Distribution and Uncertainty in Complex Event Recognition

Alexander Artikis[1,2] and Matthias Weidlich[3]

[1] Department of Maritime Studies, University of Piraeus, Greece
a.artikis@unipi.gr
[2] Institute of Informatics and Telecommunications
NCSR "Demokritos", Athens, Greece
[3] Department of Computer Science, Humboldt Universität zu Berlin, Germany
weidlima@informatik.hu-berlin.de

Abstract. Complex event recognition proved to be a valuable tool for a wide range of applications, reaching from logistics over finance to healthcare. In this paper, we reflect on some of these application areas to outline open research problems in event recognition. In particular, we focus on the questions of (1) how to distribute event recognition and (2) how to deal with the inherent uncertainty observed in many event recognition scenarios. For both questions, we provide a brief overview of the state-of-the-art and point out research gaps.

1 Introduction

Event processing has been established as a generic computational paradigm in a wide range of applications, spanning data processing in Web environments, over logistics and networking, to finance and the health sector [9]. Events, in general, report on state changes of a system and its environment, thereby enabling reactive and pro-active computing. At the very core of event processing systems is an event recognition mechanism (also known as event pattern matching [21]). It is the ability of a system to detect events that are considered relevant for processing and, as such, is the basis of realizing situation awareness in a system.

Event recognition systems are a key technology in the 'intelligent economy' that, based on the omnipresent availability of data that characterizes the information economy, provides means to analyze and act upon information. Detecting and understanding situations in computational as well as cyber-physical systems creates competitive advantage in commercial transactions, enables sustainable management of urban communities, and promotes appropriate distribution of social, healthcare and educational services [38]. By detecting relevant events also in the presence of extremely large scale data that is spread over geographical locations, event recognition systems help to extract actionable knowledge from Big Data.

The aim of this paper is to provide a brief overview of two open research questions related to event recognition. Based on a reflection of applications, we argue that distribution and uncertainty handling are of utmost importance for

effective and efficient use of event recognition. By briefly reviewing the state-of-the-art with respect to these two aspects, we carve out directions for future research in event recognition.

The rest of this paper is structured as follows. Section 2 illustrates the concept of event recognition using real-world applications. Section 3 gives details on the identified research challenges of event recognition related to distribution and uncertainty handling. Finally, Section 4 summarizes the paper.

2 Applications

Credit card fraud management is one of the applications in which complex event recognition plays a key role[4]. The goal is to detect fraud within 25 milliseconds, and even forecast it, in order to prevent the financial loss. Example fraud types include:

- 'cloned card' — a credit card is being used simultaneously in different countries;
- 'brute force attack' — multiple attempts to use a credit card per second;
- 'spike usage' — the 24-hour running sum is higher than the monthly average of the last 6 months;
- 'new high use' — the card is being frequently used in merchants or countries never used before;
- 'potential batch fraud' — many transactions from multiple cards are being used in the same point-of-sale terminal in high amounts.

The event patterns expressing fraudulent activity are highly complex involving hundreds of rules and performance indicators. They are also very diverse: fraud patterns heavily depend on the country, merchant, amount and customer. Fraud is continuously evolving — new fraud patterns appear on almost a weekly basis. Moreover, fraud detection is a needle in the haystack problem as fraudulent transactions constitute at most 0.1% of the total number of transactions. Perfect recall (finding all fraud cases) and perfect precision (never raise a false alarm) are out of reach — the state-of-practice recall and precision rates are about 60% and 10% respectively. At the same time, raising false alarms, that is, unnecessarily calling customers or blocking cards, is very costly in time and customer relationships. Missing true alarms is also very costly in terms of lost money.

Credit card fraud recognition and forecasting requires the analysis of large data streams storming from all over the world, as well as large amounts of historical data. For example, the SPEEDD project[5] will recognize fraud using up to 10,000 transactions/sec streaming from all over the world, and about 700 million events representing a 6 month history. Data streams are highly noisy: several of the data fields of credit card transactions could be left empty or contain incorrect information due to terminal misconfiguration. Examples include

[4] `https://www.feedzai.com/`
[5] `http://speedd-project.eu/`

incorrect timestamps and timezone information, incorrect merchant group codes, and missing or incorrect location information.

