Aster Data

nCluster

A New Architecture for Data Analytics

The Large Scale Data Management Experts
Executive Summary

Aster Data (www.asterdata.com) has developed and brought to market Aster Data nCluster, a new platform for data analytics. Based on the groundbreaking research of three Stanford doctoral students in computer science, nCluster features an innovative shared nothing, massively parallel architecture while running on low priced, commodity hardware.

nCluster departs from standard massively parallel (MPP) designs in several important ways. To give three examples:

nCluster employs four different node types (Queens, Workers, Loaders/Exporters, and Backup nodes), providing flexibility in the configuration of systems. Some applications may require more data loading or backup/recovery capacity and others may require more computation or query processing capacity. This configuration flexibility means that a greater range of requirements can be satisfied within a given hardware budget.

nCluster has several features designed to reduce the demands on the interconnect while efficiently redistributing data as required. Since the interconnect – the internal network connecting the nodes – is the one hardware component of a massively parallel system that is shared by all nodes, true scalability depends vitally on its efficient use.

nCluster has been designed from the beginning for high availability. For example, the system continues to run in the presence of a node failure. Unlike most other MPP systems, nCluster does not require that queries in progress at the time of the failure be restarted. The system automatically handles node failures and continues processing queries in progress, transparently restarting the components of queries that were running on the failed node.

As well as providing relational database services based on the industry standard SQL language, nCluster offers support for MapReduce, a facility for in-database analysis employing procedural programming languages such as Java, Python, and Perl. Because nCluster also supports Microsoft .NET, it has become the first product to bring MapReduce to the .NET environment and make it available to Microsoft C# developers. In addition, nCluster features other functional innovations, such as nPath, a facility for time series analysis. nCluster is available in a software only format; as an appliance; and, in the Amazon cloud.

While a young product, nCluster has successful customers, two of whose implementations are described in this paper. Among those interviewed by WinterCorp is MySpace, a leading social networking site running four nClusters of which the largest operates on 102 nodes and manages more than 145 TB (terabytes) of user data. The results of WinterCorp interviews with two nCluster customers are described in the text of this paper.

In the opinion of WinterCorp, users with challenging analytical requirements will want to evaluate nCluster as they select the platform for their next substantial implementation.
Table of Contents

1. Introduction ........................................................................................................................................ 1
   1.1. The Growing Requirements for Analytics .................................................................................. 2
   1.2. Aster Data nCluster .................................................................................................................... 2
   1.3. Report Methodology ................................................................................................................... 3
2. Key Design Principles ............................................................................................................................. 4
   2.1. MPP with Specialized Nodes ....................................................................................................... 4
   2.2. Design for Analytic Applications ................................................................................................. 6
   2.3. Use of Standard, Low Cost Components ..................................................................................... 7
   2.4. Addressing the Network Bottleneck ............................................................................................ 8
   2.5. Data Redistribution .................................................................................................................... 9
   2.6. Advanced partitioning .................................................................................................................. 11
   2.7. Data Availability ......................................................................................................................... 14
   2.8. Manageability ............................................................................................................................ 15
3. Other Key Features ................................................................................................................................... 17
   3.1. MapReduce ................................................................................................................................... 17
   3.2. .NET ............................................................................................................................................ 20
   3.3. Appliance Offering ....................................................................................................................... 21
4. Customer Cases ....................................................................................................................................... 22
   4.1. ShareThis ...................................................................................................................................... 22
   4.2. MySpace ...................................................................................................................................... 24
5. Conclusions ............................................................................................................................................... 29
List of Illustrations

Figure 1: Key Requirements for the Analytic Data Platform  Section 1.1  Page 1
Figure 2: Aster nCluster Node Level Architecture  Section 2.1  Page 4
Figure 3: Simplified Database Design Example  Section 2.5  Page 9
Figure 4: Data Redistribution in a Massively Parallel Architecture  Page 10
Figure 5: Example of Four Levels of Partitioning  Page 12
Figure 6: Example of MapReduce and SQL  Page 19
Figure 7: ShareThis Widget  Section 4.1  Page 22
Figure 8: System Architecture at MySpace  Page 25
Figure 9: Some Uses of nCluster at MySpace  Page 26
### Introduction

#### 1.1 The Growing Requirements for Analytics

A defining trend in 21st century enterprise is the rapidly increasing use of data analysis to make better, and often faster, decisions.

In many Internet based ventures, data analysis is the very basis of the business model: such analysis makes promotional dollars go further by delivering ads only to the people most likely to act on them. Data analysis tells entertainment companies how many people are listening to their music or watching their movies – and what other entertainment such people are likely to prefer.

In consumer retail and telco businesses, intensive near real time data analysis makes it possible to respond to customer behavior or interests while the customer is at the checkout counter or on the website.

In manufacturing businesses, data analysis enables better supply chain management, higher product quality and lower manufacturing cost.

| Scalable                | • Store, access and manage large, rapidly growing data volume  
|                        | • Handle rapidly growing workload  
|                        | • Perform complex analysis on data in place  
|                        | • Expand in modular units, without disruption  
| Available              | • Keep data highly or continuously available  
| Insightful             | • Handle structured and unstructured data  
|                        | • Handle data mining and embedded applications to get deep, actionable insights to the hardest questions  
| Performant             | • Deliver rapid response to requests for service (query, update, analysis, extraction)  
| Low Latency            | • Incorporate new and changed data rapidly and continuously, without interference with other operations  
| Economical Manageable  | • Be affordable in all configurations, from small to large  
|                        | • Maintain and operate the system readily, with a minimum of specialized skills  

Figure 1: **Key Requirements for the Data Analytics Platform**
In retail, data analysis is helping retailers to make better decisions stocking their stores – and helping manufacturers make better decisions in producing their product mix – because in both cases better data and better analysis is resulting in improved predictions of demand.

In many cases, the key to higher business impact depends on the ability to retain, manage, access and analyze large volumes of detailed data. And, frequently the mathematical analysis involved is quite extensive.

As a result, many enterprises find their analytic data requirements and opportunities growing rapidly. Businesses want scalable, reliable and economical solutions for these needs. They want solutions that work for a range of analytical requirements, including near real time analysis of data; computationally intensive analysis; and, complex analysis of data that is not best represented in tabular structures. In short, enterprises face an increasing variety of analytical needs, some of which may require new approaches and capabilities.

