Apache Spark™ Graph Performance with Memory1™

February 2017

Abstract

Apache Spark is a powerful open source distributed computing platform focused on high speed, large scale data processing that leverages parallel task execution across a cluster of servers to achieve more timely results. Spark provides a set of Application Programming Interfaces (API's) for various analytics algorithms including Streaming, SQL, Machine Learning and Graph processing. These operations provide a robust framework for analyzing large and complex datasets.

Previous architectures relied on storage subsystems for storing the data set, with continuous operations needed to move data from storage to memory and back to storage. While this provides capacity for big data, performance is significantly hampered. Apache Spark was designed to run as much or all of the data set in application memory as possible, thereby reducing the costly trips to storage.

In order to provide adequate performance for Spark, a very large address space is required and processing capability per server is paramount. Servers are scaled out in an effort to provide a total aggregate memory footprint large enough to handle the initial data structures, interim information and associated metadata. As cluster size requirements grow, however, network latency creates performance bottlenecks, and associated capital expenditures and operating expenses soar.

Graph processing in particular, requires significant amounts of memory. K-core decomposition is a widely used algorithm in graph processing for community detection. It is used to analyze large amounts of interrelated data to detect patterns and relationships. While it has many practical applications, significant amounts of memory are required in order to traverse large amounts of data to harvest these patterns and relationships.

Memory1™ are DDR4 modules from Diablo Technologies that expand the available application memory in a system. Inspur Memory1 Servers deliver the per server capacity and performance needed to support graph processing algorithms such as Apache Spark k-core. With most data in memory, the burden on storage is significantly lower. Even when NVMe drives are used, storage is a major bottleneck.

This paper will illustrate that: 1) more work achieved per server translates into better, more efficient performance, 2) the cluster size required to process the data is reduced, and 3) expenses are contained.

Introduction

Many large data sets consist of objects and connections between those objects. Freeway networks among cities, flight patterns between airports, social media relationships and linked power grids can all be represented by vertices (objects) and edges (connections). The graph data structure consists of vertices and edges; they are used to represent a list of items and their relationships. The use cases are extensive, from protein modelling and traffic routing to web page ranking and targeted advertising.

In these data structures, vertices can have attributes (e.g. cities have populations, land area and density). Likewise, edges can also have characteristics specific to them (e.g. freeway speed limits, number of lanes, incline or decline grades). With data represented in such a manner, various analytics can be performed to examine these relationships, predict or propose future connections and suggest more efficient means of traversing the information.

Analyzing this information can yield important results and insights. GPS (Global Positioning System) information allows us to find the fastest or shortest routes from one location to another. Social media connections help us to find new relationships with people that have similar experiences or viewpoints. Energy grid layout analysis ensures that we continue to receive electricity even when power lines come down in a storm byrerouting pathways in the most efficient manner.

The number of vertices and edges grows quickly, leading to extremely large data sets. In addition, the analysis of all this information requires tens or hundreds times more temporary data to be created in the process. Since Apache Spark was designed to utilize application memory as much and as efficiently as possible, the amount of memory per server in the cluster has a direct impact on performance. As memory per server is scaled up, the analysis is able to run more efficiently.

As servers in analytics deployments run more effectively, cluster complexity is minimized. Less space is required and simpler management is achieved. There are other important savings as well. Power, and therefore cooling, are reduced, as well as networking. Capital Expenditures and Operating Expenses are, therefore, reduced, leading to an overall lower Total Cost of Ownership.

Apache Spark™ and Graph Computations

Fast Relationship Analysis

Apache Spark is an open source processing engine for analytics, providing libraries for SQL Queries, Machine Learning algorithms, Streaming and Graph processing. These modules run on top of the Spark Core engine, which provides distributed, parallel execution of RDD's (Resilient Distributed Datasets). RDD's are the fundamental data structures in Spark, allowing for computation on datasets to be distributed among different nodes in the cluster. These RDDs allow for the faster, more efficient processing of data than is otherwise achievable with typical MapReduce and other operations.