Traffic management is another application in which complex event recognition plays a crucial role. The goal here is to detect and forecast traffic congestions, and make decisions in order to attenuate them. For example, the SPEEDD project will forecast traffic congestions 5-20 minutes before they happen, and make decisions within 30 seconds of the forecast about the adjustment of traffic light settings and speed limits. Traffic management may be realized as follows:

- *Detect* traffic flow and density patterns as well as traffic incidents and safety violations.
- *Forecast* flow, density and travel duration for different temporal horizons. The carbon print and energy consumption can also be forecast.
- *Decide* which are the optimal variable speed limits and duty cycles for the ramp metering lights.
- *Act* by automatically changing the values of the variable speed limit panels and the operation of lights on the ramp metering course.

Traffic management requires the analysis of very large noisy data streams storming from various sensors, including fixed sensors installed in highways and city streets measuring traffic flow and density, mobile sensors such as smartphones and public transport vehicles reporting on traffic conditions [4], as well as large amounts of historical data. Sensors are frequently out of order, not calibrated appropriately and inaccurate. Data is often delayed and even completely lost during transmission. The data volume is expected to grow significantly in the following years as fixed sensors are installed on an increasing number of road segments. Moreover, there is a high penetration of mobile sensors such as GPS and accelerometers mounted on public transport vehicles, and smartphones used by drivers and pedestrians.

Maritime surveillance has been attracting attention both for economic and environmental reasons [26]. As an example, preventing accidents at sea by monitoring vessel activity results in substantial financial savings for shipping companies and averts maritime ecosystem damages. Complex event recognition allows for the fusion of various streaming data expressing, among others, vessel activity, with static geographical information, for the detection of suspicious or potentially dangerous situations that may have a serious impact on the environment and on safe navigation at sea.

Maritime navigation technology can automatically provide real-time information from sailing vessels. For instance, the Automatic Identification System (AIS)[6] is a tracking system for identifying and locating vessels at sea through data exchange. AIS information is continuously emitted from over 400,000 ships worldwide[7]. AIS-equipped vessels report their position in different time scales (the frequency of AIS messages depends, for example, on the proximity to base

[6] `http://www.imo.org/OurWork/Safety/Navigation/Pages/AIS.aspx`
[7] `http://www.marinetraffic.com`

stations and the vessel type). Moreover, AIS messages are often noisy, offering contradicting information. This data source alone then creates a Big Data problem for event recognition. For effective vessel identification and tracking, additional data sources should be taken into consideration, such as weather reports and frequently updated satellite images of the surveillance areas. Furthermore, streaming data must be continuously correlated with static geographical data for detecting, among others, violations of protected areas and shipping in unsafe areas.

3 Research Challenges

To perform complex event recognition in applications such as those mentioned above, one has to deal with a series of challenges [5]. For instance, complex events may evolve over multiple scales of time and space [37]. The variety of the event stream may be reflected by sources that report events ranging from (milli-)seconds to days. Moreover, historical data spanning over long periods of time need to be taken into consideration. Taking up the credit card fraud management application from above, for instance, transaction events for a credit card may be observed within milliseconds, but to detect fraud, their occurrence needs to be related to common usage patterns ranging over weeks or even months. To cope with multiple scales of time and space a recognition system should be adaptable, computing dynamically the appropriate lengths of multi-granular windows of varying levels of detail, being able to recognize complex events from lower-level events of varying spatio-temporal granularity, without compromising efficiency [22, 25, 20].

In what follows, we focus on two challenges of complex event recognition: (1) how to distribute event recognition (Section 3.1) and (2) how to deal with the inherent uncertainty observed in many event recognition scenarios (Section 3.2). An answer to the first question is a prerequisite for coping with the continuously growing volume and velocity of event streams. Computation as well as communication resources need to be used efficiently in order to allow for large-scale event recognition. The second question is motivated by the different types of uncertainty exhibited by event recognition applications. On the one hand, event streams used as input may be incomplete, include inaccurate or even erroneous information. On the other hand, the notion of an event that shall be recognized may be imprecise, which renders any event recognition probabilistic. As such, comprehensive handling of these types of uncertainty is a prerequisite for effective event recognition.