1.2 Aster Data \(n\)Cluster

Aster Data’s \(n\)Cluster is a data warehouse platform for analytic applications, introduced to the market in 2008, with the goal of meeting emerging requirements while offering significant advances in scalability and price performance. \(n\)Cluster employs a shared nothing, massively parallel (MPP) architecture with a new approach to the design: along with other innovations described in this paper, \(n\)Cluster design aims to eliminate the network bottleneck which is often a key limitation affecting larger workloads and more complex queries on MPP platforms.

\(n\)Cluster runs on low cost, commodity hardware (e.g., standard Dell/HP/IBM/Sun servers, Cisco switches) – an essential aspect of the Aster Data strategy--which seeks to provide customers with both a technically and an economically feasible approach to implementing very large scale analytical systems. Architected for operation from hundreds of gigabytes to the petabyte scale, \(n\)Cluster is presently certified for configurations managing up into hundreds of terabytes of data.

\(n\)Cluster features other key innovations – to advance performance, data availability and ease of management – which are discussed at length in this paper. One example is a node failover mechanism which automatically completes queries in progress during a node failure – in contrast to other data warehouse products on the market, which would require restarting most or all such queries. Another example is online backup, which provides a native scale-out cluster for highly parallel backups.
Finally, Aster Data has incorporated significant functional innovations into nCluster, broadening the notion of a data management platform for analytics. For example, in addition to providing a SQL-based relational database capability -- which is now nearly universal in major market offerings -- nCluster also supports MapReduce. MapReduce is a newer framework for highly parallel data analysis made popular by Google. nCluster customers are thus offered a choice: SQL for highly structured, tabular data analysis; and, MapReduce for less structured, more procedural data analysis.

Significantly, MapReduce and SQL are integrated in nCluster. For example, MapReduce functions may be called from SQL statements. In this integrated facility, named SQL/MR, the two types of analysis can be combined in a single, pipelined data management operation.

This WinterCorp White Paper reviews the architecture and design of nCluster and the experiences of nCluster customers.

1.3 Methodology for this Report

This White Paper was sponsored by Aster Data Systems.

In developing this paper, WinterCorp operated as an independent industry expert, interviewing Aster Data customers and employees; reviewing product documentation; observing demonstrations of the product; and, critically reviewing product design, measurements and evidence in order to arrive at the descriptions and conclusions presented here. Aster Data was provided an opportunity to comment on the paper with respect to facts. WinterCorp has final editorial control over the content of this publication.
Key Design Principles

2.1 MPP with Specialized Nodes

Aster Data nCluster employs a massively parallel (MPP) architecture with some specialization of nodes, as shown in Figure 1.

Figure 2: Aster nCluster Node Level Architecture
(Source: Aster Data)
In a classic MPP database architecture, there is a single type of node which performs all data management functions. However, nCluster features three specialized node groups: queen nodes, which accept, plan and coordinate queries – and deliver answers to the requestor; loader nodes, which load new data and perform high volume data exports; and, worker nodes, which perform most other operations, including the actual processing of queries; and, backup nodes.

Only the worker nodes actually hold data. Employing a key principle of the classic MPP approach – “shared nothing” – each worker node holds a portion of the database and is the only node that reads or writes that data for normal operations (that is, the worker nodes do not share data).

Queries are submitted to the queen nodes, which plan, optimize and manage their execution, sending the necessary subqueries to each worker node.

Each worker node performs its subquery or subqueries independently of the others, acting only in accordance with the query plan(s) it has received from the queen. It is the worker nodes which manage the disk input/output, actually reading data from the database and writing data to the database. The final results of queries performed by the worker nodes are returned to the queen, where they can be combined and delivered to users.

The loader nodes do the processor intensive aspects of load and export: organizing, transforming and reformatting data. In the case of loading, this includes the generation of hash keys; compression; and, the full preparation of data to be stored in the database. In the case of export, this includes decompression; and, formatting for output. The loader nodes also do the high volume input from data sources and the high volume output to data targets such as users, reporting systems and outboard applications.

Finally, backup nodes can be considered a new fourth tier. Backup nodes store compressed data dispatched in parallel directly from worker nodes. Backup nodes are usually configured with high capacity disks to reduce the cost per unit of backup storage.

The number of nodes of each type in a given installation can be chosen independently to optimize the workload. So, if an installation has a large volume of load and extract, it can be set up with more loader nodes. If the load volume is light, but the analytic workload is heavy, nCluster can be configured with more worker nodes and fewer loader nodes. Loader nodes do not have disks and consequently cost less than worker nodes. The specialization of node types confers an advantage. In general, it means that the number of nodes of each type can be chosen for the customer’s particular requirements for throughput, response time and data availability. For example, it may be more economical to configure systems for a high volume, near real time data load requirement by deploying additional loader nodes. By contrast, in a standard MPP architecture – in which there is just one type of node – it may be necessary to configure the entire system for the peak load requirement, resulting in a comparatively higher cost.
2.2 Design for Analytic Applications

Aster nCluster is designed specifically for analytic applications – and for large scale customer needs.

That is, the designers looked beyond the common “reporting data mart” and “enterprise data warehouse” paradigms that have typified commercial data warehousing.

Rather, nCluster’s designers looked to a new generation of analytic applications that feature one or more of:

* Large data volumes
* Intensive computation
* Complex algorithms
* Unstructured data; and/or,
* Interactivity (near real time data modifications).

A characteristic of many such applications is that their requirements grow more or less continuously. Thus, systems to support them must be able to expand – virtually without limit – to handle ever larger data volumes and diverse computational workloads. And so Aster has worked to provide such a product in nCluster.

The diversity of workloads requires graceful handling of analytic applications with varying characteristics. Ad-hoc analytic queries typically involve statisticians and data miners running analysis of large structured/unstructured data volumes using complex algorithms. Reporting can range from simple to complex SQL used by SQL developers (for complex reports) and business executives (for dashboard summaries). Finally, interactive workloads introduce a new category of analytic applications that focus on “real-time” on-demand analytics. These high-concurrency, low-latency workloads involve up-to-the-second trickle-feeds and query modifications (DDL, DML statements). Leveraged for iterative SQL development, near real-time analytics (eg. trickle-fed market data for hedge fund trades), and packaged analytic applications, interactive workloads are a growing class of dynamic low-latency workloads required in scalable data warehousing. Aster’s flexible distributed systems architecture is uniquely designed to handle all of these workload classes.

The off-the-shelf commodity MPP architecture employed by Aster and others has three fundamental strengths from a hardware perspective: processor capacity, memory bandwidth and I/O bandwidth are all inherently scalable. “Bandwidth” is a measure of the rate at which data can be moved to or from a device, such as the main memory of a server or a disk drive.

