

# Environmental Scenario Search and Visualization

Mikhail Zhizhin  
Geophysical Center RAS  
3 Molodezhnaya str.  
117296 Moscow, Russia  
+7-495-9306115  
jjn@wpcb.ru

Eric Kihn  
NGDC NOAA  
325 Broadway E/GC2  
Boulder 80305 CO  
+1-303-4976346  
Eric.A.Kihn@noaa.gov

Vassily Lyutsarev  
Microsoft Research  
7 J J Thomson Avenue  
Cambridge CB3 0FB, UK  
+44 1223 479 700  
vassilyl@microsoft.com

Sergei Berezin  
Computer Mathematics Department  
MSU, Vorobiev Gory  
119899 Moscow, Russia  
+7-495-9393010  
s\_berezin@cs.msu.su

Alexey Poyda  
Geophysical Center RAS  
3 Molodezhnaya str.  
117296 Moscow, Russia  
+7-495-9306115  
poyda@wpcb.ru

Dmitry Mishin  
Geophysical Center RAS  
3 Molodezhnaya str.  
117296 Moscow, Russia  
+7-495-9306115  
dimm@wpcb.ru

Dmitry Medvedev  
Geophysical Center RAS  
3 Molodezhnaya str.  
117296 Moscow, Russia  
+7-495-9306115  
dmedv@wpcb.ru

Dmitry Voitsekhovskiy  
Computer Mathematics Department  
MSU, Vorobiev Gory  
119899 Moscow, Russia  
+7-495-9393010  
dvoits@gmail.com

## ABSTRACT

We have developed Environmental Scenario Search Engine (ESSE) for parallel data mining of a set of conditions inside distributed, very large databases from multiple environmental domains. The prime goal for ESSE design is to allow a user to query the environmental data archives in human linguistic terms. The mapping between the human language and the computer system involves fuzzy logic. Imagine for example that the end user doesn't need all of the weather data covering the Moscow region for the last 50 years, but rather needs an example of an atmospheric front near Moscow. Further imagine that this user needs satellite images of the front and he wants to know how often such fronts occur or if they have been increasing in the last 10 years. ESSE data mining and visualization portal combines metadata search, interactive fuzzy scenario query editor and event visualization system. Visualization system is implemented as MS Windows application based on NASA World Wind 3D globe and as a web application built upon OGC WMS and Microsoft Virtual Earth control.

## Categories and Subject Descriptors

H.2.8 [Database Applications]: Data mining, Image databases, Spatial databases and GIS

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

ACMGIS'07, November 7-9, 2007, Seattle, WA.

Copyright 2007 ACM ISBN 978-1-59593-914-2/07/11...\$5.00.

## General Terms

Algorithms, Management, Performance, Design, Experimentation.

## Keywords

Environment, scenario, search, fuzzy logic, spatio-temporal databases.

## 1. INTRODUCTION

We have developed Environmental Scenario Search Engine (ESSE) for parallel data mining of a set of conditions inside distributed, very large databases from multiple environmental domains.

Increasing data volumes from today's collection systems and the scientists' demand to include an integrated and authoritative representation of the natural environment in their analysis need new approaches to data mining, management and access. Recently the environmental modelling community has begun to develop several archives of continuous environmental representations. These archives take observational data and through modelling create a regular, parameterized view of the Earth system. The models use all available observation data as initial conditions for the numerical models, so the resulting data sets jointly may be considered as authoritative high-resolution representation of terrestrial weather and the near-Earth space during the last 50 years [1][2].

In this paper we use Data Resource Grid-service abstraction layer to virtualize spatio-temporal database providing time-series for our search engine. The Data Resource interface is implemented as an OGSA-DAI component [3] with a simple XML output schema.

Time-series selected from the Data Resource in XML format can be mined directly by the ESSE engine or transformed into another format with XSLT to be used by another client, for example in the MS Excel spreadsheet. We illustrate virtualization technique using NCEP/NCAR reanalysis data set [1] (Section 2).

The prime requirement of the ESSE system design is to allow the user to query the environmental data archives in human linguistic terms. The mapping between human language and computer systems involves fuzzy logic (Section 3). We use data resource grid-service abstraction layer to virtualize spatio-temporal databases providing time-series for our search engine. Results of the feasibility study were published in 2006 in the MSR Technical Report No. 1116 [4].

Major data types related to Earth sciences include time-varying vector and scalar arrays on grids and high resolution images received from satellite. In Section 4 we present a visualization system that is built as a part of the ESSE project.

Visualization system provides two applications – windows application based on NASA World Wind [5] and web applications that is built upon OGS WMS [6] and Microsoft Virtual Earth control [7]. Both applications allow researcher to show data arrays and images from various sources, including ESSE data resource web services and OPeNDAP servers [8].

All server-side components of the ESSE search engine are platform-independent and can be built either for .NET or Linux.

## 2. ENVIRONMENTAL DATA SOURCES

The first global weather reanalysis project was accomplished in late 90-s by the National Center for Environmental Protection (NCEP) and the National Center for Atmospheric Research (NCAR) [9][10]. It continues to be used with current data in near-real time, so that its products are available from 1948 to the present. The NCEP/NCAR 50-year reanalysis uses a frozen modern global data assimilation system which is identical to the global weather forecast system implemented operationally at NCEP on January 1995, except for the lower horizontal resolution (about 210 km) compared to the operationally used for weather global weather forecast (currently less than 100 km).

There are about 80 different variables, (including geopotential height, temperature, relative humidity, U and V wind components, etc.) available from 1/1/1948 till present with output every 6 hours in several different coordinate systems, such as 17 pressure level stack on 2.5x2.5 degree grids, 28 sigma level stack on 192x94 Gaussian grids, and 11 isentropic level stack on 2.5x2.5 degree grid. They are organized as different subgroups of the model output binary data files in the data archive.

### 2.1 NCEP/NCAR Reanalysis Database

For interactive data mining applications, we have converted the collection of binary files resulting from the NCEP/NCAR reanalysis project [10], into a cluster of relational databases. Before loading into the database, some of the model output data were interpolated to have the uniform spatial coordinate system based on 2.5x2.5 degree latitude/longitude grid.

In our initial design, the weather data had been stored on a cluster of 58 separate databases, each containing one year data [4]. Inside each database in the cluster the weather parameters were stored as time series in a separate binary field for each grid point. For

multiple years of data the databases were queried sequentially or in parallel threads from each server in the cluster at the application level.

