MANTIS

Cyber Physical System based Proactive Collaborative Maintenance

D1.2 Consolidated State-of-the-Art of Sensor-based Proactive Maintenance Appendix 14: High-performance stream-based processing

Work Package	WP1 - Service platform architecture requirement definition. Scenarios and use cases descriptions
Version	1.0
Contractual Date of Delivery	30/04/2016
Actual Date of Delivery	03/06/2016
Dissemination Level	Public
Responsible	Erkki Jantunen
Contributors	Tom Tourwé (SIR), Elena Tsiporkova (SIR), Mathias Verbeke (SIR), Tom Ruette (SIR), Koen Verschaeren (ATLAS), Tom Maris (ILIAS), Istvan Moldovan (BME), Pal Varga (AITIA)

The MANTIS consortium consists of:

Num.	Short Name	Legal Name	Role	Country
1	MGEP	Mondragon Goi Eskola Politeknikoa J.M.A. S.Coop.	CO	ES
2	MONDRAGON	Mondragon Corporacion Cooperativa S.Coop.	BEN	ES
3	IKERLAN	Ikerlan S.Coop.	BEN	ES
4	TEKNIKER	Fundacion Tekniker	BEN	ES
5	FARR	Fagor Arrasate S.Coop.	BEN	ES
5.1	KONIKER	Koniker S.Coop.	TP	ES
6	GOIZPER	Goizper S.Coop.	BEN	ES
7	ACCIONA	Acciona Infraestructuras S.A.	BEN	ES
8	MSI	Mondragon Sistemas De Informacion S.Coop.	BEN	ES
9	VTT	Teknologian Tutkimuskeskus VTT Oy	BEN	FI
10	LUAS	Lapin Ammattikorkeakoulu Oy	BEN	FI
11	NOME	Nome Oy	BEN	FI
12	FORTUM	Fortum Power And Heat Oy	BEN	FI
13	SQ	Solteq Oyj	BEN	FI
14	WAPICE	Wapice Oy	BEN	FI
15	AAU	Aalborg Universitet	BEN	DK
16	DANFOSS	Danfoss A/S	BEN	DK
17	UNIV	Universal Foundation A/S	BEN	DK
18	HGE	Hg Electric A/S	BEN	DK
19	VESTAS	Vestas Wind Systems A/S	BEN	DK
20	SIRRIS	Sirris Het Collectief Centrum Van De Technologische Industrie	BEN	BE
21	ILIAS	Ilias Solutions Nv	BEN	BE
22	ATLAS	Atlas Copco Airpower Nv	BEN	BE
23	3E	3e Nv	BEN	BE
24	PCL	Philips Consumer Lifestyle B.V.	BEN	NL
25	PHC	Philips Medical Systems Nederland B.V.	BEN	NL
26	PHILIPS	Philips Electronics Nederland B.V.	BEN	NL
27	S&T	Science and Technology B.V.	BEN	NL
28	TU/E	Technische Universiteit Eindhoven	BEN	NL
29	RUG	Rijksuniversiteit Groningen	BEN	NL
30	UNINOVA	UNINOVA - Instituto de Desenvolvimento de Novas Tecnologias	BEN	PT
31	ISEP	Instituto Superior de Engenharia do Porto	BEN	PT
32	INESC	Instituto de Engenharia de Sistemas e Computadores do Porto	BEN	PT
33	ADIRA	ADIRA - Metal Forming Solutions S.A.	BEN	PT
34	ASTS	Ansaldo STS S.p.A.	BEN	IT
35	CINI	Consorzio Interuniversitario Nazionale per l'Informatica	BEN	IT
36	AIT	Austrial Institute of Technology GmbH	BEN	AT
37	HBM	Hottinger Baldwni Messtechnik GmbH	BEN	AT
38	INNOTEC	Innovative Technology and Science Limited	BEN	UK
39	AITIA	AITIA International Inc.	BEN	HU
40	BME	Budaperst University of Technology and Economics	BEN	HU
41	JSI	Josef Stefan Institute	BEN	SI
42	XLAB	XLAB d.o.o.	BEN	SI
43	FHG	Fraunhofer Institute for Experimental Software Engineering IESE	BEN	DE
44	M2X	M2Xpert GmbH & Co KG	BEN	DE
45	STILL	STILL GMBH	BEN	DE
46	BOSCH	Robert Bosch GMbH	BEN	DE
47	LIEBHERR	Liebherr-Hydraulikbagger GmbH	BEN	DE

