Chapter 15

An Intelligent Archive Testbed Incorporating Data Mining

H. Ramapriyan[a], D. Isaac[b], W. Yang[c], B. Bonnlander[d], D. Danks[e][1]

[a]NASA Goddard Space Flight Center, Greenbelt, MD

[b]BPS Consulting, Inc., Bethesda, MD

[c]George Mason University, Fairfax, VA

[d]Institute for Human and Machine Cognition, Pensacola, FL

[e]Carnegie Melon University, Pittsburgh, PA

[1] This work was performed by the first author as part of his official duties as an employee of the U.S. government. It was supported by the NASA's Science Mission Directorate. The remaining authors were supported under Cooperative Agreement NCC5-645 between NASA and George Mason University. The opinions expressed are those of the authors and do not necessarily reflect the official position of NASA.

15.1 Introduction

Many significant advances have occurred during the last two decades in remote sensing instrumentation, computation, storage, and communication technology. A series of Earth observing satellites have been launched by U.S. and international agencies and have been operating and collecting global data on a regular basis. These advances have created a data rich environment for scientific research and applications. NASA's Earth Observing System (EOS) Data and Information System (EOSDIS) has been operational since August 1994 with support for pre-EOS data. Currently, EOSDIS supports all the EOS missions including Terra (1999), Aqua (2002), ICESat (2002) and Aura (2004). EOSDIS has been effectively capturing, processing and archiving several terabytes of standard data products each day. It has also been distributing these data products at a rate of several terabytes per day to a diverse and globally distributed user community (Ramapriyan et al. 2009). There are other NASA-sponsored data system activities including measurement-based systems such as the Ocean Data Processing System and the Precipitation Processing system, and several projects under the Research, Education and Applications Solutions Network (REASoN), Making Earth Science Data Records for Use in Research Environments (MEaSUREs), and the Advancing Collaborative Connections for Earth-Sun System Science (ACCESS) programs. Together, these activities provide a rich set of resources constituting a "value chain" for users to obtain data at various levels ranging from raw radiances to interdisciplinary model outputs. The result has been a significant leap in our understanding of the Earth systems that all humans depend on for their enjoyment, livelihood, and survival.

The trend in the community today is towards many distributed sets of providers of data and services. Despite this, visions for the future include users' being able to locate, fuse and utilize data with location transparency and high degree of interoperability, and being able to convert data to information and usable knowledge in an efficient, convenient manner, aided significantly by automation (Ramapriyan et al. 2004; NASA 2005). We can look upon the distributed provider environment with capabilities to convert data to information and to knowledge as an Intelligent Archive in the Context of a Knowledge Building system (IA-KBS). Some of the key capabilities of an IA-KBS are: Virtual Product Generation, Significant Event Detection, Automated Data Quality Assessment, Large-Scale Data Mining, Dynamic Feedback Loop, and Data Discovery and Efficient Requesting (Ramapriyan et al. 2004).

Large-Scale Data Mining (LSDM) plays a very important role in IA-KBS. Two important uses of LSDM are: retrospective studies covering large temporal and geographic extent; and precursor detection, where indicators of significant events are identified through analysis of historical data. Note that, once good precursors have been identified via a scientific LSDM process, computationally efficient filters on real-time data streams can be constructed so that significant events can be detected in observational data in near real-time and users can be alerted.

Especially relevant to data mining in the above discussion, there have been several research investigations in the area of Intelligent Data Understanding that were supported by NASA's Intelligent Systems Project under the Computing, Information and Communication Technology Program that can contribute to the goals of an IA-KBS. However, these investigations typically perform proofs of concept on a relatively small scale. Before their contributions can be implemented on a large scale commensurate with today's Earth science data archives, it is necessary to test them in a pseudo-operational environment. In this chapter, we describe the implementation of a testbed to accomplish this and discuss some of the observations and lessons learned from its implementation. This is a more detailed discussion than the summary of the implementation and results presented in Ramapriyan et al. 2005.

In Sect. 2, we present the basic concepts of an IA-KBS. In Sect. 3, we discuss the goals of the testbed and describe the application scenario (prediction of wild fire potential from historical and current remote sensing observations) being tested in the testbed. In Sect. 4, we provide a description of the algorithm used and implementation details. In Sect. 5, we present the results of implementation including derived fire prediction maps, processing speeds and feasibility of pseudo-operational implementation. In Sect. 6, we document our conclusions and lessons learned.

15.2 Intelligent Archives

The concept of intelligent archives was spawned out of long experience with Earth science data archives and recognition of the convergence of two factors: the accumulation of enormous quantities of valuable scientific data and the increasing practicality of applying machine learning techniques to large-scale data sets. The result was a growing belief that current data archives could and should evolve to better support the scientific process and help unlock the untapped potential of both current data holdings and ongoing observational data streams. We believe the time has come for intelligent archives, which not only store and disseminate data, but also play an active role in the knowledge building process.

Today's data archives play a key role supporting the value chain that turns data into information and knowledge. They provide common repositories for the collection and dissemination of information at all levels, from raw observations to high-level analyses. And yet it is clear that they have the potential to provide much more value in the overall value chain. For example, archives are uniquely positioned to perform the following useful functions:

- Mining archived data holdings to add metadata and thereby improve data access and usability;
- Identifying quality issues as data are being stored, while there is still the opportunity to recapture the affected observations;
- Detecting unusual patterns in data that may indicate an event of interest; and
- Facilitating collaboration and exchange among distributed research groups.

An assessment of the potential role and function of an intelligent archive identified six key capabilities such a system should exhibit: virtual product generation, significant event detection, automated data quality assessment, large-scale data mining, dynamic feedback, and data discovery and efficient requesting.

15.2.1 Virtual Product Generation

An archive does not need to produce and archive all of the derived data products that will be requested of it in advance. In many cases, only a small percentage of the products generated are actually requested. Virtual product generation allows a user to treat a product as though it were being retrieved from the archive when, in reality, the data inputs are automatically retrieved, assembled, and processed into the desired form "on the fly," in response to the request (Clausen and Lynnes 2003). This adds latency but can result in significant storage cost savings and eliminate the need for reprocessing as algorithms are improved. An intelligent archive minimizes latency by anticipating demand (e.g., based on predictive models of usage patterns and significant event detection), computing needed products just in time with a relatively small percentage of the total data.

Another aspect of virtual product generation is the ability to assemble, transparently, inputs to the production algorithm from a variety of sources and locations. This requires a global registry, interface standards, and supporting middleware (so that production algorithms receive the data in an acceptable and consistent format).

Further, a virtual product generation capability should optimize the assignment of processing resources by minimizing the need for data communications, and considering current resource utilization and availability. This means building and using a global predictive model of the interconnected network of storage and processing resources, and keeping the model current via frequent state updates (where “frequent” might mean on the order of tens of seconds).

