Doctoral Symposium: Self-Adaptive Data Stream Processing in Geo-Distributed Computing Environments

Gabriele Russo Russo
University of Rome Tor Vergata
Rome, Italy
russo.russo@ing.uniroma2.it

ABSTRACT

An ever increasing number of services requires real-time analysis of collected data streams. Emerging Fog/Edge computing platforms are appealing for such latency-sensitive applications, encouraging the deployment of Data Stream Processing (DSP) systems in geo-distributed environments. However, the highly dynamic nature of these infrastructures poses challenges on how to satisfy the Quality of Service requirements of both the application and the infrastructure providers.

In this doctoral work we investigate how DSP systems can face the dynamicity of workloads and computing environments by self-adapting their deployment and behavior at run-time. Targeting geo-distributed infrastructures, we specifically search for decentralized solutions, and propose a framework for organizing adaptation using a hierarchical control approach. Focusing on application elasticity, we equip the framework with decentralized policies based on reinforcement learning. We extend our solution to consider multi-level elasticity, and heterogeneous computing resources. In the ongoing research work, we aim to face the challenges associated with mobility of users and computing resources, exploring complementary adaptation mechanisms.

CCS CONCEPTS

• **Computer systems organization** → **Distributed architectures**; • **Software and its engineering** → *Distributed systems organizing principles.*

KEYWORDS

Distributed Data Stream Processing, Self-Adaptive Software, Elasticity, Reinforcement Learning

ACM Reference Format:
Gabriele Russo Russo. 2019. Doctoral Symposium: Self-Adaptive Data Stream Processing in Geo-Distributed Computing Environments. In *DEBS '19: The 13th ACM International Conference on Distributed and Event-based Systems (DEBS '19), June 24–28, 2019, Darmstadt, Germany*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3328905.3332304

1 INTRODUCTION

Emerging computing scenarios like Internet of Things and Smart City are significantly contributing to the "data explosion" that we have been observing for years. New pervasive, intelligent services are developed by extracting valuable information from the available data flows (e.g., coming from sensors, wearable devices). In this context, distributed Data Stream Processing (DSP) systems play a key role, as they allow for processing incoming data in a near real-time fashion, and cope with high data rates by scaling their execution on multiple computing nodes.

The increasing presence of computing resources near the edge of the network, in Fog/Edge computing landscapes, represents a great opportunity for moving data analytics applications towards the data sources and consumers, hence reducing the processing latency and improving scalability. However, several challenges must be faced in order to effectively deploy DSP systems in these emerging environments. Compared to traditional data centers, computing resources at the edge are far more heterogeneous and likely less powerful, network latencies are not negligible, bandwidth is limited, and even connectivity may be unstable in case of mobile computing devices. Moreover, the conditions of the infrastructure are highly variable, and off-line optimizations may become outdated at runtime.

In order to effectively run DSP applications in geo-distributed platforms, *self-adaptation* capabilities are needed, so that the systems are able to adjust their own deployment and behavior to face the variability of the working conditions. In this doctoral work, we study mechanisms and policies for augmenting modern DSP systems with self-adaptation. Although a significant amount of work has already been done in exploring adaptation strategies for DSP frameworks, most of the existing solutions do not target geo-distributed infrastructures. Instead, we do consider a set of requirements that characterize this kind of computing environments, including resource heterogeneity and distribution, variability of conditions, unpredictability of performance, node mobility. Considering these issues, we aim to develop decentralized self-adaptation solutions, which allow to achieve good Quality of Service (QoS), while not suffering from scalability issues in Fog/Edge scenarios.

In the following, we will provide an outline of this doctoral research work. In Section 2, we enumerate the research challenges that motivate our investigation. In Section 3, we review the existing works regarding adaptive DSP systems. Then, we describe the research methodology we adopt in Section 4, and the contributions presented so far in Section 5. In Section 6, we outline the ongoing and future research directions we consider.

2 RESEARCH CHALLENGES

Emerging Fog computing environments represent a promising opportunity for deploying DSP applications close to users and data sources. However, geo-distributed infrastructures also present several challenges that must be faced to achieve the expected QoS. This research work aims to investigate mechanisms and policies to optimize performance in such dynamic computing environments, and its motivation can be summarized by the following questions:

RQ1 - Adapting to varying conditions In addition to workload variability, DSP applications in the Fog must cope with highly dynamic computing infrastructures (e.g., varying network latency). Which mechanisms can be used to adapt applications at run-time in response to workload and infrastructure variations? How can we control the adaptation to optimize QoS over time? Do adaptation actions have a (negative) impact on performance too?