Document Revisions & Quality Assurance

Revisions:

Version	Date	By	Overview
0.1	6.8.2015	Mathias Verbeke (SIR)	First draft
0.2	24.8.2015	Koen Verschaeren (ATLAS)	Additions to Section 1
0.3	27.8.2015	Tom Maris (ILIAS)	Added ClearPriority
0.4	30.1.2015	Pal Varga (AITIA)	Conclusion
0.5	6.10.2015	Riku Salokangas	Added contractual date of delivery
0.6	30.03.2016	Tom Ruette (SIR)	Related standards
0.7	01.04.2016	Istv□n Moldov□n (BME)	ITU related standard added
1.0	02/06/2016	Mikel Muxika (MGEP)	Format correction Deliverable info update

Abstract

In MANTIS, computations need to be performed on a continuous unbounded stream of data. In this appendix, we will discuss state-of-the-art approaches for scalable stream-processing. A stream processor has a prominent role in a state-of-the-art IoT architecture as a highly scalable component to route data towards or get data from online learning algorithms, various data stores (e.g., in-memory caches, (distributed) storage for batch processing/archiving, non-production environments, etc.), and the data integration layer (e.g., lookup technical parameters, push/read from ERP/CRM systems, etc.). Stream processors also excel at basic on-the-fly filtering, pattern detection and data aggregation.

Online (incremental) learning allows to learn from these data streams as they come in, which has the advantage that the models can output a hypothesis at any time during processing, instead of only after all data is processed. However, over time some of the assumptions underlying machine learning theory can be violated, which is referred to as concept drift. A number of practical approaches to deal with this will be discussed as well.

Table of Contents

1 Scalable stream-processing

Because of the nature of the MANTIS concept and the pilots that are envisioned, large volumes of (sensor) data will be generated and computations and results are needed as soon as the data comes in (near-real-time), or computations need to be performed on a continuous unbounded stream of data. Different approaches exist for such situations, the two most relevant ones being general-purpose stream-based processing systems and dedicated complex event processing platforms.

General-purpose stream-based processing systems allow dealing with potentially infinite volumes of data streams that flow in and out of a computer system continuously and with varying update rates. Incoming data needs to be processed immediately or it is lost forever, and single-scan, on-line algorithms and analysis methods are required to avoid scanning through the data multiple times. Several open-source platforms are currently under active development, e.g.:

- Apache Samza (http://samza.apache.org) is a distributed stream processing framework. It uses Apache Kafka for messaging, and Apache Hadoop YARN to provide fault tolerance, processor isolation, security, and resource management.
- Storm (https://storm.apache.org/) is a scalable, fault-tolerant and distributed realtime computation system that makes it easy to reliably process unbounded streams of data. Storm allows developers to create topologies, i.e. graphs of computation, where each node contains processing logic, and links between nodes indicate how data should be passed around between nodes. Nodes can be spouts or bolts, where the former is a source of streams and the latter consumes streams, performs processing and possibly emits new streams.
- Spark Streaming (https://spark.apache.org/streaming/) is an extension of Spark that enables scalable, high-throughput, fault-tolerant stream processing of data streams. It provides a high-level abstraction called discretized stream or DStream, which represents a continuous stream of data, that is internally divided into batches which are processed by the Spark engine to generate the final stream of results. High-level operations on Dstreams are supported, such as filter, map, reduce, join, etc.

Stream data typically arrives in a rather low level of abstraction (e.g. sensor data), whereas most applications are interested in high-level knowledge and patterns, such as trends and deviations. At the most basic level, platforms support this through the use of queries, which can be either one-time or continuous queries. A one-time query is evaluated once over a point-in-time snapshot of the data stream, with the answer returned immediately. A continuous query is evaluated continuously as data streams continue to arrive. The answer to a continuous query is produced over time, always reflecting the stream data seen so far. Higher-level support is offered by Spark Streaming through MLlib (https://spark.apache.org/mllib/), a scalable machine learning library that implements well-known and often-used algorithms for clustering, classification, recommendation and prediction tuned to work on stream data.

