF streaming data generated by digital and analogue sensors embedded within manufacturing equipment. The analysis aims at the detection of anomalies in the behavior of such manufacturing equipment. In order to detect anomalies, the data produced by each sensor deployed in the manufacturing equipment is clustered and the state transitions between the observed clusters are modeled as a Markov chain. Based on this classification, anomalies are detected as sequences of transitions that happen with a probability lower than a given threshold. The challenge is co-organized by the HOBBIT (https://project-hobbit.eu/) project represented by AGT International (http://www.agtinternational.com/). Both the data set (presented in Section 2) and the automated evaluation platform (described in Section 4) are provided by the HOBBIT project.

2 DATA

The data stream for the 2017 Grand Challenge mimics sensor measurements and setting parameters from injection molding machines. Injection molding machines are equipped with sensors that measure various parameters of a production process: distance, pressure, time, frequency, volume, temperature, time, speed and force. All the measurements are taken at a certain point in time resulting in a 120 dimensional vector consisting of values of different types (e.g. text or numerical values).

All measurements are timestamped and provided as RDF triples or more precisely as instances of an OWL ontology that defines the semantics of the data. The ontology file is available via the HOBBIT CKAN site[1]. Measurements are simulated based on a real data set provided by Weidmüller[2]. In order to generate realistic data we have developed a data generator that is based on a model of the real data. This way we preserve the confidentiality of the data while at the same time being able to provide measurements virtually at any scale and velocity.

The example below shows a sample measurement in Turtle format to increase readability. Please note that the actual data is provided in N-Triples format.

```
debs:ObservationGroup_1
    a  i40:MoldingMachineObservationGroup ;
    ssn:observationResultTime
        debs:Timestamp_1 ;
    i40:contains  debs:Observation_1 ;
```

DEBS '17, Barcelona, Spain

978-1-4503-5065-5/17/06…$15.00
DOI: 10.1145/3093742.3096342

[1] https://ckan.project-hobbit.eu/dataset/debs-grand-challenge-2017
[2] http://www.weidmueller.de/

```
    i40:machine wmm:MoldingMachine_1;
    i40:observedCycle debs:Cycle_2.
debs:Cycle_2
    a i40:Cycle;
    IoTCore:valueLiteral "2"^^xsd:int.
debs:Timestamp_1
    a IoTCore:Timestamp;
    IoTCore:valueLiteral
        "2016-07-18T23:59:58"^^xsd:dateTime
            .
debs:Observation_1
      a i40:MoldingMachineObservation;
    ssn:observationResult debs:Output_2;
    ssn:observedProperty wmm:_9.
debs:Output_2
    a ssn:SensorOutput;
    ssn:hasValue debs:Value_2.
debs:Value_2
      a i40:NumberValue;
    IoTCore:valueLiteral
        "-0.01"^^xsd:float.
```

In addition to the measurements we provide metadata that includes information about the machine type, the number of sensors per machine and the number of clusters that must be used in order to detect anomalies in the data.

3 QUERY

The DEBS Grand Challenge addresses the problem of anomaly detection in machine data. For this task we define an anomaly detection mechanism based on Markov models. The intuition behind the mechanism is to build Markov models that reflect normal operations of a given machine. Incoming event sequences are checked against the models to determine the probability of their occurrence. The mechanisms considers event sequences as anomalies if they have - according to the model - a low probability of occurrence. Grand challenge participants had to implement the mechanism according to the details below.

Overall the anomaly detection comprises three steps: (1) finding clusters, (2) training a Markov model, and (3) finding anomalies. Machine data have multiple dimensions and the three steps must be executed for each dimension separately. Finding clusters is a preprocessing step for mapping event values to discrete states. This is a prerequisite for training a Markov model in step two, that reflects transition probabilities between the observed states. Together, step 1 and 2 create a model for anomaly detection. A requirement of the challenge is to only consider the last W events in building the model. This is to account for concept drift in the data and results in continuous model updates.

Step three uses the model to compute the probability of observing the last N received events. The mechanism reports an anomaly if the resulting value is below a given threshold.

