

# CloudSort: A TCO Sort Benchmark

The SortBenchmark Committee

Mehul A. Shah, Amiato, Chris Nyberg, Ordinal Technology Corp., Naga Govindaraju, Microsoft (Azure)  
[mehul@amiato.com](mailto:mehul@amiato.com), [chris@ordinal.com](mailto:chris@ordinal.com), [nagag@microsoft.com](mailto:nagag@microsoft.com)

August 2013, version 0.1

December 2013, version 0.2

March 2014, version 0.3

June 2014, version 0.4

**Abstract:** This short paper proposes CloudSort, a new total-cost of ownership (TCO) benchmark based on external sort. Alternatively, it's a benchmark to assess the efficacy of cloud environments for IO-intensive workloads.

## 1. Introduction

The sort benchmarks, as a collection, cover many relevant dimensions for assessing the efficacy of modern data pumps. Over the last three decades, these benchmarks have diversified, starting from the DatamationSort, a pure performance benchmark, to the inclusion of numerous variants: PennySort, Performance/Price Sort, and JouleSort. These variants measure the efficiency of external sorts along dimensions important for practical deployment in data centers: the cost of equipment (hardware and software) and energy-use. Management and maintenance cost is one dimension that remains conspicuously missing. Most importantly, there is no single yardstick that combines all these costs to assess overall efficiency.

The availability of cloud infrastructure-as-a-service platforms now gives us an opportunity to have that yardstick. Cloud platforms have proliferated in the last decade, with major vendors like Amazon, Microsoft, Google, Rackspace, VMWare, HP, as well as numerous startups. They offer on-demand compute, network, and storage at scale. More importantly, they offer amortized pricing that reflect all costs for profitably running these infrastructure services.

We, therefore, propose using the cloud to define a benchmark that measures the efficiency of external sort from a total-cost of ownership (TCO) perspective. CloudSort asks: **“What is the minimum cost for sorting a fixed number of records on any public cloud?”** The benchmark is summarized as follows:

1. Sort a fixed number (currently  $10^{12}$ , ~100TB) of randomly permuted 100-byte records with 10-byte keys
2. The sort must start with the input on a non-ephemeral, persistent store and finish with output on a non-ephemeral, persistent store.
3. All operations must be performed on a commercially available public cloud.
4. The winner is the system with the minimum cost prorated for the duration of the sort.

Similar to the other variants, there's a Daytona and Indy category for general-purpose and no-holds barred sort implementations, respectively.

## 2. Justification

A truly complete and representative TCO benchmark is elusive. Nonetheless, CloudSort will drive innovations in public cloud platforms and has the potential to become a legitimate TCO proxy for IO-intensive workloads.

**Why the public cloud?** Given the relative immaturity of cloud platforms, most are not properly provisioned for IO-intensive tasks. Moreover, over prolonged use, the public cloud is currently much more expensive than a well-

managed in-house data center for many companies. Thus, CloudSort today primarily highlights the efficacy (or deficiencies) of different cloud platforms rather than the TCO-efficiency of sort implementations.

The trend, however, is that enterprises will eventually move over to the public cloud as it matures. Benchmarks like CloudSort aim to enable this transition. It will encourage and drive cloud platforms to quickly improve support for IO-intensive operations. As cloud platforms further commoditize, the overall cost for using the public cloud will converge to and undercut the costs of self-managed data centers. Thus, in the long run, CloudSort will hopefully be an indicator for both: the effectiveness of cloud platforms and the TCO-efficiency of external sort.

**Why external sort?** External sort is representative of many IO-intensive workloads. It's a holistic workload that exercises memory, CPU, OS, file-system, IO, network, and storage. It's simple and therefore easy to port and optimize on cutting-edge technologies. As a result, it's a technology bellwether indicating which platforms are likely to make great data pumps going forward.

### 3. Advantages

A benchmark on the public cloud offers advantages that assuage some issues plaguing the other sort benchmarks.

**Accessibility.** One major concern for the larger sorts like Gray Sort is that it is a benchmark for the coalition of the willing. Only the people that have access to expensive cutting-edge clusters at national labs, well-funded academic institutions, or large companies can afford to enter the contest. CloudSort levels the playing field.

The public cloud amortizes up-front costs over time, so even groups with limited budgets can develop their implementations and run the benchmark. Moreover, given the limited cost of running the benchmark, cloud vendors may also offer to offset these costs to show-off their capabilities.

**Affordability.** A rough calculation suggests that, with current AWS pricing, the cost for running a 100TB sort (~\$650) is manageable for groups with modest budgets. Three runs for an official entry would cost approx. \$2000. Moreover, we expect entrants to run significantly scaled-down versions during development and testing.

