A Parallel Processing of Spatial Data Interpolation on Computing Cloud

Vladimír Siládi
Matej Bel University
Tajovského 40
Banská Bystrica, Slovak Republic
vladimir.siladi@umb.sk

Ladislav Huraj
University of SS. Cyril and Methodius in Trnava
Nám. J. Herdu 2,
Trnava, Slovak Republic
ladislav.huraj@ucm.sk

Eduard Vesel
Matej Bel University
Tajovského 40
Banská Bystrica, Slovak Republic
edd.ves@gmail.com

Norbert Polčák
Matej Bel University
Tajovského 40
Banská Bystrica, Slovak Republic
norbert.polcak@umb.sk

ABSTRACT

In a short span of time, cloud computing has grown, particularly for commercial web applications. But the cloud computing has the potential to become a greater instrument for scientific computing as well. A pay-as-you-go model with minimal or no upfront costs creates a flexible and cost-effective means to access compute resources. In this paper, we carry out a study of the performance of the spatial data interpolation of depth of the snow cover on the most widely used cloud infrastructure (Amazon Elastic Compute Cloud). The main characteristic of the interpolating computing is the fact that it is time-consuming and data intensive; therefore utilizing parallel programming paradigm is eligible. The geoprocessing is realized on two configuration provided by Amazon EC2 and the results as well as performance of the computing is presented in the article.

Categories and Subject Descriptors

D.1.3 [**Programming Techniques**]: Concurrent Programming—*Distributed programming, Parallel programming*; J.2 [**Computer Applications**]: Physical Science and Engineering—*Earth and atmospheric sciences*

General Terms

Measurement, Performance

Keywords

Computing clouds, cluster computing, parallel computing, inverse distance weighting, interpolation method

1. INTRODUCTION

Cloud computing represents the next evaluation step of on-demand information technology services and products. Cloud computing offers new approaches for scientific computing that forces the hardware and software investments on large data centres by major commercial players. Geographical information systems as a scientific field using data intensive computing can exploit the ongoing move towards data processing frameworks to perform parallel computations.

In this paper we present solution for parallel geoprocessing utilizing the processing power of computing cloud for computing of the spatial data interpolation of missing points in the surface. The main characteristics of the interpolating computing for such surface is the fact that it is time-consuming; therefore utilizing parallel programming paradigm is eligible.

A big attention in the world is given to the research of prognosis whether the depth of snow cover depends on the influence of global warming. The high performance computing resources are used for this purpose [10, 8]. But in most cases, the processing relates to a much larger area than continents or countries [8, 11]. Determination of the snow cover depth in a defined territory seems to be problematic because of absence of raingauge stations. Therefore, the interpolation methods for determination of the snow cover depth are necessary where the data from nearby raingauge stations are used.

Cloud computing is an emerging paradigm where computing resources are offered over the Internet as scalable, on-demand (Web) services. In this paper, we study the implementation and behaviour of inverse distance weighted interpolation (IDW) in cloud computing environment.

The rest of the paper is organized as follows. In Section 2 we present the background of spatial data interpolation of depth of the snow cover. Next we briefly introduce used model and its implementation in MPI standard. In Section 3 we describe the used Cloud and the preparation process of instances. Section 4 shows experimental results and short comparison of the models. The conclusions come in the last section with an outlook to further work.

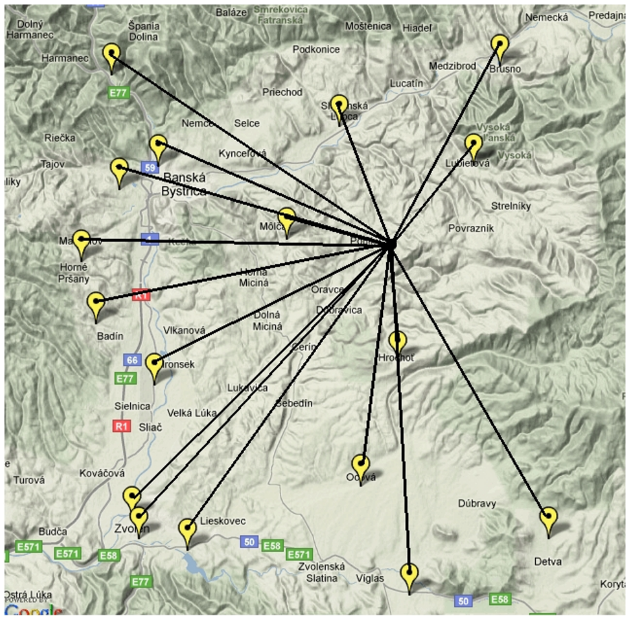


Figure 1: IDW method for one point in the location Zvolenská kotlina with 17 raingauge stations.