The data for each parameter is stored in a separate table (Table 1). There are two kinds of data tables: for surface parameters and for pressure level parameters (Figure 1). Both have common fields: latitude (lat), longitude (lon) and binary data record (obs). In addition pressure level tables have pressure level field (lvl). Each binary record contains data for the year at a particular geographical point, defined by the “lat” and “lon” fields.

The database cluster was able to deliver 50 years long time series for a given location in interactive times, but data request for multi-point or rectangular regions lasted too long to be suitable for interactive data mining applications.

**Table 1. Reanalysis variables and database tables**

	Table	Description	Units
Surface	air0	Air temperature at sigma level 995	degK
	Lftx	Surface Lifted Index	degK
	rhum0	relative humidity at sigma level 995	%
	Slp	Sea Level Pressure	Pascals
	Icec	Ice Concentration at surface	Fraction
	soil0	Volumetric Soil Moisture between 0-10 cm Below Ground Level	Fraction
	soil10	Volumetric Soil Moisture between 10-200 cm Below Ground Level	Fraction
	Skt	SST/Land Skin Temperature	degK
	temp0	Temperature between 0-10 cm below ground level	degK
	temp10	Temperature between 10-200 cm below ground level	degK
	uwnd10	u-wind at 10 m	m/s
	vwnd10	v-wind at 10 m	m/s
	Prate	Precipitation Rate at surface	Kg/m <sup>2</sup> /s
	Tcdc	Total cloud cover	%
Pressure level	Air	Air temperature	degK
	Hgt	Geopotential height	m
	Rhum	relative humidity	%
	shum	specific humidity	kg/kg
	uwnd	U-wind	m/s
vwnd	V wind	m/s	

### 2.2 Optimization of Database Schema

The first optimization stage involved database tuning for our test data mining request with long time series and several grid points:

search over 50 years of data (see data mining optimization below). Transfer overheads are significant for large datasets, because disk transfer rates are comparatively small. To save time on disk reads, we have merged binary fields from all yearly databases and packed them with GZIP algorithm to produce a single database for the whole time period. Thus, the table structure of the database remains the same, only the representation of the binary data has changed in length (all years, compressed, instead of one year, uncompressed). Due to the nature of the data values distribution, GZIP managed to pack the data significantly. Packing the whole time range is more efficient than packing monthly or yearly time series. The database size has been reduced from 327 Gb to 116 Gb.

<pl_variable_name>			<sfc_variable_name>		
PK,I3,J2,I1	lat	SMALLINT	PK,I1	lat	SMALLINT
PK,I3,J2,I1	lon	SMALLINT	PK,I1	lon	SMALLINT
PK,I1	lvl	SMALLINT		obs	VARBINARY(MAX)
	obs	VARBINARY(MAX)			

PK (lvl, lat, lon)  
I1 (lvl, lon, lat)  
I2 (lat, lon)  
I3 (lon, lat)

PK (lat, lon)  
I1 (lon, lat)

A) Pressure level data table                      B) Surface data table

Figure 1. NCEP/NCAR reanalysis data tables structure.

The new database version showed much better performance, one grid point being fetched in approximately 0.015 seconds. Considering that GZIP decompression is much faster than disk data transfer (does not exceed 0.01 sec), we gain about 30% performance increase.

An MS SQL Server version of the database has been prepared, which showed performance results generally comparable with MySQL. A more detailed comparison of MySQL and MS SQL Server databases is given in Table 2. MS SQL Server database is slightly larger than MySQL database (127 Gb). This can be explained by the fact that SQL Server storage engine maps data to 8 Kb pages, while MyISAM engine doesn't use paging.

Table 2. Database access times compared.

Database	Parameter	Query execution time, s			
		1 point	9 points lat	9 points lon	3x3 points
MySQL (compressed continuous BLOBs)	air0	0.014	0.125	0.118	0.115
	hgt	0.016	0.133	0.133	0.134
MS SQL Server (compressed continuous BLOBs)	air0	0.032	0.063	0.061	0.058
	hgt	0.028	0.102	0.081	0.089
MySQL (uncompressed BLOBs by year)	air0	0.658	1.451	1.554	2.539
	hgt	0.636	>10	>10	>10

### 2.2.1 Web Service for Data Resource

We virtualize spatio-temporal data resource using OGSA-DAI framework [3]. The technology permits to aggregate heterogeneous environmental data sources with different database schemas into a data grid with common data model and unified data access interface. It is similar to the standard OGSA-DAI data resources for SQL and XML databases, but it has array-based output data model and a metadata schema suitable for environmental applications. Multi-layer architecture of the virtual data resource is given below in Figure 2.

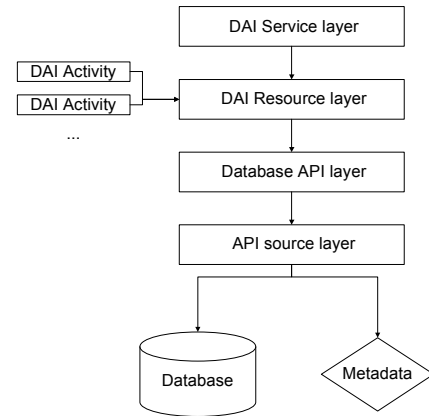


Figure 2. Multi-layer architecture of the virtual data resource.

The service level distributes the resources among the data services. The upper level allows the separation of the resources according to projects and data types. Different DAI data services can be installed and deleted independently, ensuring the flexible support of different sources.

The second abstraction layer is the resource layer. The resources distinguish data sources by means of access, but they have a common data request interface and a common output format. The activities supported by the resource are defined in resource configuration when it is added to a service.

One of the data source classes, managed by the system – are the databases of time series packed into BLOB arrays and containing measurements of weather parameters on a regular grid above the ground. The database structure may differ in BLOB packing methods and relational tables structure (representation of surface coordinates, time, and parameters). For such databases we designed a mechanism of overloaded API classes. Another function of an API class is to provide metadata for the supported database. The appropriate API class is chosen at the database API layer.

The lower level of data separation is the source. It is a unique identifier, which defines the API class, the supported database structure, the available databases, and the metadata. Different databases which share a common structure and a common API class are distinguished by this source identifier. Each database may have several metadata sets (for example, different units of measure), visible as distinct sources.

Sometimes a data processing activity requests multiple data streams from the same data resource, some of which may overlap. In this case data caching can minimize the amount of database

requests. It can improve the system performance significantly when providing data for components which perform data mining from a database and request overlapping data in different input streams. When requesting data on two slightly shifted time intervals the performance gain is about 50%.

