Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Content-based Mining for Solving Geoprocessing Problems on Grids

G. Folino, A. Forestiero, G. Papuzzo, G. Spezzano

RT-ICAR-CS-06-07 **Giugno 2006**

Consiglio Nazionale delle Ricerche, Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR)
– Sede di Cosenza, Via P. Bucci 41C, 87036 Rende, Italy, URL: *www.icar.cnr.it*
– Sezione di Napoli, Via P. Castellino 111, 80131 Napoli, URL: *www.na.icar.cnr.it*
– Sezione di Palermo, Viale delle Scienze, 90128 Palermo, URL: *www.pa.icar.cnr.it*

Content-based Mining for Solving Geoprocessing Problems on Grids

Gianluigi Folino, Agostino Forestiero, Giuseppe Papuzzo, Giandomenico Spezzano
ICAR-CNR
Via P. Bucci 41C
87036 Rende (CS) Italy
{folino, forestiero, papuzzo, spezzano}@icar.cnr.it

ABSTRACT

Geological data management and mining are critical areas of modern-day geology research. High throughput and high information content are two important aspects of any geoprocessing application. Geological data mining is efficient and faster if the geological data are indexed, stored and mined on content. A challenge for geological information mining is the distributed nature of the resources. Grid computing has emerged as an important new field in the distributed computing arena. It focuses on intensive resource sharing, innovative applications, and in some cases, high performance orientation. This paper describes how Grids technologies can be used to develop an infrastructure for developing geoprocessing applications that use content-based information mining. We present the MOSÈ system, a Grid-enabled problem solving environment (PSE) able to support the activities that concern the mining of geophysical data and modelling and simulation of spatio-temporal phenomena for analyzing and managing the identification and the mitigation of natural disasters like landslides, floods, wildfires, etc. MOSÈ takes advantages of the standardized resource access and workflow support for loosely coupled software components provided by the web/grid services technologies.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval—*Clustering*

General Terms

algorithms, design, performance

Keywords

data mining, peer-to-peer, clustering

1. INTRODUCTION

Geological data sets contain a huge amount of information about spatial phenomena. The exploitation of this knowledge with the aim to make it usable is one important aspect in developing geoprocessing applications. For example, the spatial prediction of landslide hazards is one important field of geoscientific research in which statistical classification rules have been applied. The aim of these methods is to identify areas that are susceptible to future landsliding, based on the knowledge of past landslide events and terrain parameters, geological attributes and other, possibly considering anthropogenic environmental conditions that are associated with the presence or absence of such phenomena.

Mining geophysical data for extracting knowledge is an essential task in modern geoscience applications. A challenge for spatial information mining is the lack of tools able to deal with the distributed nature of resources. Emerging geoprocessing problem-solving environments are characterized by increasing amounts of digital data and rising demands for coordinated resource sharing across geographically dispersed sites. Next generation grid technologies are promising to provide the necessary infrastructure facilitating seamless sharing of computing resources. Currently there exists no coherent framework for developing and deploying geoprocessing applications on the grid.

The MOSÈ (Spatio-Temporal **MO**delling of Environmental Evolutionary Processes by means of Geo**SE**rvices) system is a Grid-based problem solving environment (PSE) for the developing of Geoprocessing applications. MOSÈ is a PSE able to support the activities that concern the modelling and simulation of spatio-temporal phenomena for analyzing and managing the identification and the mitigation of natural disasters like floods, wildfires, landslides etc. The activities managed by MOSÈ are characterized by the necessity to handle large amount of spatio-temporal data and to support the interoperability among simulation models, distributed GIS, visualization systems, parameter estimation services, discovery of spatio-temporal patterns in preexisting data, etc. In this domain, the data conversion and the access, search, discovery and organization processes are complex problems because data are geo-referenced, stored in distributed GIS and can be used along three dimensions: temporal, spatial and referred to the physical properties. MOSÈ provides web based access to the spatial data by a browser and allows to observe and to manipulate data in a 2D/3D space by selecting regions in thematic maps. Users can examine features and patterns in a map in order to identify the region from which data must be extracted. Different physical properties can be extracted by thematic maps by data mining algorithms (clustering and classification) and used to select the most appropriated simulation model for the region analyzed. Similarly, from a temporal point of view, users can extract data that concern different parts of a temporal graph defining a period of time where the temporal data must be searched. Selected data can be used in input to a simulation model of the phenomenon that interacts with components for the parameter estimation for automated calibration of the model, 3D data visualization, and spatial data mining tools for analyzing the results.

