A Benchmark Suite for Unstructured Data Processing

Clinton Wills Smullen, IV Shahrukh Rohinton Tarapore Sudhanva Gurumurthi

Department of Computer Science
University of Virginia
Charlottesville, VA 22904
{cws3k,shahrukh,gurumurthi}@cs.virginia.edu

Abstract

A large fraction of the data that will stored and accessed in future systems is expected to be unstructured, in the form of images, audio files, etc. Therefore, it is very important to design future I/O subsystems to provide efficient storage, and access to these vast and continuously growing repositories of unstructured data. To facilitate system design and evaluation, we first need benchmarks that capture the processing and I/O access characteristics of applications that operate on unstructured data. In this paper, we present an unstructured data processing benchmark suite that we have developed. We provide detailed descriptions of the workloads in the benchmark suite and discuss the larger space of application characteristics that each of them capture.

Keywords: Benchmarks, unstructured data, I/O.

1 Introduction

In recent years, storage has undergone a major transformation. Due to advances in magnetic recording technology, the cost of storage has dropped dramatically, from over $4/Megabyte in 1990, to less than $0.001/Megabyte today [14]. These trends have led to disks being considered as commodity components, and it is now common to see disk drives that have capacities of several tens or even hundreds of Gigabytes. This growth in storage, coupled with equally dramatic technological advances in processor design, have paved the way for relatively inexpensive, but computationally powerful computers. One of the most significant outcomes of these technological advancements is the massive growth in the amount of digital data.

According to a recent study by IDC [10], a leading Information Technology market research and analysis firm, the amount of data that would be captured, stored, and replicated worldwide (what they refer to as the "digital universe") would grow from 161 exabytes in the year 2006, to over *988 exabytes* in 2010 (1 exabyte = 10^{18} bytes). Two key findings of this study are:

- A majority of this data would be in the form of images, captured from a large number of devices, such as digital cameras, camera phones, surveillance cameras, and medical imaging equipment. Most of this data would need to be stored and managed in centralized systems within organizations. The study indicates that, by 2010, although enterprises will create, capture, and replicate only 30% of the digital universe, they will have to store and manage over 85% of all data in it.
- Over 95% of the digital universe is *unstructured data* (e.g., image and music files, e-mail). According to this study, 80% of all stored data of organizations is unstructured. This growth trend is expected to continue into the future, thereby mandating the need for efficient ways of searching, structuring, and providing security for unstructured data [10]. Similar trends about the importance of unstructured data processing have also been reported by other market research and advisory firms, such as, the Gartner Group and the Butler Group [36].

Unstructured data processing is therefore a very important emerging class of applications. There are a number of unstructured data processing applications that are already in use today. These applications include text searches (exact and approximate searches) [5], content-based searches of image, video, and audio files [26], and data fusion [19]. Although some of these applications are used in relatively niche domains (e.g., geo-spatial data fusion is used in urban planning and forestry [35]), the core *methods* used in these applications are expected to become commonplace across a wider range of applications in the future. For example, Content-Based Image Retrieval (CBIR) [26], which is very processing and I/O intensive, is now used in the field of medicine for querying biomedical digital libraries [18].

However, CBIR has been identified by Intel as an important emerging application even for the home user [7], say, for content-based querying of digital photo collections stored on her personal computer, or on a photo repository, such as, Flickr™. This growing demand for unstructured data management has already started creating a market for hardware appliances that are specifically designed for searching and processing unstructured data [8, 32, 30].

Given this shift in momentum towards unstructured data processing, it is imperative to develop benchmarks that can aid in the design and evaluation of future systems that would have to run these applications efficiently. A common feature across several unstructured data processing applications are that they are very I/O intensive, and therefore the place heavy demand on the storage system to deliver high performance. Moreover, unstructured data processing workloads exhibit a variety of unique data access patterns that are not sufficiently captured by traditional server I/O workloads, such as, the TPC [33] benchmarks. Therefore, in order to design storage systems for this important emerging class of applications, we need a benchmark suite that can capture their processing and I/O characteristics. In this paper, we present a benchmark suite that we have developed for unstructured data processing.

Our benchmark suite consists of four workloads:

- Edge detection
- Proximity search
- Data scanning
- Data fusion

The edge detection workload applies the Smallest Univalve Segment Assimilating Nucleus (SUSAN) [27] algorithm on a set of images from the MIT CBCL Face Recognition Database [22]. The proximity search application performs the k-Nearest Neighbor [17] search algorithm over a database of meteorological records [16]. Data scanning applies the Boyer-Moore string matching algorithm [2] over a synthetically generated file-system of unstructured files and captures the behavior of an anti-virus engine [29]. Data fusion implements pixel-based fusion of a collection of images from the NASA World Wind database [24].