In contrast, Rule Engines, Complex Event Processing (CEP), and Streaming Analytics platforms exist that combine data from multiple sources [1] and offer higher-level and dedicated support for inferring patterns that suggest more complicated circumstances, aiming to identify meaningful events (e.g. anomalies, threats, etc.) within an event cloud. The CEP Market Player Survey [22] provides an overview of this market [22].

They often offer a domain-specific language, GUI or SQL-like language that allows a developer to specify evaluation conditions or constraints over an event set and express event correlation and abstraction, event hierarchies, and relationships between events such as causality, membership, and timing, and abstracting event-driven processes [2]. These systems also need to be highly available, scale with the amount of data coming in, be able to parallelize their processing and make use of distributed processing power in order to provide near real time results. A variety of tools and engines have emerged in recent years, among which the most originate from:

- Drools Export/Fusion (http://www.drools.org/), an open-source business rule engine and complex event solution able to understand and handle events or stream of events in a cloud, detect relevant patterns and take appropriate actions based on the patterns detected.
- StreamBase CEP (http://www.streambase.com/products/streambasecep), a platform that performs rapid development via a graphical event-flow language, providing also broad connectivity to real-time and historical data.
- Oracle Event Processing (http://www.oracle.com/technetwork/middleware/complex-event-processing/overview/index.html) which is built on industry standards and provides an open architecture for sourcing, processing, and publishing complex events.
- Esper (Java based) and NEsper (.NET based) (http://esper.codehaus.org/) open source development library that enable rapid development of applications that process large volumes of incoming events in real-time.
- SAP□s SYBASE Event Stream Processor (http://www.sybase.com/products/financialservicessolutions/complex-event-processing), a market-leading platform that delivers continuous intelligence for fast and intelligent decision-making.
- Skyline (https://github.com/etsy/skyline) is a dedicated anomaly detection platform that takes an alternative approach for continuously monitoring time series data and automatically detecting anomalies. By design, it relies upon a limited set of different predefined algorithms that should reach consensus about what is an anomaly. User-defined algorithms can be integrated as well, if needed.
- Software AG Apama (https://www.softwareag.com/corporate/products/apama˙webmethods/analytics/overview/default.asp) The Apama platform rapidly correlates, aggregrates and detects patterns across large volumes of fast-moving data. Apama offers a mature domain-specific language (EPL) and GUI (Queries), native Matlab integration and various connectors towards message buses and (in-memory) storage solutions.
- SQL Stream (http://www.sqlstream.com/) is a real-time data hub for operational intelligence and IoT that is built on a distributed SQL stream processing engine.
- ClearPriority Intelligence (http://www.clearpriority.com/) is a real-time Operational Intelligence Platform that empowers business users to analyse and anticipate critical situations in real-time, so that they can act on opportunities, threats and problems as they occur. It consolidates risk-related data from multiple network, platform and application sources. Near real-time data extraction, transformation and correlation capabilities let you keep tabs on operational activity. Also, it allows you to rapidly define new sources and rules, and implement controls in the form of real-time alerts and Key Performance Indicators (KPIs) without IT intervention. ClearPriority□s Development Studio reduces the effort needed to create new monitoring applications. An intuitive, event-based graphical notation lets you express business and risk situations in a natural and straightforward manner.

The Forrester Research Big Data Streaming Analytics Platforms Q3 2014 report currently mentions Software AG, IBM, SAP, Tibco and Informatica as leading solution providers [15].

The uptake of stream-based processing technology in the predictive analytics domain has been rather limited, mainly due to the fact that the major focus in recent years was on deploying and integrating the required hardware and software infrastructures for capturing and storing the huge volumes of generated data.

At Atlas Copco, several multiple business rules / CEP / Streaming Analytics platforms were reviewed, resulting in the observation that the open-source solutions are still very development / technical driven solutions, as is also mentioned by Forester Research [15]. This can be an advantage or disadvantage depending on your specific business case, budget and skills within your team. The tool selection also needs to take into account the actual and future use cases, e.g., to distinguish between the need for stream processing versus tools with a micro-batch or decision tree approach [21].