Capacity is added to the architecture by adding nodes. For example, each worker node has processors; dedicated memory; dedicated disks; and, dedicated paths to disk. So, adding processor capacity (that is, adding a node) always adds memory bandwidth and disk bandwidth in proportion. This is one of the fundamental advantages of shared nothing over shared memory and shared disk architectures: processor capacity, disk bandwidth and memory bandwidth remain in balance as the configuration is expanded or contracted.
Further, in the MPP architecture, more nodes can, in principle, always be added. (In practice, each version is certified up to some maximum number of nodes, presently in the hundreds. Aster will expand this maximum over time, aiming to keep it large enough so that it is not a practical concern for customers). So, a second fundamental advantage of the MPP architecture is that there is no fixed upper limit in I/O capacity or processing power.

Both of these fundamental advantages are in contrast to an SMP architecture in which: (a) processor capacity, memory bandwidth and disk I/O bandwidth can be independently varied and are not necessarily in balance for a given application; and, (b) there are fixed upper limits on all dimensions of capacity.

2.3 Use of Standard, Low Cost Components

A major challenge in large scale analytics is cost. At hundreds of thousands of dollars per terabyte of data for most workloads – the price of several widely used analytical platforms – cost can quickly become an insurmountable hurdle. When the required investment is large, a major business opportunity may be lost simply because the upfront investment required cannot be justified with enough certainty.

Understanding this, and yet recognizing that analytic requirements are constantly growing in scale, Aster decided to build nCluster to run on low cost, commodity hardware such as standard rack-mount or rack-optimized servers with direct-attached storage (DAS). This helps the customer control cost and further frees the customer to use the hardware that his or her enterprise has chosen as a standard. Aster’s goal is to deliver nCluster based solutions at a fraction of the price of comparable solutions on the most widely used data warehouse platforms.

At the time this paper was published, in the third quarter of 2009, the typical nCluster worker node required less than $10,000 of hardware and managed 1-2 terabytes of data. For example a typical 10-node nCluster with Dell PowerEdge servers would cost $70K in hardware (10 servers x $7K per server) and provide 24TB of disk capacity (300GB x 8 disks per server x 10 servers). Typically, a customer would start with a 2-3 node nCluster and scale-out to tens -- or sometimes one hundred or more -- of servers over 1-3 years.

The use of low cost components is applied in the use of standard servers; disks; network cards; network switches; and, underlying software. To this point, Aster systems have been deployed by customers on servers from Dell, HP, and IBM.
The database engine used in the nCluster nodes is PostgreSQL, a widely used open source product. The default operating system is Gentoo Linux, although Aster also supports other Linux OS distributions including RHEL (RedHat Enterprise Linux), CentOS, Ubuntu, and SUSE. This allows IT organizations to maintain their corporate-standard Linux distribution and install Aster nCluster software on top of the OS.

Aster has invested in creating unique value in nCluster. However, this investment is concentrated in the software layered above the database engine. The designers of nCluster adopted the view that the basic problems of building a relational database engine had been solved well enough over the preceding 35 years. They have envisioned an architecture in which the database engine is deployed on hundreds or thousands of nodes—and the major opportunity for innovation concerns how those many database engines are coordinated, used and managed as one system to accomplish complex, large scale queries, analyses and loads.

2.4 Addressing The Network bottleneck

Section 2.2 describes the inherent scalability of MPP architectures, which eliminate fixed upper limits in processor capacity, memory bandwidth and disk I/O bandwidth. But, there is still one hardware limitation in many MPP architectures: network bandwidth.

In comparison to standard MPP designs, nCluster offers an additional network advantage: the separation of loader nodes and worker nodes. As a result of this separation, inbound data enters the system over links that are not being used to process queries. Similarly, outbound data leaves the system over links that are not being used to process queries. Particularly in applications that continuously or frequently load or export large volumes of data, this design serves to increase the effective network bandwidth available for query processing—an important benefit. In larger systems, this effect can be enhanced by dedicating entire racks and/or switches to the loader nodes, thus providing them with an entirely separate scalable network infrastructure.
2.5 Data Redistribution

In analytic applications, data redistribution is a common operation which can stress the capacity of the interconnect.

An example of a situation requiring frequent data redistribution is as follows. Suppose that the sales table is partitioned on its primary key, salesid. And, suppose that the customer table is partitioned on its primary key, customerid. Suppose that the two tables are linked because each sales record contains the customerid for the customer who made that purchase. A ten node system with four cores per node would probably have 40 partitions of each table. This database design is illustrated in Figure 3.
Assume there are one hundred million customer records of 10,000 bytes each and five billion sales records of 200 bytes each. Then there is one TB of customer data and there are 10 TB of sales data.

Now consider the query, “Return the customer data for all customers along with the data for all purchases by each customer.” This is a simplified version of a common type of query. The output of such a query might be sent to an application that analyzes each customer’s purchases and demographics and decides from them what promotional message to send to that customer.

To satisfy the query, however, the analytic engine has to bring each customer’s data together with the data about that customer’s purchases (data in the sales table). However, the purchase data is not mainly in the same partition – probably not mainly even on the same node – as the customer data it relates to.

In order to bring a customer and its data together in one place, an analytic engine will typically repartition one of the two tables. The simplest plan calls for sales to be repartitioned on customerid, after which each customer’s purchases would be located near the data for that customer.

In our 10 TB sales table with 40 partitions, 97.5% of the data on average in each partition will be moved across the interconnect in such an operation. Thus, the redistribution processes associated with each partition will send and receive 243.75 GB of data in order to accomplish the operation. To minimize the elapsed time, all of these data movements will be done in parallel. Thus, redistributing this table results in the movement of a total of 9.75 TB of data by more than a thousand concurrent processes, each moving an average of 6.25 GB of data. This type of redistribution operation – moving entire tables or fractions of tables over the interconnect – occurs frequently in an analytic database of average complexity.

**Figure 4: Data Redistribution in a Massively Parallel Architecture**

Now customer #42891’s purchases are all in the same partition — and in a location close to the matching customer record.

Before customer #42891’s purchases could be anywhere in the sales table, since their location is determined by salesid, not customer id.
Therefore, it is a key requirement of an analytical engine to be able to efficiently redistribute data, even as the system grows to have many nodes and the volumes of data to be redistributed grow large. The larger the scale and the higher the complexity, the more critical this requirement becomes. The requirement is also intensified by the need to service multiple concurrent queries.

Compression.