### 3. FUZZY SCENARIO SEARCH FOR ENVIRONMENTAL EVENTS

In this section we present algorithms and software toolbox for the parallel data mining for a set of conditions inside distributed very large databases from multiple environmental domains. The software toolbox is called Environmental Scenario Search Engine (ESSE). The prime requirement of the ESSE system design is to allow the user to query the environmental data archives in human linguistic terms. The mapping between human language and computer systems involves fuzzy logic.

#### 3.1 Methods

The base data model in our study is vector-valued time-series

$$\mathbf{X} = \{\mathbf{x}(t_i)\}, i = 1 \dots N, \mathbf{x}(t_i) = (x_1(t_i), \dots, x_M(t_i)),$$

where  $N$  is the number of time samples, and  $M$  is the number of observed parameters. It can be represented as a trajectory in the  $M$ -dimensional phase space  $\mathbf{R}^M$ . For example, in Figure 1 we have two-dimensional trajectory in the pressure-temperature (P-T) space. A (fuzzy) state  $S$  in a phase space  $\mathbf{R}^M$  is a fuzzy set which can be described by fuzzy logic expression composed of predicates describing in numerical or linguistic terms the parameter values in each of  $M$  dimensions. For example, the state  $S_1$  corresponding to the “red” region in Figure 1 can be described by the fuzzy expression:

$$S_1 = (\text{Very Large}(P)) \text{AND} (\text{Very Large}(T)),$$

where the linguistic term  $\text{Very Large}()$  is a predicate, and the operator AND stands for the fuzzy logic conjunction. In the same way, the state  $S_2$  corresponding to the “blue” region is

$$S_2 = (\text{Very Small}(P)) \text{AND} (\text{Very Small}(T)),$$

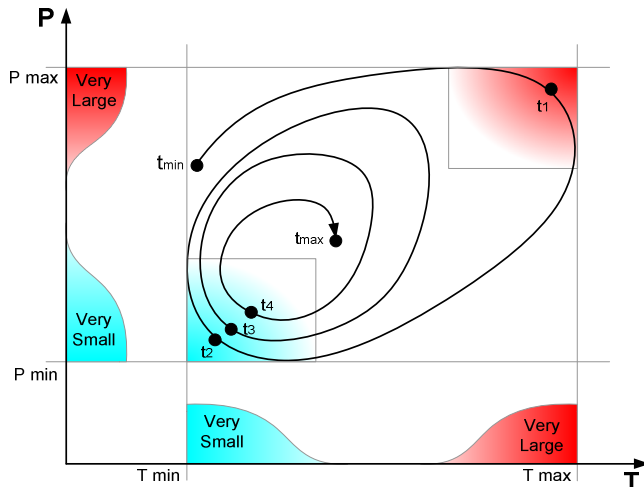


Figure 3. Time-series as a trajectory in the two-dimensional phase space (P – pressure, T – temperature).

Now, combining the descriptions of the states with the “time shift” operator  $\text{Shift}(dT, \cdot)$  to describe transitions between the states, we can write the following symbolic expression for the environmental scenario “very low temperature and pressure after very high temperature and pressure”:

$$\text{Scenario} = (\text{Shift}(dT=1, S_1)) \text{AND} (S_2).$$

The only pair of observations in Figure 3 which fit the above scenario is the pair  $(t_1, t_2)$ . Our environmental scenario search engine, ESSE, is designed to mine for the phase space transitions like that in very large scientific databases.

#### 3.1.1 Fuzzy environment states

Fuzzy environment states in the multidimensional spatio-temporal data sources are specified as logical expressions involving operators {AND, OR, NOT} applied to a set of linguistic {Very Large, Large, Average, Small, Very Small} and numeric {Less Than, Equal, Greater Than, Between} predicates formed for each variable (dimension) using the simple trapezoid or more smooth bell function

$$\mu_{gbell}(\tilde{x}; a, b, c) = \frac{1}{1 + \left| \frac{\tilde{x} - c}{a} \right|^{2b}}$$

Here,  $\tilde{x}$  stands for normalized for range [0,1] scalar data variable,  $c$  stands for center of the symmetrical “bell”,  $a$  for its half-width, and  $b/2a$  controls its slope. We use here simple range normalization  $\tilde{x} = (x - x_{\min}) / (x_{\max} - x_{\min})$  from the climatology analysis of the variable  $x_{\min} \leq x \leq x_{\max}$ . The normalization limits are set to the minimum and maximum parameter values observed within the continuous or seasonal intervals given by the time constraints of the fuzzy search. Fuzzy membership functions (MFs) for a linguistic term set are plotted in Figure 4.

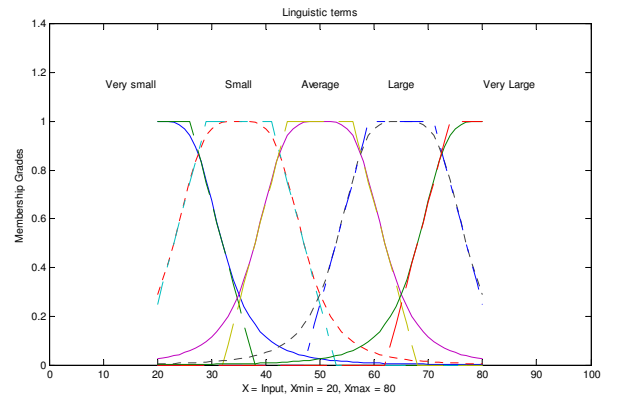


Figure 4. Trapezoid and bell membership functions for linguistic fuzzy term set.

One of the simplest generalizations from classical to fuzzy set theory for two one-dimensional MFs  $\mu_A(x), \mu_B(y)$  is to use minimum of MFs  $\min(\mu_A, \mu_B)$  for the intersection of fuzzy sets (fuzzy logic AND), maximum of MFs  $\max(\mu_A, \mu_B)$  for fuzzy

sets union (fuzzy logic OR), and  $\mu_{\bar{A}} = 1 - \mu_A$  for fuzzy set complement (fuzzy logic NOT).

In 1980 R. Yager introduced a parametric family of T-norms, T-conorms and fuzzy complements [11][12]. Parameterized by  $q$  a family of fuzzy AND aggregations for two one-dimensional MFs  $\mu_A(x), \mu_B(y)$  is defined by Yager's T-norm operator:

$$T_Y(\mu_A(x), \mu_B(y), q) = 1 - \min \left\{ 1, \left[ (1 - \mu_A(x))^q + (1 - \mu_B(y))^q \right]^{\frac{1}{q}} \right\} \quad q \geq 1$$

More general formula for the parametric Yager's T-norm operator for fuzzy AND aggregation of any  $M > 1$  one-dimensional MFs  $\mu_n(x), n = 1 \dots M$  is

$$T_Y(\mu_n(x), q) = 1 - \min \left\{ 1, \left[ \sum_{n=1}^M (1 - \mu_n(x))^q \right]^{\frac{1}{q}} \right\}, \quad q \geq 1$$

The resulting surface of values for the multi-dimensional MF is more smooth than using a simple minimum of the aggregating MFs, which is the limit case of Yager's T-norm for  $q=1$ .