3.1 Distributed Event Recognition

Distributed deployment of event recognition enables scalability and allows for reaching the throughput that is required by contemporary applications, such as credit card fraud detection. Systems that exploit distribution and realise event recognition by independent processors, may be classified as *clustered* or

networked [9]. In a clustered system, the processors realising event recognition are strongly connected (e.g. part of the same local area network), meaning that the link between them is faster than the link between the system and the event sources. In the case of credit card fraud detection, for instance, processing may be shared in a large cluster of machines, connected by high-speed network connections, that receives the input event streams from remote sources. Examples for clustered recognition systems include Borealis [1], NextCEP [30], or commercial offerings such as IBM System S [41]. Networked systems, in turn, try to push computation to the sources in order to reduce communication costs [3], e.g., based on publish-subscribe middleware [19, 27]. The advent of scalable infrastructures for distributed event processing, such as Storm [35] or Spark Streaming [42] further provides opportunities for scalable event recognition. That is, the detection logic for a composite event may be encoded directly as a processor or instances of wrapped engines for centralised event recognition can be used as processors in these infrastructures.

Distribution strategies determine how the event recognition task is split up among different processors, may they be clustered or networked. Many of these techniques are *query-driven*, i.e., they apply distribution schemes that leverage the syntax and semantics of the composite event that shall be recognised. Examples includes the row/column scaling and pipelining, see [8], to distribute the execution of automata expressing queries in the Cayuga event language [7]. Semantic dependencies between composite events can be used to identify strata of independent queries, which are then executed on different nodes of a distributed system [18]. Other work showed how event recognition can be distributed to nodes while reusing operators that are part of the detection of multiple composite events [30]. Yet, distribution strategies may be even more fine-granular. That is, *instance-driven* techniques are not guided by the definition of the composite event, but focus on its partial materialisations (i.e., partial matches). For instance, input events that belong to individual run instances of the finite state machine of a composite event may be distributed to different nodes [6], thereby providing fine-grained partitioned data parallelism [13].

Distribution strategies for networked systems particularly aim at reducing the volume of data sent between the processors realising event recognition. Such strategies are particular valuable in sensor networks, such as those mentioned earlier in the context of traffic management. Methods proposed in this space decompose the event recognition task into a set of local constraints that can be verified at the event sources that generate the input data. The definition of these constraints typically relates to the absence of a composite event, i.e., as long as the local constraints are satisfied, it can be concluded that the composite event of interest has not occurred. As such, the constraints avoid unnecessary communication between the processor in situations where the composite event cannot be detected. Existing techniques following this idea have been tailored for events that are defined as a function over aggregate values derived at the event sources. In traffic management, for instance, such an approach enables to check whether the aggregated traffic flow in a certain neighbourhood stays

above a threshold, even though, most of the time, the sensed values are only locally checked at each sensor. Specific methods to realise this approach include sketching [24] and geometric reasoning [31, 12, 15].

Open issues. Despite much work on the distribution of event recognition, there are notable research gaps:

Distribution of probabilistic event recognition. As will be detailed below, event recognition is inherently uncertain, e.g., because of manual data input (credit card transactions) or noisy sensor data (traffic management, maritime surveillance). One way of handling this uncertainty is to rely on probabilistic instead of deterministic techniques. However, this renders the vast majority of existing distribution techniques inapplicable and calls for new deployment models and distribution strategies that are geared towards probabilistic methods.

Networked distribution of complex composite events. Techniques that aim at minimisation of communication in networked event recognition have focussed on composite events that are defined as functions over aggregate values. Yet, composite events that correlate events based on logical, temporal, and spatial conditions cannot be addressed with existing methods. Broadening the set of types of composite events that can be considered in the minimisation of communication between event recognition processors is an important direction for future research. For the above mentioned example of monitoring traffic flow, for instance, it may be relevant to not only detect that a threshold is exceeded, but to identify a sequence of spatially related violations of such a threshold.