$n$Cluster employs data compression for internode transfers of data. That is, data is compressed by the sending node and uncompressed by the receiving node. Since data flowing over the interconnect is compressed, this has the effect of further increasing interconnect bandwidth in comparison to most other analytic engines on the market. For example, if we assume 3-to-1 compression – a conservative estimate for many large tables – then redistribution of the 10 terabyte table described above will require moving 3.25 TB (rather than 9.75 TB) of data over the interconnect: an effective 3-to-1 expansion of interconnect capacity.

Dual Optimizers: Aster Data has designed a 2-stage optimizer to increase query performance and avoid unnecessary data shuffling over the network. The first stage is a Global optimizer residing in the queen-node layer which provides a global plan including which steps are required, what order they should occur, and what network transfers (if any) are required. The second stage is a local optimizer that runs on each individual Worker node. This deferred local planner provides fine-grained optimization based on the specific characteristics of the node and its data.

2.6 Advanced partitioning

Each unit of parallel operation in a shared nothing architecture is responsible for one part of the database, customarily called a partition. It turns out that the methods used in database partitioning – and the forms of partitioning supported – are quite significant in MPP architecture, and influence performance, scalability, availability and manageability. The significance of good partitioning increases with the size of the database.

From its first release, $n$Cluster has had strong partitioning capabilities – a significant product strength.
Top Level Hash Partitioning.

Hash partitioning is used at the top level for any partitioned table in an nCluster database. That is, a hash function is applied to the partition key for each row to yield a numeric value. This value is used to select a partition of the database where the row is physically stored.

Later, if the system receives a request for the row with the same partition key, the hash function will be applied again, yielding the same numeric value and causing the system to look in the same partition for the row.

With a well chosen partition key, rows will be distributed in a roughly uniform fashion among the partitions. There will be one partition for each unit of parallelism in the system – typically, one for each processor core. By requiring hash partitioning at the top level – and therefore achieving uniform distribution of the rows of most tables among the units of parallelism – nCluster ensures good parallelism on many database operations.

Each table can have multiple levels of partitioning – as many as may be required to produce the desired performance. Below the top level, partitioning can be by hash, range or list at the option of the database administrator.

Multi-level partitioning can be useful when the data in large tables is frequently segmented or aggregated by a few well known attributes. For example, it may be useful to partition a billion transactions first by salesid; then by month; then by store. In all cases, the top level of partitioning is by hash. In this example, the second level is by month, where month would probably be represented by a number and set up as range partitioning. The third level is by store, most likely using store names and list partitioning. This is illustrated in Figure 5.
Assuming a billion rows and ten hash partitions at the first level, each top level partition contains one hundred million rows. Assuming 60 months and 100, then the average fourth level partition contains 1,667 rows. So, a query that concerns the electronics sold at the Kansas City Store last month would require that only an average of 1,667 rows be read by each of the database instances in the system. Each instance would be reading from only one fourth level partition, so the disk I/O would be serial, and therefore faster than random I/O. Further, this operation would be performed ten ways in parallel. This is approximately sixty thousand times faster than reading the entire billion row table. This optimization is possible because of the advanced partitioning supported by nCluster.

Note also, that nCluster allows the use of hash, range and list partitioning at the second level and all subsequent levels. The choice made at one level does not constrain the choices at subsequent levels. This makes it possible to suit the partitioning to the data and its characteristic uses.

Some data warehouse engines support only one or two levels of partitioning and therefore do not offer a comparable performance advantage in this case. Others have a limited menu of partitioning options or combinations and could not come close to the sixty thousand times benefit in this example and in many other cases.

Optional table replication.

The database administrator can optionally request that a table be replicated, rather than partitioned. With this option, a complete copy of the table is stored by the system in each top level partition. When the table is changed, nCluster automatically propagates the changes to each replica of the table. Table replication is a good choice for small tables that are frequently used in queries – and that are not frequently changed. An example would a table of the states of the United States. The state table has only 50 rows. Replicating it will not take much space but will save time on queries where information about the state – or the name of the state – must be joined to another table, such as customer or store.

“Between” Partitioning and Replication.

For most tables, either partitioning or replication will be the obvious design choice. Tables such as state and country – which have fewer than a thousand values – will often be replicated. Tables such as customer, which can have millions – or even hundreds of millions -- of values in a large business, will often be partitioned.

But for some tables, which are large but are frequently joined with certain other tables, it may seem that both partitioning and replication are desirable. nCluster offers two answers to this. First, nCluster does permit the database designer to choose to both partition and replicate a table. This may sometimes be the right choice. Second, nCluster is capable of caching redistributed data in the destination partition and exploiting this in subsequent queries. This may be thought of as combining the advantages of partitioning and replication.
2.7 Data Availability

High data availability is built into nCluster at a fundamental level of the design.

First, each top level (hash) partition is replicated on a peer node. Thus, if each worker node has four top level database partitions as its primary data, then it also has four replicas of partitions from other nodes. Updates are propagated synchronously at commit time. Thus, the application does not see a response from a commit request until the moment when nCluster can guarantee that both the primary copy of the data and its replica have been updated.

Aster’s design for low-latency replication provides for commits to occur within milliseconds of a change to the primary copy of the data. As a result, Aster aims to provide high-concurrency, low-latency queries and trickle-feed loading, with maximum transactions per second approaching that of OLTP databases.

Peer replication assures that data remains available for use, even in the event that a node is lost. When a node fails, the replica of each partition of the failed node becomes the primary active copy. Processing of queries continues uninterrupted. The queen automatically redirects new queries as required. Statements that were in progress on the failed node at the moment of failure are automatically restarted by the queen without any user-perceived disruption or error.

A distinctive feature of nCluster is this: only the statements in progress on the failed node need to be restarted, and this is done automatically by the queen. In some MPP systems, node failures require that the entire database system be restarted; and that all queries in progress on the entire system then be restarted (rather than just the query statements that were running on the failed node). Further, some systems require that the user or application restart any failed queries.

Aster nCluster differs in that: (a) no system restart is required after a node failure: the system keeps on running; (b) the system will simply continue to process queries; (c) only the portions of queries (the local subquery statements) running on the failed node at the moment of failure need be restarted; and, (d) the system restarts such failed statements automatically. As a result, the volume of work that must be re-processed in the event of a node failure is much lower than with typical MPP designs. In addition, there is little or no interruption of service with a node failure and no action is required of users.

Aster has also focused on avoiding “planned” downtime. In particular, Aster nCluster will continue to:

* Process queries while loading fresh data;
* Process queries or loads while backing up data
* Process queries or loads while exporting data.

