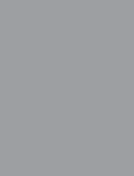
FUNL™

COMPLEX BIG DATA ON A BUDGET

MISSION

The goal of FUNL is to bring meaning to data at an affordable price. FUNL incorporates commodity hardware and novel software techniques to exploit the hardware capabilities. Efficient I/O mechanisms for GPU-based graph processing, a library of graph analytic and machine learning solutions, and visualization are integrated into a unified system that provides end-to-end solutions to Big Data problems.



FUN

GRAPH
SOLUTION
MODEL
BIG DATA
NETWORK
INSIGHT
AFFORDABLE
ANALYTICS

Technica provides professional services, products, and innovative technology solutions to the Federal Government. We specialize in network operations and infrastructure; cyber defense and security; government application integration; systems engineering and training; and product research, deployment planning, and support.

Independent Research IR&D and Development

Technica®

Technica Dulles, VA (HQ)
22970 Indian Creek Drive, Suite 500
Dulles, VA 20166

703.662.2000

technicacorp.com

Technica®

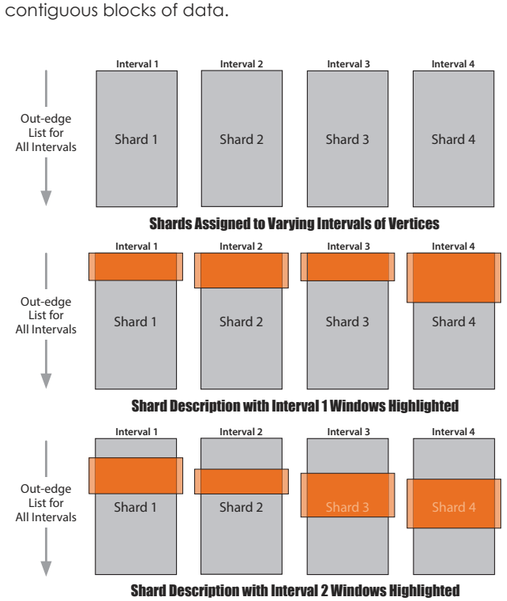
PROBLEM

Data production is increasing exponentially, creating a corresponding demand for Big Data processing. Complex processing with significant data interdependencies, such as graph analysis, typically requires expensive investments in hardware and software. These solutions use specialized hardware with large amounts of RAM, high-speed interconnections, and many CPUs to fit the data into memory.

INSPIRED BY PROCESSING WITH PARALLEL SLIDING WINDOWS

As Big Data Graphs are irregular, processing requires random access throughout the entire graph which can greatly increase processing time. Parallel Sliding Windows (PSW)[1] is an out-of-core graph processing technique that organizes graph data into partitions to be loaded separately into memory and processed iteratively. This enables large data sets to be processed efficiently on commodity hardware

PSW divides the graph into *P* intervals of vertices with consecutive IDs. The list of edges is then organized such that for each interval of vertices, the process only needs to access *P* contiguous blocks of data.



Given a graph of 1.4 billion vertices and 6.6 billion edges, a CPU-only solution using PSW was able to compute belief propagation in 27 minutes on a Mac Mini, while a Hadoop-based graph mining library, distributed over 100 nodes runs the same computation in 22 minutes.

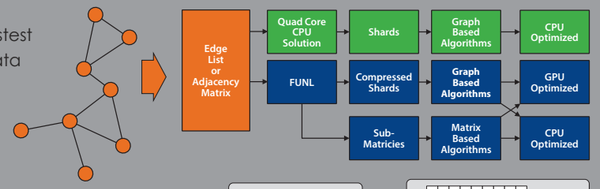
[1]Kyrola, Blelloch, and Guestrin, GraphChi: Large-Scale Graph Computation on Just a PC, 2012

SOLUTION

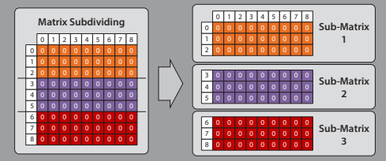
The FUNL graph analytics solution combines the I/O efficiency of PSW, hybrid CPU/GPU algorithms, and compression techniques to deliver high performance at a much lower cost, with a low barrier to entry. Using off-the-shelf desktop hardware, FUNL performs comparably to dedicated solutions for many scenarios.

FUNL vs CPU-ONLY, SINGLE MACHINE SOLUTION

FUNL has a flexible framework that chooses the best approach to get the fastest performance for any data set and algorithm.



For some scenarios, a linear algebra approach performs better than PSW. In these cases, the FUNL system divides the input matrix into smaller sub-matrices to fit into CPU or GPU memory for processing.

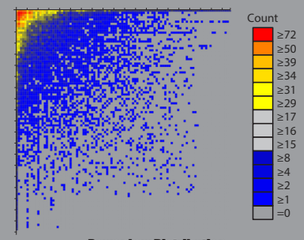


Matrix Subdividing

I/O EFFICIENT ALGORITHMS

Reducing I/O is key to improving the performance of out-of-core graph algorithms. The limited memory of the GPU means that multiple shards can be stored in CPU memory. FUNL incorporates multiple caching schemes to optimize data reuse from CPU memory for each algorithm.