Semantic distribution. Most distribution approaches are guided by the definition of the composite event (query-driven) or its partial materialisations (instance-driven). However, in many event processing scenarios, the input event streams also exhibit characteristics that enable effective distribution of event recognition tasks. Recently, it was shown how regularities in the occurrences of events can be leveraged to rewrite composite event patterns, see [10, 40]. For instance, the knowledge that events of one type may only be followed, but never preceded, by events of another type enables rewriting of a conjunction pattern over these types into a sequence pattern. Similarly, such knowledge about stream characteristics may be exploited to spread the event recognition task among the nodes of a distributed system.

3.2 Event Recognition under Uncertainty

Event recognition applications exhibit various types of uncertainty. Sensor networks introduce uncertainty due to reasons that range from inaccurate measurements through local network failures to unexpected interference of mediators. As mentioned earlier, for example, several of the data fields of credit card transactions could be left empty or contain incorrect information due to terminal misconfiguration. Similarly, Automatic Identification System (AIS) messages in the maritime domain are often noisy with contradicting information. For all of these reasons, input event streams lack veracity. Furthermore, in many application

domains, we only have imprecise knowledge about the pattern/definition of a complex event, or the available events and context information are insufficient for expressing a complex event.

Noisy input streams are handled, to various extents, by several approaches. For instance, the Lahar system [28], which is based on Cayuga, has an inference mechanism for answering queries over probabilistic data streams, that is, streams whose events are tagged with a probability value. In [32], each input event is defined as a set of alternatives, each with its occurrence probability, with all alternatives summing to a probability value of 1, or less than 1 if non-occurrence is possible. Tran and Davis [36] assume a computer vision setting where input events are detected by visual information processing algorithms with some degree of belief. This degree is propagated to a Markov Logic Network [11] expressing the complex event patterns using weighted utility formulas. Syntactically, each formula F_i in Markov logic is represented in first-order logic and it is associated with a weight w_i. The higher the value of the weight, the stronger the constraint represented by F_i. Semantically, a set of Markov logic formulas (F_i, w_i) represents a probability distribution over possible worlds. A world violating formulas becomes less probable, but not impossible as in first-order logic. Skarlatidis et al. [33] represent and reason over probabilistic data streams using the ProbLog logic programming framework [16]. ProbLog allows for assigning probabilities to events and can compute the 'success' probability of a query by summing the probabilities of all the subprograms that entail it.

Imperfect event patterns are naturally handled by techniques based on probabilistic graphical models. Morariu and Davis [23] use Markov Logic Networks in combination with Allen's interval algebra [2] to determine the most consistent sequence of complex events, based on the observations of low-level classifiers. Sadilek and Kautz [29] propose a method based on hybrid-Markov Logic Networks [39] in order to recognize (un)successful human interactions using noisy GPS streams. Other work relies on a hierarchy of Markov Logic Networks where mid-level networks process the output of low-level computer vision algorithms, and high-level networks fuse the output of mid-level networks to recognize complex events [14]. An attempt for generic event recognition in Markov Logic Networks is presented in [34]. In this setting, a dialect of the Event Calculus [17] is combined with probabilistic domain-dependent rules. Consequently, the approach supports probabilistic inertia. In other words, in the absence of relevant information the probability of a complex event may increase or decrease over time. The inertia behaviour of a complex event may be customized by appropriately adjusting the weight values of the corresponding rules.

Open issues. Probabilistic event recognition has been recently attracting attention. However, there are still open issues — below we discuss two of them.

Real-time event recognition under uncertainty. Although there is considerable work on optimising probabilistic reasoning techniques, the imposed overhead does not allow for real-time performance in a wide range of applications. To deal with Big Data, the focus has to shift to *distributed* and *parallelised* probabilistic reasoning. While we mentioned the challenge of distributing

probabilistic techniques in the previous section, this challenge also refers to the creation of new reasoning models and algorithms that offer greater potential for distribution than existing methods and which can exploit the full potential of infrastructures for highly-parallel execution.