15.2.2 Significant Event Detection

With a constant stream of several terabytes of data per day entering an archive, it is likely that manual analyses will miss some significant events, or at least miss them until the opportunity to perform focused collection of additional information related to the event has passed.

Significant event detection helps identify phenomena of interest within very large data sets and data streams. This in turn enables not only near-real-time reporting of geophysical events in an ingest data stream, but also content-based queries if the events are stored as metadata. The idea here is to place matched filters or pattern recognition algorithms on an input data stream (or streams) to automatically detect the occurrence of an event. Some examples of significant events are hurricanes, wild fires, volcanic eruptions, and failure of a sensor or some other part of the information processing chain. The event detection, in turn, could trigger a variety of actions such as notification of subscribers, generation and distribution of associated products, retasking of sensor assets, reallocation of system resources, and self-repair.

15.2.3 Automated Data Quality Assessment

As in the case of significant events, experience has shown that data quality issues can lay undiscovered for long periods, hidden in the flood of data. In addition to surfacing such issues in a timely manner, the potential exists to circumvent a variety of complex data quality issues.

Automated data quality assessment maintains the algorithmic processing pedigree of a data product, and ensures the scientific and algorithmic consistency of the underlying modeling and processing assumptions (Isaac and Lynnes 2003). An intelligent archive can take responsibility (to a greater or lesser extent) for monitoring and perhaps correcting the quality of the products it delivers to a requestor or a consuming process. This includes both the algorithms and inputs used to generate the product (that is, a sophisticated kind of product provenance and configuration control) as well as internal inspection of the products to ensure that they meet a variety of user-specified characteristics (e.g., regarding cloud cover, dynamic range, or sampling resolution). Other sources of error (e.g., bit errors, compression/decompression lossiness and mistakes in indexes or metadata) can also be detected prior to delivery and, in some cases, corrected or ameliorated (e.g., through interpolation or other data modeling approaches).

15.2.4 Large Scale Data Mining

While data mining has already proven to have utility in Earth science data analysis, an intelligent archive must perform data mining efficiently to enable the analysis of large data volumes. Here, the primary meaning of the term is a process that finds higher-level emergent causal relationships at a modeling level above the level at which the inputs exist. Typically, data mining sits at the top of the value chain – taking information as input, and producing knowledge. Two important examples of this type of analysis are (1) retrospective studies covering large temporal and geographic extent, and (2) precursor detection, where indicators of significant events are identified through analysis of historical data. Note that once good precursors have been identified via this scientific data mining process, computationally efficient filters on real-time data streams can be constructed. The output of a successful data mining process is typically a model that can serve as the basis for prediction, event detection, classification, quality assessment, or other purposes. The knowledge derived from successful data mining may also have other value or utility: pure science (discovering or confirming previously unverified correlations or relationships – e.g., in global climatology); efficiency (discovering that only some inputs contribute significantly to an output); and instrument or spacecraft health and safety (detection or prediction of anomalies).

15.2.5 Dynamic Feed-back

The ability to stochastically optimize the allocation of the storage, network, and computing resources of the archive using dynamic feed-back loops supports the other archive functions by increasing throughput and reducing user-experienced latency in product delivery (Morse et al. 2003). An intelligent archive can be modeled (McConaughy and McDonald 2003) as one component in a complete system that includes sensor tasking, sensors and collection, ground station early level product generation, archiving and higher-level production, real-time or near real-time applications, and user request satisfaction and associated production. There are two low-latency feedback loops of interest. The first is for retasking of resources to support time critical scenarios (e.g., fire detection). The second is for system resource optimization to maximize throughput and minimize latency of delivered user-requested products.

15.2.6 Data Discovery and Efficient Requesting

Some advanced capabilities of an intelligent archive, particularly virtual product generation, have the potential to put heavy loads on cooperating systems. For that reason, an intelligent archive should act as an intelligent requestor of data, exploiting its knowledge of information interrelationships and computing resources to minimize its load on cooperating systems. In addition, intelligent archives of the future should be able to detect the presence of newly available data anywhere in the world (Isaac and McConaughy 2004), determine the usefulness of the data, learn how to access them and ultimately provide the data to users or applications in a usable form. This involves an ongoing search process that keeps constantly and persistently aware of potential data sources and their changing status, and capabilities to retrieve and reformat the data transparently to meet the users' data interface requirements or preferences. In this way, the intelligent archive becomes an interface not only to its own holdings, but also to the broad array data sources of interest across the accessible web. Together or independently, these capabilities have the potential to significantly improve support of the knowledge-building value chain.

15.3. Testbed and Scenario

The concept of an intelligent archive is intriguing. But it is even more interesting if it is feasible to implement the concept using current computing technology. A test using a realistic scenario on realistic data volumes suggests that it is.

Perhaps the most important capability identified for an intelligent archive is the ability to facilitate the transformation of data into knowledge in a distributed environment using large-scale data mining. Data mining algorithms are generally viewed as unsuited for large-scale use disciplines like Earth science that involve very high data volumes. There have been many research projects that have developed algorithms with promising results. But they have been tested on data subsets of only a few gigabytes, while large-scale datasets tend to be multiple terabytes in size. To bridge the gap between research and operations, we constructed a testbed to see how a typical algorithm would perform on full-scale data sets.

The testbed provides the computational and data resources required for implementing IA-KBS concepts on a scale that provides concrete evidence about the associated benefits and risks prior to implementing these concepts in operational systems. Such evidence is important both to the users of information and to data managers. The testbed provides a capable and flexible infrastructure for exploring a variety of data mining scenarios, though the focus in this discussion is primarily on performance and utility outcomes rather than system and software components or frameworks.

Using the six key capabilities identified above as a guide, several NASA-funded research projects and their algorithms were surveyed and assessed for their applicability to the testbed. Initially, a "reference architecture" for the IA-KBS was prepared (Morse and Yang 2004). In this reference architecture, key interfaces and dependencies were specified, overlapping or redundant functionality was eliminated, and sample use cases and associated operations' concepts were created and described. The reference architecture makes evident where a given research algorithm or capability might reside. The goal was to find research that shows both scientific and operational relevance, and that is applicable to as large a subset of the IA-KBS functional capability as possible. Other evaluation criteria that entered into the selection process included implementation feasibility, source data availability, and collaboration potential.