MOSÈ uses a Web service based computing portal architecture to coordinate the access to the resources. Workflow technology is used to compose the services. The main components of MOSÈ are simulation services, geographic information (GI) services, geographic data and catalogues providing ontologies and metadata on the data and services. Each component has a wrapper and an XML interface for a simple composition.

In content networks, the problem of grouping information, coming from different sites requires the adoption of a distributed approach. Services of data mining supplied by MOSÈ can be profitably applied when data are in different sites on a network, especially in grid environment.

This first version of the MOSÈ [3] system regards the analysis of landslide hazard areas in the Region Campania near the Sarno area. The main actor in this scenario is a manager who wants to get an overview of the Sarno area with the indication of the regions which are currently slid down and those which are susceptible to slide down (landslide hazard areas) within a fixed time. For each scenario, the manager generates a workflow that orchestrates the web services necessary to obtain the outcome, and submitted it to the MOSÈ workflow enactment engine, which takes care of its execution. Some of the components that constitute the MOSÈ system use results of previous research developed in the past years and guarantee high performance and accuracy of the results [7].

The paper presents the software architecture of the MOSÈ system (section 2), an example of a knowledge grid service that enables mining of geophysical data using a distributed multi-agent spatial clustering algorithm (section 3), an application scenario in which the knowledge service for spatial clustering is used together with other grid services to obtain clusters of highly damaged buildings (section 4). Finally, conclusions are presented (section 5).

2. SOFTWARE ARCHITECTURE OF MOSE

The MOSÈ middleware is built on existing web/grid services technologies and standards [6]. This section provides a brief information on the MOSÈ architecture. A Web based interface, shown in Figure 1, is used to access the services offered by MOSÈ. The Web based portal supplies access to the spatial data by the client browser and allows observing, selecting, and manipulating data in a 2D/3D space selecting regions in thematic maps. Users can examine features and patterns in a map in order to identify the region from which data must be extracted and/or analysed. The MOSÈ architecture employs a service-oriented architecture. The architecture, shown in Figure 2, includes some components exported as web/grid services, each with an associated repository preserving historical (or previously created)

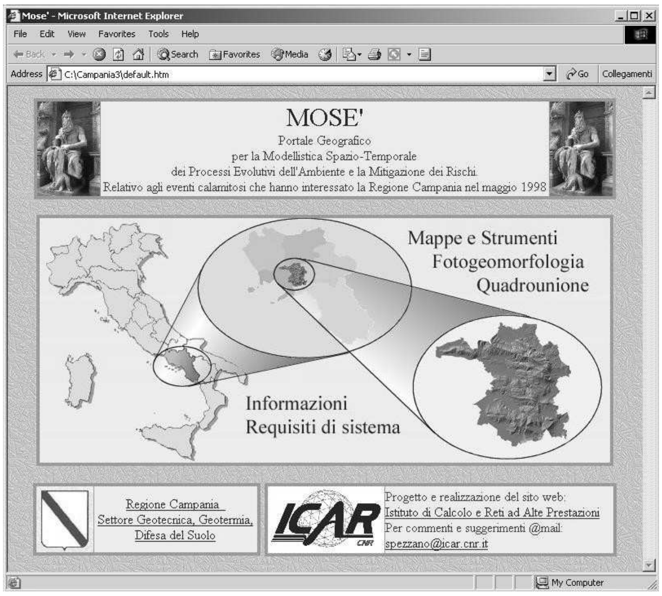


Figure 1: MOSÈ Web based GUI.

information, a workflow executor and Web-based access to a Geographical Information System (GIS).

The main components exported as web/grid services are:

- Data extractor component, to extract raster maps from the GIS by Geomedia Web Map tool.
- Visualization component, based on AVS-Express, to implement 2D/3D visualizations and virtual reality representations of one or more layers of the data extracted from the GIS.
- CamelotGrid [4], a cellular automata (CA) simulation tool running on the computational grid.
- Estimation of model parameters component, based on a parallel genetic algorithm running on a parallel machine available on the Grid. The CA models simulated with CamelotGrid are calibrated with the parameters that are estimated by this component.
- Data mining component, performing operations of spatial clustering, classification, etc.