Machine learning. Estimating manually the confidence values of the complex event patterns is a tedious and error-prone process. Using machine learning techniques, it is possible, in Markov Logic Networks for instance, to estimate the weights of the rules expressing a complex event pattern, given a set of training data. Weight learning in Markov Logic Networks is performed by optimising a likelihood function, which is a statistical measure of how well the probabilistic model fits the training data. In addition to weight learning, the structure of a Markov Logic Network, that is, the rules expressing complex events, can be learned from training data. Currently, the structure of a complex event is constructed first, and then weight learning is performed. However, separating the two learning tasks in this way may lead to suboptimal results, as the first optimisation step needs to make assumptions about the weight values, which have not been optimized yet. Better results can be obtained by combining structure learning with weight learning in a single stage.

4 Summary

In this paper, we took three applications for event recognition—credit card fraud management, traffic management, and maritime surveillance—as a starting point to motivate two research challenges: the distribution of event recognition and uncertainty handling. For both challenges, we gave a brief overview of the state-of-the-art and then identified research gaps. Those relate in particular to the distribution of probabilistic event recognition, networked distribution of composite events, semantic distribution, real-time event recognition under uncertainty, and machine learning in event recognition.

Acknowledgements

The authors have received financial support from the EU FP7 project SPEEDD (619435), the project *"AMINESS: Analysis of Marine INformation for Environmentally Safe Shipping"* which is co-financed by the European Fund for Regional Development and from Greek National funds through the operational programs "Competitiveness and Entrepreneurship" and "Regions in Transition" of the National Strategic Reference Framework - Action: "COOPERATION 2011 – Partnerships of Production and Research Institutions in Focused Research and Technology Sectors", and the German Research Foundation (DFG) in the Emmy Noether Programme (4891).

References

1. Abadi, D.J., Ahmad, Y., Balazinska, M., Çetintemel, U., Cherniack, M., Hwang, J., Lindner, W., Maskey, A., Rasin, A., Ryvkina, E., Tatbul, N., Xing, Y., Zdonik, S.B.: The design of the borealis stream processing engine. In: CIDR. pp. 277–289 (2005), http://www.cidrdb.org/cidr2005/papers/P23.pdf
2. Allen, J.: Maintaining knowledge about temporal intervals. Communications of the ACM 26(11), 832–843 (1983)
3. Artikis, A., Baber, C., Bizarro, P., de Wit, C.C., Etzion, O., Fournier, F., Goulart, P., Howes, A., Lygeros, J., Paliouras, G., Schuster, A., Sharfman, I.: Scalable proactive event-driven decision-making. IEEE Technology and Society Magazine 33(3), 35–41 (2014)
4. Artikis, A., Weidlich, M., Schnitzler, F., Boutsis, I., Liebig, T., Piatkowski, N., Bockermann, C., Morik, K., Kalogeraki, V., Marecek, J., Gal, A., Mannor, S., Gunopulos, D., Kinane, D.: Heterogeneous stream processing and crowdsourcing for urban traffic management. In: International Conference on Extending Database Technology (EDBT). pp. 712–723 (2014)
5. Artikis, A., Gal, A., Kalogeraki, V., Weidlich, M.: Event recognition challenges and techniques: Guest editors' introduction. ACM Trans. Internet Techn. 14(1), 1 (2014), http://doi.acm.org/10.1145/2632220
6. Balkesen, C., Dindar, N., Wetter, M., Tatbul, N.: Rip: run-based intra-query parallelism for scalable complex event processing. In: DEBS. pp. 3–14 (2013)
7. Brenna, L., Demers, A.J., Gehrke, J., Hong, M., Ossher, J., Panda, B., Riedewald, M., Thatte, M., White, W.M.: Cayuga: a high-performance event processing engine. In: SIGMOD Conference. pp. 1100–1102 (2007)
8. Brenna, L., Gehrke, J., Hong, M., Johansen, D.: Distributed event stream processing with non-deterministic finite automata. In: DEBS (2009)
9. Cugola, G., Margara, A.: Processing flows of information: From data stream to complex event processing. ACM Computing Surveys 44(3), 15 (2012)
10. Ding, L., Works, K., Rundensteiner, E.A.: Semantic stream query optimization exploiting dynamic metadata. In: Abiteboul, S., Böhm, K., Koch, C., Tan, K. (eds.) Proceedings of the 27th International Conference on Data Engineering, ICDE 2011, April 11-16, 2011, Hannover, Germany. pp. 111–122. IEEE Computer Society (2011), http://dx.doi.org/10.1109/ICDE.2011.5767840
11. Domingos, P., Lowd, D.: Markov Logic: An Interface Layer for Artificial Intelligence. Morgan & Claypool Publishers (2009)
12. Giatrakos, N., Deligiannakis, A., Garofalakis, M., Sharfman, I., Schuster, A.: Distributed geometric query monitoring using prediction models. ACM TODS (2014)
13. Hirzel, M.: Partition and compose: parallel complex event processing. In: DEBS. pp. 191–200 (2012)
14. Kanaujia, A., Choe, T.E., Deng, H.: Complex events recognition under uncertainty in a sensor network. arXiv:1411.0085 [cs] (Nov 2014), arXiv: 1411.0085
15. Keren, D., Sagy, G., Abboud, A., Ben-David, D., Schuster, A., Sharfman, I., Deligiannakis, A.: Geometric monitoring of heterogeneous streams. IEEE TKDE (2014)
16. Kimmig, A., Demoen, B., Raedt, L.D., Costa, V.S., Rocha, R.: On the implementation of the probabilistic logic programming language ProbLog. Theory and Practice of Logic Programming 11, 235262 (2011)
17. Kowalski, R., Sergot, M.: A logic-based calculus of events. New Generation Computing 4(1), 67–96 (1986)