2. SPATIAL DATA INTERPOLATION

Interpolation is a mathematical function or method that estimates the values at points where no measured values are obtainable. It is not able to measure the values of the particular phenomenon in all points of the space, but only in sample points. Spatial interpolation supposes the attribute data points are continuous over space whicht allows for the estimation of the prediction points at any location within the data boundary. Furthermore the spatial interpolation assumes that the phenomenon is spatially dependent, indicating the values closer together are more likely to be similar than the values farther apart.

The primary disadvantage of using the spatial interpolation, especially for large and complex datasetsis is the fact that it can still be an iterative, more time-consuming task, requiring an adequate knowledge of underlying methods and their implementation.

For our research we use the spatial interpolation for prediction of depth of the snow cover. The depth of the snow cover is a very variable meteorological element in the landscape. Many factors influence the depth, mainly snow precipitation, altitude, air temperature, profile of the relief, solar power, cloudiness, air temperature inversion, etc. Snow depth is measured by meteorological, climatological and precipitation stations once a day. However, during periods of snowfall, it is measured every six hours to determine the amount of recent snowfall.

The analysis of the snow cover is based on many continuous observations and measurements at the specific climatological stations. All these gauging places can be geographically strictly characterized by the altitude, latitude and longitude, as well as by the detailed characteristics of the relief shape.

The aim of this work is the depth of the snow cover computing in any point based on the geographical characteristics of a specific geographical point in a modelled area. The outcome is derived from the available data, which have been obtained from available meteorological or climatological stations from a defined landscape area.

An actual geomorphological entity Zvolenská kotlina as a part of Slovak Republic (see Figure 1), which is exactly defined by its borders, has been chosen as an example of the application of our designed method. We use the digital terrain model of this entity. The 17 meteorological stations of Slovak Hydrometeorogical Institute are situated in this defined area and their long-term measurements are available for our research. The input data are stored in large matrices. The output values depend on the time-consuming computing process.

2.1 Inverse Distance Weighting Interpolation Method

As the basic interpolating method we used an inverse-distance-weighting algorithm to interpolate the snow cover measurements. This deterministic model, compared to e.g. Kriging interpolating method, is relatively fast and easy to compute, and straightforward to interpret. The IDW method as a deterministic spatial interpolation model is one of the most popular methods adopted by geoscientists and geographers partly because it has been implemented in many GIS tools [7].

2.2 Parallel Implementation of IDW Interpolation

The general assumption of this method is that the attribute values of any given pair of points are related to each other, but their similarity is inversely related to the distance between the two locations. The general idea of IDW is that the attribute value of an un-sampled point is the weighted average of known values within the neighbourhood, and the weights are inversely related to the distances between the location of un-sampled point and the location of its neighbours. The value of inverse-distance weight is modified by a constant power with increasing distance. This dependence can be expressed by the following relationship (1).

$$y_0 = \frac{\sum_{i=1}^{n} \frac{y_i}{d_i^k}}{\sum_{i=1}^{n} \frac{1}{d_i^k}} \tag{1}$$

where y_0 is value of un-sampled point; d_i denotes the distance between un-sampled point and sampled location i and y_i is given value at sampled locations i; n is the total number of known points used in interpolation.

Value k is arbitrary positive real number called the weighting exponent; if the parameter k equals to zero, there is no decrease with distance, the weights will be the same and the prediction will be the mean of all measured values; if the k value is very high, only the immediate few surrounding points have an influence on the prediction. For our experiments we used k equals to 2 named as the method of inverse square distance weighting, where the weighting function depends on Euclidean distance and is radially symmetric about each scatter point.

Our previous experience with parallel implementation of interpolation methods on various systems, e.g. General-Purpose computing on Graphics Processing Units [3], Multicore CPU design or Computational grid [2] indicates the use of the cloud computing model based on MPI for distributed systems. Moreover, to use for example Amazon Elastic

Compute Cloud for anything other than bag-of-tasks type of scientific applications, one would need to use some parallel middleware. MPI is a language-independent communications technique used to program parallel computers, i.e. system with distributed memory. Amazon Elastic Compute Cloud requires using a "free" MPI implementation. So we used an open implementation OpenMPI of the MPI which eases the creation of distributed applications by abstracting the underlying network system.