RQ2 - Decentralized control Given the large scale of modern infrastructures, and the possibly large number of applications being executed concurrently, how can we control the execution of multiple DSP applications in a *decentralized*, scalable fashion?

RQ3 - Dealing with uncertainty As accurate performance models of the applications and the computing infrastructures are rarely available, how can we optimize QoS dealing with parameters uncertainty?

RQ4 - Multi-level adaptation Modern computing infrastructures allow for fine-grained and dynamic resource allocation, leveraging, e.g., container-based virtualization or Software Defined Networking (SDN). Can infrastructure-level adaptation be exploited to jointly optimize application-level QoS and resource utilization?

RQ5 - Mobility How should DSP systems be extended in order to work in presence of mobile data sources and/or consumers, and possibly mobile computing nodes? As mobile devices have usually limited energy capacity, which energy-aware adaptation strategies could be introduced?

3 RELATED WORK

To face the variable working conditions of modern computing environments, and effectively deploy and run DSP systems, different research directions have been explored in recent years. Several adaptation mechanisms have been considered in the literature as reviewed in [7, 28, 29].

A lot of effort has been spent exploring *elasticity*, that is the capability of adjusting the operator parallelism and/or the number of used computing nodes at run-time so as to avoid both over- and under-provisioning. Elasticity, which has undergone intense study in the context of cloud computing [22, 35], is particularly challenging for DSP because (i) latency-sensitive applications cannot tolerate long deployment reconfiguration periods, and (ii) stateful operators require state migration and synchronization protocols [15]. An evaluation of the reconfiguration overhead associated with elasticity in DSP systems is shown in [33], where different protocols are considered. As surveyed in [7] and [29], a variety of different techniques have been used to define elasticity policies for DSP, including heuristics [10, 11], control theory [8], linear programming [5], queueing theory [20], fuzzy logic [24], reinforcement learning [16]. However, most of these approaches rely on centralized controllers, and none considers heterogeneous computing infrastructures. Interestingly, Lombardi et al. [21] also propose a multi-level elasticity solution, which is able to scale operator parallelism along with the number of used computing nodes. However, their solution does not target geo-distributed environments.

Other works study how to optimize the application *placement* at run-time, hence migrating operators so as to guarantee good performance under varying infrastructure conditions. Pietzuch et al. [27] describe a fully decentralized algorithm to minimize network usage, exploiting a latency space as a search space to find the best placement solution. Eskandari et al. [9] propose a hierarchical approach based on graph partitioning, which aims at reducing inter-node traffic. The same goal is pursued by Aniello et al. [1], using a first-fit greedy heuristic. Cardellini et al. [5] describe an integer linear-programming model for determining the optimal operator placement and replication at run-time, considering application latency, deployment cost, and adaptation cost as objectives. Recently, Li et al. [19] proposed a model-free algorithm that aims to minimize application latency using deep reinforcement learning.

Increasing interest is being devoted to the deployment of DSP applications at the edge of the network, aiming to reduce latency. To this end, several issues must be faced, as discussed in [7] and [34], mainly due to the limited amount of computing resources provided by the devices at the edge. A few approaches have been presented so far to exploit simple mobile devices for streaming computation. For example, mobile clouds are used to deploy DSP applications in [36] and [26], hence exploiting multiple local devices to carry out heavy computation. Graubner et al. [13] instead aim to exploit specialized hardware on mobile phones for executing streaming operations on data emitted by onboard sensors. Considering this emerging deployment scenario, Yang et al. [37] propose a framework for partitioning DSP applications between mobile devices and cloud, so as to optimize performance.

Furthermore, various adaptation mechanisms related to the exchange of data streams among operators have been investigated. *Load shedding* techniques aim at avoiding overloading by discarding some input items before processing [12, 17]. Various techniques for the *adaptive routing* of the data streams have been presented, e.g., for achieving dynamic load balancing [32, 38], for reducing communication overhead [23], or for exploiting data locality [2]. A few works investigate *dynamic batch sizing*, that is adjusting the size of exchanged data batches so as to optimize the trade-offs between communication efficiency and latency [14, 39]. Chen et al. [6] instead propose a *backpressure* controller, which avoids application overloading by throttling the rate at which data are sent to operator instances.