In this paper, we present details about the implementation of each of these workloads, and discuss how workload models a larger application space. These benchmarks capture the *core operations* that are common across a wide spectrum of unstructured data processing applications. Each of these core operations encapsulate an unique processing and data access pattern. These benchmarks are valuable to computer architects and system designers, especially those who work in the area of storage system design.

The outline for the rest of the paper is as follows. The next section discusses related work and Section 3 presents a detailed description of the benchmark suite. Section 4 concludes this paper.

2 Related Work

A large number of benchmarks are available for evaluating different aspects of computer performance. Some popular benchmarks include: SPEC [28] – for evaluating processor and cache performance, STREAM [21] – for evaluating memory system performance, and IOMeter [15] – for benchmarking I/O performance. Benchmarks also exist for evaluating particular classes of systems, e.g., TPC [33] and EEMBC [31], which target server and embedded platforms respectively. None of these benchmarks specifically capture the characteristics of unstructured data processing.

Recently, there have been a number of efforts to create benchmarks for specific applications. These applications include bioinformatics [1, 20], biometrics [6], and data mining [23]. Our unstructured data processing benchmark suite is intended to contribute to this growing pool of benchmarks for emerging workloads.

3 Benchmark Description

Our primary goal in assembling this workload suite was to identify and study a wide range of unstructured data processing applications, distill their key processing and I/O access characteristics, and compose workloads that embody these characteristics. To identify these workloads (and their associated datasets), we did extensive literature surveys and also talked to experts in various fields such as, scientific computing, business analytics, and computer security. We were also careful in crafting the workloads in a such a way that they could be easily ported onto an architecture simulator and run to completion within a reasonable amount of time.

We have identified four workloads that capture a wide spectrum of applications within the domain of unstructured data processing. We now provide a brief description of each of these workloads and highlight the larger application space that each of them represent. The name that we coined for each of these workloads is given in parenthesis.

- **Edge Detection (`susan`):**

  Edge detection is a basic operation that is used in applications, such as content-based image recognition [26], and speech recognition [12]. Edge detection is representative of a class of applications that highlight or collect points on high contrast in the input. These points of interest can be the edges of an object in an image, or individual words in a conversation (in the case of

speech recognition systems). The primary use of edge detection is as a filter of less relevant information in the data, leaving only the structural properties of the data that are of interest to the user.

We model this space of applications by composing a workload that performs the edge detection operation on images. The objective of this workload is to detect and extract the facial features of subjects (which are edges) from a set of images. The edge detection is performed using the Smallest Univalve Segment Assimilating Nucleus (SUSAN) algorithm [27]. SUSAN is a low level image processing suite, which can perform edge detection, corner detection, and structure preserving noise reduction. A non-linear filtering approach is taken to associate a local region with each pixel of similar intensity. Our dataset consists of a set of images from the MIT CBCL Face Recognition Database [22]. In this workload, the processor requests an image from the storage system, performs the edge detection algorithm, and then requests the next image.

- **Proximity Search (`nn`):**

  Given a search criteria and a distance function a proximity search will return any occurrence of the search criteria along with any neighbors as defined by the distance function. Proximity searches contrast point-based searches in that the latter will return only an exact match of the search criteria, if it exists, and nothing more. There are many variants of proximity searches, such as k-nearest neighbor search and ϵ-approximate nearest neighbor search [17], but they all perform the same task: classifying a specific point in the search space with respect to its neighbors. Proximity searches are often used in Internet search engines [3], and also in content-based search applications, such as, song intersection [4], which is used to search for songs based on the acoustic properties of an input song, and scene completion [13], which is a technique to seamlessly patch a photograph by automatically detecting and using parts from other images that preserve the semantics of the original image.

  Our workload implements the k-nearest neighbor search procedure. The procedure works as follows: given a search criteria and non-negative integer k, the nearest neighbor search returns the k database records that match or come closest to matching the specified criteria. We use a database from the National Oceanic and Atmospheric Administration (NOAA) National Hurricane/Tropical Prediction Center as our dataset [16]. This database contains information about tropical cyclones in the North Atlantic for a period of 155 years, starting from the year 1851, with storm progressions recorded at six-hour intervals. Each record in this database has fields for the date and time of the storm, the latitude and longitude of the storm center, storm surge levels, and variety of barometric data. For a given (latitude,longitude) pair, we compute the k tropical storms that occurred closest to the given coordinate. The search procedure scans through the entire database, calculating the Euclidean distance between the search coordinates and the (latitude,longitude) pair of each database record, maintaining a list of the k closest matches. We resort to a full scan of the database for each query, since scanning has been shown to be efficient for such multi-attribute searches [25]. The workload reads in the entire database from disk and performs the search as the records come in, maintaining the k closest matches. In our workload, we set k=3.