18. Lakshmanan, G.T., Rabinovich, Y.G., Etzion, O.: A stratified approach for supporting high throughput event processing applications. In: Gokhale, A.S., Schmidt, D.C. (eds.) DEBS. ACM (2009)
19. Li, G., Jacobsen, H.: Composite subscriptions in content-based publish/subscribe systems. In: Alonso, G. (ed.) Middleware 2005, ACM/IFIP/USENIX, 6th International Middleware Conference, Grenoble, France, November 28 - December 2, 2005, Proceedings. Lecture Notes in Computer Science, vol. 3790, pp. 249–269. Springer (2005), http://dx.doi.org/10.1007/11587552_13
20. Lijffijt, J., Papapetrou, P., Puolamäki, K.: Size matters: Finding the most informative set of window lengths. In: ECML/PKDD (2). pp. 451–466 (2012)
21. Luckham, D.: The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems. Addison-Wesley (2002)
22. Maier, D., Grossniklaus, M., Moorthy, S., Tufte, K.: Capturing episodes: may the frame be with you. In: DEBS. pp. 1–11 (2012)
23. Morariu, V.I., Davis, L.S.: Multi-agent event recognition in structured scenarios. In: CVPR. pp. 3289–3296 (2011)
24. Papapetrou, O., Garofalakis, M.N., Deligiannakis, A.: Sketch-based querying of distributed sliding-window data streams. PVLDB 5(10), 992–1003 (2012)
25. Patroumpas, K.: Multi-scale window specification over streaming trajectories. J. Spatial Information Science 7(1), 45–75 (2013)
26. Patroumpas, K., Artikis, A., Katzouris, N., Vodas, M., Theodoridis, Y., Pelekis, N.: Event recognition for maritime surveillance. In: Alonso, G., Geerts, F., Popa, L., Barceló, P., Teubner, J., Ugarte, M., den Bussche, J.V., Paredaens, J. (eds.) Proceedings of the 18th International Conference on Extending Database Technology, EDBT 2015, Brussels, Belgium, March 23-27, 2015. pp. 629–640. OpenProceedings.org (2015), http://dx.doi.org/10.5441/002/edbt.2015.63
27. Pietzuch, P.R., Bacon, J.: Peer-to-peer overlay broker networks in an event-based middleware. In: Jacobsen, H. (ed.) Proceedings of the 2nd International Workshop on Distributed Event-Based Systems, DEBS 2003, Sunday, June 8th, 2003, San Diego, California, USA (in conjunction with SIGMOD/PODS). ACM (2003), http://doi.acm.org/10.1145/966618.966628
28. R, C., Letchner, J., Balazinksa, M., Suciu, D.: Event queries on correlated probabilistic streams. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data. pp. 715–728. SIGMOD '08, ACM, New York, NY, USA (2008), http://doi.acm.org/10.1145/1376616.1376688
29. Sadilek, A., Kautz, H.A.: Location-based reasoning about complex multi-agent behavior. J. Artif. Intell. Res. (JAIR) 43, 87–133 (2012)
30. Schultz-Møller, N.P., Migliavacca, M., Pietzuch, P.R.: Distributed complex event processing with query rewriting. In: DEBS (2009)
31. Sharfman, I., Schuster, A., Keren, D.: A geometric approach to monitoring threshold functions over distributed data streams. In: SIGMOD Conference. pp. 301–312 (2006)
32. Shen, Z., Kawashima, H., Kitagawa, H.: Probabilistic event stream processing with lineage. In: Proc. of Data Engineering Workshop (2008)
33. Skarlatidis, A., Artikis, A., Filippou, J., Paliouras, G.: A probabilistic logic programming event calculus. Theory and Practice of Logic Programming 15(2), 213–245 (2015)
34. Skarlatidis, A., Paliouras, G., Artikis, A., Vouros, G.: Probabilistic event calculus for event recognition. ACM Transactions on Computational Logic 16(2), 11:1–11:37 (2015)