Parametric family of fuzzy OR aggregations for any  $M > 1$  one-dimensional MFs is described by Yager's T-conorm operator

$$S_Y(\mu_n(x), q) = \min \left\{ 1, \left[ \sum_{n=1}^M (\mu_n(x))^q \right]^{\frac{1}{q}} \right\}, \quad q \geq 1$$

Yager's fuzzy complements are defined by formula

$$N_Y(\mu_n(x), q) = (1 - \mu_n(x))^{\frac{1}{q}}, \quad q > 0$$

The ESSE search engine is designed to support different libraries of T-norm, T-conorm and complement operators. Below we will compare search time when using the Yager's formulas with the orders  $q=1$  (min-max aggregation) and  $q=5$  (Yager aggregation).

### 3.1.2 Fuzzy event scenario

In applications we search for events in the changing environment, where the input variables and the one-dimensional MFs-predicates  $\mu_n(x(t))$  depend on time, as well as the fuzzy expressions  $E_Y(\mu_n(x(t)), q)$ , which are composed of T-norms, T-conorms and complements over the predicates  $\mu_n(x(t))$ . We consider the values of the resulting time series  $E_Y(\mu_n(x(t)), q)$  as the "likeliness" of the environmental state to occur at the time moment  $t$ , or, in other terms, to visit a subregion of the phase space described by the fuzzy expression (see Figure 1). We search for the highest values of the  $E_Y(\mu_n(x(t)), q)$  and consider these to be the most likely candidates for the environmental event.

To be able to search for events like a "hot day" or a "hot week" we introduce the concept of *event duration*  $k\Delta t$  which may be any multiple  $k = 1, 2, \dots$  of the time step  $\Delta t$  of the input. For example, the time step in the NCEP/NCAR reanalysis is  $\Delta t = 6$  hours, so the minimum event duration is also 6 hours, but the event duration may be also 1 day, 1 week, etc. We do a moving average of the input parameters with the time window of the event

duration before calculation of the one-dimensional MFs in the fuzzy expression:

$$\bar{x}(t_i) = \frac{1}{k} \sum_{j=i}^{i+k-1} x(t_j), \quad t_i = t_0 + i\Delta t.$$

When searching for a "hot day" in the NCEP/NCAR reanalysis, first we have to smooth the air temperature with the time window of 1 day ( $k = 4$ ), then calculate the linguistic predicate "low"  $\mu_{low}(\bar{x}(t_i))$ , sort the fuzzy scores in descending order, and finally take the several first times with the highest scores as the candidate events.

We need the time shift operator to define multiple-state event scenario:

$$\text{shift}(\mu(x(t_i)), p) = \mu(x(t_{i-p})), \quad p \in N.$$

For any fuzzy logic expression  $E$  we have:

$$\text{shift}(\text{shift}(\mu, p), q) = \text{shift}(\mu, p + q),$$

$$\text{shift}(E(\mu_1, \mu_2, \dots), p) = E(\text{shift}(\mu_1, p), \text{shift}(\mu_2, p), \dots)$$

For example, to find an abrupt air pressure drop, we can use a two-state scenario with fuzzy AND of the "very large" and time-shifted "very low" predicates for pressure  $P(t)$  (see Figure 9):

$$S(t) = T(\mu_{\text{very large}}(P(t)), \text{shift}(\mu_{\text{very low}}(P(t)), 1), q).$$

Following this example, a two-state scenario with the fuzzy expressions for states  $E_1(\mu(x(t)))$  and  $E_2(\mu'(x(t)))$  with the time delay between the steps  $p\Delta t$  can be defined as a Yager T-norm conjunction of the time-shifted expressions

$$S(t) = T(E_1(\mu(x(t))), \text{shift}(E_2(\mu'(x(t))), p), q).$$

Generalization of the formula for more than two states is straightforward.

To have the result of the fuzzy search in the form of a ranked list of the  $K$ -most likely dates (times) of the events, we sort the scenario MF  $S(t)$  and select the times of the  $K$  maximum values of the aggregated MF  $\{t_{i_1}, t_{i_2}, \dots, t_{i_K}\}$  separated in time by the intervals longer than the event duration:  $t_{i_{k+1}} - t_{i_k} > k\Delta t$ .

### 3.1.3 Importance of the input parameters

The fuzzy search request may contain conditions which never or very rarely take place at the same time at the specified location, although they can be observed there separately at different time moments. For example, very high precipitation rate and very high air pressure are unlikely to occur simultaneously. The fuzzy search for such a combination of conditions may return an empty set of candidate dates and times. We decrease the probability of the empty fuzzy search results by introducing the concept of *importance* of the input parameters. The importance  $w_n$  is a constant weight of a given parameter in the range between 0 and 1. More important parameters are given higher weight, with the condition that the highest priority is then normalized to one. Then instead of the one-dimensional MFs  $\mu_n(x(t_i))$  in the fuzzy expressions we use "optimistic" values  $\max[\mu_n(x(t_i)), 1 - w_n]$ .

For parameters with the importance 1 we use the original MFs as before, and the parameters with the importance 0 are not used in the search at all.

### 3.2 Search time optimization

During the optimization of the whole system the performance analysis of the fuzzy search components has been carried out. For performance analysis a test request has been composed: “find an atmospheric front near Moscow in the last 50 years”. The formal description of this scenario follows:

$$\begin{aligned}
 & \left[ \text{Small}(P_{\text{point}1}) \text{AND Large}(\text{shift}_{1d} P_{\text{point}1}) \right] \\
 & \text{AND} \\
 & \left[ \text{Large}(v\text{Wind}_{\text{point}1}) \text{AND Small}(\text{shift}_{1d} v\text{Wind}_{\text{point}1}) \right] \\
 & \text{AND} \\
 & \left[ \text{ANY}(u\text{Wind}_{\text{point}1}) \text{AND Small}(\text{shift}_{1d} u\text{Wind}_{\text{point}1}) \right] \\
 & \dots \\
 & \left[ \text{Small}(P_{\text{point}9}) \text{AND Large}(\text{shift}_{1d} P_{\text{point}9}) \right] \\
 & \text{AND} \\
 & \left[ \text{Large}(v\text{Wind}_{\text{point}9}) \text{AND Small}(\text{shift}_{1d} v\text{Wind}_{\text{point}9}) \right] \\
 & \text{AND} \\
 & \left[ \text{ANY}(u\text{Wind}_{\text{point}9}) \text{AND Small}(\text{shift}_{1d} u\text{Wind}_{\text{point}9}) \right]
 \end{aligned}$$

The timing diagram of the fuzzy search execution before optimization for the given scenario over 9 spatial points is given in Figure 5. In Table 3 we give details on the diagram segments.