The problem selected for demonstration is fire prediction (Bonnlander 2005). Fire prediction is very relevant socially. Also, it is a challenging problem, since there is a large component of stochastic uncertainty. Fuel type and availability, moisture, sources of ignition, temperature and precipitation –over considerable lengths of time and seasonal conditions – influence the predicted fire potential. Analysis shows that the algorithm can exercise most of the functional areas identified in the IA-KBS reference architecture. Finally, the scientific goals of the research will benefit from the testbed, since the testbed can process a large variety of geographic areas, fuel types, and seasonal conditions, and hence significantly extend the scientific relevance of the algorithm into new and previously unexplored aspects of the underlying phenomena.

15.3.1 Design Issues

The IA-KBS project had identified a number of technical issues associated with implementation of the intelligent archive concepts, most notably scalability issues (McConaughy and McDonald 2003; Isaac and McConaughy 2004). These issues needed to be addressed in a manner that was operationally relevant and yet could be implemented on a limited budget.

- Scalability and Parallelization. Two approaches to parallelization were considered: coarse grained and fine grained. Coarse grained parallelism was chosen because, although it requires partitioning the data and generating multiple independent models, it is much easier to run multiple instances of a data mining algorithm on each node than it is to implement a truly distributed algorithm. In the fire prediction scenario, it was convenient to partition data along fuel type (land-cover) and month of the year.
- Source Data Restructuring. Most data mining algorithms require observational data with one record per observation having all the relevant independent variables, while remote sensing data are typically stored as files with parameters covering contiguous areas in space and time. The fire prediction algorithm was no exception to this. Although an indexing scheme could be used to map from one representation to another, the resulting physical data access would "bounce" around the source data, resulting in poorer performance. Therefore, we chose to physically re-order the data into the form and format expected by the data mining algorithms.
- Representation of Time. It is obvious that, since prior precipitation has a significant effect on fire potential, prediction of fires involves time explicitly. However, there is no explicit mechanism in the algorithms selected for the testbed for handling time. Instead, observations for the same variable at different times are simply transposed into multiple variables within a single record for input into the data mining algorithm. This temporal "flattening" is straightforward unless the time series is very long, in which case the dimensionality of the data space would grow too large. For the fire prediction scenario, we kept the time series short by reducing measurements into a few averages for the prior day, week, month, etc.

15.3.2 Testbed Design

Prior work in the IA-KBS project identified general capability needs, challenges, and opportunities (Ramapriyan et al. 2004). The testbed design provides an opportunity to explore these general concepts at a practical level that would be relevant to an operational system. The testbed design is discussed briefly below as three different views: a system network view, a functional view, and a software component view.

System Network View

One of the most important aspects of the testbed is that it should demonstrate intelligent archive concepts in an operationally-relevant environment without interfering with the production operations of an actual operational system. Features included to make the environment "operationally relevant" include a high-performance node for the data mining and event detection components, use of pre-production and production archive nodes for source data, and high-speed networks for node connections. Fig. 15.1 shows a simplified view of the components from the system network's point of view. The peer archives shown here are two of several Distributed Active Archive Centers (DAACs) that operationally archive and distribute NASA's Earth science data. For the purposes of the testbed, data are obtained from these DAACs and stored in a separate IA storage node.

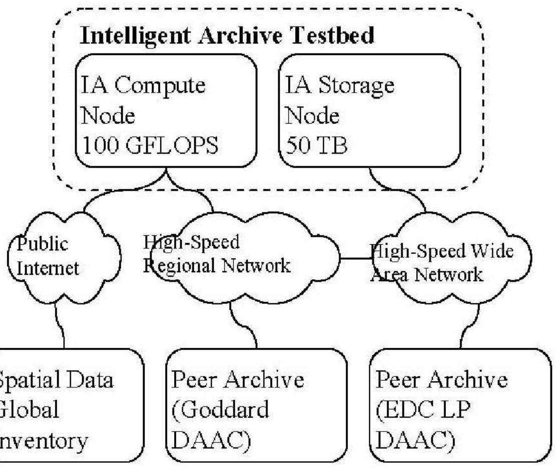
Intelligent Archive Testbed
IA Compute Node 100 GFLOPS
IA Storage Node 50 TB
Public Internet
High-Speed Regional Network
High-Speed Wide Area Network
Spatial Data Global Inventory
Peer Archive (Goddard DAAC)
Peer Archive (EDC LP DAAC)

Fig. 15.1. Intelligent Archive Testbed System Components

Functional View

The testbed includes a subset of the functional components identified in the IA-KBS reference architecture (Morse and Yang 2004), which were derived directly from the envisioned IA capabilities. The primary focus of the testbed is on the data mining and event detection components. These two components work together: the data mining component examines historical data to extract a statistical model of fire potential; the event detection component then uses this model to scan current data to assess current fire potential.

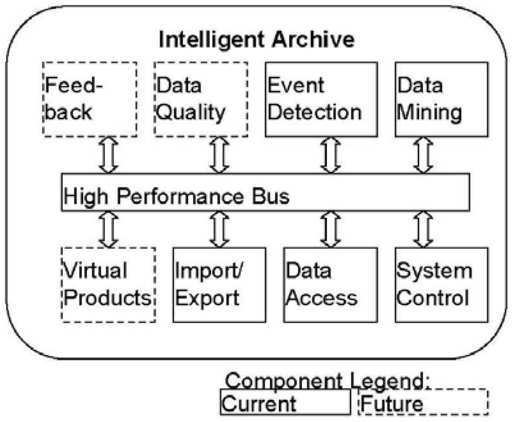

Fig. 15.2. Intelligent Archive Functional Components

The data mining algorithms process the prepared data and extract a statistical model of fire potential based on the available remote sensing parameters. The algorithm implemented is logistic regression, since it performs well in terms of both accuracy and computational effort. The data mining algorithms are implemented in MATLAB.

Software Component View

The testbed includes a variety of software components that together provide the infrastructure needed to manipulate and mine large volumes of data. These are grouped as shown in Fig. 15.3 into layers of services at differing levels of abstraction.

The Local Application Platform Layer provides Job Management and Data Access services. Job Management includes the MATLAB Distributed Computing Toolbox/Engine for dispatching different data pre-processing, data mining, and event detection tasks to different IA Computational Node processors. Data Access services include Hierarchical Data Format (HDF) libraries for reading NASA remote sensing data files, and MATLAB I/O for managing pre-processed data.

The Grid Services Layer provides a variety of services for locating and accessing distributed computing and storage resources. The testbed uses these services mainly to identify and obtain source data for use by the data mining and event detection components in the Application Layer. The Collective services employ the EOS Clearinghouse (ECHO) for identifying specific files that contain the remote sensing parameters for the times and locations of interest. The Resource services are used primarily to access data from the grid using GridFTP. The Connectivity services include services for authenticating the local server to the grid, plus low-level communication services.