An advantage of a domain specific language or GUI is that it enables rules and computations to be prototyped and developed by engineers. The presence of efficient temporal operators (time-/location-based patterns and windows) is a major requirement in this respect. Combined with easy-to-use replay functionality and the ability to replicate the real-time stream towards a development environment, this can highly increase the development productivity. Some of the tools also offers connectors, an SDK or plug-in architecture to integrate with R, Matlab, SAS, and message buses without or with only minor custom development.

The more established solutions also claim and prove a higher performance and easier to deploy architecture, resulting in a lower TCO. This is certainly the case if an Hadoop/Spark environment is not part of your IoT architecture [16] [20].

Recently in-memory databases start to challenge the approach taken by commercial and open-source stream processing platforms [17].

The major cloud players continue investing in stream processing solutions-as-a-service :

- Microsoft - Stream Analytics (http://azure.microsoft.com/en-gb/services/stream-analytics/)
- Google - Cloud Dataflow (https://cloud.google.com/dataflow/; [18],[19])
- Amazon - Kinesis (https://aws.amazon.com/kinesis/)

All platforms offer integration with other big data and machine learning services available on the specific cloud platforms and various Hadoop-based frameworks.

2 Online (incremental) learning

Many learning algorithms start from the assumption that the set of training examples is completely known in advance and that a learning algorithm can access these examples in arbitrary order, and arbitrarily many times (batch learning). In the MANTIS setting, however, we have to deal with incremental algorithms that deal with the examples one by one (online or incremental learning) [3]. The main advantage of such algorithms is that they can output a hypothesis at any time during processing, instead of only after all data is processed.

In the last decade, the prevalence of data streams has fuelled interest in the development of online algorithms. In the field of stream mining, the focus lies especially on the high volume/high speed characteristics of data streams and the corresponding trade-off between accuracy and performance [4]. The challenge is that streams are potentially unbounded in size, cannot be stored efficiently, and are subject to evolving concepts. Taking these restrictions into account, Domingos and Hulten [5] proposed a set of requirements that a stream mining algorithm should fulfil: (a) require a small constant time per record, (b) require a fixed amount of memory independent of the records seen so far, (c) use at most one scan of the data, (d) have a usable model at any point in time, (e) produce, ideally, a model that is equivalent to the batch version, and (f) be able to cope with changes in the data generation process (i.e. concept drift).

Much of the work in online learning is focused on translating the batch mining and learning settings to an online setting. For example, there exist streaming pattern mining algorithms [4, 6], decision tree algorithms for classification [4, 5, 7], clustering approaches [4], and many more. Because many problems cannot be solved exactly without allowing multiple passes over the data, many of the techniques are based on sampling, simplification (sketching), summarization and aggregation.

3 Concept drift

There are two assumptions that underlie most of the theory on machine learning and data mining. The first assumption states that examples are drawn (independently) from the instance space according to a fixed distribution (i.i.d.). The second assumption states that the target concept remains fixed. Concept drift occurs when either of these constraints is violated and most of the guarantees offered by computational and statistical learning theory are invalidated.

This problems have been studied in computational learning theory and have led to theoretical insights and practical approaches. Theoretical results include the extent of concept drift to analyze bounds [8] and the impact of the rate of drift [9]. These have led to frameworks like FLORA [10], approaches that are capable of learning under concept drift [11] and can detect changes in the data [12, 13]. A full overview on concept drift adaption can be found in a recent survey by Gama et al. [14].

4 Conclusion

The first set of issues addressed in this document is the scalability of stream-processing. General-purpose stream-based processing systems allow dealing with potentially infinite volumes of data streams that flow in and out of a computer system continuously and with varying update rates. The most current approaches use Big Data analytics, and in-memory databases within cloud architecture. The various approaches and solutions for specific problems are listed in this document.

The second set of issues is regarding online, incremental learning; since the prevalence of data streams has fuelled interest in the development of these approaches. The challenge is that streams are potentially unbounded in size, cannot be stored efficiently, and are subject to evolving concepts. There are various solutions referenced and briefly reviewed in this document that overcome these issues.

Furthermore, the concept drift may be needed due to the fact that some basic assumptions of machine learning are invalid for current applications. Current theoretical and practical results are also referenced in this document.