Further, Aster nCluster requires no downtime for either restoring replicas after node failure or redistributing data after new nodes are added.
2.8 Manageability

As analytic systems grow large and come to encompass many components, they can become increasingly difficult to manage.

For example, on a small system with a few tables and a few applications, it may be a simple matter for the database administrator to analyze performance problems and respond to them. If two frequently accessed tables are on the same disk drive, this is readily observed and it is not difficult to move one of the busy tables to another drive.

But, change the scale – so that there are hundreds of applications; hundreds of tables; and a thousand or more drives – and it is no longer simple to detect; analyze; or, correct the problems. Highly parallel architectures -- with their techniques for distributing work over multiple processors and spreading tables and other structures over multiple drives -- can increase the difficulty of understanding the system and its performance behavior. The problem of managing the system -- its ever changing demands -- its behavior -- and its resources -- can quickly escalate beyond the capabilities of unaided humans to manage.

Additionally, the use of low-cost standard commodity hardware in large scale MPP systems creates new supportability challenges. In a system with many commodity nodes and disks, failures will occur -- and, failure recovery can easily escalate to become a management nightmare. The ability to gracefully recover from errors and physical failures becomes increasingly important as the system size increases.

Complexity aside, the sheer cost of managing larger systems – in staff time – can become a barrier to effective use. The owners of some data warehouses report that they must employ multiple DBAs (database administrators) just to maintain and tune indexes, aggregates or other artifacts of performance in large scale analytic systems.

Thus, industry experience has shown that manageability is a key aspect of scalability: even a platform that typically delivers good performance on large analytic databases will have limited utility unless it can also make such systems manageable.

And where does manageability come from? Primarily from three sources: (a) inherent system simplicity; (b) automatic management of complexity by the system itself; and, (c) tooling to help people with the interventions and decisions that require human judgment.

Aster approached the development of nCluster with recognition of this requirement for manageability in large scale analytic systems.

First, the design of nCluster puts a strong emphasis on the first point: inherent system simplicity. The number of physical database design decisions required is minimized: the primary physical design choices for a table are the choice of a partitioning scheme and partitioning keys.

Second, nCluster is largely self-configuring: via autosensing, the system can set itself up on a cluster following the entry of a MAC address (unique hardware identifier) for one component. In fact, a GUI-based administration console allows for “one-click” scale-out (adding nodes) and scale-in (removing nodes). Complexities such as re-tuning, re-indexing, re-partitioning are all handled behind the scenes for ease of use and simplified automation.
Third, Aster says that all management functions are performed via a single, web based, graphical user interface in which the database administrator is dealing primarily with a single system image.

Fourth nCluster includes several features designed to simplify management and increase system resilience, including: automatic self-healing for transient software/firmware errors; incremental (delta) replication to restore the system to “multiple copy” state after a node failure; and, extensive historical log data to aid in diagnosing and correcting problems.

Finally, Aster nCluster supports mixed workload management and dynamic workload prioritization. The nCluster administrator can specify a user, priority, and “fairshare” (ratio of # of queries or time allowance relative to other users in the same priority bucket) to govern when queries are accepted by the system for execution. By controlling priority and fairshare, the administrator can increase performance predictability for each of several classes of work.

A dynamic workload manager provides distributed systems resource provisioning of processor and disk I/O to each “service class”. A service class is comprised of workloads with a given set of rules, termed a “predicate”. The predicate can range from coarse to fine-grained (eg. all non-modify SELECT statements from user Tom from IP address xyz that accesses the ExecutiveDashboard table in the Executives database from 9am to 5pm weekdays will get “highest” priority). All of this can be defined and updated using standard SQL or a GUI-based console.

Dynamic workload manager can re-prioritize workloads currently running in the system. For example, assume CPU/disk resources have been allocated to workloads A, B, and C. If a higher priority query is then submitted, dynamic workload manager will automatically re-prioritize workloads-in-flight so that the new higher priority query gets performed quickly.

In another scenario, assume workload C is a data mining query that is taking longer than anticipated. Dynamic workload manager can throttle down the distribute compute and disk I/O resources on-the-fly to allow other workloads to progress more quickly.

Thus, Aster has paid attention to manageability issues throughout the development of nCluster, aiming to provide a system that remains manageable even as the number of nodes grows large.
Other Key Features

Section 2 of this paper focused on the fundamental design features of Aster’s shared nothing MPP architecture, outlining basic strengths and highlighting innovations within the foundation of the product.

The present section discusses some additional features of particular interest.

3.1 MapReduce

Aster nCluster features support for MapReduce, a popular and powerful facility for in-database analytics. That is, instead of exporting data from nCluster to a separate system for analysis, MapReduce enables programmers to perform a wide range of analytical tasks, in parallel, directly in the database where the data resides. With large volumes of data, a great deal of time and resources can be saved because the data does not have to be moved to another system. Further, all of the parallel infrastructure of the nCluster database system is brought to bear on behalf of the programmer working in MapReduce, as opposed to the typical serial processing that occurs on a mid-tier application running on a single SMP server outside the database tier.

Definition of MapReduce.

From the paper, MapReduce: Simplified Data Processing on Large Clusters, by Jeffrey Dean and Sanjay Ghemawat of Google:

MapReduce is a programming model and an associated implementation for processing and generating large data sets...Many real world tasks are expressible in this model.

Programs written in this functional style are automatically parallelized and executed on a large cluster of commodity machines. The run-time system takes care of the details of partitioning the input data, scheduling the program’s execution across a set of machines, handling machine failures, and managing the required inter-machine communication. This allows programmers without any experience with parallel and distributed systems to easily utilize the resources of a large distributed system.

MapReduce provides a facility for analyzing data on a large scale and in parallel, complementing the capabilities of SQL, the structured query language employed in nCluster and most other modern relational database systems.
Contrast to SQL.

The emphasis in SQL is on nonprocedural access to tabular data -- defining what data is wanted rather than how to navigate to it. SQL was developed to free programmers and other users from the details of navigation and data structure, where the data to be used is readily organized into tables: structures of unordered rows and fixed, named columns. SQL, developed in the 1970’s by IBM, has become an industry standard and grown through the contribution of many people, so that it now provides good solutions to data definition and access for a wide range of data types and analyses. Further, in parallel database systems such as nCluster, those who create SQL requests for data are freed of concern for how the data is partitioned; how parallel execution is to be efficiently accomplished; as well as how failures are handled. All aspects of inter-machine communication within the system are hidden from the user of SQL. Thus, the advantages attributed to MapReduce in the above paragraph from the Google paper are already present for users of nCluster and most other MPP database systems, via SQL. All such advantages are available within such systems before the introduction of MapReduce.