Power law distributions are common in real world graph data. This means that there are many vertices with few edges, and few vertices with many edges. FUNL's compression scheme improves the PSW approach by exploiting the large number of small integers, potentially cutting the file size in half. Parallelized decompression reduces the overhead that is introduced.



Power Law Distribution

MASSIVE PARALLELISM

FUNL employs a GPU with 1000s of compute cores to accelerate computationally intensive algorithms. The massive parallelism of GPUs speeds up analysis by as much as an order of magnitude or more, depending on the computational intensity of the algorithm as well as properties of the input data. To take full advantage of the GPU, data can be regularized for even more efficient computation. The benefit gained from the use of a GPU depends heavily on the ratio of computation to memory access. For low computation to memory access scenarios, the CPU may be more efficient. FUNL distributes the processing between the CPU and the GPU as necessary.

DEEP LEARNING

DeepInsight is a deep learning application for analyzing graph data. Based on the DeepWalk[2] algorithm, it uses the GPU to generate many random walks on the graph in parallel, which are used to learn about relationships within the graph. The result is a compressed representation of each node that can be used for many tasks including classification and link prediction. Additionally, DeepInsight can use natural language processing to incorporate text associated with each node to improve the results.

[2]Bryan Perozzi, Rami Al-Rfou, Steven Skiena, DeepWalk: Online Learning of Social Representations, 2014

PERFORMANCE RESULTS

To test performance, FUNL was evaluated against three different benchmarks:

- Quad Core CPU benchmark, a CPU-only solution using PSW and the same hardware as FUNL
- Spark cluster benchmark using Amazon Web Services
- 16-Core Server benchmark, a CPU-only solution using PSW

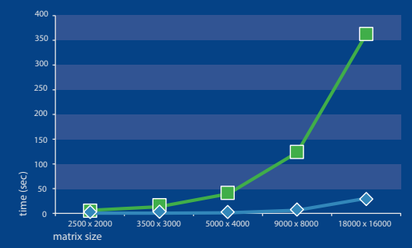
ALGORITHM PERFORMANCE

		FUNL	Quad Core CPU	Spark
Algorithm	Data Set	Performance Time (seconds)		
Page Rank	4.8M Vertices 69M Edges	5.23	11.46	110.4
Triangle Counting	41M Vertices 1.4B Edges	563	3960.49	N/A
Collaborative Filtering Using Alternative Least Squares	Sparse Matrix: 20K x 10K Non-Zeros: 53M	190.15	1444.85	N/A
Support Vector Machines	Dense Matrix: 25M x 100	456.41	731.09	N/A

FUNL / Quad Core CPU
GPU: GeForce GTX TITAN, 2688 CUDA Cores, 928 MHz, 6GB vRAM
CPU: Core i7, Quad Core, 3.40 GHz
RAM: 16GB (4x4GB), 1333 MHz
Storage: HDD, ~ 180 MB/sec

Spark Cluster
System: AWS EC2 m1.large
Nodes: 10
Network: Moderate Performance

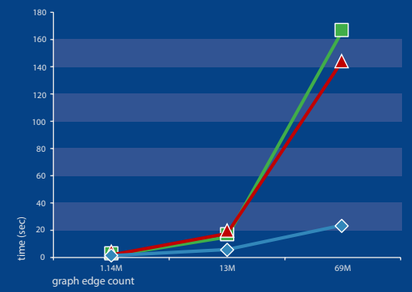
Singular Value Decomposition is a matrix factoring algorithm used for many applications including Principal Component Analysis. PCA uses orthogonal transformation to find a linear projection of high dimensional data into a low dimensional subspace. The resulting linearly uncorrelated variables are called principal components, and represent the dimensions in which the greatest variance in data exists.



Scaling of Out-of-Core SVD as a Function of Matrix Size

FUNL / Quad Core CPU
GPU: GeForce GTX TITAN, 2688 CUDA Cores, 928 MHz, 6GB vRAM
CPU: Core i7, Quad Core, 3.50 GHz
RAM: 16GB (4x4GB), 1333 MHz
Storage: HDD, ~ 150 MB/sec

Belief Propagation is a message passing algorithm performed on graph models as a method to infer information about unknown vertices based on known information about other vertices. FUNL implements BP using compressed PSW.



Scaling of Belief Propagation Performance as a Function of Graph Size

FUNL / Quad Core CPU
GPU: GeForce GTX TITAN, 2688 CUDA Cores, 928 MHz, 6GB vRAM
CPU: Core i7, Quad Core, 3.40 GHz
RAM: 16GB (4x4GB), 1333 MHz
Storage: HDD, ~ 180 MB/sec

16 Core Server
CPU: Xeon E5-2690, 16 Cores, 2.9 GHz
RAM: 64GB, 1600 MHz
Storage: HDDx6, RAID0, 690MB/sec