- **Data Scanning (`malware`):**

  This workload captures the behavior of text search engines, which scan through large unstructured datasets to find patterns of interest. One large class of applications that perform such scan-based searches are security software, such, as anti-virus engines, spyware and rootkits scanners, and intrusion-detection systems. These software use a technique known as *signature-matching*, where the bytes of a file are compared against a set of known-to-be malicious byte-sequences, known as "signatures" [29]. In fact, malware detection is considered a significantly important application that there are co-processors that are commercially available that are designed specifically for signature-matching [30].

  The key characteristic of malware scanning applications is that they scan a large amount of unstructured file data against a database of signatures to detect the presence of specific byte-sequences. We implement this form of signature-matching using the efficient Boyer-Moore string searching algorithm [2]. We use a database of randomized signatures where each signature is, on average, 64 bytes in length. The dataset to be scanned is randomly generated with each file being 256 KB, on average. Exponential distributions are used to determine whether a file has either a partial signature or a complete signature. The partial signatures prevent the Boyer-Moore algorithm from achieving its best-case running time. Only one file in the entire dataset has a real signature. An instance of the Boyer-Moore algorithm is run for each signature. Data streaming is provided by the use of circular buffer to hold the section of the file

currently under inspection. As data is passed over by every instance of the Boyer-Moore algorithm, data is removed from the circular buffer and replaced by new data from the file. If any one of the signatures matches, the name of the file is returned, and we skip to the next file. The Boyer-Moore algorithm generates bad-character and good suffixes tables, for which the size depends on the size of the signature set. Thus there is significant bookkeeping overhead associated with each execution of the algorithm in addition to the data to be scanned.

- **Data Fusion: (`geo`):**

  Data fusion is a technique that is used to obtain information originating from different sources by combining their individual data into a composite entity. There are many examples of the use of data fusion in high performance computing, such as in strategic defense [9], where data coming in from multiple sources needs to be evaluated to make decisions in real-time. In medicine, image fusion techniques are used in Magnetic Resonance Imaging (MRI) to build an accurate three-dimensional model of a subject from hundreds of "slice" images [34]. Geo-spatial data fusion is used for analyzing changes to a particular geographical region using data from sources (e.g., different satellite images), possibly gathered at different points of time. Geo-spatial data fusion is used to aid urban planning, in forestry, and in disaster relief, e.g., to assess damage due to flooding [35].

  Data fusion is becoming a very popular Internet-based application, especially in the context of web-based maps (e.g., Google Maps™). In this application, roadmap data is combined with other information, such as, satellite images and traffic updates, to present a wealth of information to the user about a particular geographical region. Given the popularity of web-based maps, it is highly conceivable that future versions of these applications would use much richer content, such as, images and videos from traffic-cameras and sensors to provide real-time detailed visualizations. The servers that run such an application would need to support very efficient data fusion capabilities, which will place a heavy demand on the storage system to deliver the required data streams to the server processor(s), where the fusion operations are performed to provide the composite information requested by the user.

  Our workload implements pixel-based geo-spatial data fusion [11] on a dataset that consists of several satellite images, which we obtained from the NASA World Wind database [24]. For a given pair of images, the workload performs a sequence of three processing steps: (i) image enhancement, (ii) edge detection, and (iii) pixel-based image fusion. Image enhancement applies a combination of low-pass and high-pass filters to the images, which are then fed into the edge detection step, which uses the same algorithm as our `susan` workload to extract the edges of the images. The third processing stage combines common pixels in the two images to produce a new image that highlights the differences between the two that are input to the workload.

4 Conclusions

We have presented four benchmarks that capture the processing and data access patterns of the core operations that are used in a wide spectrum of unstructured data processing applications. These applications are I/O intensive and place heavy demands on the storage system. We plan to release these benchmarks to the research community shortly.

5 Acknowledgements

This research has been supported in part by NSF grants: CAREER Award CCF-0643925, CNS-0627527, CNS-0551630, and a grant from HP.

References

[1] K. Albayraktaroglu, A. Jaleel, X. Wu, M. Franklin, B. Jacob, C.-W. Tseng, and D. Yeung. BioBench: A Benchmark Suite for Bioinformatics Applications. In *Proceedings of the International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pages 2–9, March 2005.

[2] R. Boyer and J. Moore. A Fast String Searching Algorithm. *Communication of the ACM*, 20(10):762–772, 1977.

[3] S. Brin and L. Page. The Anatomy of A Large-Scale Hypertextual Web Search Engine. In *Proceedings of the World Wide Web Conference (WWW)*, pages 107–117, April 1998.

[4] M. Casey and M. Slaney. Song intersection by approximate nearest neighbor search. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR)*, pages 144–149, October 2006.

[5] R. Chamberlain, M. Franklin, and R. Indeck. Exploiting Reconfigurability for Text Search. In *Proceedings of the Workshop on High Performance Embedded Computing (HPEC)*, September 2006.