GraphX, the Spark API for graph and graph parallel computations, allows for fast, iterative processing on large datasets. This API provides a set of common algorithms for efficiently building, transforming and developing insights from graph structured data. GraphX extends the Spark RDD with a Resilient Distributed Property Graph, providing associations of relevant data to edges and vertices. Additionally, multigraphs are available that allow multiple properties (multiple edges of relationships) between vertices.

The GraphX library contains various functions for manipulating and analyzing graph data structures. Page rank, triangle counting and k-core decomposition are common algorithms for traversing relationship-based data. The concept of k-core decomposition was introduced to study the clustering of social networks and has applications toward many additional workloads. These communities rely on members remaining highly connected. Members that are less connected are likely to leave the community, further risking additional members.

A k-core is a subgraph for which all vertices have degree at least k. The k-core of a graph G is the largest induced subgraph of G for which every vertex has degree of at least k. The goal in analyzing these networks is to determine the "coreness" (or relevance) of each vertex within G.

Traversing and analyzing the relationships between objects to determine this "coreness" requires complex processing of very large data sets. A tremendous amount of interim data must be created during the graph processing, requiring copious amounts of memory. It is very common for a 200GB dataset to require 2TB or 3TB of memory. For maximum performance, this data should remain in memory until the operations complete, posing significant challenges. Lack of memory restricts how much data can be analyzed, as well as the length of time required, and leads to massive horizontal scaling of the cluster in order to access enough needed memory.

Diablo Technologies' Memory1

High Capacity Application Memory

Memory1 modules from Diablo Technologies are the first memory DIMMs to expose NAND flash as standard application memory. It is the first Storage Class Memory product on the market and provides high density memory for servers, with up to 4X more capacity than standard DRAM DIMMs. Memory1 modules are JEDEC standard DDR4 DIMMs that interface seamlessly with existing hardware and software. No modifications are required for processors, motherboards or applications.

The DIMMs are deployed directly into the DDR4 memory slots of a server and interface with existing processor memory controllers. Enterprise and cloud applications automatically leverage the additional memory. This expansion of memory per server is particularly beneficial for memory-intensive applications such as Apache Spark.

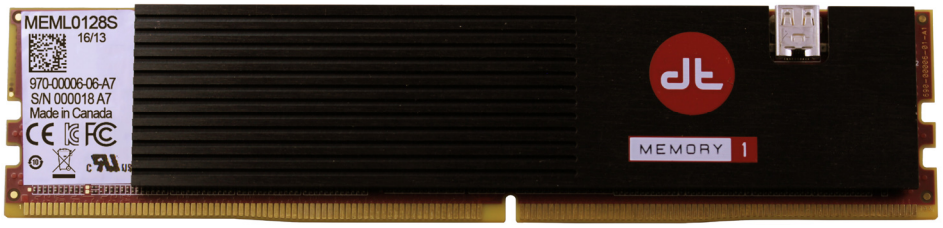

Figure 1: 128GB Memory1 DIMM

Diablo Memory Expansion (DMX) software is a combination of firmware and software that enables Memory1 DIMMs to expand the memory available to an application. The software creates a new tier, managing data movement between DRAM and Diablo flash memory. It intelligently handles application memory access, managing the performance and endurance of Memory1 modules.

Memory1 provides intelligent scaling, smarter data placement and unique flexibility by:

- Scaling up memory resources per node, requiring fewer servers for real time analytics, faster business decisions and more transactions completed in less time
- Dynamically allocating, managing and placing data across hybrid memory and system resources to balance price and performance
- Delivering the flexibility to address evolving business needs, applications and technologies

Inspur Memory1 Servers

System Details

Inspur Memory1 Servers are standard dual-socket x86 1U and 2U servers that are pre-tested and pre-qualified to work seamlessly with Memory1 technology. Pre-configured with 1TB or 2TB of Memory1, these servers can be stacked up to provide up to 40TB of system memory in a single full size rack. Inspur Memory1 Servers can be deployed in single server or cluster of 3 configurations, and scale up (from 1TB to 2TB) and scale out to single rack or multi rack levels.