Figure 1 provides an overview of the three described query steps. Once started, the activities for each step are executed continuously and never stop. This means that a changed cluster center must be considered in the subsequent steps right after the centers have changed. An event that causes a change of a cluster center first causes the update of the centers, then an update of the Markov model and is finally used in anomaly detection. Further specifics of the three query steps are described below.

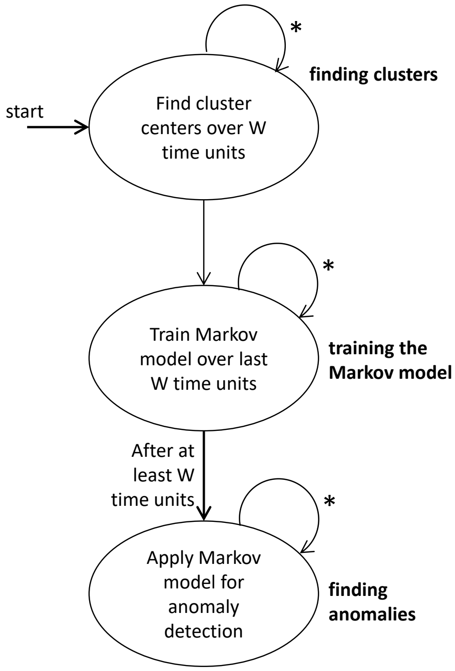
start
Find cluster centers over W time units
*
finding clusters
Train Markov model over last W time units
*
training the Markov model
After at least W time units
Apply Markov model for anomaly detection
*
finding anomalies

Figure 1: Query States

3.1 Finding Clusters

For each stateful dimension, one needs to find and maintain up to K cluster centers, using the numbers 1 to K as seeds for the initial K centroids. The number K is defined in the metadata for each dimension of each individual machine. It is required to use all measurements in the last W time units to find the cluster centers.

The initial cluster centers for each dimension of measurements are determined by the first K distinct values for that dimension in the stream. When recomputing the clusters after shifting the time window, the cluster centers are determined by the first K distinct values for that dimension in the given window.

If a given window has less than K distinct values, then the number of clusters to be computed must be equal to the number of distinct values in the window. If a data point has the exact same distance to more than one cluster center, it must be associated with the cluster that has the highest center value. The algorithm must compute M (e.g.: 50) iterations to find a clustering, unless it terminates earlier.

3.2 Training the Markov Model

To build the Markov model, one needs to determine the transition probabilities by maintaining the count of transitions between all states in the last W time units. For determining a transition at time t, one must use the cluster centers that are valid at time t, i.e., no remapping of past observations to clusters in retrospect is required. The current state that was reached prior to t, does not need to be reevaluated at t. No two tuples for the same dimension have the same time stamp.

3.3 Finding Anomalies

One needs to output an alert about a machine, if any sequence of up to N state transitions for that machine is observed, that has a probability below Td.

3.4 Continuous Processing

Time is always defined as application time, i.e., as given by the timestamp of arriving tuples. Each new event is (1) first used to update the cluster centers, (2) then to update the Markov model, and (3) to compute the probability of the last up to N state transitions.

3.5 Parameters

All grand challenge solutions must be able to accommodate the following parameters:

- W: window size for finding cluster centers with k-means clustering and for training transition probabilities in the Markov model.
- N: number of transitions to be used for the combined state transition probability.
- M: number of maximum iterations for the clustering algorithm.
- Td: the maximum probability for a sequence of N transitions to be considered an anomaly. The value of Td is specified for each dimension d for which the clustering is performed.

4 EVALUATION PLATFORM

The HOBBIT platform is designed to benchmark Linked Data systems on a cluster. It allows for the deployment of several components designed as Docker containers. The components communicate to each other through RabbitMQ communication service. There is platform controller component which orchestrates message passing to benchmarks and a web user interface.

For the DEBS 2017 Grand Challenge, the platform is able to compute metrics, generate data, and evaluate the benchmarked solution output at the same time, in a streaming fashion.

