Implementation of CCSDS Lossless Data Compression in HDF

Pen-Shu Yeh[1], Wei Xia-Serafino[2], Lowell Miles[3], Ben Kobler[1] , Daniel Menasce[4]
[1]Goddard Space Flight Center, Greenbelt, MD 20771
[2]Global Science and Technology Inc., Greenbelt, MD 20771
[3]Microelectronics Research Center, U. of New Mexico, Albuquerque, NM 87131
[4]George Mason University, Fairfax, VA 22030

***Abstract*- The Earth Science Data and Information System (ESDIS) handles over one terabyte (10 [12] bytes) of data daily and is using the Hierarchical Data Format (HDF) for data archiving and distribution. This report provides the progress and status of our effort to alleviate bandwidth and storage burdens by first performing compression studies on various science data products and later integrating the selected compression scheme into HDF.**

I. INTRODUCTION

A. Why data compression?

The Earth Science Data and Information System (ESDIS) Project provides scientific and other users access to data from NASA's Earth Science Enterprise. The ESDIS Project provides this access through the development and operation of the Earth Observing System (EOS) Data and Information System (EOSDIS). EOSDIS captures and processes telemetry data, generates higher-level standard data products, and performs mission operations for instrument and spacecraft control. Data products from EOS and other NASA Earth science missions are stored at several Distributed Active Archive Centers (DAACs) to support interactive and interoperable retrieval and distribution of data products.

EOSDIS manages data from NASA's past and current Earth science research satellites and field measurement programs, providing data archiving, distribution, and information management services for missions including:

- EOS missions Landsat-7, QuikSCAT, Terra and ACRIMSAT
- Pre-EOS missions (UARS, SeaWIFS, TOMS-EP, TOPEX/Poseidon and TRMM)
- All of the Earth Science Enterprise legacy data (e.g., pathfinder data sets)

EOSDIS uses the Hierarchical Data Format (HDF) [1] for EOS (HDF-EOS), developed as an extension to the HDF supported by the National Center for Supercomputing Applications (NCSA) at the University of Illinois.

About 1,000 project scientists are front-line users of the data. In addition there are an estimated 30,000 secondary users including researchers, federal/state/local governments, application users, the commercial remote sensing community, teachers, museums and the general public.

The transmission and archiving of such an unprecedented amount of data requires tremendous computing time, computer storage, and I/O bandwidth.

One technique that can reduce the total data archive storage and network connection time requirement without compromising data fidelity is lossless data compression. Lossless data compression is a mature technology that has been used extensively in a variety of applications.

B. Lossless Data Compression

Lossless data compression techniques such as zip, gzip, and winzip are widely used for compressing files residing on PCs. These techniques are all based on the Lambel-Ziv-Welch (LZW) [2] algorithm or its variations, and would generally yield poor compression ratios on data originating from spacecraft instruments. A second well-established technique is arithmetic coding [3]. This technique works on most types of data, but exhibits relatively slow speed due to the need to update statistics along the process. The Consultative Committee on Space Data Systems (CCSDS) has adopted the extended-Rice algorithm as the recommendation for international standards for space applications [4][5][6]. This technique was developed specifically for science instrument data by a joint effort between NASA GSFC and Jet Propulsion Laboratory (JPL) based on requirements of high speed for real time processing, low complexity, and quick adaptation to statistics. It has been implemented on many space missions in either instruments or data systems and baselined for many future satellites as well.

II. OBJECTIVE

The objective of this project is to implement an efficient (in both data reduction and processing speed) lossless data compression into the EOSDIS system. There are three phases of the project.

Phase I objective is to perform a comparison study on different lossless data compression techniques and to recommend the most suitable compression algorithm to EOSDIS based on the study results. Lossless data compression algorithms, Szip (an implementation of the CCSDS algorithm), Gzip and Arithmetic Coding (Az) were evaluated. Compression ratio and compression speed were used as comparison criteria.

Phase II objective is to evaluate the feasibility of using Szip in EOSDIS operations based on its speed and data reduction performance. Gigabytes of MODIS real sensor data were tested on an EOS CORE SYSTEMS (ECS) operational machine and a scalability analysis was performed.

Phase III objective is to implement Szip into the HDF library.

III. PROJECT DESCRIPTION

A. PhaseI (6/98-12/98)

In this phase, the performance of different lossless data compression techniques was compared on typical EOSDIS HDF-EOS data files. Two measurements, speed and compression ratio (CR), were used as the comparison criteria. The algorithms evaluated were Unix compress (Cz), Gzip (Gz), JPEG Arithmetic coding (Az) [7], and CCSDS lossless data compression (Sz). They were tested on a Sun Sparc20 workstation running the Unix operating system. The prototype study results demonstrated that the performance of CCSDS lossless data compression technique is superior both in the compression ratio and the compression speed for science data. Az is too slow in compression and decompression time, and the Cz has poor compression ratio.