The core of the system is the Workflow Executor (WE) that receives a workflow built by BPEL Designer and executes it on the Grid. A repository is associated to each component to reuse results or models previously obtained. CamelotGrid maintains a repository of the models of simulations, the parameter estimation service retains the parameters estimated for different regions for future reuse, the data extractor keeps data in the data repository, the 3D visualization component maintains a repository of 3D models of its simulations and finally the data mining component uses a knowledge repository to save acquired knowledge. Note that models can be obtained from previously executed simulations or can be inserted ex-novo using apposite tools of the PSE. MOSÈ can be used to execute simulations of different complex natural phenomena. Users must specify

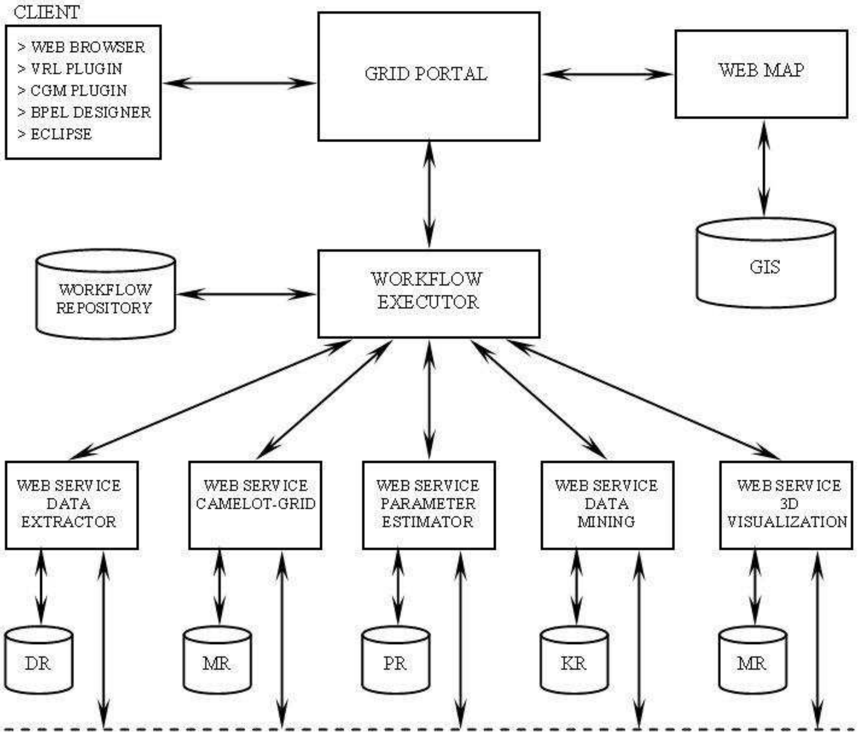


Figure 2: The MOSÈ software architecture.

the new CA model of the phenomenon and transfer it to CamelotGrid. New metadata and ontologies describing the data must be introduced and new GIS containing the geographical data for the application can be linked by the Web Map tool.

3. AN EXAMPLE OF KNOWLEDGE SERVICE: DISTRIBUTED SPATIAL CLUSTERING

In content networks, the problem of grouping information, coming from different sites requires the adoption of a distributed approach. MOSÈ supplies services of data mining that can be profitably applied when data are in different sites on a network, especially in grid environment. In order to give a more comprehensive explanation of the entire process, in the following subsections, we will describe a significant example of a data mining service that can be exploited in order to perform the task of clustering, i.e. aggregation of contents. The service, called P-SPARROW, executes the task of clustering on spatial data distributed along the network. It is based on a multi-agent paradigm that exhibit a collective intelligent behavior (swarm intelligence [1]) and combines the stochastic search of an adaptive flocking with a density-based clustering method.

Note that the emergent collective behavior is the outcome of a process of self-organization, in which insects are engaged through their repeated actions and interaction with their evolving environment. Intelligent behavior frequently arises through indirect communication between the agents using the principle of stigmergy [5].