To parallelize the snow cover model, the domain decomposition method is used, where all parallel computing elements (threads or processes) have equal or almost equal amount of data to be processed. The parallel algorithm is very similar to the serial algorithm with some additional routines added to facilitate the communication between parallel computing elements. All the parallel computing elements involved in the parallel calculations basically perform the same computational operations [6].

The cloud computing implementation based on MPI uses both point-to-point and collective communication. The MPI interface is meant to provide essential virtual topology, synchronization, and communication functionality between a set of processes in a language-independent way. In MPI, the entire code is launched on each node and it is controlled what each code executes based on its node number in the MPI universe along with an algorithm that distributes work, e.g. a master/slave model. Programmers have to divide the code in two parts for the master and for the slaves or to recognize which code they have almost the same.

Master node and slaves' nodes act as independent entities communicating through passing messages. These communications can be asynchronous or synchronous.

Once the data types are defined, distributing of data among processes is done by collective communication through functions MPI_Bcast and MPI_Scatter and, if necessary, by sending and receiving messages through functions MPI_Send and MPI_Recv of MPI implementation. Data distribution is performed in the Master node. Further mutual communication between Slave nodes is not necessary. Slave nodes send the particular results to the Master node which calculates the final result.

```
/*--- Distributed MPI Slave Code ----*/
...

MPI_Recv(&index, 1, MPI_INT, 0, indexmsg,
                         MPI_COMM_WORLD, &status);

for(i = index; i < index + chunksize; i++) {
    sum_up = 0;
    sum_down = 0;

    for(j =0; j < data_data_count; j++) {
      distance = SQR(raster_x[i] - data_x[j]) +
                 SQR(raster_y[i] - data_y[j]);
      sum_up += (data_value[j]/distance);
      sum_down += (1.00/distance);
    }

    raster_value[i] = sum_up/sum_down;
}

MPI_Send(&raster_value[index], chunksize,
            MPI_FLOAT,0, arraymsg, MPI_COMM_WORLD);
```

The following cloud computing model is just based on this scheme. The computing speed-up of all techniques was detected on different kinds of raster. The two-dimensional raster changed in pixel density. One-dimensional matrices represented each raster, because of format of GIS Grass (Geographic Resources Analysis Support System) input file. The sizes of the matrices varied from 1000×1000 to 10000×10000. In one case (cloud large configuration), we made experiment with matrix size 20000×20000.

3. USE OF COMPUTING CLOUD

Authors Mells and Grance define for the National Institute of Standards and Technology in [9] the cloud computing as: The Cloud Computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the datacenters that provide those services.

Cloud computing means both the applications delivered as services over the Internet and the hardware and systems software in the datacenters that provide those services. Private companies offer various solutions of commercial computer clouds, e.g. Amazon's Elastic Compute Cloud EC2, Windows Azure, IBM's Blue Cloud, etc. The Window Azure is primarily designed to run .NET framework programs. Our previous programs have been written in low-level programming language C. Therefore, Amazon's Elastic Compute Cloud is used to design a virtual computer cluster identical with the before used real computer cluster.

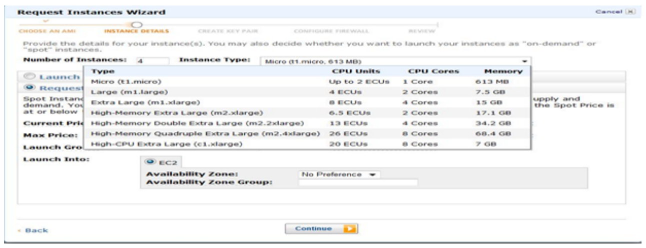
3.1 Amazon Elastic Compute Cloud

The major differences between the Amazon Web Services environment and a typical supercomputing center are for example; almost all HPC applications assume the presence of a shared parallel filesystem between compute nodes, and a head node that can submit MPI jobs to all of the worker nodes. Running these applications in the cloud requires either that the features of a typical HPC environment are replicated in the cloud, or that the application is changed to accommodate the default configuration of the cloud [5, 12].

