# Big Data Processing at Microsoft: Hyper Scale, Massive Complexity, and Minimal Cost

Hiren Patel, Alekh Jindal, Clemens Szyperski
{firstname.lastname}@microsoft.com
Microsoft

## ABSTRACT

The past decade has seen a tremendous interest in large-scale data processing at Microsoft. Typical scenarios include building business-critical pipelines such as advertiser feedback loop, index builder, and relevance/ranking algorithms for Bing; analyzing user experience telemetry for Office, Windows or Xbox; and gathering recommendations for products like Windows and Xbox. To address these needs a first-party big data analytics platform, referred to as *Cosmos*, was developed in the early 2010s at Microsoft. Cosmos makes it possible to store data at exabyte scale and process in a serverless form factor, with SCOPE [4] being the query processing workhorse. Over time, however, several newer challenges have emerged, requiring major technical innovations in Cosmos to meet these newer demands. In this abstract, we describe three such challenges from the query processing viewpoint, and our approaches to handling them.

**Hyper Scale.** Cosmos has witnessed a significant growth in usage from its early days, from the number of customers (starting from Bing to almost every single business unit at Microsoft today), to the volume of data processed (from petabytes to exabytes today), to the amount of processing done (from tens of thousands of SCOPE jobs to hundreds of thousands of jobs today, across hundreds of thousands of machines). Even a single job can consume tens of petabytes of data and produce similar volumes of data by running millions of tasks in parallel. Our approach to handle this unprecedented scale is two fold. First, we *decoupled and disaggregated* the query processor from storage and resource management components, thereby allowing different components in the Cosmos stack to scale independently. Second, we scaled the data movement in the SCOPE query processor with quasilinear complexity [2]. This is crucial since data movement is often the most expensive step, and hence the bottleneck, in massive-scale data processing.

**Massive Complexity.** Cosmos workloads are also highly complex. Thanks to adoption across the whole of Microsoft, Cosmos needs to support workloads that are representative of multiple industry segments, including search engine (Bing), operating system (Windows), workplace productivity (Office), personal computing (Surface), gaming (XBox), etc. To handle such diverse workloads, our approach has been to provide a *one-size-fits-all experience*. First of all, to make it easy for the customers to express their computations, SCOPE supports different types of queries, from batch to interactive to streaming and machine learning. Second, SCOPE supports both structured and unstructured data processing. Likewise, multiple data formats, including both propriety and open source source such as Parquet, are supported. Third, users can write business logic using a mix of declarative and imperative languages, over even different imperative languages such as C# and Python, in the same job. Furthermore, users can express all of the above in simple data flow style computation for better readability and maintainability. Finally, considering the diverse workload mix inside Microsoft, we have come to realization that it is not possible to fits all scenarios using SCOPE. Therefore, we also support the popular Spark query processing engine. Overall, the one-size-fits-all query processing experience in Cosmos covers very diverse workloads, including data formats, programming languages, and the backend engines.

**Minimal Cost.** While scale and complexity are hard by themselves, the biggest challenge is to achieve all of that at minimal cost. In fact, there is a pressing need to improve Cosmos efficiency and reduce operational costs. This is challenging due to several reasons. First, optimizing a SCOPE job is hard considering that the SCOPE DAGs are super large (up to 1000s of operators in single job!), and the optimization estimates (cardinality, cost, etc.) are often way off from the actuals. Second, SCOPE optimizes a given query, while the operational costs depend on the overall workload. Therefore workload optimization becomes very important. And finally, SCOPE jobs are typically interlinked in data pipelines, i.e., the output of one job is consumed by other jobs. This means that workload optimization needs to be aware of these dependencies. Our approach is to develop a *feedback loop* to learn from past workloads in order to optimize the future ones. Specifically, we leverage machine learning to learn models for optimizing individual jobs [3], apply multi-query optimizations to optimize the costs of overall workload [1], and build dependency graphs to identify and optimize for the data pipelines.

**ACM Reference Format:**
Hiren Patel, Alekh Jindal, Clemens Szyperski. 2019. Big Data Processing at Microsoft: Hyper Scale, Massive Complexity, and Minimal Cost. In *ACM Symposium on Cloud Computing (SoCC '19), November 20–23, 2019, Santa Cruz, CA, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3357223.3366029

## REFERENCES

[1] Alekh Jindal, Shi Qiao, Hiren Patel, Zhicheng Yin, Jieming Di, Malay Bag, Marc Friedman, Yifung Lin, Konstantinos Karanasos, and Sriram Rao. 2018. Computation Reuse in Analytics Job Service at Microsoft. In *SIGMOD*.

[2] Shi Qiao, Adrian Nicoara, Jin Sun, Marc Friedman, Hiren Patel, and Jaliya Ekanayake. 2019. Hyper Dimension Shuffle: Efficient Data Repartition at Petabyte Scale in SCOPE. In *VLDB*.

[3] Chenggang Wu, Alekh Jindal, Saeed Amizadeh, Hiren Patel, Shi Qiao, Wangchao Li, and Sriram Rao. 2019. Towards a Learning Optimizer for Shared Clouds. In *VLDB*.

[4] Jingren Zhou, Nicolas Bruno, Ming-Chuan Wu, Per-Åke Larson, Ronnie Chaiken, and Darren Shakib. 2012. SCOPE: parallel databases meet MapReduce. *VLDB J.* 21, 5 (2012), 611–636.


*SoCC '19, November 20–23, 2019, Santa Cruz, CA, USA*

ACM ISBN 978-1-4503-6973-2/19/11.
https://doi.org/10.1145/3357223.3366029