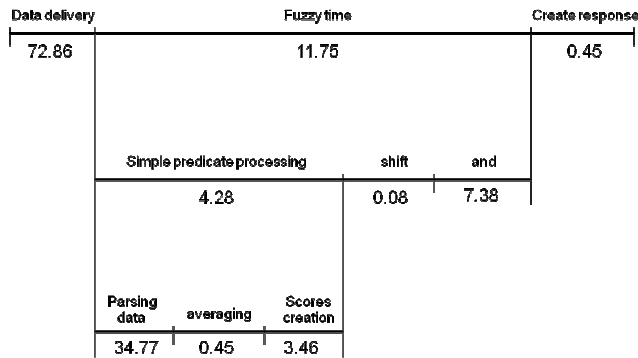


Figure 5. Time diagram of the search system before optimization.

In the first version of the search engine most of the execution time was spent on the XML input data transfer and parsing. We decided to switch to binary data transfer using data objects similar to OPeNDAP streams. It greatly reduced the data delivery time from 72 s to 2 s.

The timing diagram in Figure 5 shows that the next bottleneck is the evaluation of AND and OR fuzzy logic expressions. The evaluation complexity depends on the T-norm functions library.

During the performance test we used bell MFs and Yager functions with  $q=5$ . There are two ways to reduce computation complexity: to simplify the functions or to tabulate the existing functions library. We have tried both approaches: 1) we have created a simple version of the fuzzy functions library based on minimax T-norm evaluation and trapezoid MFs; 2) we have tried the quantization of the Yager fuzzy membership functions.

Table 3. Search timing diagram steps.

Data delivery	Time of the raw data delivery from the data source to the search engine
Fuzzy time	Total time of scenario search
Create response	Time of XML-response construction
Simple predicate processing	Total time of the evaluation of one-dimensional predicates
Shift	Total time of the SHIFT operator evaluation
And	Total time of the AND and OR operators evaluation
Parsing data	Time needed to transform the data into internal representation
Averaging	Total averaging time
Scores creation	Total time of the evaluation of one-dimensional predicates, excluding averaging and data transform

#### 3.2.1 Quantization of fuzzy membership values

The most time consuming operations in these formulas are calculations of powers  $p$  and  $1/p$ , where  $p$  is a parameter of the Yager operator family, which we consider to be fixed after initialization of the function library. We know that values of the fuzzy operands  $\mu_A(x), \mu_B(y)$  are in the range of  $[0, 1]$ . Thus we can use a lookup table to accelerate calculation of the powers in the Yager functions.

We have divided the unit segment  $[0,1]$  into  $2^k - 1$  equal steps. At each step we pre-calculate the values of  $x^p$  and  $x^{1/p}$ , and store them in a table. Later, when we need to calculate the T-norm or T-conorm functions, we simply select them as the nearest tabular value. The question is how to choose the number of quantization steps  $k$  to have both a good approximation and a short calculation time.

To see how the approximation error depends on the quantization step, we need to define how to estimate the discrepancy between the fuzzy search results obtained with and without quantization of the Yager functions. As a quantitative measure of the difference between the time series of the fuzzy search scores we can use a root mean square error *RMSE*. To measure the change (overlap) in a ranked list of the  $K$ -most likely date-times of the events found by the fuzzy search, we have introduced a special norm *DIFF*.

*RMSE* (root mean square error) is defined as

$$RMSE = \sqrt{\frac{\sum_{i=1}^N (x(t_i) - y(t_i))^2}{N}},$$

where  $N$  is the total number of the fuzzy search scores. To take in account that only the first  $size$  candidates are considered from a potentially very long  $N$  fuzzy search scores list, we had to modify this well-know formula for  $RMSE_{size}$  as follows:

$$RMSE_{size}(X, Y) = \sqrt{\frac{\sum_{i=1, size} (x(t_i) - y(t_i))^2}{size}}, \quad t_i \in TimesX$$

$$RMSE_{size}(Y, X) = \sqrt{\frac{\sum_{i=1, size} (y(t_i) - x(t_i))^2}{size}}, \quad t_i \in TimesY$$

$$RMSE_{size} = \frac{(RMSE_{size}(X, Y) + RMSE_{size}(Y, X))}{2}$$

Here  $TimesX$  is the set of the first  $size$  most likely candidates for the search result  $X$ , and  $TimesY$  is the same  $size$  subset for the search result  $Y$ . We have to symmetrize  $RMSE_{size}$  because in general  $RMSE_{size}(X, Y) \neq RMSE_{size}(Y, X)$ .

The above norm  $RMSE_{size}$  measures the difference between the multiple score values from the two candidate lists. But in many search cases the order of the selected candidates is more important than their scores: like in the social choice, we don't care about absolute score values, but we want to have the candidate order unchanged. To measure the stability of the order in the candidates list, we introduce another norm  $DIFF$ :

$$D_{X \cap Y} = \frac{\sum_{i=0, size-1} (f(i) - f(j))^2}{size}, \quad \text{if } i \in TimesX \cap TimesY$$

$$D_{X / Y} = \frac{\sum_{i=0, size-1} (f(i) - f(const))^2}{size}, \quad \text{if } i \in TimesX / TimesY$$

$$D_{Y / X} = \frac{\sum_{j=0, size-1} (f(j) - f(const))^2}{size}, \quad \text{if } j \in TimesY / TimesX$$

$$Diff = D_{X \cap Y} + D_{X / Y} + D_{Y / X}$$

$$f(x) = arctg(x)$$

Here  $i$  and  $j$  are the ranks of the same candidate in the top  $size$  subsets of the two fuzzy search results  $X$  and  $Y$ ,  $TimesX$  and  $TimesY$  are the same as for the  $RMSE_{size}$  norm, and the "logistic" function  $f$  is used to enhance the importance of the rank difference for the high-ranked candidates.

We have measured the quantization error on the same test query for the atmospheric front search over 9 grid points nears Moscow for the 50 years time period. In Figures 6 and 7 we compare norms  $RMSE_{size}$  and  $Diff$  for the search results with and without quantization for different number of quantization steps  $k$  for the first 100 event candidates list.