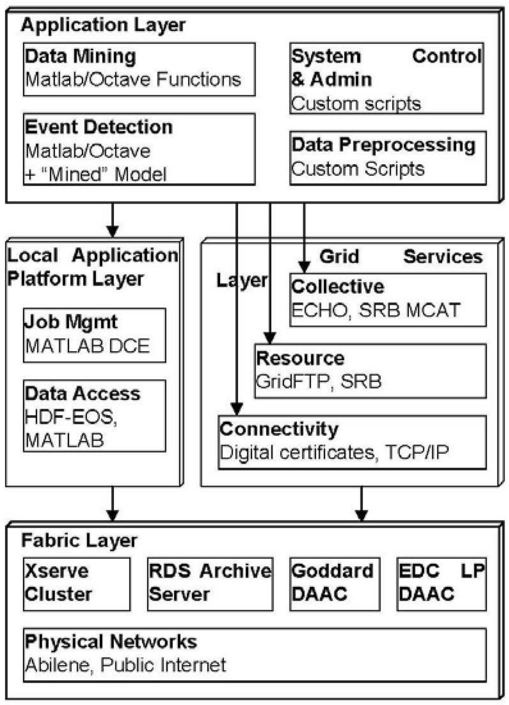

Fig. 15.3. Intelligent Archive Software Components

Data Preparation and Mining

As noted above, the data mining problem selected for demonstration of the IA testbed is wildfire prediction. Fire potential is determined by a number of parameters for which good remote sensing data exist, including temperature, precipitation history, fuel type and availability, and fuel moisture. Ignition events, including lightning and human activities, are the final factor in the occurrence of wildfires. However, these are excluded from the predictive model because these are immediate causes of fires rather than predictors of fire potential several days ahead of time.

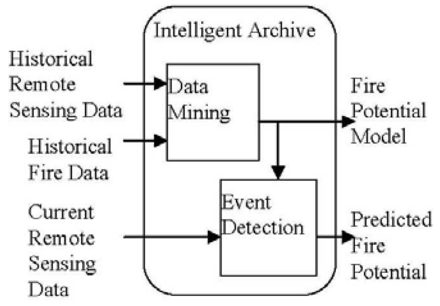

Fig. 15.4. Research scenario

15.4. Testbed Implementation

15.4.1 The Testbed Algorithm Implementation

Algorithm: The algorithm follows a set of steps that reflect a fairly standard approach to statistical forecasting. This approach involves defining a collection of independent variables (in this case, variables representing climate and historical fire occurrence data) and a single dependent variable (the occurrence of at least one fire within the next N days) in order to construct a model that produces a probability of fire conditioned on the independent variables. In abstract terms, the algorithm assumes that all available climate and fire occurrence data can be spatially and temporally co-registered, so as to produce a temporal sequence of data grids for each data source (one grid per day for each data source) over a specific date interval covering several years. In these terms, the independent variables for the statistical model correspond to the data values associated with a particular grid cell location on a specific date, and the independent variable corresponds to the presence or absence of fire within the next N days at that same grid cell location (a binary value).

Important details of the algorithm include a description of the particular variables chosen, the preprocessing steps used to standardize the data, and the particular statistical modeling approach used. The choice of variables was guided in part by a visual analysis of the fire occurrence data, which showed that both land cover type (e.g., grasslands versus coniferous forests) and time of year were important factors in determining fire frequency. For this reason, the algorithm parameters were chosen to produce a unique model for each combination of 17 different land cover types and the 12 different months in the year. The main preprocessing steps involved co-registering data values onto a 8km-resolution grid covering the coterminous United States, creating separate "training sets" for each statistical model based on land cover type and month of the year, and rescaling values for the independent variables in each training set to have zero mean and unit variance. More details regarding preprocessing are given in the next section.

The choice of logistic regression as the statistical modeling approach was based upon an extensive study comparing the out-of-sample forecasting accuracy of models based on logistic regression, binary classification trees, and Support Vector Machines (Bonnlander 2005). Results of the comparative study indicate that logistic regression provides consistently superior forecasting accuracy over the other two approaches, and it had the extra benefit of being the fastest of the three approaches by several orders of magnitude. The logistic regression implementation used here came from the glmfit() function provided in the Matlab Statistical Toolbox, Release 13.

Two-phase implementation: The implementation of the wildfire prediction algorithm in the testbed underwent two phases. The goal of the first phase was to build the testbed's hardware and system software environment and to replicate the algorithm originally implemented in the IHMC and compare the algorithm performances in the IHMC machine and the testbed machine. During the first phase, the source code, written in Matlab, and the input data sets was rehosted to the testbed's head node and its local file system. The goal of the second phase was to investigate the performance and feasibility of the algorithm in an operationally-relevant environment with distributed parallel computing and EOS' production data products. During the second phase, most of the data were obtained from RDS through its subscription to operational NASA DAACs. The data were stored in the Xsan administered RAID system in the testbed. The parallelization was achieved through the use of the Matlab Distributed Computing Engine (MDCE). With MDCE, one could start a job manager to manage a number of workers, which perform computation tasks. When multiple workers were started and run at different computing nodes in a computing cluster to collectively perform a subset of a larger computing task, it achieved the goal of parallelizing the computing task. The fire prediction algorithm involved training and building hundreds of models for different land surface types and at different prediction time frames. The Matlab code was written such that model building process could be performed for multiple models or for any subset of models. The code also employed a lock mechanism which would lock a task being processed so that an available computing process could skip a locked task and proceed to the next task. These coding techniques made MDCE very suitable to achieve parallelization. MDCE licenses were installed in four of the six dual-processor computing nodes, which allowed a maximum of eight workers to be run at the same time.

Source Data used in Phase I: Source data used in the algorithm in IHMC and during the first phase of the testbed included four types of information: weather parameters, vegetation condition parameters, national fire occurrence database, and fuel code map.

a) Weather parameters: the weather parameters include four Global Surface Summary of the Day (GSSD) variables from National Climate Data Center (NCDC). They were daily minimum air temperature (TMIM), maximum air temperature (TMAX), precipitation (PRECIP), and vapor pressure deficit (VPD). There were about 6000 observation points globally, among which about 1200 were in the continental U.S.
b) Vegetation condition parameters: the vegetation condition parameters included leaf area index (LAI) and Fractional Photosynthetically Active Radiation (FPAR) derived from MODIS measurements. LAI and FPAR were two of the standard MODIS land products. They were available at 1km spatial resolution.
c) Fire occurrence information: fire occurrence data included the location and size of fire for a specific day. These data were obtained from the National Interagency Fire Management Integrated Database (NIFMID) managed by USFS. The time coverage of the data was from 1986 to 2004, among which 2004 data were 90% complete.
d) Fuel code map: the fuel code map included 24 fuel types among which 20 were vegetation types (Burgan et al., 1998). The map was developed primarily based on the 1-km land cover map derived from AVHRR NDVI by Loveland et al. 1991.

