Latency-Aware Strategies for Placing Data Stream Analytics onto Edge Computing

Alexandre da Silva Veith, Marcos Dias de Assunção, Laurent Lefèvre
Inria, LIP, ENS Lyon, University of Lyon

Abstract

Much of the "big data" generated today is received in near real-time and requires quick analysis. In Internet of Things (IoT) [1, 9], for instance, continuous data streams produced by multiple sources must be handled under very short delays.

As a result, several stream processing engines have been proposed. Under several engines, a stream processing application is a directed graph or dataflow whose vertices are operators that execute a function over the incoming data and edges that define how data flows between them. A dataflow has one or multiple sources (*i.e.*, sensors, gateways or actuators), operators that perform transformations on the data (*e.g.*, filtering, mapping, and aggregation) and sinks (*i.e.*, queries that consume or store the data).

In a traditional cloud deployment, the whole application is placed in the cloud computing to benefit from virtually unlimited resources. However, processing all the data in the cloud can introduce latency due to data transfer, which makes near real-time processing difficult to achieve. In contrast, *edge computing* has become an attractive solution for performing certain stream processing operators, as many *edge devices* have non-trivial compute capacity.

The deployment of data stream processing applications onto heterogeneous infrastructure has been proved to be NP-hard [2]. Moving operators from cloud to edge devices is challenging due to limitations of edge devices [5]. Existing work often proposes placements strategies considering user intervention [8]. Many models do not support memory and communication constraints [6, 4] while others consider all data sinks to be located in the cloud, with no feedback loop to actuators located at the edge of the network [3, 7]. There is a lack of solutions covering scenarios involving smart cities, precision agriculture, and smart homes comprising various heterogeneous sensors and actuators, as well as, time-constraint applications.

We model the data stream processing placement problem considering heterogeneous computational and network resources, and computing and communication as M/M/1 queues (*i.e.*, Poisson arrival distribution, exponential service time and single server). Events are handled in a First-Come, First-Served fashion both by the computation and communication services, guaranteeing the time order of events; an important requirement in many data stream processing applications. The model allows us to calculate the waiting and service times for each message in computation and communication queues allowing for estimating the response time.

We then propose two strategies to minimize the application response time by splitting the dataflow graph dynamically and distributing the operators across cloud and edge infrastructure. We focus on real-time analytics applications with multiple sources and sinks distributed across resources. In particular, we first decompose the application graph by considering behaviors such as *forks* and *joins* (*i.e.*, split points), and by identifying the operator dependencies recursively.

The Response Time Rate (RTR) strategy takes the decomposed graph and organizes the deployment sequence and consecutively calculates the response time for each operator by considering the previous mappings, resource capabilities, and operator requirements. RTR with Region Patterns (RTR+RP) strategy extends RTR by exploiting the split points to first find candidate operators for edge or cloud and then estimates the response time for the edge operators.

Comprehensive simulations considering multiple application configurations demonstrate that our approach can improve the response time up to 50%. For future work, we will investigate further techniques to deal with CPU-intensive operators and their energy consumption.

References

[1] ATZORI, L., IERA, A., AND MORABITO, G. The internet of things: A survey. *Computer Networks 54*, 15 (2010), 2787–2805.

[2] BENOIT, A., DOBRILA, A., NICOD, J.-M., AND PHILIPPE, L. Scheduling linear chain streaming applications on heterogeneous systems with failures. *Future Gener. Comput. Syst. 29*, 5 (July 2013), 1140–1151.

[3] CARDELLINI, V., GRASSI, V., PRESTI, F. L., AND NARDELLI, M. Distributed QoS-aware scheduling in Storm. In *9th ACM International Conference on Distributed Event-Based Systems* (New York, USA, 2015), DEBS '15, ACM, pp. 344–347.

[4] CHENG, B., PAPAGEORGIOU, A., AND BAUER, M. Geelytics: Enabling on-demand edge analytics over scoped data sources. In *IEEE International Congress on Big Data (BigData Congress)* (June 2016), pp. 101–108.

[5] DE ASSUNÇÃO, M. D., DA SILVA VEITH, A., AND BUYYA, R. Distributed data stream processing and edge computing: A survey on resource elasticity and future directions. *Journal of Network and Computer Applications 103* (2018), 1 – 17.

[6] HOCHREINER, C., VOGLER, M., WAIBEL, P., AND DUSTDAR, S. VISP: An ecosystem for elastic data stream processing for the internet of things. In *20th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2016)* (Sept 2016), pp. 1–11.

[7] NI, L., ZHANG, J., JIANG, C., YAN, C., AND YU, K. Resource allocation strategy in fog computing based on priced timed petri nets. *IEEE Internet of Things Journal PP*, 99 (2017), 1–1.

[8] SAJJAD, H. P., DANNISWARA, K., AL-SHISHTAWY, A., AND VLASSOV, V. Spanedge: Towards unifying stream processing over central and near-the-edge data centers. In *2016 IEEE/ACM Symposium on Edge Computing (SEC)* (Oct 2016), pp. 168–178.

[9] UCKELMANN, D., HARRISON, M., AND MICHAHELLES, F. An architectural approach towards the future internet of things. In *Architecting the internet of things*. Springer, 2011, pp. 1–24.