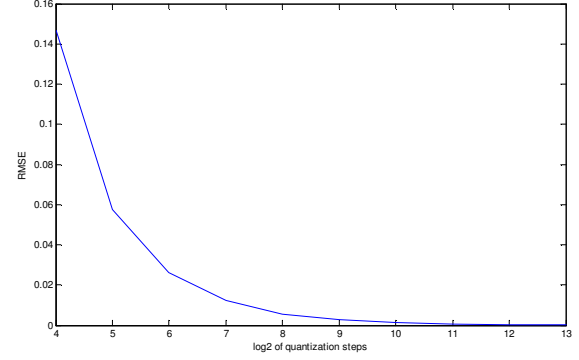


Figure 6.  $RMSE$  of the test search results as a function of the number of quantization steps for the first 100 events list

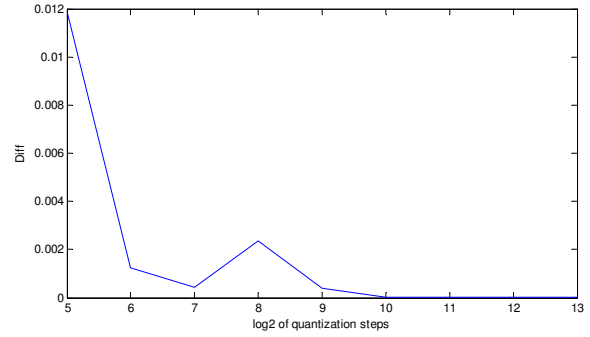


Figure 7.  $Diff$  of the test search results as a function of the number of quantization steps for the first 100 events subset

As it was expected, the search results difference vanishes when the number of quantization steps  $k$  tends to infinity. When the search results difference was compared to the time required for calculations, we have chosen the "optimal" number of steps  $k = 10$ . Comparison of the timing diagram below (Figure 8) with the one from the previous section shows three-fold fuzzy search time increase when the quantization of the fuzzy membership functions is used.

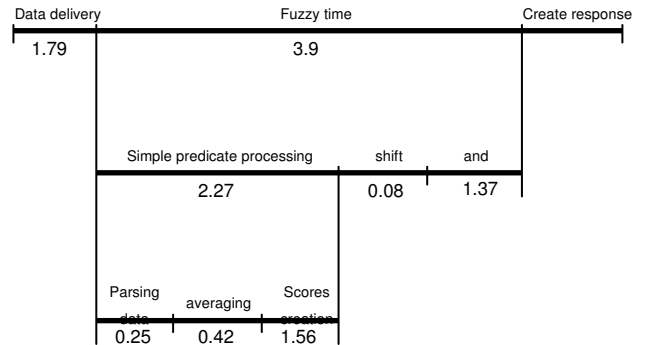


Figure 8. Time diagram for the test search after quantization of fuzzy membership functions

### 3.2.2 *Minimax norms and trapezoid membership functions*

Another way to accelerate the fuzzy search engine is to change the bell functions family used in fuzzy predicates and the Yager norms family in the fuzzy expressions for another, less computationally-expensive, function families. Such an easy-to-calculate functions can be trapezoid membership functions minimax T-norms and -conorms. For our test search there is no time gain when compared to timing diagram of the bell/quantized Yager functions library. However, in the current version of the fuzzy search engine we have implemented all types of fuzzy functions, including bell and trapezoid membership and Yager and minimax norms.

### 3.3 Fuzzy search and statistics of extreme weather events

In February 2007 the major results from the new Fourth Assessment Report (FAR) of the IPCC on climate change have been released. The Policymaker's Summary of the FAR's Physical Science Basis [13] is a collection of statements regarding climate change observations and forecast, which are formulated in following fuzzy terms upon agreement of the board of scientific experts. To quote some of the main statements in the summary:

“At continental, regional, and ocean basin scales, numerous long-term changes in climate have been observed. These include changes in Arctic temperatures and ice, widespread changes in precipitation amounts, ocean salinity, wind patterns and aspects of extreme weather including droughts, heavy precipitation, heat waves and the intensity of tropical cyclones (including hurricanes and typhoons).”

“It is very likely that hot extremes, heat waves, and heavy precipitation events will continue to become more frequent.”

Combination of the ESSE weather reanalysis data sources with the fuzzy search engine becomes a powerful tool for quantitative formalization and rapid validation of such kind of fuzzy statements. Below we present a use case for distribution of temperature extremes in space and time for the last three decades 1975-2006.

First, we have to formalize the weather “hot extreme”. For this demonstration, we use a very simple fuzzy expression “very high temperature”. Then we select from the NCEP/NCAR reanalysis database air temperature time series for the time period 1976 - 2005 with 6 hours time step on the grid with 5 degrees step in latitude and longitude. Fuzzy engine search for the “very high temperatures” over this data set returns 10 hot extreme weather events at each 5 degrees grid point, including event times and fuzzy scores. Finally we sort the events occurrence time into three decadal bins: 1976-1985, 1986-1995 and 1996-2005 (Appendix 10.6). It takes about 1.5 hour to run the search in a loop for each grid point over the 5 degree grid.

To visualize the global decadal distribution of the “hot extremes”, we implement the following algorithm. The grid point color is selected based on the decadal bin with the maximum number of the weather extremes: blue (getting colder) for the decade 1976-1985, yellow (neutral) for the 1986-1995, and red (getting hot) for

the decade 1996-2005. The size of the grid point depends on the “certainty” of the weather extremes distribution: the maximum size grid point is plotted for the locations where 9-10 from 10 events have occurred in the same decade, medium size grid point is plotted for the locations where 7-8 events have occurred in the same decade, and minimum size grid point is used to indicate 4-6 events occurrences (Figure 9).

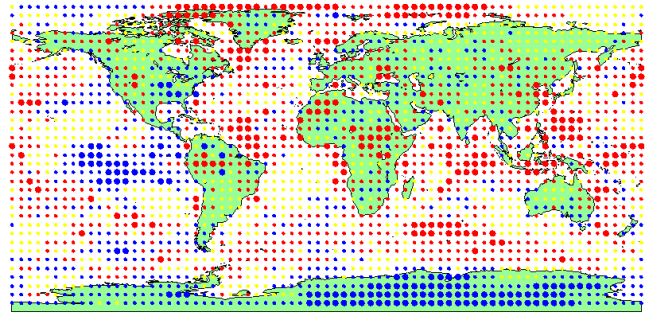


Figure 9. Distribution of 10 hottest days by decade. Blue circles for decade 1976-1985, yellow circles for 1986-1995, and red for 1996-2005.