Table 1 below shows the reference configurations for two Inspur Memory1 servers specifically configured for Spark applications.

Server Type	CPU	Memory	Storage	Networking	Options
1U	E5-2660 v4 E5-2690 v4 E5-2697 v4	1TB 2TB	2x 256GB SSD 4x 1TB SSD LSI3008 HBA	2x 1GbE, 2x 10GbE SFP+	Up to 10x 2.5” SSD/HDD
2U	E5-2660 v4 E5-2690 v4 E5-2697 v4	1TB 2TB	2x 128GB M.2 4x 1TB SSD LSI-3008 HBA	2x 1GbE, 2x 10GbE SFP+	Up to 12x 3.5” HDD

Table 1: Inspur Memory1 Server Reference Configurations

Image 1: Inspur Memory1 Server NF5180M4

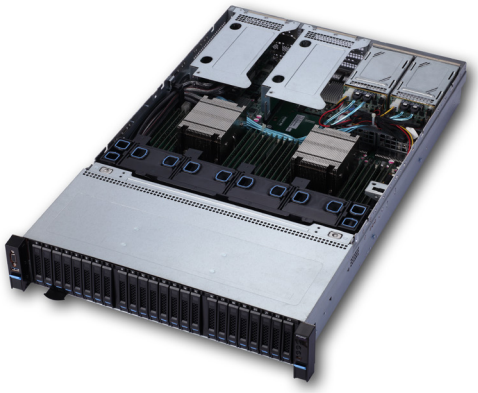

Image 2: Inspur Memory1 Server NF5280M4

Increasing Performance on Spark K-core

Benchmarking K-core Performance

Spark performance is critical for processing graph data structures, as most use cases for graph are time sensitive. Updating travel routes, balancing energy grids and modelling complex biological systems are all performance sensitive cases due to their extremely large datasets, fast response time requirements, or both. To achieve the necessary performance, Apache Spark GraphX (the API for Spark graph applications) keeps all working data in memory, along with the very large amounts of additional data required to adequately lay out and process the initial dataset.

K-core decomposition on Spark is an iterative process, where iterations generate significant working set data, requiring very large total memory capacity within the cluster to maintain optimal processing time and cluster stability. Benchmarking performance, then, relies on testing systems with differing amounts of memory.

Spark Graph Memory Requirements

In order to appropriately test k-core performance on Inspur Memory1 servers, it was necessary to calculate the amount of memory required to support all of the additional working set data generated. This is determined largely by the ratio of vertices to edges in the initial dataset, as shown in table 2 below.

Vertices (Millions)	Edges (Billions)	E/V Ratio	Initial Dataset (GBs)	Working Data-set (GBs)	Memory Requirement Increase
300	30	100:1	516	8,000	16x
100	10	100:1	164	2,600	16x
100	20	200:1	221	2,600	12x
100	30	300:1	320	2,600	8x

Table 2: Increased Memory Requirements

As this chart illustrates, providing sufficient memory per node is highly beneficial to k-core processing and ensuring the cluster has the total required memory it needs is paramount.

The three data set sizes used each had a ratio of 100:1 in terms of edges to vertices, requiring total memory of 16x the initial data set. As can be seen from Figure 2 below, a ratio of 100:1 on a data set of 164GB requires total memory capacity of 2.6TB.