Latency and throughput are the main metrics measured by the benchmark. Besides latency and throughput, the platform also check the correctness of the results. The latency is calculated as the difference between the system clock time when the anomaly produced by the solution is received in the benchmark and the system clock time when the last contributing input tuple is sent to the solution. In order to identify the last contributing tuple, the benchmark computes anomalies itself ("gold standard") and matches them against the solution's ones. Throughput is defined as the total amount of bytes processed by the solution divided by total processing time.

Every solution benchmarked by the Hobbit platform is uploaded together with an adapter. The latter is design for the solution to interact with the benchmark. More concretely, the role of the adapter is to interact with RabbitMQ by reading input data and sending anomalies.

The solution adapter might be part of the same Docker container of the solution or located in a different container. Since the solution adapter can create additional containers, both variants are supported by the Hobbit platform. Note that this functionality allows to evaluate distributed systems, as done in the challenge.

5 ADDITIONAL REMARKS

The evaluation of each submitted solution is performed by the evaluation platform. For each solution, the evaluation platform keeps track of the different injection rates that can be sustained by the solution and their respective latency. The score of each solution is proportional to the sum of the sustained rates and inversely proportional to their respective latency, as presented in Equation 1.

$$\text{score} = \frac{\text{rate}_1}{\text{latency}_1} + \frac{\text{rate}_2}{\text{latency}_2} + \dots \qquad (1)$$

ACKNOWLEDGMENTS

This year's Grand Challenge is co-organized by the HOBBIT project (https://project-hobbit.eu/) represented by AGT International (http://www.agtinternational.com/). Both the data set and the automated evaluation platform are provided by the HOBBIT project.

We would like to thank Weidmüller (http://www.weidmueller.de/) for the provisioning of the original data set that AGT International used to generate a realistic data set for the purpose of this challenge. The HOBBIT project has received funding from the European Union's H2020 research and innovation action program under grant agreement number 688227.

REFERENCES

[1] Vincenzo Gulisano, Zbigniew Jerzak, Spyros Voulgaris, and Holger Ziekow. 2016. The DEBS 2016 Grand Challenge. In *Proceedings of the 10th ACM International Conference on Distributed and Event-based Systems (DEBS '16)*. ACM, New York, NY, USA, 289–292. DOI: https://doi.org/10.1145/2933267.2933519

[2] Zbigniew Jerzak, Thomas Heinze, Matthias Fehr, Daniel Gröber, Raik Hartung, and Nenad Stojanovic. 2012. The DEBS 2012 grand challenge. In *Proceedings of the Sixth ACM International Conference on Distributed Event-Based Systems, DEBS 2012, Berlin, Germany, July 16-20, 2012*, François Bry, Adrian Paschke, Patrick Th. Eugster, Christof Fetzer, and Andreas Behrend (Eds.). ACM, 393–398. DOI: https://doi.org/10.1145/2335484.2335536

[3] Zbigniew Jerzak and Holger Ziekow. 2014. The DEBS 2014 grand challenge. In *The 8th ACM International Conference on Distributed Event-Based Systems, DEBS '14, Mumbai, India, May 26-29, 2014*, Umesh Bellur and Ravi Kothari (Eds.). ACM, 266–269. DOI: https://doi.org/10.1145/2611286.2611333

[4] Zbigniew Jerzak and Holger Ziekow. 2015. The DEBS 2015 Grand Challenge. In *Proceedings of the 9th ACM International Conference on Distributed Event-Based Systems, DEBS '15, Oslo, Norway, June 29 - July 3, 2015*, Frank Eliassen and Roman Vitenberg (Eds.). ACM, 266–268. DOI: https://doi.org/10.1145/2675743.2772598

[5] Christopher Mutschler, Holger Ziekow, and Zbigniew Jerzak. 2013. The DEBS 2013 grand challenge. In *The 7th ACM International Conference on Distributed Event-Based Systems, DEBS '13, Arlington, TX, USA - June 29 - July 03, 2013*, Sharma Chakravarthy, Susan Darling Urban, Peter Pietzuch, and Elke A. Rundensteiner (Eds.). ACM, 289–294. DOI: https://doi.org/10.1145/2488222.2488283