Why MapReduce in a Database System. However, no single language provides the best solution to every conceivable data analysis problem. Certainly no single language can address the preferences of every variety of user.

MapReduce now has a rapidly growing following in information technology, particularly for applications that require procedural computation (i.e., computations not readily expressed in a single formula applying to a row or a set of rows -- and therefore requiring a multi-step procedure to define them). Further, when a complex procedural computation must analyze data having a structure that is not straightforwardly or efficiently defined in SQL (due to the lack of efficient turingcompleteness), then the complications of working in SQL are compounded.

Aster’s Approach.

To address both the growing popularity of MapReduce and the need to address requirements for computations and data structures that may be awkward in SQL, Aster has built a MapReduce capability directly into nCluster. In addition, Aster has made the MapReduce functions seamlessly callable from SQL as an SQL extension. This allows the user to apply the capabilities of both SQL and MapReduce to the solution of a problem – a nice touch, especially for DBAs and SQL developers that are highly proficient and experienced in SQL but less so in procedural languages like Java or C++.
Examples.

For example, consider the case of a database in which each record contains several structured attributes (time, location, size, weight, color, etc.) of a chemical sample followed by a large, unstructured data object containing a waveform. With nCluster one could select the set of records relevant to a given analysis applying the powerful SELECT statement of SQL to the structured attributes. Once the records were selected, one could analyze the waveforms retrieved with MapReduce, all done in a pipelined, single pass over the data set. This analysis might produce structured data results, which could then be written back to the database with SQL. In fact, the power of this integrated SQL/MapReduce approach goes beyond structured database formats and extends to unstructured files. Using the example above, the unstructured waveforms need not be stored as BLOBs in a database but could in fact be stored as unstructured raw file objects in the distributed nCluster system. In a single pipelined query, standard SQL could select filtered data from structured schema tables and feed them into a MapReduce function that combines this structured data with the unstructured waveform files to yield structured (or unstructured) query results.

Another interesting scenario concerns the sharing of data among analysts in different disciplines. Thus, a group of data analysts fluent in SQL might share a database with a group of engineers or statisticians more comfortable with MapReduce and related analytical tools. With nCluster, both groups can use the same data; each using the tool it prefers to access the data.
3.2 .NET

In June 2009, Aster introduced .NET (pronounced dot-net) support for nCluster. This means that customers developing in the Microsoft environment can use nCluster, employing both SQL and MapReduce. nCluster previously supported Java, Python, Perl and C++ for such functions. With the addition of .NET support, nCluster also supports in-database analytic functions written in C#.

Definition of .NET.

According to the Microsoft .NET developer’s guide:

The .NET Framework is an integral Windows component that supports building and running the next generation of applications and XML Web services. The .NET Framework is designed to fulfill the following objectives:

* To provide a consistent object-oriented programming environment whether object code is stored and executed locally, executed locally but Internet-distributed, or executed remotely.
* To provide a code-execution environment that minimizes software deployment and versioning conflicts.
* To provide a code-execution environment that promotes safe execution of code, including code created by an unknown or semi-trusted third party.
* To provide a code-execution environment that eliminates the performance problems of scripted or interpreted environments.
* To make the developer experience consistent across widely varying types of applications, such as Windows-based applications and Web-based applications.
* To build all communication on industry standards to ensure that code based on the .NET Framework can integrate with any other code.

The .NET Framework has two main components: the common language runtime and the .NET Framework class library... You can think of the runtime as an agent that manages code at execution time, providing core services such as memory management, thread management, and remoting, while also enforcing strict type safety and other forms of code accuracy that promote security and robustness...

The class library, the other main component of the .NET Framework, is a comprehensive, object-oriented collection of reusable types that you can use to develop applications ranging from traditional command-line or graphical user interface (GUI) applications to applications based on the latest innovations provided by ASP.NET, such as Web Forms and XML Web services.

WinterCorp believes that Aster nCluster is the first system to offer MapReduce in the .NET environment, an important software development environment for the web, both in small and large companies. In addition, nCluster provides an MPP architecture relational database system supporting SQL for the .NET environment – one of the few such products available on commodity hardware.
3.3 Appliance Offering

Also in June, 2009, Aster introduced its MapReduce Data Warehouse Appliance, which offers in one integrated, pre-configured and pre-tested package:

* Aster’s nCluster SQL/MR database;
* Dell hardware;
* Microstrategy, a leading business intelligence platform for query, reporting and OLAP (online analytical processing); and,
* Aqua Data Studio for GUI-based database administration and data modeling.

Note that the Aster MapReduce Data Warehouse Appliance is built entirely of standard components available separately, distinguishing it from several other appliance offerings in the industry.

Moreover, this is the first appliance to natively package MapReduce analytical power out-of-the-box. As we are in the formative years of MapReduce, the initial deployments have primarily required large numbers of technically-savvy software developers to set up, administer, and maintain open-source MapReduce systems. The Aster MapReduce Data Warehouse Appliance offers pre-canned MapReduce algorithms (called SQL/MR functions), a Software Development Kit (SDK) and APIs, sample retail industry data set, and extensive documentation. As a result, customers are presumably able to start using MapReduce out-of-the-box within hours, as opposed to days or weeks with highly technical open-source MapReduce.
4.1 ShareThis

ShareThis (www.sharethis.com) is an Internet-based company that provides services to help people share online content. The company offers a widget (Figure 7) that makes it easy for you to share online content with your friends and associates.

Figure 7: ShareThis Widget with Example of its Use by a Publisher
Anyone who creates web content can make the widget available on their website. Individuals can also have the widget constantly available to themselves via a browser plugin.

When any user clicks on the widget anywhere in the world, he or she receives automated assistance in making the related content (e.g., a photo, an article, a video) available to contacts via social networks, email or other means. Shared content is also logged in the user’s “Share Box” and can be easily accessed again (e.g., if you remember later that you want to share that item with someone else; if you see your boss and he or she doesn’t remember seeing it, so you want to send it again or send it a different way...). In general, ShareThis facilitates the sharing of web content among networks of people.

As a result of this sharing activity, the ShareThis company ends up with a database that exceeded 100 million page views a day, as of a recent interview with WinterCorp. In that interview, Lenin Gali, Director of Business Intelligence at ShareThis, described the company’s interest in analyzing the most recently received several months of this data. Reporting and analysis on the data makes it possible for the company to track how much use is being made of its services; which content is being shared; frequency of content sharing; trends in content sharing; and, so on.

