Aether: leveraging linear programming for optimal cloud computing in genomics

Citation

Published Version
doi:10.1093/bioinformatics/btx787

Permanent link
http://nrs.harvard.edu/urn-3:HUL.InstRepos:37160380

Terms of Use
This article was downloaded from Harvard University’s DASH repository, and is made available under the terms and conditions applicable to Other Posted Material, as set forth at http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA

Share Your Story
The Harvard community has made this article openly available. Please share how this access benefits you. Submit a story.

Accessibility
Genome analysis

Aether: leveraging linear programming for optimal cloud computing in genomics

Jacob M. Luber1,2,3,4,†, Braden T. Tierney1,2,3,4,†, Evan M. Cofer1,2,4,5, Chirag J. Patel3,* and Aleksandar D. Kostic1,2,4,*

1Section on Pathophysiology and Molecular Pharmacology, 2Section on Islet Cell and Regenerative Biology, Joslin Diabetes Center, Boston, MA 02215, USA, 3Department of Biomedical Informatics, 4Department of Microbiology and Immunobiology, Harvard Medical School, Boston, MA 02115, USA and 5Lewis-Sigler Institute for Integrative Genomics, Princeton University, Princeton, NJ 08544, USA

*To whom correspondence should be addressed.
†The authors wish it to be known that, in their opinion, the first two authors should be regarded as Joint First Authors.

Associate Editor: Bonnie Berger

Received on August 16, 2017; revised on November 7, 2017; editorial decision on November 30, 2017; accepted on December 7, 2017

Abstract

Motivation: Across biology, we are seeing rapid developments in scale of data production without a corresponding increase in data analysis capabilities.

Results: Here, we present Aether (http://aether.kosticlab.org), an intuitive, easy-to-use, cost-effective and scalable framework that uses linear programming to optimally bid on and deploy combinations of underutilized cloud computing resources. Our approach simultaneously minimizes the cost of data analysis and provides an easy transition from users’ existing HPC pipelines.


Contact: chirag_patel@hms.harvard.edu or aleksandar.kostic@joslin.harvard.edu

Supplementary information: Supplementary data are available at Bioinformatics online.

1 Aether

Data accumulation is exceeding Moore’s law, which only still progresses due to advances in parallel chip architecture (Esmaeilzadeh et al., 2013). Fortunately, the shift away from in-house computing clusters to cloud infrastructure has yielded approaches to computational challenges in biology that both make science more reproducible and eliminate time lost in high-performance computing queues (Beaulieu-Jones and Greene, 2017; Garg et al., 2011); however, existing off-the-shelf tools built for cloud computing often remain inaccessible, cumbersome, and in some instances, costly.

Solutions to parallelizable compute problems in computational biology are increasingly necessary; however, batch job-oriented cloud computing systems, such as Amazon Web Services (AWS) Batch, Google preemptible Virtual Machines (VMs), Apache Spark and MapReduce implementations are either closed source, restrictively licensed, or locked in their own ecosystems making them inaccessible to many bioinformatics labs (Shvachko et al., 2010; Yang et al., 2007). Other approaches for bidding on cloud resources exist, but they neither provide implementations nor interface with a distributed batch job process with a backend implementation of all necessary networking (Andrzejak et al., 2010; Tordsson et al., 2012; Zheng et al., 2015).

Our proposed tool, Aether, leverages a linear programming (LP) approach to minimize cloud compute cost while being constrained by user needs and cloud capacity, which are parameterized by the number of cores, RAM, and in-node solid-state drive space. Specifically, certain types of instances are allocated to large web service providers (e.g. Netflix) and auctioned on a secondary market when they are not fully utilized (Zheng et al., 2015). Users bid amongst each other for use of this already purchased but unused compute time at extremely low rates (up to 90% off the listed price; https://aws.amazon.com/ec2/pricing/). However, this market is not without its complexities. For instance, significant price fluctuations,
up to an order of magnitude, could lead to early termination of multi-hour compute jobs (Fig. 1A). Clearly, bidding strategies must be dynamic to overcome such hurdles.

Aether consists of bidder and batch job processing command line tools that query instance metadata from the vendor application programming interface (APIs) to formulate the LP problem. LP is an optimization method that simultaneously solves a large system of equations to determine the best outcome of a scenario that can be described by linear relationships. The Aether bidder, described in detail in the Supplementary Methods, generates and solves a system of 140 inequalities using the simplex algorithm (Fig. 1B). For the purposes of reproducibility, an implementation of the bidder using CPLEX is also provided as an optional command line flag.

Subsequently, the replica nodes specified by the LP result are placed under the control of a primary node, which assigns batch processing jobs over transmission control protocol, monitors for any failures, gathers all logs, sends all results to a specified cloud storage service level agreements (Andrzejak et al., 2010). Future directions include training the bidding algorithm to predict its own effect on pricing variability when being utilized at massive scale as well as distributing compute nodes across datacenters when enough resources are being spun up to strongly influence the market.

Our work was funded by National Institutes of Health/National Human Genome Research Institute (NIH/NHGRI) T32 HG002295, PI: Park, Peter J.

Acknowledgements

We thank Thomas Lane and Chengwei Luo for their feedback and review of the manuscript. We thank Tommi Vatanen for helping us with data access.

Funding

This work was funded by National Institutes of Health/National Human Genome Research Institute (NIH/NHGRI) T32 HG002295, PI: Park, Peter J.
References