4 RESEARCH APPROACH

In order to answer the research questions presented in Sec. 2, we investigate self-adaptation strategies for DSP systems. As most of the considered challenges are not uniquely related to DSP, we aim to develop solutions that could be applied in more general distributed systems scenarios. Given the complexity of the envisioned computing environment, we do not try to tackle all the outlined challenges at the same time. Instead we proceed by incrementally extending the set of considered research questions, building new solutions on top of already completed work.

As a first step, we study and analyze the challenges and available adaptation mechanisms for distributed DSP. Having observed that most of the approaches for adaptive DSP in the literature are not decentralized, as our first contribution, we define a framework for organizing self-adaptation in a decentralized fashion.

Among the available adaptation mechanisms, we first focus on elasticity, which has attracted a lot of interest in the community, and represents a key feature for modern DSP systems. We investigate decentralized elasticity policies to be integrated in our adaptation framework. We first consider the simplifying assumption of homogeneous computing resources, and then we enlarge our focus targeting heterogeneous infrastructures. In order to deal with uncertainty about the underlying system model, we take into consideration the adoption of reinforcement learning techniques.

Observing that elasticity alone does not allow to face all the research challenges we consider, we also aim at integrating other adaptation mechanisms in our framework. For instance, backpressure and load shedding could be integrated with elasticity to cope with short-term workload variations with reduced overhead. We also turn our attention to infrastructure-level adaptation mechanisms, so as to optimize the allocation of computing resources along with application QoS. In particular, infrastructure-level elasticity can be exploited to avoid both under- and over-provisioning of the computing resources. As a result, we investigate multi-level adaptation frameworks and policies.

Finally, we focus on the challenges related to mobility of users and computing nodes in Fog/Edge scenarios. We study the key issues to be tackled in order to guarantee acceptable QoS in this scenario, and investigate further adaptation mechanisms (e.g., dynamic batch sizing, mobile-cloud computation offloading) to enhance existing DSP frameworks. We again resort to multi-level solutions, leveraging the tools provided by computing infrastructures and networks (e.g., Software-Defined-Networking) to assist application-level adaptation.

During the various phases of this research work, we evaluate the proposed solutions both theoretically through analytical and simulation models, and experimentally by implementing our mechanisms and policies on top of existing DSP frameworks (i.e., Apache Storm and Apache Flink).

5 CURRENT STATUS

In this section we describe the contributions presented so far within this research project.

Elastic and Distributed DSP Framework. To efficiently control the execution of self-adaptive DSP applications in a Fog-like environment, we propose Elastic and Distributed DSP Framework (EDF) [4]. The adaptation in EDF is organized according to the *Monitor, Analyze, Plan and Execute* (MAPE) architectural pattern for self-adaptive systems [18]. In particular, EDF relies on a two-layered hierarchical solution, where a high-layer centralized MAPE-based *Application Manager* coordinates the run-time adaptation of subordinated, distributed MAPE-based *Operator Managers*. The latter components in turn locally control the adaptation of single DSP operators. The two layers operate with separation of concerns and different time scales. On on hand, the hierarchical structure of the framework leads to increased scalability compared to centralized approaches, while, on the other hand, it avoids the instability that often affects fully decentralized solutions.

Reinforcement Learning-based Elasticity Policies. The first adaptation mechanism we explore is elasticity. Considering the long-running nature of DSP applications, and the strict latency requirements to which they are often subject, optimizing the amount of used computing resources is critical to guarantee acceptable QoS in a cost-effective manner. We aim at overcoming common limitations of state-of-the-art elasticity policies, which often neglect the adaptation cost in making decisions, and are either centralized or based on very simple heuristics (e.g., threshold-based policies).

Leveraging the decentralized architecture of EDF, we define new elasticity control policies. In order to deal with uncertainty about the system model parameters, we resort to reinforcement learning techniques to improve the policy at run-time. In particular, we employ model-based RL algorithms that exploit the partial available knowledge about the system to achieve faster convergence. We first formulate the elasticity control problem for a single operator in isolation [3], and then for whole DSP applications [4]. In order to evaluate our solution, we implement it on top of Apache Storm. Our experiments demonstrate that the RL-based algorithm outperforms popular threshold-based approaches, also relieving the user from the need of tuning policy parameters by hand. Moreover, by including a reconfiguration cost in the model, our policies can take into account the adaptation overhead, and limit the number of deployment reconfigurations.