The 15 TB of data that ShareThis carries in its data warehouse today is likely to grow rapidly as the business grows. According to Mr. Gali, they started with an implementation of the data warehouse on Oracle, found it took too long to load the data, and decided it would become too costly as the data volumes grew larger. At the time he began looking for an alternative solution, Mr. Gali said it was taking 4-6 hours a day to load the data on Oracle.

When Mr. Gali invited Aster to tell him what they could do, Aster volunteered to perform a proof-of-concept. They came back in a week with a 4-node solution that could load a day of data in 30 minutes – and that ran the queries faster than they had run on Oracle.

A key requirement was to build the company’s infrastructure in the Amazon cloud. Aster demonstrated that it could do this and ShareThis presently runs its nCluster data warehouse in the cloud. Mr. Gali feels that one of the greatest strengths of nCluster is the ease with which it can be set up and used. He and one other associate support the use of the data warehouse by the entire company. The hardware and system software support is provided by Amazon as an integral element of its cloud services. Mr. Gali can set up an Aster cluster in the cloud himself. He can expand the cluster with a new node very simply; new nodes will be populated automatically with data within a few hours.

At the time of our interview with Mr. Gali, ShareThis was running a 10-node nCluster in the Amazon cloud for production. For the hardware nodes, he was paying $400/node/month. For the storage he was paying $100/ TB/month. Aster delivered the nCluster software to his systems in the cloud for a separately negotiated price. He also has development and test systems in the cloud, which he pays for only as they are used, which consequently makes them very cost effective.

Mr. Gali feels that the ease of administration of nCluster, along with the flexibility of operating in the Amazon cloud, provides a very good solution for a small company such as ShareThis, faced with formidable data management requirements which are likely to grow rapidly – and to a much larger scale.
4.2 MySpace

A leading social networking and entertainment site, MySpace generates 3 to 4 terabytes of data each day concerning activity on its website by its approximately 130 million unique visitors. To effectively manage its business, MySpace must be able to analyze this data for insights into customer preferences; online customer experience; and, customer activity, such as which videos a customer watched or which music a customer listened to. Furthermore, as a result of the rapid rate at which trends grow, peak and decline on the internet, MySpace needs the results of this analysis rapidly, often seeking to react to new developments in the same day they occur.

MySpace was aware that its requirements were very demanding: the website users generate about 10 billion events a day, which must be analyzed as rapidly as possible after they occur.

In addition to enormous data volumes, for its clickstream data warehouse MySpace needed:

* A scale out architecture that could grow incrementally along with MySpace requirements;
* An architecture that could run on hardware compliant with existing company standards;
* A powerful analytical capability;
* Continuous, 7 by 24 operation; and,
* Low data latency, to address near real time query requirements.

After an examination of alternative architectures, MySpace selected Aster nCluster. A strong consideration was a company commitment to hardware vendor independence – in fact, the MySpace data warehouse uses heterogeneous hardware, with servers coming from both Dell and HP. Further, MySpace wanted to avoid the cost of a storage area network. It was very important to MySpace that nCluster not only had the needed capabilities, but could deliver them on standard, commodity hardware. In fact, overall, MySpace put a strong emphasis on having a cost effective solution, realizing that it was going to be implementing a very large system.

After a pilot system that proved nCluster would work in practice with the data from MySpace Video, the company implemented a total of 4 clusters which together meet a range of needs for volume data loading, real time data collection and analysis, business intelligence dashboards, in depth analysis and end user query and reporting with Microstrategy. Of the four clusters, one is used for development, one for staging, one for the data warehouse; and one for the data marts.
As a result of their Aster nCluster implementation, MySpace has new capabilities that were out of reach before, due to the rapid growth and large scale of their operation. The MySpace system architecture, with Aster nCluster as the platform for the data warehouse and the data marts, is shown in Figure 8.

For example, MySpace now gets timely data on which new features and content are getting used and therefore can react rapidly to focus development resources on the items that customers value the most. They are now able to conduct nearly real time A/B tests of experimental features. That is, they can make a feature available to a control group and rapidly measure the behavior of customers who do (the “A” group) and do not (the “B” group) have access to the feature. Both of these capabilities contribute to the agility and payoff of MySpace development efforts. As another example, MySpace now has greatly enhanced customer retention capabilities. They can now readily identify customers who have not been active – or have been making shorter visits – and take action when appropriate to encourage such customers to use the site more.

Figure 8: System Architecture at MySpace
(Source: Hala Al Adwan, MySpace, Gartner BI Conference, 2009)
Cluster is used to calculate the royalties due to each music label, based on the frequency and duration of each piece of music played in each time period. This information also helps MySpace to increase its focus on specific varieties of music and/or artists that its customers want to hear. This is a key capability as the company moves to increase its emphasis on MySpace as an entertainment destination.

In discussing the MySpace use of nCluster at the Gartner BI Conference, Hala Al-Adwan, Vice President of Data at MySpace, emphasized several points.

First, that MySpace chooses to move very rapidly in implementing systems – this is part of their organizational style and a critical element in the fast moving world of social networking on the internet. Their initial acquisition and implementation of a data warehouse based on nCluster was accomplished in 8 weeks – an unthinkably short period of time for most larger companies and a challenge for a company of any size. Ms. Al-Adwan reported that MySpace went from having 0 nCluster nodes onsite to having 90 installed and fully functional in six weeks. She noted that this required outstanding performance from the Aster organization but also involved an exceptional effort on the part of several groups within MySpace. Ms. Al-Adwan also stated that the data warehouse support for MySpace Music was implemented on nCluster in “about two weeks, an amazing feat”.

---

**Figure 9: Some Uses of the MySpace Data Warehouse on nCluster**
(Source: Hala Al Adwan, MySpace, Gartner BI Conference, 2009)

| Web Analytics                     | • Provide insight to 130 million monthly uniques  
|                                  | • Search engine referrals: pageviews, uniques  
|                                  | • A/B split tests for improved usability  
| Executive Reports                | • Executive dashboard: C-level and product team  
|                                  | • MySpace Music royalty reports for music labels  
|                                  | • MySpace TV Dashboard  
| Advanced Analytics               | • User retention queries  
|                                  | • More  

"nCluster is used to calculate the royalties due to each music label, based on the frequency and duration of each piece of music played in each time period. This information also helps MySpace to increase its focus on specific varieties of music and/or artists that its customers want to hear. This is a key capability as the company moves to increase its emphasis on MySpace as an entertainment destination."
Second, Ms. Al-Adwan said that she chose Aster in part because she believed they would make a good partner for MySpace in an extremely aggressive, very large scale implementation. She wanted a partner that would be “as committed as we were to a successful outcome”. That has indeed proved to be the case in a partnership that is now about fifteen months old. She reports that Aster has provided steady, extremely responsive support as new challenges have surfaced and new problems have had to be solved, summing up her view of Aster as “amazingly supportive”.