3.1 The P-SPARROW clustering algorithm

Density-based clustering methods try to find clusters on the basis of the density of points in regions. Dense regions that are reachable from each other are merged to formed clusters. DBSCAN [2] is one the most popular density based methods and it is based on the idea that all the points of a data set can be regrouped into two classes: *clusters* and *noise*. Clusters are defined as a set of dense connected regions with a given radius (*Eps*) and containing at least a minimum number (*MinPts*) of points. The two parameters, Eps and MinPts, must be specified by the user and allow to control the density of the cluster that must be retrieved. The algorithm defines two different kinds of points in a cluster: *core points* and *non-core points*. A core point is a point with at least MinPts number of points in an Eps-neighborhood of the point. The non-core points in turn are either *border points* if they are not core points but they are *density-reachable* from another core point or *noise points* if they are not core points and are not density-reachable from other points. To find the clusters in a data set, DBSCAN starts from an arbitrary point and retrieves all points that are density-reachable from that point. A point p is density reachable from a point q, if the two points are connected by a chain of points such that each point has a minimal number of data points, including the next point in the chain, within a fixed radius. If the point is a core point, then the procedure yields a cluster. If the point is on the border, then DBSCAN goes on to the next point in the database and the point is assigned to the noise. DBSCAN builds clusters in sequence (that is, one at a time), in the order in which they are encountered during space traversal. The retrieval of the density of a cluster is performed by successive spatial queries. Such queries are supported efficiently by spatial access methods such as R*-trees.

DBSCAN is not suitable for finding *approximate* clusters in very large datasets neither it is adapt to work in a distributed environment. In fact, DBSCAN starts to create and expand a cluster from a randomly picked point. It works very thoroughly and completely accurately on this cluster until all points in the cluster have been found. Then another point outside the cluster is randomly selected and the procedure is repeated. This method is not suited to stopping early with an approximate identification of clusters.

As in DBSCAN, **P-SPARROW** finds cluster performing region-queries on core points but it replaces the exhaustive search of the core points with a stochastic multi-agent distributed search that discovers in parallel the points. P-SPARROW is constituted of two phases: a local phase for the **discovery** of the core points on each peer and a **merge** phase that concerns a global relaxation process in which nodes exchange cluster labels with nearest neighbors until a fixed point (i.e. all nodes detect no change in the labels) is reached.

```
for i=1 . . . MaxIterations
        foreach agent (yellow, green)
                if (not visited (current_point))
                        density = compute_local_density();
                        mycolor= color_agent(density);
                endif
        end foreach
        foreach agent (yellow, green)
                dir= compute_dir();
        end foreach
        foreach agent (all)
                switch (mycolor){
                        case yellow, green: move(dir, speed(mycolor));
                        break;
                        case white: stop ();generate_new_agent();break;
                        case red: stop (); merge();
                        if (new_red()) clone_agent();break; }
        end foreach
        if ((bag_out.dimension()> threshold)or(i%IterMigr==0))
                send_bag();
        if (bag_in_full()) notify_changes();
end for
```

Figure 3: The pseudo-code of P-SPARROW executed on every peer.

All the data are partitioned among the peers, proportionally to the computing power and to the cpu-load of the peer itself. Each peer implements the flocking algorithm, described in figure 3, using a fixed number of agents that initially occupy a randomly generated position in the space. Each agent moves testing the neighborhood of each object (data point) it visits in order to verify if the point can be identified as a *core point*. Then, P-SPARROW uses a flocking algorithm, inspired by the principles of the Geographical Analysis Machine [8], with an exploring behavior in which individual members (agents) search some goals, whose positions are not known *a priori*, in parallel and signal the presence or the lack of significant patterns into the data to other flock members, by changing color. The entire flock then moves towards the agents (*attractors*) that have discovered interesting regions to help them, avoiding the uninteresting areas that are instead marked as obstacles. The color is assigned to the agents by a function associated to the data analyzed during the exploration, according to the DBSCAN density-based rules and with the same parameters: *MinPts*, the minimum number of points to form a cluster and *Eps*, the radius of the circle containing these points. In practice, the agent computes the local density (*density*) in a circular neighborhood (with a radius determined by its limited sight, i.e. *Eps*) and then it chooses the color (and the speed) in accordance to some simple rules, better described in table 1.

$property > threshold$	$\Rightarrow$	$mycol = red\ (speed = 0)$
$\frac{threshold}{4} < prop. \leq threshold$	$\Rightarrow$	$mycol = green\ (speed = 1)$
$0 < property \leq \frac{threshold}{4}$	$\Rightarrow$	$mycol = yellow\ (speed = 2)$
$property = 0$	$\Rightarrow$	$mycol = white\ (speed = 0)$

Table 1: Assigning speed and color to the agents

So *red* agents reveal a high density of interesting patterns in the data, *green*, a medium one, *yellow*, a low one and white agents indicate a total absence of patterns. The color is used as a communication mechanism among flock members to indicate them the roadmap to follow. The main idea behind our approach is to take advantage of the colored agent in order to explore more accurately the most interesting regions (signaled by the red agents) and avoid the ones without clusters (signaled by the white agents). Red and white agents stop moving in order to signal these regions to the others, while green and yellow ones fly to find clusters. Green agents will move more slowly than yellow agents in order to explore more carefully zones with a higher density of points. The variable speed introduces an adaptive behavior in the algorithm. In fact, agents adapt their movement and change their behavior (speed) on the basis of their previous experience represented from the red and white agents. Anyway, each flying agent computes its heading by taking the weighted average of alignment, separation and cohesion.