Moreover, the right cloud service that best suits the program must be chosen. The requirements include availability, reliability, configurability, offered hardware solutions and price. Of the total services offered on the market, we chose a solution from Amazon, Amazon Web Services in particular Amazon Elastic Compute Cloud (EC2). EC2 is a Web service interface that provides computing and its change in size of the cloud. It is designed to configure compute capacity as simply as possible for developers which gives the users complete control over their computing resources and lets them run programs on the Amazon best computing environments. Amazon EC2 reduces the time needed to acquire and run a new instance of the server in seconds, allowing to quickly adapt capacity, as upwards, so below [1]. Amazon EC2 also introduced an interesting new pricing policy using for computing capacity pay no monthly or activation fees. Price is for real time use, when an instance is running, i.e. users pay only for the capacity that their applications actually need (pay-as-you go).

3.2 Preparation of Instances

Choosing the right operating system for instances as well as configuration of system resources for the cloud is required before starting to use a program for parallel computing.



Provide the details for your instance(s). You may also decide whether you want to launch your instances as "on-demand" or "spot" instances.

Number of Instances: 4 Instance Type: Micro (t1.micro, 613 MB)

Type	CPU Units	CPU Cores	Memory
Micro (t1.micro)	Up to 2 ECUs	1 Core	613 MB
Large (m1.large)	4 ECUs	2 Cores	7.5 GB
Extra Large (m1.xlarge)	8 ECUs	4 Cores	15 GB
High-Memory Extra Large (m2.xlarge)	6.5 ECUs	2 Cores	17.1 GB
High-Memory Double Extra Large (m2.2xlarge)	13 ECUs	4 Cores	34.2 GB
High-Memory Quadruple Extra Large (m2.4xlarge)	26 ECUs	8 Cores	68.4 GB
High-CPU Extra Large (c1.xlarge)	20 ECUs	8 Cores	7 GB

Spot Instan demand. You at or below upply and the Spot Price is

Current Pri

Max Price:

Launch Gro

Launch Into: EC2

Availability Zone: No Preference

Availability Zone Group:

Figure 2: Instances in Amazon EC2. The ECU is the CPU performance unit defined by Amazon.

There are different physical locations of the resources where each of these locations contains different availability zones being independent of each other in the event of failure. In our case the instances were located in the area of West EU (Ireland). Furthermore, a choice of operating system for the cloud is necessary for using of the MPI middleware. Commercial offerings of cloud services are not limited to the specific operating systems. We have chosen Linux distribution, Ubuntu specifically. Ubuntu was chosen mainly because of the previous experience of clusters solutions. There are many different versions of Ubuntu available in EC2, for our examinations we use instances ami-0e0f3d7a image (ubuntu-hardy-10.10- amd64- server-2011100l). Instances are in EC2 classified into three categories that reflect their computing power: standard instances (suitable for most applications), high-memory instances (especially for high throughput applications), and high-cpu instances (suitable for compute intensive applications). We have considered small instances in our experiments, because they are the default instance size and frequently demanded by users. We opted two instances configuration:

- *Micro* – Intel (R) Xeon (R) Processor E5507 (4M Cache, 2.26 GHz, 4.80 GT / s Intel (R) QPI), 613 MB RAM, 8 GB HDD,
- *Large* – Intel (R) Xeon (R) Processor E5645 (12M Cache, 2.40 GHz, 5.86 GT / s Intel (R) QPI), 7.5 GB RAM, 8 GB HDD.

The number of instances has been created at 4, because of the need for a master and three worker nodes to compare with our previously published experiments, see Figure 2. In addition, the number of instances has been determined by the price, because the economic side of the cloud services plays an important role. The price of each use depends on the applied configuration, so the Micro configuration is the bottommost and charged of $ 0.009 per hour when the instance is running. The next step of the preparation of instances is setting of security including private key generation, setting of ports, setting of security group, etc.

The following step of the cluster creation is the configuration of instances to co-operate in the cluster. Each instance represents one node of the cluster. One instance (labeled master) was configured as the master node. The rest of instances (labeled wn1, wn2, wn3) were configured as the worker nodes. The configuration includes an installation of some resource manager, scheduler, basic compiler and OpenMPI. The Torque resource manager and the Maui scheduler were used in this cluster to emulate the previously used cluster with physical nodes.