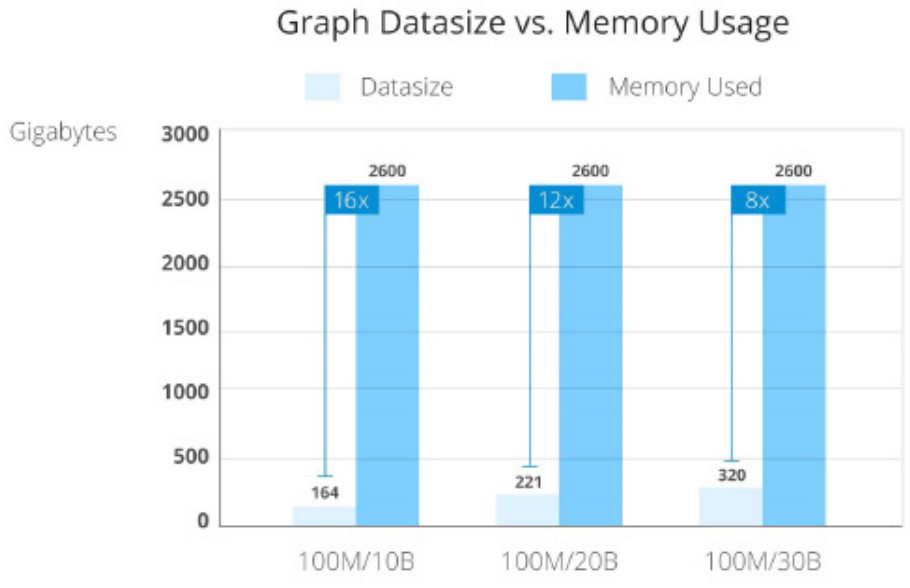

Figure 2: Memory Usage by Edge/Vertex Ratio

In order to size the cluster with enough Memory1, we used the following guidelines:

- A five node cluster with 2 TB of Memory1 per server (10 TB total) can support up to 500 GB of initial graph data with 30 billion edges and 300 million vertices
- Each 2 TB Inspur Memory1 server can support roughly 100 GB to 200 GB of graph data depending on the edge to vertex ratio

Cluster Configuration

To test k-core decomposition performance of Memory1 against a DRAM only configuration, the same set of five servers were used in both cases. First, the servers were configured to use only the installed DRAM to process multiple data sets. Next, the same systems were configured to run with 2 TB of Memory1 each, again processing the same sets of data. In each case, a cluster was set up using Inspur Memory1 servers (Inspur NF5180M4) with the following hardware and software configurations:

Hardware	Software
2 Intel(R) Xeon(R) CPU E5-2683 v3 processors	Java version 1.8.0_65
28 cores, 56 hyper threads per host	Spark version 1.5.2
Clock speed 2.00GHz	Hadoop 2.6.4
256GB DRAM	DMX version: 2.1.2
2TB Memory1	Linux 3.18.3 with DMX 2.1.2.29 modules
1TB NVME drive for storing graph data	
1 10Gb Ethernet NIC per host	

Table 3: Inspur Memory1 Server Cluster Configuration

Comparative Testing

Benchmarking Graph Performance

To compare the DRAM based servers against the Inspur Memory1 servers, three datasets of varying sizes were run on the five node cluster described above. First, a 164GB data size was used consisting of 100 million vertices and 10 billion edges. Next, a 340GB initial data size was tested made up of 200 million vertices and 20 billion edges. Finally, 516GB data was used with 300 million vertices and 30 billion edges. Each test was performed on the DRAM only cluster, then on the Memory1 cluster.

The working set in each case grew to sixteen times the size of the initial dataset. This is based on the edge to vertex ratio described above. This means that we can fit essentially all of the initial data and the working data in Memory1, while the DRAM servers will need to continually persist data to the NVMe drives.

Testing Results

The Spark k-core decomposition algorithm was run on both setups with similar configurations. The time for completion was recorded in each case for comparative purposes. In addition, disk throughput was recorded during each benchmarking run in order to monitor the amount of data being moved to and from storage.

Three dataset sizes were used for testing: 164GB, 340GB and 516GB of initial data. The ratio of vertices to edges used was 100:1. Spark, therefore, increased the total dataset size by a factor of 16x, reaquiring sixteen times the amount of memory. Figure 3 shows the graph performance results of testing the three workloads.