Green and yellow agents compute their movement observing the positions of all the agents that are at most at some fixed distance (*dist_max*) from them and applying the rules of Reynolds' [9] with the following modifications: *alignment* and *cohesion* do not consider yellow agents, since they move in a not very attractive zone; *cohesion* is the resultant of the heading towards the average position of the green flockmates (centroid), of the attraction towards red agents, and of the repulsion by white agents; a *separation* distance is maintained from all the agents, whatever their color is.

New red agents executes the merge procedure; i.e., a temporary label will be given to these agents and to all the points of their neighborhood, if they are not already labeled. Otherwise the minimum of all the labels will be assigned to all the core points in this neighborhood, in order to make them belong to the same cluster. In this way, on each peer the set of red agents (core points) determinates the local model of clustering. Neighboring peers must be informed about the new core points or about the new labels in order to merge all the points belonging to the same cluster. To this end, red agents create clone agents and put them in

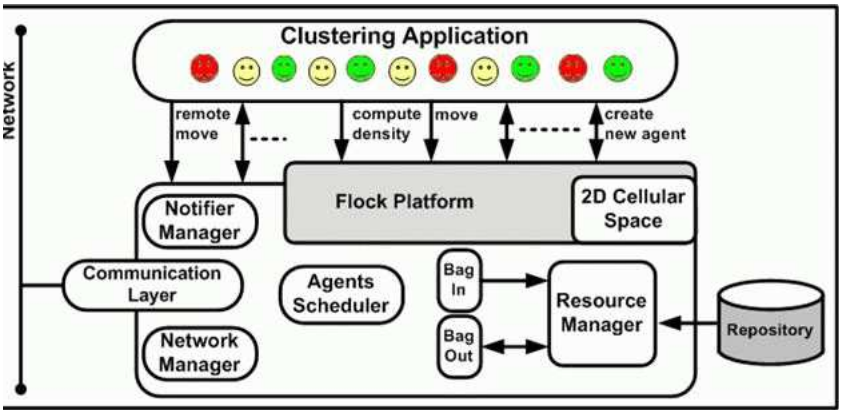


Figure 4: The software architecture of P-Sparrow.

an apposite bag and, when a fixed number of clone agents is achieved (i.e. a bag of agents has reached the desired dimension) or a certain number of iterations have been performed, each peer will send the bag containing the cloned red agents to the neighboring peers. Consequently, the agents received from the other peers will be put in another bag that will be used in the next iteration (or when it becomes full) for the merge phase. In practice, the new agents continuously update the labels as multiple clusters take shape concurrently. This continues until nothing changes, by which time all the clusters will have been labeled with the minimum initial label of all the sites containing the data. All the points having the same label form a cluster.

3.2 The software architecture

The software architecture of P-SPARROW is described in figure 4.

The **flock platform** manages the cellular space in which the agents move. Furthermore, it supplies the main procedures concerning the agents (move, remote move, create new agent, clone agents, etc..) using the underlying levels. Agents of different colors will be scheduled by means of the **agents scheduler**.

The **resource manager** (RM) execute efficiently range queries (i.e. compute density) in the dataset, accessing the repository, in order to choose the new color of the agents. The RM is also responsible of putting new agents received by the neighboring peers in the appropriate zone in order to start a new phase of merge. The arrival of a new bag of agents is signaled by the **notifier manager** that supplies also information about new events such as the fall of a peer, the convergence of the algorithm, etc... The **network manager** handles the send and the receive of the bags of agents on the basis of the topology of the system, depending on the characteristics of the network, using JXTA sockets.

3.3 Sequoia benchmark

To evaluate the performance of P-Sparrow, we used a spatial dataset, SEQUOIA [10], composed by 62556 names of landmarks (and their coordinates), and extracted from the US Geological Survey's Geographic Name Information System. In practice, the points in figure 5, represent points of interest in the sequoia area and the three main clusters, discovered from our algorithm, correspond respectively to the areas of S. Francisco, Sacramento and Los Angeles.