Table 1: Execution Time Depending on Problem Size (Raster Size) and Parallel Computer Architecture

Matrix size	Measured time (s)			
	Micro config.	Large config.	Intel Dual Core	Computer cluster
$1.0E+06$	0.7200	0.1600	0.2154	0.7900
$4.0E+06$	2.5300	0.6100	0.8602	3.0000
$9.0E+06$	6.5700	1.3500	1.9350	6.6700
$1.6E+07$	11.3800	2.2500	3.4410	12.350
$2.5E+07$	–	3.5000	5.3730	18.960
$3.6E+07$	–	5.0400	7.7360	27.400
$4.9E+07$	–	7.0300	10.530	37.660
$6.4E+07$	–	6.7000	13.750	48.590
$8.1E+07$	–	11.280	17.400	61.230
$1.0E+08$	–	13.620	21.490	76.960
$4.0E+08$	–	62.770	–	–

4. EXPERIMENT AND RESULTS

In the experimental evaluation of our computing we focused on the performance improvements from the Amazon EC2 using one small standard instance and one large standard instance in EU location.

To compare EC2 outcomes with a meaningful set of data, we used our previously published results [4]. These results were obtained from programs, which ran in our local cluster having physical nodes and our multi-core processor. To achieve comparable conditions, we used the same multi-core processor, as used in cluster nodes (Intel Pentium D, Dual Core, 2.66GHz, 2GB RAM, 100 GB HDD). As we had full control of the cluster, there was no additional workload on the cluster during our experiments. The input data are stored in the large matrices where are stored input data from 17 gauging stations (data includes the amount of snow cover and location coordinates – latitude, longitude, altitude). The output values depend on the time-consuming computing process. Performance time is measured from the assignment of workers to transfer the calculated results to the master. In all cases it is assumed that the data was already present in the frameworks preferred storage location. Measured time is recorded in the file *time.txt*. Outputs of our experiments can be straightforward visualized by the GIS Grass tool. We must note that the MPI implementation does not have strict limits of the number of processors and the program can be run on larger number of nodes in the future.

4.1 Measuring Time

The same way of obtaining of the measurement values was applied for Micro configuration as well as for the Large configuration. Since Amazon EC2 provides a Grid service, it should be possible to predict the delay of the service provider. Therefore, we performed three measurements and we have chosen the smallest measured time (the differences were at times in hundredths of seconds). The measurements started from matrix of size 1000×1000 and gradually increased for each measurement by 1000, to matrix size 10000×10000 what constitutes sufficient measured values to form some conclusions of the scheme. The greater matrices were tested just to confirm what the cloud load is able to calculate on the Large configuration. For exam-

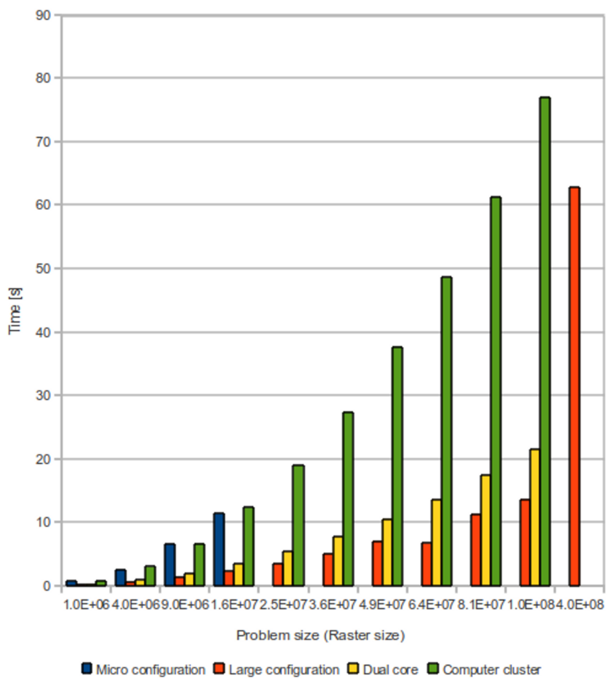


Figure 3: Chart of execution time depending on problem size (raster size) and parallel computer architecture.

ple, for the matrix of size 20000 × 20000 we found that the output text file contains measured times the size to 1 GB. Unfortunately we were not able to determine the outcomes for the micro configuration bigger than 4000 × 4000 matrix size, because the Amazon EC2 has not enough processing power to execute such configuration.

Results of our experiments are matched in the chart, Figure 3, and in the Table 1.

The results in Table 1 show that the use of the computer cloud gave comparable performance for geo-processing as the use of multi-core processor. It should be noted that the hardware used for comparison is less powerful than the actually available hardware. This fact emphasizes the advantage of cloud. Cloud does not require users to upgrade hardware. They have still modern and powerful hardware available.