## 4. VISUALIZATION OF ENVIRONMENTAL EVENTS

We need to interactively render environmental event data from various sources over the surface of the Earth. Data falls in one of two similar classes: gridded data arrays and geolocated images. Gridded data are a scalar or vector two-dimensional arrays of points usually (but not always) aligned with meridians and parallels with a constant step for latitude and longitude; each dimension having hundreds of points. Geolocated image is a high resolution image (thousands by thousands pixels) with mapping to latitude and longitude defined for each pixel. Currently we use only one type of geolocated images – DMSP satellite visual and infrared images.

The ESSE visualization engine is easily extensible not only for the new data sources, but also for the new types of visualization. At the moment our module supports the following visualization types: color maps, isolines and vector fields for gridded data rendering and texturing of the Earth surface with geolocated images possibly with transparency effects.

### 4.1 Visualization engine

In terms of the client-server architecture, the visualization engine can be deployed either on a server or on a client side (thin or thick clients respectively), providing different capabilities to display the data and different hardware requirements for the client workstation. The system architecture is split into three modules (Figure 10). Two of them, namely VisualEsse.DataEngine and VisualEsse.Visualization, are shared by all visualization clients. The VisualEsse.Renderer module is client-dependent.

The VisualEsse.DataEngine module provides data upon request in the same way irrespective of the data source type. Thus a new data source (NetCDF, WCS, etc.) can be added simply by extending this module with a new data provider. To increase the efficiency of the data request processing, the VisualEsse.DataEngine is charged to cache the provided data.

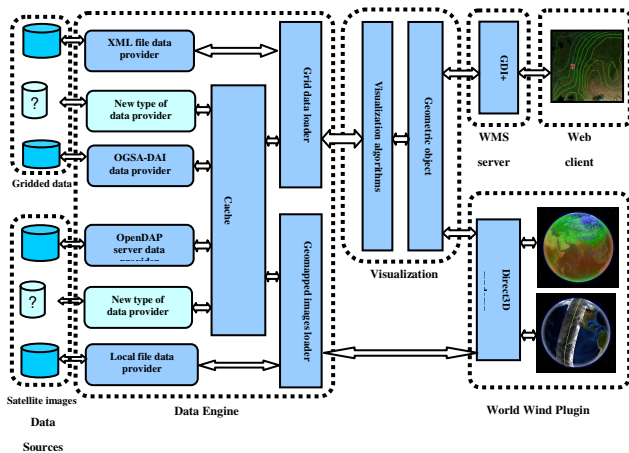


Figure 10. Visualization engine architecture.

The VisualEsse.Visualization module is responsible for applying of visualization algorithms to the data, at this stage being no matter how the data was retrieved. All algorithms implement the same interface describing a single method to convert certain abstract data model into a set of abstract geometric objects. An implementation of the geometric objects depends on the computer graphics API (DirectX, OpenGL or GDI+). Therefore the VisualEsse.Visualization module can be easily extended with new visualization types and algorithms as well as with support for a new computer graphics API.

#### 4.2 NASA World Wind plugin

The VisualEsse.Visualization module is used as a part of the World Wind plugin to visualize the data, therefore all its functionality is inherited by the plugin. Thus it is possible to build animated transparent color maps and isolines upon the 3D Earth surface. Additionally, a user can select different palettes for such visualization. Besides, VisualEsse.Plugin allows mapping of DMSP OLS images to the Earth surface.

To be more efficient and interactive, the World Wind plugin renders data only over a part of the Earth surface which is visible for the user. If the visible part of the Earth is changed, the new data request is dispatched in background allowing the user to continue to work with the World Wind application while the new data are loading. To increase the data processing efficiency, a data request caching is used for networked data sources.

In our plugin we have implemented progressive visualization, where the data are requested on a sparse grid first which to show a preview picture to the user as quickly as possible. At the same time a background request for full-resolution data is performed. Progressive visualization currently is used only for visualization of geolocated images; it is not used for gridded data sources because even for the full Earth coverage the grid size is relatively small for currently available data sources.

In Figure 11 we give example of a visualization mash-up for data from two different sources: ESSE OGSA-DAI resource for the wind speed and OPeNDAP service for DMSP OLS orbit. Our plugin can synchronize time for different data layers, in that case it is the only 2006 hurricane Ernesto.

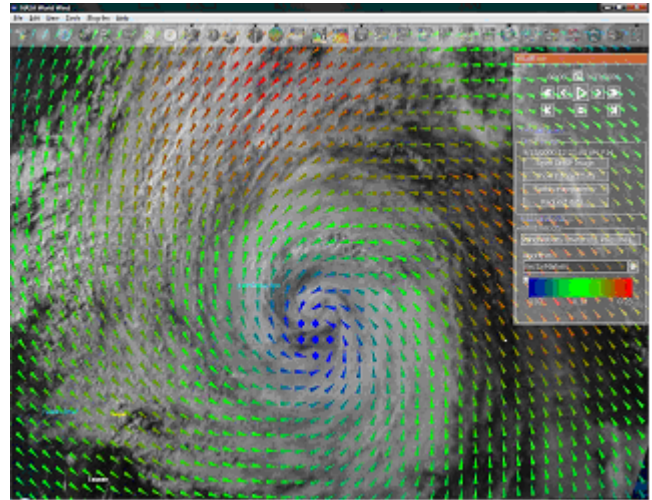


Figure 11. NASA World Wind VisualESSE plugin screenshot.

#### 4.3 OpenGIS server and thin client

We have implemented the “thin” client approach as a web application based on Microsoft Virtual Earth service. In this case the data retrieval and visualization tasks are performed by a special web map server, compliant with the OpenGIS specifications [6]. The resulting images are displayed in a regular web browser. Interaction with the web application requires JavaScript interpreter on the browser side.

The thin client can visualize environmental data available from ESSE sources over road maps, satellite images or mixed images of the Earth available from the Microsoft Virtual Earth. We took the Microsoft Virtual Earth control as a base for building the thin client. The Virtual Earth control renders the Earth images in background and they are combined with the data visualization images from our WMS server. Currently we support the data visualization with color maps and isolines and time navigation controls.

Microsoft Virtual Earth SDK [7] can display vector graphics containing segments and polygons over combined road maps and the Earth surface images. It provides a special event mechanism activated when a visible part of the map is changing. It works well with Microsoft Internet Explorer and Firefox browsers, but it has some problems with vector graphics in Opera browser.