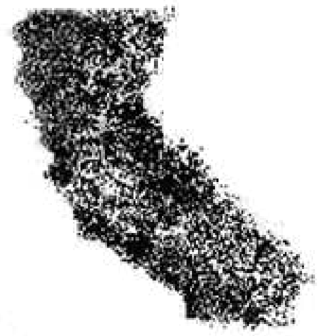
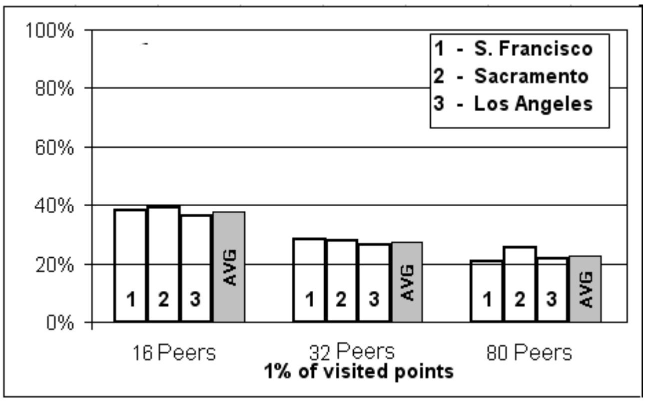


Figure 5: The Sequoia dataset.

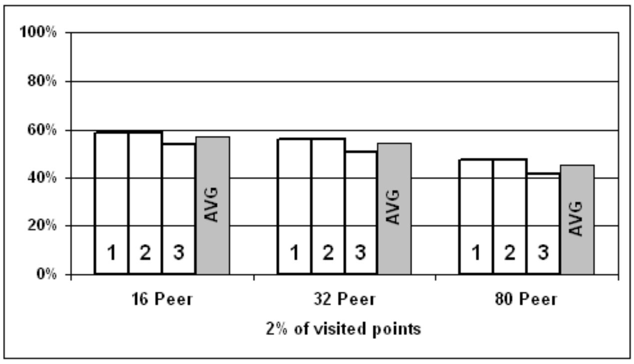
The dataset was partitioned respectively on 16, 32 and 80 peers to simulate a real environment in which the data are distributed on different sites.

We run our algorithm using 100 agents working until they explore the 1%, 2%, 5% and 10% of the entire data set, using 16, 32 and 80 peers. All the experiments were averaged over 30 trials. Our algorithm uses the same parameters as DBSCAN. Therefore, if we visited all the points of the dataset, we would obtain the same results as DBSCAN, as the merge phase is the same. Then, in our experiments we consider as 100% the cluster points found by DBSCAN (note DBSCAN visit all the points). We want to verify how we come close to this percentage visiting only a portion of the entire dataset and that must be effective for different number of peers involved in the computation. Note that the dominant operation in the computation is the execution of the range queries, performed each time a point is visited, while the time of the other operations is negligible. So, the fact of reducing the percentage of visited points considerably reduces the execution time.

For a large number of peers, the density of points for cluster for peer necessarily decreases; so we have to choose a different value of the parameter MinPts to keep into account this aspect. In practice, we choose a value of MinPts inversely proportional to the number of peers (i.e. if we fix MinPts as 8 on 16 peers, we have to fix as 4 on 32 peers and so on). In figure 6 (a and b) and 7 (a and b) we show the experimental results concerning the accuracy and scalability

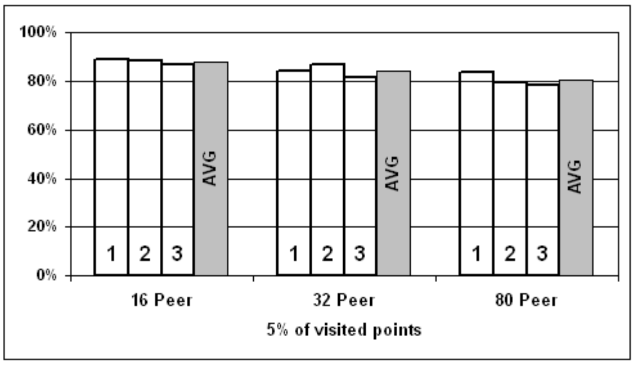


(a)