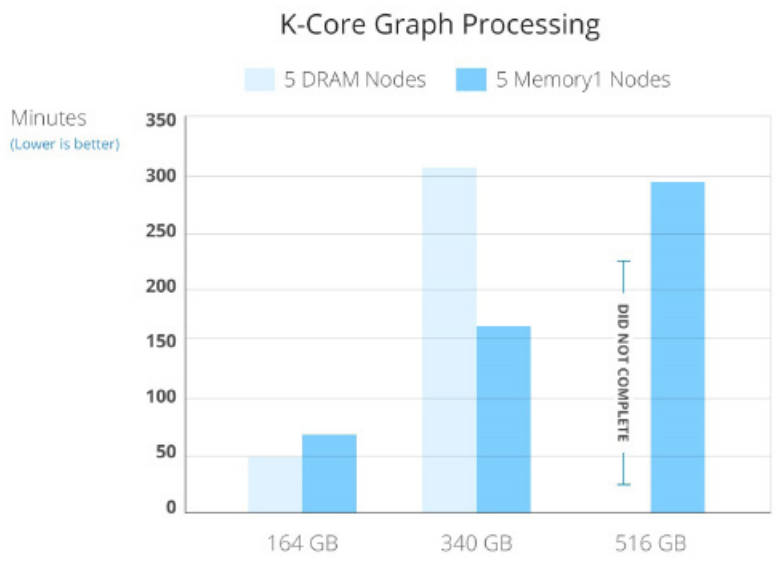

Figure 3: Results of Graph Testing on Inspur Memory1 servers vs DRAM-only Servers

In the case of the 164GB data size, the DRAM only configuration outperformed the Memory1 configuration by approximately 27%. The DRAM cluster completed the tests in 50 minutes, whereas the Memory1 cluster completed them in 67 minutes. For small data sets, DRAM may be sufficient, though graph analysis workloads rarely operate on such a small amount of information.

When the initial data grew to 340GB, the DRAM cluster took approximately 100% more time to complete, or twice as long. The DRAM cluster took 306 minutes (5 hours and 6 minutes), while the Memory1 cluster completed the tests in only 156 minutes (2 hours and 36 minutes). These results are highly significant as graph analysis typically requires fast processing on big data.

Finally, with data of 516GB, the DRAM cluster could not complete the tests at all. The Memory1 cluster was able to complete these tests in 290 minutes (4 hours and 50 minutes). This illustrates the dependence that big data processing, particularly graph analysis, has on memory requirements. Additionally, this shows that the Memory1 cluster completed the analysis of 516GB of data in less time than the DRAM-only cluster completed the analysis on 340GB of initial data.

With more capacity per server, a greater amount of the data set can be contained in application memory. If all or nearly all data remains in memory, the burden on storage is significantly lower. Even when NVMe drives are used, storage is a major bottleneck.

In our observations, we measured the expected corresponding decrease in disk IO. The average disk throughput seen on the Inspur Memory1 servers was around 200 MB per second (see Figure 4 below). The average disk throughput recorded on the DRAM only machines measured 3.5x higher at approximately 700MB per second.