All data were co-registered to gridded 8-km resolution matrices in Lambert Azimuthal equal area projection with center latitude being 45.0 degrees and central meridian being -100.0 degrees. Each grid matrix has 361 rows and 573 columns, covering the entire conterminous U.S. Both weather and vegetation condition parameters were preprocessed by the Terrestrial Observation and Prediction System (TOPS) project at AMES (http://www.ntsg.umt.edu/tops/webpages/right/flowchart/index.php).

MODIS data were aggregated from 1km to 8km while the weather data were interpolated to 8km grid using techniques described in Jolly et al. 2005. These data covered time periods between March 5, 2000 and December 31, 2004. The National Interagency Fire Management Integrated Database (NIFMID) provides records of fires occurring on USFS land that required suppressive action for the years 1986-2003 (USDA 2003), including fire location, ignition date and final fire size. The fuel code map, given at 1-km resolution with the same map projection, was aggregated to 8-km resolution using Matlab code. Preprocessing code was written to convert the data into Matlab matrices containing standardized values for independent variables.

Source Data used in Phase II: During the second phase, data obtained directly from the operational NASA DAACs were used to replace some of the input data used in phase I. These included MODIS snow/ice, FPAR/LAI, land cover, Thermal/fire, and TRMM 3-hr gridded precipitation data. Among the above operationally available data, the MODIS FPAR/LAI product was also used in the phase I implementation, but during phase II the original preprocessed data were replaced by data directly from the NASA DAAC. For the other data products, MODIS snow/ice was used as a new prediction variable; MODIS land cover was used to replace the fuel code; TRMM precipitation was used to replace the original precipitation data; and the MODIS thermal/fire was used to replace the dependent fire occurrence variable. The minimum and maximum air temperature and the vapor pressure deficit data used in phase I were still used in Phase II model training and forecast. It was originally planned to also replace these preprocessed data using NASA operational data such as MODIS land surface temperature and atmospheric profile products. After some initial implementation tests, it was determined that the resources required would exceed those available to the project.

Preprocessing of NASA operational data: The operational NASA data were in different resolutions, both spatial and temporal, and in different coordinate reference systems (CRS) as compared to the data sets used in Phase I. Preprocessing was needed to transform the operational data into a form that could be input into the IHMC algorithm. The preprocessing functionalities included a) CRS transformation to convert MODIS sinusoidal and TRMM geographic CRS to Lambert Azimuthal Equal Area projection; b) spatial resolution transformation to resample/interpolating 500-m (for the snow/ice data) and 1-km MODIS data and the 0.25-degree TRMM data into 8-km grid; c) temporal resolution transformation to convert TRMM 3-hour measurements to daily values; d) mosaicking of multiple MODIS tiles into one single coterminous US grid; and d) file reformatting to convert MODIS HDF-EOS and TRMM native HDF formats to generic binary format. Although it was possible to develop one single tool for all the products because the many aspects of the preprocessing were the same for different products, it was decided that one tool for each product was appropriate because it was easier to configure them for individual products. These tools were placed in the training/prediction process flow so that fire prediction models could be retrained when needed and be used to make real-time or near real-time forecast when new data products become available. Preprocessing of all the five operational products added less than 40 seconds overhead to the model forecast.

Construction of wild fire prediction models: After all input data were preprocessed into a co-registered 8-km grid, they were separated into independent and dependent variables. The independent variables for a given day, T, and a given year, Y, were the following:

a) TMIN, TMAX, VPD, PRECIP, and SNOW on day T in year Y,
b) Averages of TMIN, TMAX, VPD, PRECIP, and SNOW over the days [T-7,T-1] in year Y,
c) Averages of TMIN, TMAX, VPD, PRECIP, and SNOW over the days [T-30,T-1] in year Y,
d) FPAR on day T-1 in year Y,
e) LAI on day T-1 in year Y, and
f) Number of fires in the previous year (all of Y-1)

Where TMIN, TMAX, VPD, PRECIP, and SNOW are minimum temperature, maximum temperature, vapor pressure deficit, precipitation, and percent snow cover, respectively.

The corresponding dependent variables were the following:

a) Fire occurrences (zero or one) in the days [T,T+29] for 30-day models
b) Fire occurrences (zero or one)in the days [T,T+6] for 7-day models
c) Fire occurrences (zero or one) in the day T for 1-day models

That is, for an N-day model, if no fire occurred in the days [T, T+N-1], the dependent variable would have a value of zero, and a value of one otherwise.

The independent and dependent variables were used to train three model types, each for 1-, 7-, and 30-day prediction. If required input data on and before day D-1 are available, the models can predict fire potential respectively, for day D, for a 7-day period staring from D (i.e., [D, D+6]), and for a 30-day period staring from day D (i.e., [D, D+29]). The model building, i.e., the generation of the logistic regression models, is done by calling related Matlab function glmfit(), which is provided in Matlab's Statistics Toolbox.

Because the behavior of fires is different at different times of a year and over different land cover types, one single model is not likely to predict fire at all times and locations. The models were separately built for different land cover types and for each of the 12 months in a year. The MODIS land cover product identifies 17 different land cover types, among which 14 occur in the 2001 data used in the tests reported here. Thus, a complete model building process generates 504 models (i.e., 3 model types; 14 land cover types; 12 months; 3 x 14 x 12 = 504).

15.4.2 The Testbed hardware and software configuration

The testbed was built using an Apple G5 Xserve cluster. The head node of the cluster consisted of dual 2.0GHz G5 processors, 4GB of DDR SDRAM and two 250GB hard disks. There were five cluster nodes, each having dual 2.0 or 2.3 GHz G5 processors with 2GB DDR SDSRAM and 8GB hard disk. The combined peak performance of all 12 processors was 103 GFLOPS. The storage system consisted of five RAID arrays with total capacity of 22 Terabytes. The RAID arrays and cluster nodes were interconnected via 2Gbps Fiber Channels. A gigabyte Abilene network connected the testbed, Remote Data Storage (RDS) facility, the NASA DAACs, and other resources. The RDS contained a transient data storage system of 47 TB, of which 2 TB were allocated to the testbed activities, and a persistent data storage system of 185 TB. The subscription to NASA Data Pool for data products needed by the wildfire prediction algorithm and a notification mechanism to the testbed were set up in RDS. The automatic notification processing and data pulling from RDS to the testbed were implemented in the testbed machine. Several standard open interface protocols and proprietary software were used for data transfer between the testbed and RDS machines and NASA data pools, which included the Nirvana Storage Resources Broker (SRB) software, GridFtp, and Open Geospatial Consortium Web Coverage Service. The operating system of the testbed was Tiger OSX version 10.4 and its RAID system was administrated using the Xsan version 1.1. The software used to implement the wildfire prediction algorithm was MDCE.