Don Watters, Principal Architect, adds that the partnership extends into the realm of strategy, to include the mutual sharing of roadmaps. Aster has been responsive the strategic requirements of MySpace, shaping its product direction to make sure that MySpace can continue to succeed in its data warehouse program. In a variety of ways, Mr. Watters feels that Aster has demonstrated its commitment to the success of the MySpace data warehouse.

Third, Ms. Al-Adwan emphasized that much of the special value of the system derives from the “advanced analytics” that MySpace is able to run on nCluster. In addition to using the SQL based capabilities of the platform, MySpace has made heavy use of the MapReduce facility now built into nCluster, as well as nPath, A MapReduce function for time series analysis now built into nCluster.

It is the integration of SQL and MapReduce that produces the key advantage here. For example, in calculating the time a user spends on the site, MySpace first uses SQL to select the set of page views comprising a given user session. It then uses MapReduce and nPath to calculate the associated times – a calculation which is expressed as a record-by-record operation. With either SQL alone or MapReduce alone, this would be a far more complex calculation and far more difficult to express. SQL doesn’t have the procedural looping construct which would make it straightforward to express the calculation. MapReduce doesn’t have the powerful set selection expressions that make it straightforward to identify the data to be operated on.

Further, Aster’s parallel implementation of SQL/MR means that the calculation can be expressed by means of a straightforward Java procedure, but be executed in parallel on the data across all the nCluster worker nodes.
MySpace develops in the Microsoft .NET framework and, partly in response to MySpace’s interest in this, Aster has delivered .NET support. Also of interest, Ms. Al-Adwan said, was nCluster’s forthcoming capability to expand the cluster and redistribute the data over the expanded configuration, while the system remained online.

For the next phase of development, Ms. Al-Adwan plans to expand the role of nCluster. Whereas currently the system is employed to understand and measure the behavior of website visitors, in the next phase nCluster will also be used to generate content for MySpace customers. Such content is expected to personalize the user experience; add value for the user; and, strengthen the relationship of the user to MySpace.

An example of this would be to provide a recording artist with a chart showing visits to the artist’s page; plays of the artist’s music; and so on – over time.

Or, the generated content might concern how many people shared certain interests, preferred songs, etc., with the user. These are examples in which the data stored and analyzed in nCluster is applied to add value to the experience of a customer visiting the site. Here, the scalability, performance and favorable economics of Aster nCluster are critical to the proposition that MySpace can beneficially employ the platform to do more than report on what has happened.

Overall, it is clear from the information provided by MySpace that Aster nCluster has played a valuable role to this point and promises to contribute in yet more significant ways in the future, as MySpace moves on to its next level of development as a leading Internet business.
Conclusions

Aster Data has entered the market for large scale data analytic platforms with nCluster, a product designed for scalability, high performance and high data availability.

Affordable.

nCluster runs on commodity hardware to make deployment convenient and affordable for its customers.

Innovative Design.

The system employs innovative design techniques and features to offer an appealing solution for the next generation of analytic applications, which Aster believes will be characterized by large and rapidly growing data volumes; in depth analysis; and, new analytic techniques. Aster also believes it will be increasingly important for analytic systems to be continuously available – both to accept new data and to respond in near real time to queries.

Advanced Analytics.

nCluster employs a nicely designed combination of proven approaches (e.g., massively parallel, shared nothing relational database architecture) and new techniques, such as the integration of MapReduce into its database engine. MapReduce extends the SQL-based capabilities of nCluster to address a broad range of analytical requirements, where the analysis can be performed on data in place within the database engine. Customers are using MapReduce to attack problems which would be more difficult to solve within SQL and to analyze data that is not as readily modeled in SQL.

Attacking the Network Bottleneck.

The design philosophy of nCluster represents a departure in the design of massively parallel analytic engines. From the outset, Aster has concentrated on addressing the network bottleneck that is the key limiting factor in many designs.

Designed for High Availability.

Aster has focused on high availability from the outset. For example, a node failure does not require manual intervention and does not require that users restart queries. Rather the system handles the failure and automatically restarts any subqueries that were in progress on the failed node.
Designed for Manageability.

Manageability has also been addressed from the beginning. For example, the system is designed to largely self configure on the basis of a single MAC address (the hardware address of one component, such as a server). Also, adding nodes or taking nodes out of service is a straightforward online operation. Upon the addition of nodes, the system automatically redistributes data to take advantage of the expanded hardware configuration.

Successful Customers.

Customers interviewed for this paper expressed high satisfaction with Aster Data and nCluster. These range from small companies running on a few nodes to provide services to fewer than ten users – to quite a large company – MySpace – operating four clusters, of which the largest was running on 102 nodes. All report that they have been successful in their endeavors with nCluster; that they like the product; and, that Aster has been an excellent partner – quick to respond and effective in its support.

Recommendation.

Overall, Aster’s nCluster brings a forward-looking, exciting new design for analytic data requirements. With its massively parallel architecture; its use of commodity hardware; its focus on scalability, availability and manageability; and, its rapid innovation via the integration of MapReduce and other features, Aster’s nCluster offers users a distinctive set of capabilities in a promising new design. Though nCluster is a young product, it has successful customers – including some with large scale, demanding requirements.

In the opinion of WinterCorp, users with challenging analytical requirements will want to evaluate nCluster as they select the platform for their next substantial implementation. As with any young product, nCluster will prove to be less than fully mature in some ways. Interested customers should therefore do a thorough evaluation accompanied by a realistic proof of concept, testing the performance of the platform under the most demanding conditions it will face in production. But through such an evaluation, WinterCorp believes that quite a few users will find that they like what nCluster offers them.
WinterCorp is an independent consulting firm that specializes in the performance and scalability of terrabyte—and petabyte—scale data management systems throughout their lifecycle.

Since our inception in 1992, we have architected many of the world’s largest and most challenging databases in production today. Our consulting services help organizations define business-critical database solutions, select their platforms, engineer their implementations, and manage their growth to optimize business value.

With decades of experience in large-scale database implementations and in-depth knowledge of database products, we deliver unmatched insight into the issues that impede performance and the technologies that enable success.

245 First Street, Suite 1800
Cambridge, MA 02142
617 695-1800

visit us at www.wintercorp.com

The Large Scale Data Management Experts