Assume that we run a parallel two-pass sort across 100 x m1.large EC2 instances each with 2 x 1.5 TB EBS volumes attached. We expect disk I/O to be the bottleneck, and we assume each volume provides 50MB/s sequential bandwidth. In that case, a two-pass sort will finish in less than 12 hours (11.1 hrs). Currently, on-demand pricing for an AWS Linux m1.large EC2 is \$0.24/hr, and EBS is \$0.05/GB-month. Thus, prorated cost for EBS for a 12-hour sort run is  $300,000 \text{ GB (input, temp, output)} * 0.05 \text{ (\$/GB-month)} / 30 \text{ (days/month)} / 24 \text{ (hours/day)} * 12 \text{ (hours)} = \$250$ . This comes to \$538 for the entire 100 node, 12-hrs sort run: \$288 for EC2 and \$250 for EBS. Assuming an additional 20% overhead for IOPS and other charges, the total cost is approximately \$645.

**Auditability.** With the previous sort benchmarks, auditing a result involves good detective work on the part of auditors. The actual runs are hardly ever reproduced. On the public cloud, entrants can easily offer a limited trial of their sort software as virtual machine images, so anyone can attempt to rerun the experiments.

### 4. Rules

We list detailed rules for CloudSort.

**Public Cloud.** What defines the public cloud? A public cloud is any commercially available infrastructure-as-a-service (IaaS). These include services that allow one to provision (spin-up) and de-provision (spin-down) compute, storage, and network resources on-demand. CloudSort entries on such services would likely implement external sort from scratch using these raw, low-level resources.

We also include services that offer infrastructure applications on-demand like operational databases, data warehouses, and Hadoop. For these, CloudSort entries could reuse their built-in sorting functionality.

The maximum allowable time for provisioning and de-provisioning is 1 day.

**Pricing & Total Cost.** The total cost will include the prorated cost for renting the cloud infrastructure as well as the prorated cost for the software. Beyond fixed on-demand pricing, there are a number of discounted pricing schemes available currently: reserved instance, spot market, pricing discounted over time with extended use.

In the spirit of previous cost-efficiency benchmarks, we want to use the undiscounted, “list” price of the service. So, all costs will be calculated from fixed, on-demand pricing without any discounts. To be more precise, the on-demand price must be a constant price published for the prior three months. There should be no variability in each resource’s price based on time-of-day, week, month, or season. So, spot-market pricing, for example, is not allowed. The on-demand pricing must have a maximum billing granularity of 1 day.

When calculating total cost, the cost will be prorated to the nearest second for the duration of the sort run, even if the billing granularity is larger (e.g. per day, per month, etc.). Moreover, we will include all costs incurred during the sort run: compute, storage (bytes and IO), and network I/O.

Since we expect cloud-pricing schemes to evolve with perhaps increasing complexity, the committee reserves the right to determine that a particular pricing scheme applies.

**Storage & Reliability.** There are several types of storage provided by cloud vendors. For example, there are long-term, persistent, highly-available stores like Amazon S3 or Google Cloud Storage. There are virtual arrays like elastic-block store (EBS). Finally, there are ephemeral stores like local disks on virtual machines.

The input and output must reside on non-ephemeral storage that has the same reliability requirements as GraySort Daytona. That is, this non-ephemeral storage must survive single-node failures where a single node is a compute node or disk. Thus, S3 buckets, EBS volumes, and reduced redundancy storage all qualify. Ephemeral virtual machine disks do not.

There is no explicit availability requirement. The store must have some type of file-system API for opening, reading, writing, and closing files.

**Measurement Variability.** Since physical resources are often shared in cloud environments, load and interference can vary and affect the consistency of the measurement. Entrants can take enough trials to get a consistent set of runs. As in the other benchmarks, we will accept the average of any three consecutive runs.

In the future, we may consider running the sort implementations ourselves. Dedicated resources from the cloud providers specifically for the sort contest would greatly improve our ability to fairly crown winners.

**Provisioning.** Provisioning can take a significant time, especially when running in a large distributed configuration. We will not include in total costs, the time and cost to provision VMs, storage, network, and setup the sort software. We will also not include the cost to provision and populate the input data set.

**Reporting.** We will report the total time to complete the sort, as well as, the total costs and breakdown of costs during the sort the run.

## 5. Conclusion

The sort benchmarks are missing a total-cost-of-ownership category. Moreover, cloud platforms today are poorly provisioned for IO-intensive workloads. We propose CloudSort that asks what is the cheapest you can sort a fixed

number of records on the public cloud. This benchmark will help drive innovation in the public cloud for supporting IO-intensive tasks. Over time, as the cloud matures, this benchmark will not only point out the best platforms for building data pumps, but also find the most efficient sort implementations from a total-cost-of-ownership perspective.

### **Acknowledgements**

We would like to thank Partha Ranganathan, Christos Kozyrakis, George Porter, Suzanne Rivoire, and Paul Newson for reviewing and providing critical feedback on drafts of this benchmark proposal.