15.5 Results

15.5.1 Computation Speed

As mentioned before, prediction models were built for different land surface types and different model types (i.e., time frames). The MODIS land cover product identified fourteen different vegetative cover types. For each prediction type, 168 models, each for a particular month and a particular vegetative surface type, could be built. With three model types providing 1-day, 7-day, and 30-day predictions, a total of 504 were generated in the testbed. Table 15.5-1 lists times used to generate the three model types, each containing 168 individual models. In the table, the time for the 30-day model included the time used in the variable preparation, which generated independent and dependent variable arrays from input grid data. The time needed to complete this preparation was 7.1 hours using a single worker node. If this time had been deducted, the training time for the single worker 30-day model would have been 15.8 hours, which was equivalent to the other two model types.

Table 15.5-1. Times used in generating the three types of prediction models (in hours)

	Single Worker	Eight Workers
1-day model	16.5	1.6
7-day model	16.3	1.5
30-day model	22.9	2.8

(Note: The time in the 30-day model includes preprocessing time. See text)

It is to be noted that the times used in 8-worker distributed computing were less than one eighth of the time used in the single-worker computing. Two factors contribute to these results. The first is that the single worker sessions were performed in the head node of the testbed, on which many other processes from other projects were also running. The second is that the three cluster nodes, on which six of the eight workers were running, were 2.3GHz G5 dual-processor machines while the head node was a 2.0GHz G5 dual-processor machine.

Once a model was constructed, the time needed to generate forecast result for the entire conterminous U.S., at 8-km grid cell size, was about 12 seconds for one model prediction using the head node. This time included reading the input variables from disk files and writing prediction result to the output disk files. With less than 40 seconds of preprocessing from

NASA operational data included, obtaining the three predictions (i.e., 1-day, 7-day, and 30-day) each day would take less than two minutes with a single node.

The prediction computations scale linearly with the number of grid cells. Thus, if prediction models are used at a 1-km resolution, the time needed to generate the three predictions will be about two hours since the number of grid cells increase by a factor of 64. This would still meet operational daily forecast requirements. Of course, the time used to train and build the forecast models would be much longer than those shown in Table 15.5-1.

15.5.2. Forecast results

The forecast results by the 1-day, 7-day, and 30-day models were probabilities of fire occurrences in the next day, within the next 7 days, and within the next 30 days, respectively. Although the probabilities, floating point values ranging from 0.0 to 1.0, indicated the likelihood of fire occurrences, they could not be thresholded to definitive 1 or 0 values to predict if there would or would not be wildfires. Therefore, it was not possible to directly compare the forecast results to the known fires to determine the accuracy of the forecasts. Hence, visual comparisons between forecast images and known fire images and the Receiving Operating Characteristic Curve (ROC) analyses were used to assess the forecasts. A few examples of predicted images and ROC curves are presented here to show the forecast results. Figs. 15.5 and 15.6 are 30-day forecast results for the winter and summer seasons and Fig. 15.7 is a 7-day forecast result for the spring season, all in 2004. The known fire occurrences are plotted as black dots in the figures. The color bar at the bottom of each figure shows the natural log values of the forecast probabilities. These figures indicate that most known fire occurrences, which were not used in model training, fall into the high probability areas. The results demonstrate that the forecasts are visually satisfactory.

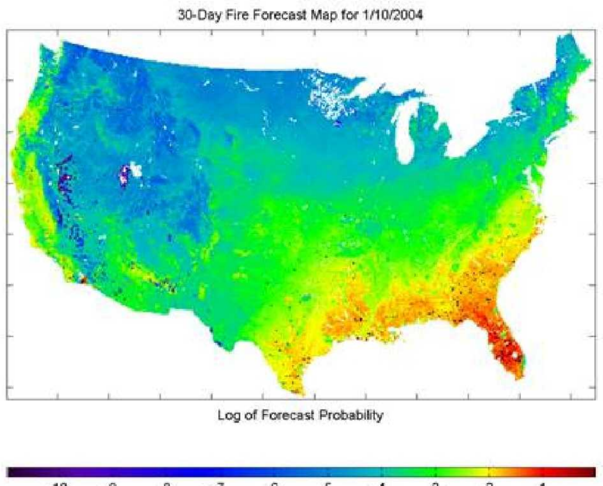
30-Day Fire Forecast Map for 1/10/2004
Log of Forecast Probability
-10
-9
-8
-7
-6
-5
-4
-3
-2
-1

Fig. 15.5. 30-day forecast for 1/11/2004-2/9/2004

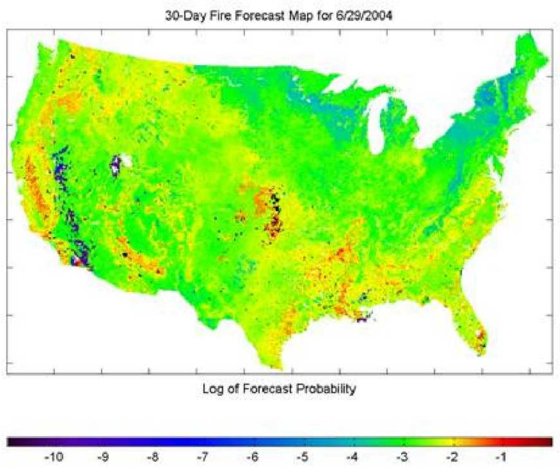
30-Day Fire Forecast Map for 6/29/2004
Log of Forecast Probability
-10
-9
-8
-7
-6
-5
-4
-3
-2
-1

Fig. 15.6. 30-day forecast for 6/30/2004-7/29/2004

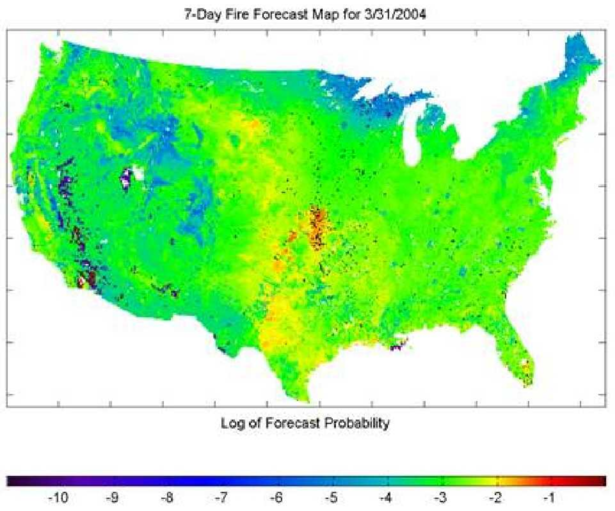
7-Day Fire Forecast Map for 3/31/2004
Log of Forecast Probability
-10
-9
-8
-7
-6
-5
-4
-3
-2
-1