35. Toshniwal, A., Taneja, S., Shukla, A., Ramasamy, K., Patel, J.M., Kulkarni, S., Jackson, J., Gade, K., Fu, M., Donham, J., Bhagat, N., Mittal, S., Ryaboy, D.V.: Storm@twitter. In: Dyreson, C.E., Li, F., Özsu, M.T. (eds.) International Conference on Management of Data, SIGMOD 2014, Snowbird, UT, USA, June 22-27, 2014. pp. 147–156. ACM (2014), http://doi.acm.org/10.1145/2588555.2595641
36. Tran, S.D., Davis, L.S.: Event Modeling and Recognition Using Markov Logic Networks. In: Proceedings of the 10th European Conference on Computer Vision (ECCV). Lecture Notes in Computer Science, vol. 5303, pp. 610–623. Springer (2008)
37. Vespier, U., Nijssen, S., Knobbe, A.J.: Mining characteristic multi-scale motifs in sensor-based time series. In: He, Q., Iyengar, A., Nejdl, W., Pei, J., Rastogi, R. (eds.) 22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013. pp. 2393–2398. ACM (2013), http://doi.acm.org/10.1145/2505515.2505620
38. Vesset, D., Flemming, M., Shirer, M.: Worldwide decision management software 2010–2014 forecast: A fast-growing opportunity to drive the intelligent economy. IDC report 226244 (2011)
39. Wang, J., Domingos, P.: Hybrid markov logic networks. In: AAAI. pp. 1106–1111 (2008)
40. Weidlich, M., Ziekow, H., Gal, A., Mendling, J., Weske, M.: Optimizing event pattern matching using business process models. IEEE Trans. Knowl. Data Eng. 26(11), 2759–2773 (2014), http://doi.ieeecomputersociety.org/10.1109/TKDE.2014.2302306
41. Wu, K., Yu, P.S., Gedik, B., Hildrum, K., Aggarwal, C.C., Bouillet, E., Fan, W., George, D., Gu, X., Luo, G., Wang, H.: Challenges and experience in prototyping a multi-modal stream analytic and monitoring application on system S. In: Koch, C., Gehrke, J., Garofalakis, M.N., Srivastava, D., Aberer, K., Deshpande, A., Florescu, D., Chan, C.Y., Ganti, V., Kanne, C., Klas, W., Neuhold, E.J. (eds.) Proceedings of the 33rd International Conference on Very Large Data Bases, University of Vienna, Austria, September 23-27, 2007. pp. 1185–1196. ACM (2007), http://www.vldb.org/conf/2007/papers/industrial/p1185-wu.pdf
42. Zaharia, M., Das, T., Li, H., Hunter, T., Shenker, S., Stoica, I.: Discretized streams: fault-tolerant streaming computation at scale. In: Kaminsky, M., Dahlin, M. (eds.) ACM SIGOPS 24th Symposium on Operating Systems Principles, SOSP '13, Farmington, PA, USA, November 3-6, 2013. pp. 423–438. ACM (2013), http://doi.acm.org/10.1145/2517349.2522737