Table 1 shows summary comparison results from the Phase I study. Values in the table represent the averages over all data products.

Test data used in this table are real sensor data and derived products from MODIS Airborne Simulation (MAS), TRMM, AVHRR, TOVS, ASTER, SeaWifs, and TOMS with a total data volume of 930 Mbytes. The data product levels include level 1 through 4, and the data types include 8-bit, 16-bit and 32-bit floats. Based on this study, CCSDS lossless data compression was recommended as the most suitable data compression technique for archiving and distributing EOSDIS data.

TABLE 1
PHASE I RESULT SUMMARY
TESTED ON SUN SPARC20

	Sz	***Gz***	***Cz***	***Az***
CR	3.24	2.44	2.06	2.38
Compress (Seconds)	353	8112	1973	10516
Decompress (seconds)	394	1264	790	7341

B. Phase II (1/99-6/01)

In the second phase, the feasibility of using CCSDS compression for real-time processing was evaluated using Moderate Resolution Imaging Spectroradiometer (MODIS) simulation data and the satellite data on the ECS operational production machine (SGI Power Challenge). Results for the Phase II study are shown in Tables 2 and 3. Table 2 contains results using Szip compression on MODIS data, and Table 3 contains results using Gzip compression.

TABLE 2
LOSSLESS DATA COMPRESSION RESULTS USING SZIP ON MODIS REAL SENSOR DATA
TESTED ON SGI POWER CHALLENGE

MODIS Data	**File Size(bytes)**	**Szipped (bytes)**	**CR**	**Szip Time**	**Sunzip Time**
Level-1B	895612046	387861914	2.31	79.7	86.1
Level-2	1185674280	101563451	11.67	55.7	54.7
Level-3	1457438720	449184123	3.24	143	141.2
Level-4	11520000	1432733	8.04	0.4	0.5
Total	3550245046	940042221	3.78	278.8	282.5

TABLE 3
LOSSLESS DATA COMPRESSION RESULTS USING GZIP ON MODIS REAL SENSOR DATA
TESTED ON SGI POWER CHALLENGE

MODIS Data	**File Size**	**Gzipped (bytes)**	**CR**	**Gzip Time**	**Gunzip Time**
Level-1B	895612046	491677162	1.82	957.70	116.10
Level-2	1185674280	89762530	13.21	445.30	82.70
Level-3	1457438720	394182203	3.70	1949.40	140.50
Level-4	11520000	597784	19.27	7.30	0.20
Total	3550245046	976219679	3.64	3359.70	339.50

TABLE 4
TOTAL SAVINGS USING SZIP AT GSFC DAAC

Scenario	Additional Cost ($K)	Storage Savings ($K)	Network Savings ($K)	Net Savings ($K)
SZDC	$-$100	1,547	486	2,033
SZDU		1,547	0	1,447

At the time of Phase II testing, the acquired Level 2, 3 and 4 data sets contained a large percentage of fill values which were not typical of true data products. Thus the Szip data reduction is far from being the achievable optimal result. However, Phase II results did show that the Szip compression time on the ECS production machine for a typical set of MODIS products (one granule including 1km, 500m and 250m resolutions) is about 3 minutes. It takes almost one hour to compress the same data sets with Gzip. Only the Szip compression speed is suitable for the ECS operations since the compression time for a granule is relatively small (~10%) compared to the time it takes to generate one granule of Level-1B data from Level 0 data.

In Phase II, a scalability analysis model was established to assess the cost savings from using CCSDS lossless data compression in the ECS operational system. This was done based on the data volumes archived and distributed at the GSFC DAAC. The scalability model is used to assess cost savings from using Szip as a data compression algorithm for two scenarios: (1). SZDC - compress before storing and distribute in compressed form; and (2). SZDU - compress before storing and uncompress before distributing. SZDC saves bandwidth and network transmission time over SZDU but requires users to decompress files. Assuming utilizing the DAAC hardware tape compression at 1.5:1 compression for Level 0 and the use of Szip compression for Level 1 and above data, the savings in storage and network over an 8 year period at the GSFC DAAC are shown in the Table 4.

Based on the results from Phase I and Phase II studies, ESDIS recommended implementing the lossless data compression algorithm, *Szip* into the HDF library at NCSA.