Fig. 15.7. 7-day forecast for 4/1/2004-4/7/2004

More quantitative assessments of the forecasts can be performed by ROC analyses. ROC indicates how the forecast probability separates positive (fire) samples from negative (no-fire) samples. The larger the area under the ROC curve (AUC), the better the probability separates the samples. The maximum AUC is 1.0, indicating perfect separation between positive and negative samples. An AUC value of 0.5 is random prediction while values smaller than 0.5 indicate worse than random prediction. The smaller-than-0.5 case usually occurs when there are not enough samples (Hopley and van Schalkwyk 2007). Figs. 15.8 and 15.9 are ROC graphs, respectively, for the 30-day and 7-day model forecasts calculated from the evergreen needle leaf forest land cover type. These graphs show that the model forecast results are much better than random prediction in most cases. The average AUC values over twelve months for these two graphs are 0.72 and 0.74, respectively.

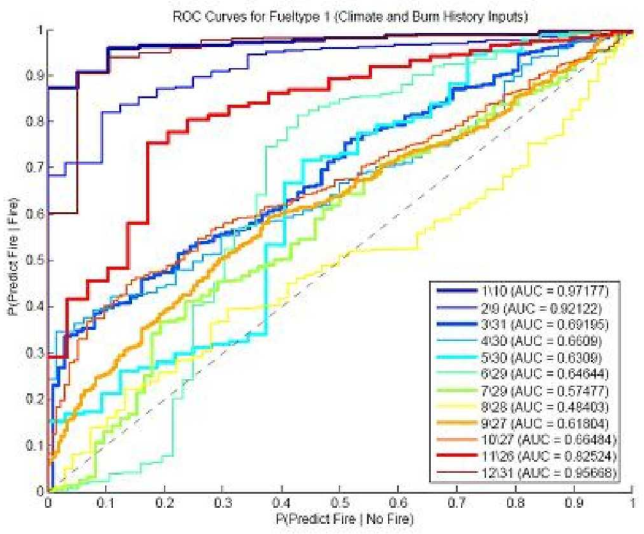

Fig. 15.8. 30 day Model ROC Curves for the Evergreen needle Leaf Forest

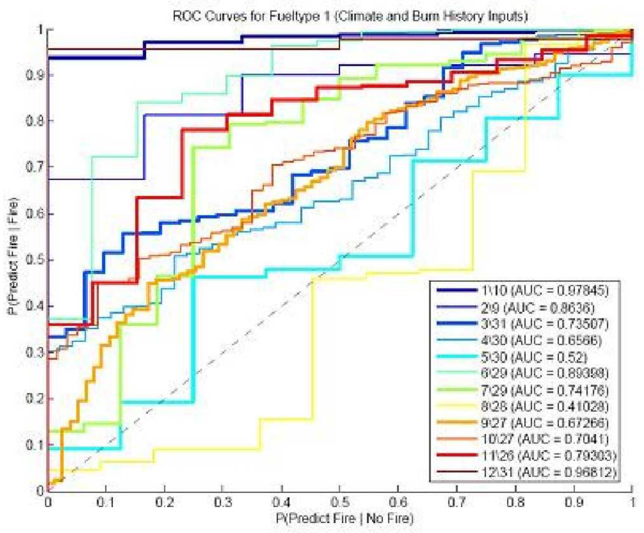

Fig. 15.9. 7-day Model ROC Curves for the Evergreen needle Leaf Forest

15.6. Observations and Conclusions

We have discussed the implementation of a testbed for an intelligent archive to demonstrate the feasibility of using data mining algorithms on a large scale with remotely sensed data in an operationally relevant environment. The testbed used for this work consisted of hardware and software in a distributed environment that was distinct from, but interfacing with, NASA's operational Distributed Active Archive Centers, so that there would be no interference with the operational systems. Several of the previously identified key functions of an Intelligent Archive were exercised through this testbed. A data mining algorithm employing logistic regression was used to develop a fire prediction model from time series of a variety of remotely sensed data and derived products. The results from the testbed were encouraging from several points of view. Some of the lessons learned and observations are given below from the points of view of: Science/Algorithm and Execution Efficiency

15.6.1 Science/Algorithm

Algorithm and parameter selection is science-driven. This implies that substantial, on-going, active guidance by science subject matter experts is essential. Automation can greatly reduce the workload of the trained investigator, but cannot replace the investigator's expertise.

Interpretation of data mining results requires domain expertise. The development and validation team must include members with a broad range of technical and scientific skills appropriate to the problem. A selected mix of algorithm experts, domain scientists, and statisticians need to be involved, perhaps on an ad hoc basis in the development and validation phase.

Logistic Regression makes interpretation of results relatively easy. A significant consideration in the selection of logistic regression (given that predictive performance was not sacrificed) was the heuristic significance of the model produced by this algorithm. The parameters produced by the Logistic Regression model have a natural meaning which can be read, interpreted, and understood by a knowledgeable person without the need for significant additional transformation.

Correlated variables complicate interpretation of results. The logistic regression model takes a weighted sum of its input values to produce an output value, using weights derived from regression over a sample. If the sample has a pair of highly correlated variables, the resulting model will contain a pair of weights whose sum is well determined, but whose individual values are not. Therefore, analysis of the influences of individual variables is limited to those that are uncorrelated, and the remaining variables should be analyzed in groups.

15.6.2 Execution Efficiency

Data mining is feasible on large data volumes. The conceptual architecture for the IA-KBS envisions an ongoing algorithm development and validation process; and the testbed results confirm that this process is computationally manageable. Given scientific collaboration indicated above, the experience from the testbed suggests that the data mining development, validation, and extension of such algorithms is well within the state of the art and the computational resources of commercial off-the-shelf systems.

Mined models can be computationally efficient. Near real-time event detection (or prediction) is well within the current, modest computer system capabilities, based on the timings indicated in Sect. 15.5.1. This also suggests that content-based retrieval is very feasible.

Performance & flexibility of pre-processing is important. A flexible and powerful development environment is important because the preprocessing code can be much larger than the actual data mining code. Efficiency of the preprocessing code is important because this code must run not only in the model training phase (which can be performed against a sample and without significant time constraints), but also in the model execution phase (in near real-time).