[6] C.-B. Cho, A. Chande, Y. Li, and T. Li. Workload Characterization of Biometric Applications on Pentium 4 Microarchitecture. In *Proceedings of the International Workload Characterization Symposium*, pages 76–86, October 2005.

[7] P. Dubey. Recognition, Mining, and Synthesis Moves Computers to the Era of Tera. *Technology@Intel Magazine*, February 2005.

[8] Exegy TextMiner (Whitepaper). http://www.exegy.com.

[9] G. Fountain and S. Drager. High performance real-time fusion architecture. In *Proceedings of the Fifth International Conference on Information Fusion*, pages 1478–1485, 2002.

[10] J. Gantz and et al. The Expanding Digital Universe - A Forecaset of Worldwide Information Growth Through 2010, March 2007. IDC Whitepaper.

[11] R. Gens. Geospatial Data Fusion - Seminar Talk, Geophysical Institute, University of Alaska Fairbanks, February 2004.

[12] D. Haisheng, Z. Xiaoyan, L. Yupin, and Y. Shiyuan. Robust Edge Detection Method for Speech Recognition. In *Proceedings of the International Conference on Signal Processing (ICSP)*, pages 609–612, August 2004.

[13] J. Hays and A. Efros. Scene completion using millions of photographs. *ACM Transactions on Graphics (SIGGRAPH 2007)*, 26(3), 2007.

[14] Hitachi Global Storage Technologies - HDD Technology Overview Charts. http://www.hitachigst.com/hdd/technolo/overview/storagetechchart.html.

[15] Iometer. http://www.iometer.org/.

[16] B. Jarvinen, C. Neumann, and M. Davis. A Tropical Cyclone Data Tape for the North Atlantic Basin, 1886-1983: Contents, Limitations, and Uses. Technical Report NWS NHC 22, National Oceanic and Atmospheric Administration (NOAA), 1984.

[17] J. Kleinberg. Two algorithms for nearest-neighbor search in high dimensions. In *Proceedings of the Symposium on the Theory of Computing (STOC)*, pages 599–608, May 1997.

[18] T. Lehmann and et al. Content-based Image Retrieval in Medical Applications. *Methods of Information in Medicine*, 43(4):354–361, 2004.

[19] LexisNexis Special Services Extends its Intelligence Analysis Solutions with Inxight ThingFinder Professional, April 2006. LexisNexis Press Release http://www.lexisnexis.com/about/releases/LNSS-Inxight.asp.

[20] Y. Li, T. Li, T. Kahveci, and J. Fortes. Workload Characterization of Bioinformatics Applications on Pentium 4 Architecture. In *Proceedings of the International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, pages 15–22, September 2005.

[21] J. McCalpin. The STREAM2 Home Page. http://www.cs.virginia.edu/stream/stream2.

[22] MIT Center for Biological and Computational Learning (CBCL) Face Recognition Database. http://cbcl.mit.edu/software-datasets/heisele/facerecognition-database.html.

[23] R. Narayanan, B. Ozisikyilmaz, J. Zambreno, G. Memik, and A. Choudhary. MineBench: A Benchmark Suite for Data Mining Workloads. In *IEEE International Symposium on Workload Characterization (IISWC)*, pages 182–188, October 2006.

[24] NASA World Wind. http://worldwind.arc.nasa.gov/.

[25] E. Riedel, G. Gibson, and C. Faloutsos. Active Storage for Large-Scale Data Mining and Multimedia. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, pages 62–73, August 1998.

[26] A. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based Image Retrieval at the End of the Early Years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380, December 2000.

[27] S. Smith and J. Brady. SUSAN - A New Approach to Low Level Image Processing. *International Journal of Computer Vision*, 23(1):45–78, May 1997.

[28] SPEC - Standard Performance Evaluation Corporation. http://www.spec.org/.

[29] P. Szor. *The Art of Computer Virus Research and Defense*. Addison-Wesley, 2005.

[30] Tarari Anti-Virus Content Processor. http://www.tarari.com/antivirus/index.html.

[31] The Embedded Microprocessor Benchmark Consortium (EEMBC). http://www.eembc.org/.

[32] The Netezza Performance Server System. http://www.netezza.com/products/products.cfm.

[33] TPC - Transaction Processing Performance Council. http://www.tpc.org/.

[34] R. Wasserman, R. Acharya, C. Sibata, and K. Shin. A data fusion approach to tumor delineation. *icip*, 02:2476, 1995.

[35] A. Waxman and et al. Information Fusion for Image Analysis: Geospatial Foundations for Higher-Level Fusion. In *Proceedings of the International Conference on Information Fusion (ISIF)*, pages 562–569 Vol. 1, July 2002.

[36] C. White. Consolidating, Accessing, and Analyzing Unstructured Data, December 2005. Business Intelligence Network article.