In order to move all complex code away from the web browser’s JavaScript interpreter, we have implemented the following scheme. When the Virtual Earth or time control is used or a visible part of the map is changing, a special JavaScript handler is invoked. It retrieves a new visible part of the map using the Microsoft Virtual Earth API and requests new data visualization from our special WMS server. The WMS server in response generates an image that renders the requested environmental parameter using a user specified or default visualization type. The WMS server can return either a vector image in SVG format or a raster image with transparency in GIF or PNG format. The resulting WMS image is added to the Virtual Earth control as additional layer (Fig. 32). We use Prototype [JS] library to perform AJAX requests for asynchronous processing of isolines in SVG format.

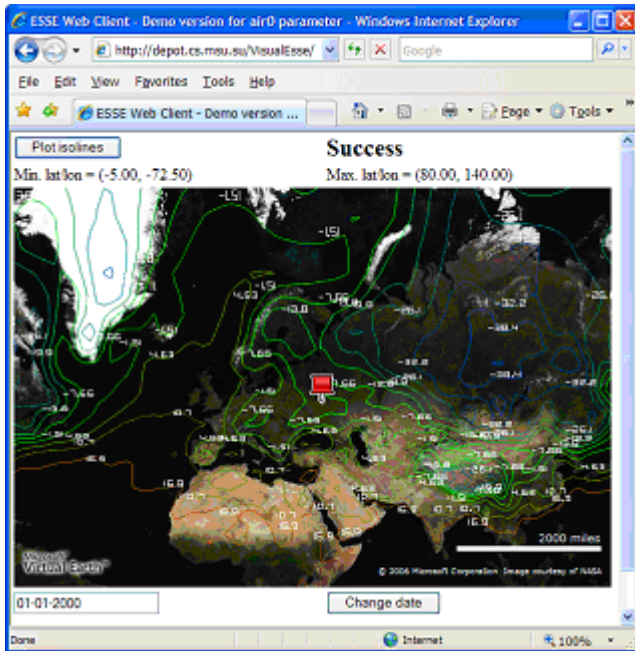


Figure 32. Web client sample screenshot.

Our WMS server implementation is based upon the Data Engine with data requests cache and Visualization modules. The data are transformed within the Data Engine to the common model *GeometricObject* using specified visualization algorithm. Then the *GeometricObject* is rendered to a raster image or to vector SVG image inside the Visualization module using GDI+ API.

## 5. CONCLUSIONS AND FUTURE PLANS

The idea of mining numeric archives using fuzzy terms has numerous potential applications both in environmental sciences and beyond. But in order to enable its routine use by wide communities we need to purify abstractions presented to a user, enhance considerably the performance and make it compatible with existing practices of data stewardship and application development.

Current implementation of our search engine uses fuzzy AND over discrete time shifts of the membership scores of each state in multiple states scenario. For example, if we search for a calm and warm week in May followed by a cold and rainy one, we will calculate fuzzy scores separately for each week, then time shift the second scores one week back, and calculate for each date in May the total score as (SCORE(warm week)) AND (LEFT\_SHIFT (SCORE(cold week))).

This approach is computationally efficient, but it is not generic enough to be used to search for states with fuzzy time interval boundaries: we can use time in the same way as all other parameters, providing fuzzy intervals for duration of each scenario state. With such generalization, we can consider the scenario state as fuzzy minimum rectangular boundary in the parameters + time coordinate system.

The ESSE engine is capable of mining distributed data located in different parts of the Internet. Remote database servers could be used for data collection and computing virtual parameters. Even

parts of ESSE expressions could be evaluated remotely and results transmitted to root node for final processing. The way in which a query is distributed to data sources affects dramatically CPU and network loads. Today this distribution is done manually by the client application. We will research algorithms to automatically distribute fuzzy search over available network to optimize the whole query performance.

ESSE software tools including data source virtualization API, fuzzy search engine, and VisualESSE clients are open-source under BSD licence. Platform-independent Java libraries are hosted by SourceForge ([esse.sf.net](http://esse.sf.net)). MS Windows version of the ESSE engine and the VisualESSE are hosted by CodePlex.

## 6. REFERENCES

- [1] Kalnay, E., et al. The NCEP/NCAR 40-year reanalysis project. *Bull Am. Meteorol. Soc.*, 77, 1996, 437-471; Web site: <http://www.cdc.noaa.gov/cdc/reanalysis/>
- [2] ECMWF Re-Analysis ERA-40: <http://www.ecmwf.int/research/era/>
- [3] M. Antonioletti, M.P. Atkinson, R. Baxter, A. Borley, N.P. Chue Hong, B. Collins, N. Hardman, A. Hume, A. Knox, M. Jackson, A. Krause, S. Laws, J. Magowan, N.W. Paton, D. Pearson, T. Sugden, P. Watson, and M. Westhead. The Design and Implementation of Grid Database Services in OGSA-DAI. *Concurrency and Computation: Practice and Experience*, Volume 17, Issue 2-4, Pages 357-376, February 2005; K. Karasavvas, M. Antonioletti, M.P. Atkinson, N.P. Chue Hong, T. Sugden, A.C. Hume, M. Jackson, A. Krause, C. Palansuriya. Introduction to OGSA-DAI Services. *Lecture Notes in Computer Science*, Volume 3458, Pages 1-12, May 2005; Web site: <http://www.ogsadai.org.uk/>
- [4] M. Zhizhin, A. Poyda, D. Mishin, D. Medvedev, E. Kihn, V. Lutsarev. Scenario Search on the Grid of Environmental Data Sources. MSR Technical Report, July 2006. Online at <http://research.microsoft.com/research/pubs/view.aspx?type=Technical%20Report&i d=1116>
- [5] NASA WorldWind viewer web site <http://worldwind.arc.nasa.gov/>
- [6] OpenGIS standards and specifications. Online at <http://www.opengeospatial.org/standards>
- [7] Microsoft Virtual Earth SDK web site <http://dev.live.com/virtualearth/sdk/>
- [8] OpenDAP network array-oriented data access protocol: <http://www.opendap.org/>
- [9] Kalney, E. Atmospheric modeling, data assimilation and predictability. Cambridge University Press, 2002
- [10] Kalnay, E., and Coauthors. The NCEP/NCAR 40-Year Reanalysis Project. *Bull. Amer. Meteor. Soc.*, 77, 1996, pp. 437-471
- [11] Yager, R. On a general class of fuzzy connectives. *Fuzzy Sets and Systems*, 4, 1980, 235-242
- [12] Climate Change 2007: The Physical Science Basis. Summary for Policymakers. Online <http://www.ipcc.ch/SPM2feb07.pdf>