We extend the algorithm to work in presence of heterogeneous computing resources [30]. To deal with the significantly larger state space of the resulting model, we exploit Function Approximation techniques. The resulting approximate algorithms lead to near-optimal policies in small problem instances, and allow to solve large problem instances where exact methods are not applicable.

Multi-level Elasticity. We investigate the opportunity of jointly optimizing elasticity at application- and infrastructure-level in a wide-area computing environment. We extend EDF and propose Multi-Level Elastic and Distributed DSP Framework (E2DF) [25]. In E2DF, a second hierarchy of MAPE controllers is used to manage the resource allocation in the infrastructure, by running decentralized controllers in the geographically distributed regions (e.g., micro-data center in the Fog). We equip the new components with RL-based policies which aim to minimize the amount of resources used, while satisfying the demand coming from applications [31].

6 EXPECTED CONTRIBUTIONS

In this section we briefly outline the future directions we will follow to complete the research project.

Multi-tenancy scenarios. The control policies we studied so far do not explicitly take into account the presence of multiple applications, possibly owned by different users, in execution on the same infrastructure. Each application may be subject to different QoS requirements and need an appropriate adaptation policy. In this context we aim at meeting the QoS goals of each application, while optimizing the usage of computing resources. Exploiting game theory and multi-agent optimization techniques, we will define proper resource allocation policies.

Other adaptation mechanisms. We plan to analyze and integrate other adaptation mechanisms in conjunction with elasticity. A major limitation of elasticity is the significant adaptation overhead currently imposed by DSP frameworks whenever the deployment has to be changed. For this reason, elasticity is not well suited to react to short-term workload variations, and different mechanisms (e.g., backpressure, load shedding) may be investigated to overcome these limitations. In particular, we aim to develop a solution that jointly controls elasticity and backpressure. By identifying the most valuable adaptation action at each decision time, we could guarantee the desired QoS level with reduced adaptation overhead.

Mobility. Specific adaptation strategies should be explored in order for DSP systems to guarantee an acceptable service level in mobile environments, where unstable network connectivity represents a critical issue, along with energy consumption. First of all, we aim to evaluate the practical impact of these issues on existing DSP frameworks. Then, we will investigate solutions to mitigate the negative effects of mobility on QoS, leveraging, e.g., dynamic batch sizing, computation offloading. We will also extend our adaptation frameworks with fault-tolerance mechanisms and strategies.

Deep Learning-based policies. The convergence of the adaptation mechanisms discussed above will lead to ever more complex policies to be defined. To overcome the lack of accurate system models, we will continue investigating RL techniques. Given the complexity of the new tasks to face, we will turn our attention to *Deep RL* techniques, which have been successfully applied in many domains, and can cope with very large state spaces.

ACKNOWLEDGMENTS

The author would like to thank his supervisors Prof. Valeria Cardellini and Prof. Francesco Lo Presti for all their support and advice.

REFERENCES

[1] L. Aniello, R. Baldoni, and L. Querzoni. 2013. Adaptive Online Scheduling in Storm. In *Proc. ACM DEBS '13*. ACM, New York, NY, USA, 207–218.

[2] M. Caneill, A. El Rheddane, V. Leroy, and N. De Palma. 2016. Locality-Aware Routing in Stateful Streaming Applications. In *Proc. Middleware '16*. ACM, New York, NY, USA, Article 4, 13 pages.

[3] V. Cardellini, F. Lo Presti, M. Nardelli, and G. Russo Russo. 2018. Auto-scaling in Data Stream Processing Applications: A Model Based Reinforcement Learning Approach. In *InfQ 2017 – New Frontiers in Quantitative Methods in Informatics (CCIS)*, Vol. 825.

[4] V. Cardellini, F. Lo Presti, M. Nardelli, and G. Russo Russo. 2018. Decentralized Self-Adaptation for Elastic Data Stream Processing. *Future Gener. Comput. Syst.* 87 (2018), 171–185.