ACKNOWLEDGMENT

The authors would like to thank the following individuals for their assistance and contributions of ideas to this work: G. McConaughy and C. Lynnes (NASA Goddard Space Flight Center-GSFC) – IA-KBS concepts, S. Morse (SOSACorp) – Intelligent Data Understanding (IDU) research assessment and testbed conceptual design, L. Di (GMU) – IA-KBS concepts, X. Li (GMU) – modeling and forecasting software execution on the testbed, T. Chu (IHMC) – assessment of software porting to testbed, and P. Smith (MacDonald, Dettwiler & Associates, Ltd.), B. Koenig (Electronic Data Systems Corporation - EDS) and C. Yee (Raytheon) – support for RDS/testbed interfaces. They would also like to thank C. Bock, D. Lowe and M. Esfandiari (Earth Science Data and Information System ESDIS Project, NASA GSFC) for their encouragement and support.

The data used in this study were obtained from several sources as indicated in Sect. 15.4.1. The sources include NOAA's National Climate Data Center, U.S. Forest Service, NASA's Ames Research Center, and three of NASA's EOSDIS Data Centers – Land Processes DAAC, National Snow and Ice Data Center and the Goddard Earth Science Data and Information Services Center.

REFERENCES

Bonnlander BV (2005) "Statistical Forecasts of Wildfire: A Baseline Approach"; (2005) CMU Laboratory for Symbolic Computation Technical Report #172

Burgan R, Klaver R, Klaver J (1998) Fuel models and fire potential from satellite and surface observation. International Journal of Wildland Fire 8:159-170

Clausen M, Lynnes C (July 2003) Virtual Data Products in an Intelligent Archive, White Paper prepared for the Intelligent Data Understanding program, http://daac.gsfc.nasa.gov/intelligent_archive/presentations.shtml

Hopley L, Van Schalkwyk J (March 3, 2007) "The magnificent ROC (Receiver Operating Characteristic curve)", http://anaesthetist.com/mnm/stats/roc/Findex.htm

Isaac D, Lynnes C (January 2003) Automated Data Quality Assessment in the Intelligent Archive, White Paper prepared for the Intelligent Data Understanding program, http://disc.gsfc.nasa.gov/IDA/

Isaac, D, McConaughy, G (September 2004) "Intelligent Archives in the Context of Knowledge Building Systems: Data Volume Considerations", White Paper prepared for the Intelligent Data Understanding program, http://disc.gsfc.nasa.gov/IDA/

Jolly W, Graham J, Michaelis A, Nemani R, Running S (2005) A flexible, integrated system for generating meteorological surfaces derived from point sources across multiple geographic scales, Environmental Modeling & Software, 20:873-882

Loveland T, Merchant J, Ohlen D, Brown J (1991) Development of a land-cover characteristics database for the conterminous U.S. Photogrammetric Engineering and Remote Sensing, 57:1453-1463

McConaughy G, McDonald K (September 2003) "Moving from Data and Information Systems to Knowledge Building Systems: Issues of Scale and Other Research Challenges," White Paper prepared for the Intelligent Data Understanding program, http://disc.gsfc.nasa.gov/IDA/

Morse S, Isaac D, Lynnes C (January 2003) "Optimizing Performance in Intelligent Archives," White Paper prepared for the Intelligent Data Understanding program, http://disc.gsfc.nasa.gov/IDA/

Morse S, Yang W, (October 2004) A Conceptual Specimen Architecture for an Intelligent Archive in a Knowledge Building System, White Paper prepared for the Intelligent Data Understanding program, http://disc.gsfc.nasa.gov/IDA/

NASA (February 3, 2005) "Evolution of EOSDIS Elements", Study Team Briefing to NASA, http://eosdis-evolution.gsfc.nasa.gov/

Ramapriyan H, McConaughy G, Morse S, Isaac D (August 2004) "Intelligent Systems Technologies to Assist in Utilization of Earth Observation Data," presented at Earth Observing Systems IX, SPIE Meeting, http://disc.gsfc.nasa.gov/IDA/

Ramapriyan H, Isaac D, Yang W, Morse S, (2005) "Large Scale Data Mining to Improve Usability of Data – an Intelligent Archive Testbed", IGARSS, Seoul, Korea

Ramapriyan H, Behnke J, Sofinowski E, Lowe D, Esfandiari M (2009) "Evolution of the Earth Observing System (EOS) Data and Information System (EOSDIS)", Chapter 7, Standards Based Data and Information Systems for Earth Observations, (Eds) Di L, Ramapriyan H, and Bai Y, Springer-Verlag,.

USDA Forest Service (1993) National Interagency Fire Management Integrated Database (NIFMID) reference manual; U.S. Department of Agriculture, Forest Service, Fire and Aviation Management. Washington, D.C., USA., p 43

Acronyms

ACCESS	Advancing Collaborative Connections for Earth System Science
AUC	Area under the ROC curve
AVHRR	Advanced Very High-Resolution Radiometer
CRS	Coordinate Rederence Systems
DAAC	Distributed Active Archive Center
DCE	Distributed Communication Environment
DDR	Double Data Rate
ECHO	EOS ClearingHOuse
EDC	EROS Data Center
EDS	Electronic Data Systems
EOS	Earth Observing System
EOSDIS	Earth Observing System Data and Information System
FPAR	Fractional Photosynthetically Active Radiation
FTP	File Transfer Protocol
GFLOPS	Giga (10**9) Floating Point Operations Per Second
GMU	George Mason University
GSFC	Goddard Space Flight Center
GSSD	Global Surface Summary of the Day
HDF	Hierarchical Data Format
IA	Intelligent Archive
IA-KBS	Intelligent Archive in the Context of a Knowledge Building System

IDU	Intelligent Data Understanding
IHMC	Institute for Human and Machine Cognition
LAI	Leaf Area Index
LP DAAC	Land Processes DAAC
LSDM	Large-Scale Data Mining
MCAT	Metadata Catalog
MDCE	MatLab Distributed Computing Engine
MEaSUREs	Making Earth Science Data Records for Use in Research Environments
MODIS	Moderate-Resolution Imaging Spectroradiometer
NASA	National Aeronautics and Space Administration
NCDC	National Climate Data Center
NDVI	Normalized Difference Vegetation Index
NOAA	National Oceanic and Atmospheric Administration
PRECIP	precipitation
RAID	Redundant Array of Independent Disk
REASoN	Research, Education and Applications Solutions Network
RDS	Remote Data Storage
ROC	Receiving Operating Characteristic Curve
SDRAM	Synchronous Dynamic Random Access Memory
SRB	Storage Resources Broker
TB	Terabyte
TCP/IP	Transmission Control Protocol/Internet Protocol
TMIM	Temperature, Minimum
TMAX	Temperature, Maximum
TOPS	Terrestrial Observation and Prediction System
U.S.	United States
USFS	United States Forest Service
VPD	Vapor Pressure Deficit