5. CONCLUSION

This study evaluates the behaviour of the inverse distance weighted interpolation to accelerate snow cover depth prediction on the current most popular cloud computing provider Amazon EC2. We have demonstrated that clouds offer attractive parallel computing paradigm for scientific computation applications employing the spatial interpolation method.

Power and scalability as well as low charge of cloud data center, compare with the similar performance that GPGPU or cluster computing, suggests that scientific computation applications will increasingly be implemented on clouds. Cloud computing offers convenient user interfaces and the scientist or scientific institution is avoided only with little cost overhead of keeping and managing their own computing system.

Cloud computing proves some disadvantages as well. Since cloud services are often remote, they require a constant Internet connection and manipulation with cloud doesn't work well with low-speed connection. Additionally, there is not direct management of the cloud resources for the user and it is not clear if and how many other tasks are running on the allocated computer and network influencing the performance of the system. Moreover, a transfer and storage of large geodata sets can mean a time or cost burden of the computing process. Other issue of cloud computing is the question of security and privacy that needs to be ensured when executing the tasks in mostly remote locations which are controlled by external parties, (especially for health data sets). But the problems regarding transfer and storage of data as well as trustworthiness of infrastructures are not specific only for cloud infrastructures. They must be addressed for all kinds of distributed architectures.

The cloud computing as a new parallel computing technique has wide prospect in field such as spatial interpolation, so in the future we are planning to implement more efficient algorithms via implementing other interpolation methods as well as to implement different ways of decomposition for this type of computing environment.

6. ACKNOWLEDGMENTS

The authors acknowledge the data support of Slovak Hydrometeorogical Institute.

7. REFERENCES

[1] A. Eliáš, M. Strémy, and M. Kudla. Communication interface for virtual devices. In *Ecumict 2008: Proceedings of the Third European Conference on the Use of Modern Information and Communication Technologies*, pages 133–138, 2007.

[2] L. Huraj, V. Siládi, and J. Siláči. Comparison of design and performance of snow cover computing on gpus and multi-core processors. *WSEAS TRANSACTIONS on INFORMATION SCIENCE and APPLICATIONS*, 7(10):1284–1294, October 2010.

[3] L. Huraj, V. Siládi, and J. Siláči. Design and performance evaluation of snow cover computing on gpus. In *Proceedings of the 14th WSEAS International Conference on Computers: Latest Trends on Computers*, pages 674–677, 2010.

[4] L. Huraj, V. Siládi, and J. Siláči. Performance evaluation and comparison of mpi and openmp paradigms for interpolating computation. *Aktualnyje problemy i innovacii v ekonomike, upravlenii, obrazovanii, informacionnych technologijach*, 1(6):185–188, October 2011.

[5] K. R. Jackson, L. Ramakrishnan, K. Muriki, S. Canon, S. Cholia, J. Shalf, H. J. Wasserman, and N. J. Wright. Performance analysis of high performance computing applications on the amazon web services cloud. In *IEEE Second International Conference on Cloud Computing Technology and Science CLOUDCOM*, 2010.

[6] J. Škrinárová and M. Melicherčík. Measuring concurrency of parallel algorithms. In *Proceedings of the 2008 1st International Conference on Information Technology*, pages 289–292, 2008.

[7] G. Lu and D. Wong. An adaptive inverse-distance weighting spatial interpolation technique. *Computers & Geosciences*, 34(9):1044–1055, September 2008.

[8] A. Šmakin, D. Turkov, and A. Michjlov. Model' snežnogo pokrova s učetom sloistoj struktury i evol'ucii. *Kriosfera Zemli.*, XIII(4):69–79, 2009.

[9] P. Mells, T. Grance, N. I. of Standards, and Technology). The nist definition of cloud computing. Technical report, NIST Special Publication 800-145, September 2011.

[10] A. Shukinov, S. Butenkov, and A. Zhukov. Blanket of snow state physical model for clustering calculations based on information granulation. *Izvest'ja JUFU. Techničskie nauki.*, 97(8):213–223, 2009.

[11] M. Takala and J. Pullainen. Detection of snow melt using different algorithms in global scale. In *Proceedings of 2008 IEEE International Geoscience & Remote Sensing Symposium*, pages 674–677, 2008.

[12] P. Tanuska and T. Skripcak. The proposal of functional user requirements generation. In *ICCRD 2011: 3rd International Conference on Computer Research and Development*. IEEE, March 2011.