[5] V. Cardellini, F. Lo Presti, M. Nardelli, and G. Russo Russo. 2018. Optimal Operator Deployment and Replication for Elastic Distributed Data Stream Processing. *Concurr. Comput.: Pract. Exper.* 30, 9 (2018), e4334.

[6] X. Chen, Y. Vigfusson, D. M. Blough, F. Zheng, K. Wu, and L. Hu. 2017. GOVERNOR: Smoother Stream Processing Through Smarter Backpressure. In *Proc. IEEE ICAC '17*. 145–154.

[7] M.D. de Assunção, A. da Silva Veith, and R. Buyya. 2018. Distributed Data Stream Processing and Edge Computing: A Survey on Resource Elasticity and Future Directions. *J. Netw. Comput. Appl.* 103 (2018), 1 – 17.

[8] T. De Matteis and G. Mencagli. 2017. Elastic Scaling for Distributed Latency-sensitive Data Stream Operators. In *Proc. PDP '17*. IEEE, 61–68.

[9] L. Eskandari, Z. Huang, and D. Eyers. 2016. P-Scheduler: Adaptive Hierarchical Scheduling in Apache Storm. In *Proc. ACM ACSW '16*. ACM, New York, NY, USA, Article 26, 10 pages.

[10] A. Floratou, A. Agrawal, B. Graham, S. Rao, and K. Ramasamy. 2017. Dhalion: Self-regulating Stream Processing in Heron. *Proc. VLDB Endow.* 10, 12 (Aug. 2017), 1825–1836.

[11] B. Gedik, S. Schneider, M Hirzel, and K. Wu. 2014. Elastic Scaling for Data Stream Processing. *IEEE Trans. Parallel Distrib. Syst.* 25, 6 (2014), 1447–1463.

[12] B. Gedik, K. Wu, P.S. Yu, and L. Liu. 2005. Adaptive Load Shedding for Windowed Stream Joins. In *Proc. ACM CIKM '05*. ACM, New York, NY, USA, 171–178.

[13] P. Graubner, C. Thelen, M. Körber, A. Sterz, G. Salvaneschi, et al. 2018. Multimodal Complex Event Processing on Mobile Devices. In *Proc. ACM DEBS '19*. 112–123.

[14] B. Heintz, A. Chandra, and R. K. Sitaraman. 2015. Optimizing Grouped Aggregation in Geo-Distributed Streaming Analytics. In *Proc. ACM HPDC '15*. ACM, New York, NY, USA, 133–144.

[15] T. Heinze, L. Aniello, L. Querzoni, and J. Zbigniew. 2014. Cloud-based Data Stream Processing. In *Proc. ACM DEBS '14*. 238–245.

[16] T. Heinze, V. Pappalardo, Z. Jerzak, and C. Fetzer. 2014. Auto-scaling Techniques for Elastic Data Stream Processing. In *Proc. IEEE ICDEW '14*. 296–302.

[17] E. Kalyvianaki, M. Fiscato, T. Salonidis, and P. Pietzuch. 2016. THEMIS: Fairness in Federated Stream Processing Under Overload. In *Proc. ACM SIGMOD '16*. ACM, New York, NY, USA, 541–553.

[18] J. O. Kephart and D. M. Chess. 2003. The Vision of Autonomic Computing. *Computer* 36, 1 (Jan 2003), 41–50.

[19] T. Li, Z. Xu, J. Tang, and Y. Wang. 2018. Model-free Control for Distributed Stream Data Processing using Deep Reinforcement Learning. *Proc. VLDB '18* 11, 6 (2018), 705–718.

[20] B. Lohrmann, P. Janacik, and O. Kao. 2015. Elastic Stream Processing with Latency Guarantees. In *Proc. IEEE ICDCS '15*. 399–410.

[21] F. Lombardi, L. Aniello, S. Bonomi, and L. Querzoni. 2018. Elastic Symbiotic Scaling of Operators and Resources in Stream Processing Systems. *IEEE Trans. Parallel Distrib. Syst.* 29, 3 (2018), 572–585.

[22] T. Lorido-Botran, J. Miguel-Alonso, and J.A. Lozano. 2014. A Review of Auto-scaling Techniques for Elastic Applications in Cloud Environments. *J. Grid Comput.* 12, 4 (2014), 559–592.