5 Related standards

- ISO 17789: Information technology -- Cloud computing -- Reference architecture
- ITU-T Recommendation Y.3600 : Big data - Cloud computing based requirements and capabilities

References

[1] Ivy Schmerken: Deciphering the myths around complex event processing. Wall Street and technology (2008)

[2] Opher Etzion: Temporal Perspectives in Event Processing. Principles and Applications of Distributed Event-Based Systems 2010: 75-89

[3] Avrim Blum: On-line algorithms in machine learning. Springer (1998)

[4] Charu C. Aggarwal (Ed.): Data Streams - Models and Algorithms. Advances in Database Systems 31, Springer (2007)

[5] Pedro Domingos, Geoff Hulten: Catching up with the Data: Research Issues in Mining Data Streams. DMKD 2001

[6] Moses Charikar, Kevin Chen, Martin Farach-Colton: Finding Frequent Items in Data Streams. ICALP 2002: 693-703

[7] Geoff Hulten, Laurie Spencer, Pedro Domingos: Mining time-changing data streams. KDD 2001: 97-106

[8] David P. Helmbold, Philip M. Long: Tracking Drifting Concepts By Minimizing Disagreements. Machine Learning 14(1): 27-45 (1994)

[9] Anthony Kuh, Thomas Petsche, Ronald L. Rivest: Learning Time-Varying Concepts. NIPS 1990: 183-189

[10] Gerhard Widmer, Miroslav Kubat: Learning in the Presence of Concept Drift and Hidden Contexts. Machine Learning 23(1): 69-101 (1996)

[11] Geoff Hulten, Laurie Spencer, Pedro Domingos: Mining time-changing data streams. KDD 2001: 97-106

[12] Matthijs van Leeuwen, Arno Siebes: StreamKrimp: Detecting Change in Data Streams. ECML/PKDD (1) 2008: 672-687

[13] Daniel Kifer, Shai Ben-David, Johannes Gehrke: Detecting Change in Data Streams. VLDB 2004: 180-191

[14] Jo□o Gama, Indre Zliobaite, Albert Bifet, Mykola Pechenizkiy, Abdelhamid Bouchachia: A survey on concept drift adaptation. ACM Comput. Surv. 46(4): 44 (2014)

[15] Mike Gualtieri, Rowan Curran : The Forrester Wave: Big Data Streaming Analytics Platforms, Q3 2014. 17/07/2014, Forester Research, Inc.

[16] SQLStream : SQLstream Blaze and Apache Storm - A benchmark comparison, 2015

[17] Dough Henschen : MemSQL, VoltDB vie for In-Memory Big Data Role (http://www.informationweek.com/big-data/big-data-analytics/memsql-voltdb-vie-for-in-memory-big-data-role/d/d-id/1319071)

[18] Tyler Akidau, Alex Balikov, Kaya Bekiroglu, Slava Chernyak, Josh Haberman, ˘ Reuven Lax, Sam McVeety, Daniel Mills, Paul Nordstrom, Sam Whittle: MillWheel: Fault-Tolerant Stream Processing at Internet Scale (http://static.googleusercontent.com/media/research.google.com/en/us/pubs/archive/41378.pdf)

[19] Tyler Akidau, Robert Bradshaw, Craig Chambers, Slava Chernyak, Rafael J. Fernandez-Moctezuma, Reuven Lax, Sam McVeety, Daniel Mills, J Frances Perry, Eric Schmidt, Sam Whittle: The Dataflow Model: A Practical Approach to Balancing Correctness, Latency, and Cost in Massive-Scale, Unbounded, Out-of-Order Data Processing (http://www.vldb.org/pvldb/vol8/p1792-Akidau.pdf)

[20] Check the DEBS (Distributed Event-Based Systems) conference proceedings for the yearly stream processing challenge (http://debs.org/ - http://dl.acm.org/event.cfm?id=RE268&CFID=539786736&CFTOKEN=44420144)

[21] Charles Brett, Mike Gualtieri. Must you choose between business rules and complex event processing platforms. 21/01/2009, Forrester Research

[22] Paul Vincent, David Luckham. http://www.complexevents.com/2014/12/03/cep-tooling-market-survey-2014/