C. Phase III (7/01 – 2/02)

In Phase III[1] , we worked towards integrating the CCSDS lossless data compression algorithm into the HDF library. HDF-4 was chosen for prototyping due to its wide acceptance in the GSFC science community.

HDF-4 currently supports the following compression schemes: Run-length encoding (RLE), Adaptive Huffman (SKPHUFF) and Gzip compression (deflate). It is only natural that the CCSDS lossless data compression algorithm is implemented in the same manner as the other existing algorithms. For this prototyping activity, the version of HDF-4 with Szip subroutine is named HDF-4-Szip.

MODIS Level-1B data was acquired from the GSFC DAAC and used for testing HDF-4-Szip. Sample utility programs for using HDF-4-szip were written in C to facilitate usage. These include routines to compress binary arrays and store them in HDF format, compress an existing HDF file, and extract binary data from a compressed HDF file.

A comparison of results using existing HDF-4 lossless compression routines and HDF-4-Szip is shown in Table 5 on one MODIS granule of size of 343 Mbytes. The test was performed on Pentium II 300 Mhz processor with Linux-7.1.

TABLE 5
LOSSLESS COMPRESSION RESULTS USING VARIOUS COMPRESSION TECHNIQUES ON MODIS REAL SENSOR DATA TESTED ON PENTIUM II 300MHZ PROCESSOR

Technique	Rle	Huff	Szip	Gzip
Compression Ratio	1.60	2.28	2.80	2.37
Compress Time(sec)	85.7	558.4	71.6	273.1
Decompress Time(sec)	41.6	574.9	63.6	68.3

IV. BENEFITS

The most significant impact of this project is that the lossless compression processing will reduce the archive storage requirement to less than one half of the original data size. This would amount to significant savings in cost and maintenance. A second major impact is that the compressed files will require less than half of the time to be transported between data centers and users. In cases when data transmission across a congested leased line or Internet is involved, this would amount to the reduction of the total operational cost for data distribution. Implementing the CCSDS lossless data compression into the HDF library provides a transparent method for users to capitalize on reduced data volumes since the HDF library contains an interface for storing and retrieving compressed or uncompressed data. Because the data compression and decompression times of szip compression

[1] Jointly funded by ESDIS Prototype Program and the GSFC standards program

are significantly less than those for other compression techniques, use of szip in operational EOS scenarios becomes very feasible.

V. FUTURE WORK

In the prototype study, we have demonstrated that Szip is the most suitable compression algorithm for EOSDIS and that the most efficient way to use Szip is to implement it into the HDF library. Before the integrated HDF-4 is distributed, the following work still needs to be completed:

1. Perform a complete test of HDF-4-Szip on the EOSDIS operational product machine;
2. Perform tests on typical Level 2, 3 and 4 data;
3. Provide additional format support (32-bit integer and float) in Szip;
4. Support implementation of Szip into HDF-5;
5. Advocate usage in GSFC science community.

ACKNOWLEDGEMENTS

Special thanks are due to Karen Moe at GSFC for her encouragement and support, to Al Fleig of MODIS MODAP for his advice from the MODIS system point of view, to Chris Lynnes and George Serafino of GSFC DAAC for their suggestions and providing test data, to Robert Wolf for his advice from the MODIS scientist point of view, and to Mike Folk, Robert McGrath of NCSA, for their collaborations. The authors would also like to thank the following colleagues for their various support: Gail McConaughy, Michael King, Ed Masuoka, Gary Roth, Robert Westbrook, Jian-Chun Qin, Liping Di, Doug Ig.

ACRONYMS

Sz (Szip) = one software implementation of the CCSDS lossless data compression algorithm
Sunzip = Szip decompression
Gz = LZW based Gzip
Gunzip = Gz decompression
Cz = LZW based Unix Compress
Az = JPEG arithmetic coding lossless data compression

REFERENCES

[1] HDF Users Manual and HDF Reference Manual
[2] "A technique for high performance data compression," Welch, T, IEEE Computer, v.17, no. 6, 6/1984.
[3] "An introduction to arithmetic coding," Langdon, G., BM J. Res. Develop, 28(2), 3/1984.
[4] CCSDS 121.0-B-1: Lossless Data Compression. Blue Book. Issue 1. 5/1997.
[5] CCSDS 120.0-G-1: Lossless Data Compression. Green Book. Issue 1. 5/1997.
[6] CCSDS document available from http://www.ccsds.org/ccsds/ccsds_document_access.html
[7] JPEG Still Image Data Compression Standard, Pennebaker, W. & Mitchell, J., 1993.