[23] R. Mayer, M. A. Tariq, and K. Rothermel. 2017. Minimizing Communication Overhead in Window-Based Parallel Complex Event Processing. In *Proc. ACM DEBS '17 (DEBS '17)*. ACM, New York, NY, USA, 54–65.

[24] G. Mencagli, M. Torquati, and M. Danelutto. 2018. Elastic-PPQ: A Two-Level Autonomic System for Spatial Preference Query Processing Over Dynamic Data Streams. *Future Gener. Comput. Syst.* 79 (2018), 862 – 877.

[25] M. Nardelli, G. Russo Russo, V. Cardellini, and F. Lo Presti. 2018. A Multi-level Elasticity Framework for Distributed Data Stream Processing. In *Euro-Par 2018: Parallel Processing Workshops (LNCS)*, Vol. 11339. Springer, 53–64.

[26] Q. Ning, C. Chen, R. Stoleru, and C. Chen. 2015. Mobile Storm: Distributed Real-Time Stream Processing for Mobile Clouds. In *IEEE CloudNet '15*. 139–145.

[27] P. Pietzuch, J. Ledlie, J. Shneidman, M. Roussopoulos, M. Welsh, and M. Seltzer. 2006. Network-Aware Operator Placement for Stream-Processing Systems. In *Proc. ICDE'06*. 49–49.

[28] C. Qin, H. Eichelberger, and K. Schmid. 2019. Enactment of Adaptation in Data Stream Processing with Latency Implications—A Systematic Literature Review. *Inf. Softw. Technol.* 111 (2019), 1 – 21.

[29] H. Röger and R. Mayer. 2019. A Comprehensive Survey on Parallelization and Elasticity in Stream Processing. *ACM Comput. Surv.* 52, 2, Article 36 (April 2019), 37 pages.

[30] G. Russo Russo, V. Cardellini, and F. Lo Presti. 2019. Reinforcement Learning Based Policies for Elastic Stream Processing on Heterogeneous Resources. In *Proc. ACM DEBS '19*.

[31] G. Russo Russo, M. Nardelli, V. Cardellini, and F. Lo Presti. 2018. Multi-Level Elasticity for Wide-Area Data Streaming Systems: A Reinforcement Learning Approach. *Algorithms* 11, 9 (2018), 134.

[32] S. Schneider, J. Wolf, K. Hildrum, R. Khandekar, and K. Wu. 2016. Dynamic Load Balancing for Ordered Data-Parallel Regions in Distributed Streaming Systems. In *Proc. ACM Middleware '16*. ACM, New York, NY, USA, Article 21, 14 pages.

[33] A. Shukla and Y. Simmhan. 2018. Toward Reliable and Rapid Elasticity for Streaming Dataflows on Clouds. In *Proc. IEEE ICDCS '18*. 1096–1106.

[34] F. Starks, V. Goebel, S. Kristiansen, and T. Plagemann. 2018. Mobile Distributed Complex Event Processing—Ubi Sumus? Quo Vadimus? In *Mobile Big Data: A Roadmap from Models to Technologies*. Springer, 147–180.

[35] L.M. Vaquero, L. Rodero-Merino, and R. Buyya. 2011. Dynamically Scaling Applications in the Cloud. *SIGCOMM Comput. Commun. Rev.* 41, 1 (Jan. 2011), 45–52.

[36] H. Wang and L. Peh. 2014. MobiStreams: A Reliable Distributed Stream Processing System for Mobile Devices. In *Proc. IEEE PDPS '14*. 51–60.

[37] L. Yang, J. Cao, Y. Yuan, T. Li, A. Han, and A. Chan. 2013. A Framework for Partitioning and Execution of Data Stream Applications in Mobile Cloud Computing. *ACM Perf. E. R.* 40, 4 (2013), 23–32.

[38] N. Zacheilas, N. Zygouras, N. Panagiotou, V. Kalogeraki, and D. Gunopulos. 2016. Dynamic Load Balancing Techniques for Distributed Complex Event Processing Systems. In *Proc. IFIP DAIS '16*. Springer-Verlag, Berlin, Heidelberg, 174–188.

[39] Q. Zhang, Y. Song, R. R. Routray, and W. Shi. 2016. Adaptive Block and Batch Sizing for Batched Stream Processing System. In *Proc. IEEE ICAC '16*. 35–44.