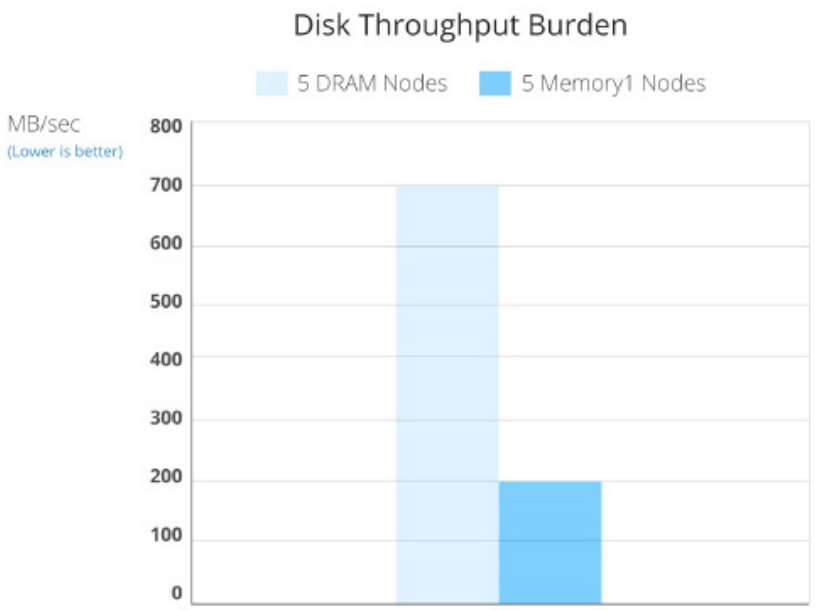

Figure 4: DRAM Nodes Move More Data Between Storage and Memory Reducing Performance

When disk throughput is high, as in the case of the DRAM-only cluster, performance suffers. The system is forced to wait while data is moved from storage to memory and back to storage. Additionally, the processor must be involved when handling these IO events, further straining valuable system resources.

What We've Shown

Many use cases exist for graph data structures as they describe the interrelationships between a variety of objects. The ability to represent this data in a meaningful and actionable manner allows for more complex analysis of significantly larger volumes of data. Apache Spark is gaining wide adoption as the best means of processing this data and deriving useful insights with speed and accuracy.

The k-core decomposition algorithm has many practical applications such as the analysis of social networks, missing link preditions, and fraud detection. Analyzing this data, however, requires significant investment in application memory and additional servers, driving up expense and complexity. Relying on DRAM is costly and does not offer the required capacity per server. Additional servers are scaled out in order to provide the required memory footprint, increasing management, networking, power, cooling and other expenses. Solutions are needed to provide this larger address space, as available memory has a huge, direct impact on performance.

As we've shown in this paper, Diablo Technologies' Memory1 modules via the Inspur Memory1 servers provides this much needed capacity, increasing the amount of work achieved per server and decreasing execution time by more than half or better. Utilizing Memory1, we observe significant increases in performance over servers using only DRAM.

With an initial data set of greater than approximately 200GB, Inspur Memory1 servers outperform DRAM-only based servers. At 340GB of initial data, the k-core algorithm completed in approximately half the time on the Inspur Memory1 server. Larger data sets using DRAM only creates an unstable system that cannot complete the graph analysis at all as the system is exhausted of resources.

Performance is critical on job related analyses like k-core decomposition. The ability to increase the amount of available application memory per server is the only feasible solution and, as we've shown, Inspur Memory1 servers deliver the memory capacity and performance required to appropriately handle this analysis.

The result is the ability to size a cluster for graph processing according to the data set and the requirements for analysis. By increasing the amount of work achieved per server, Memory1 allows for either:

- Significantly faster processing of large datasets on fewer machines

Or

- Significantly increasing processing capability of an existing cluster

About Inspur

Inspur Systems Inc., located in Fremont, CA, is part of Inspur Group, a leading Cloud Computing and global IT Solutions Provider. Inspur was founded in 1945 and has since provided IT products and services for over 85 countries in the world. Inspur is ranked by Gartner as one of the Top 5 largest server manufacturers in the world and #1 in China. Inspur provides our global customers with data center servers and storage solutions which are Tier1 quality and performance, energy efficient, cost effective and built specific to actual workloads and data center environments. As a leading total solutions and services provider, Inspur is capable of providing total solutions at IaaS, PaaS and SaaS level with high-end servers, mass storage systems, cloud operating system and information security technology. For more information, visit www.inspursystems.com.

About Diablo Technologies

Diablo Technologies is a leading developer of high-performance memory products that solve urgent business problems by wringing more performance out of fewer servers. Diablo's Memory1™ combines the highest capacity memory modules with their leading Software Defined Memory platform. Memory1 enables a dramatic reduction in datacenter expenses with significant increases in server and application capability. Diablo's products and technology are included in solutions from leading server vendors such as Inspur. Diablo is best known for its innovative Memory Channel Storage™ (MCS™) architecture. Memory Channel Storage dramatically decreased storage access times by more than 80% by attaching flash storage directly to the CPU's memory controller.