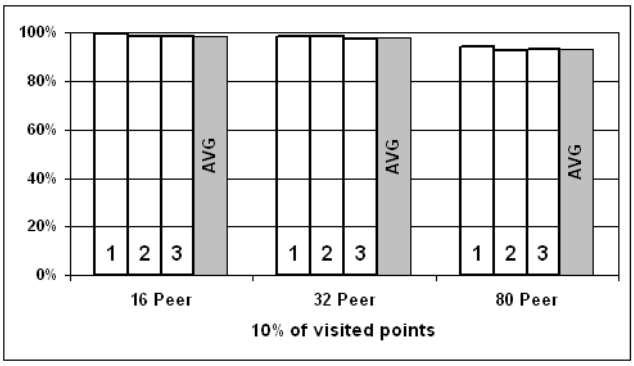


(b)

Figure 6: Number of points for cluster for Sequoia dataset (percentage in comparison to the total number of points for cluster) when P-SPARROW analyzes 1% and 2% of total points, using 16, 32 and 80 peers.



(a)



(b)

Figure 7: Number of points for cluster for Sequoia dataset (percentage in comparison to the total number of points for cluster) when P-SPARROW analyzes 5% and 10% of total points, using 16, 32 and 80 peers.

of the algorithm by varying the number of peers for Sequoia dataset.

For instance, on 80 peers, visiting only the 5% of points, on average, we obtain an accuracy of about 80% and visiting the 10% of data we reach 93% of accuracy.

Furthermore, the scalability (i.e. the effect on the accuracy of increasing the number of peers and so reducing the number of data points for peer) is quite good. In fact, if look at the Sequoia dataset, for the 5% case, we obtained a reduction from 88% for 16 peers to 81% for 80 peers while for the 10% case, we have a little reduction from 99% to 94%. Visiting only 1% of the dataset we have low percentage of points found; however they are sufficient to have an approximate idea of the shape of the clusters.

4. AN APPLICATION SCENARIO

One of the difficulties that civil protection authorities have to deal with in order to confront emergency conditions such as a landslide, is the management of the information coming up from the area where the landslide takes place. The difficulty becomes bigger due to the fact that after an landslide the demand for urgent intervention is huge. Emergency response actions must be taken immediately by civil protection authorities and a framework plan for planning and execution of post landslide operations is essential. One of the most critical actions that must be taken by civil engineers after a landslide is the discovery of post landslide damaged buildings. Data concerning the location of the buildings, their main characteristics and damage to different parts of the structure are collected, and can be compared to landslide map and historical damage locations.

MOSÈ can help a member of the civil protection to discover areas named hot spots which may represent the regions of highly damaged buildings.

Consider a scenario in which spatial data concerning the location of damaged buildings are sent by detectors and stored in different nodes. A decision-maker of the civil protection could use MOSÈ to recognize the highest density areas with damaged building in order to prevent the access to the area's inhabitants. We have applied MOSÈ on data concerning the landslide hazard areas in the Region Campania near the Sarno area. Results of the entire process are shown in figure 8, obtained using the overlap visualizer of MOSÈ. In the red circles, you can observe the three clusters representing the clusters of damaged buildings, obtained from a complete execution of P-SPARROW.

Obviously, the information must be obtained as quickly as possible. P-SPARROW, permits to find approximate clusters even if we do not explore all the points of the data sets. So the user can receive a first information about the

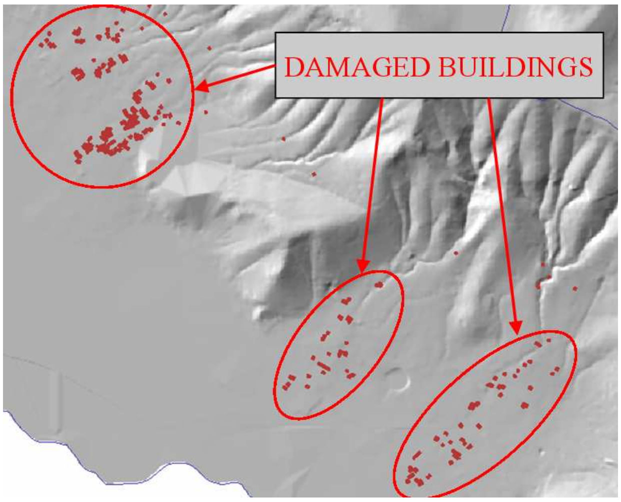


Figure 8: Visualisation of the workflow result.

interesting areas and can immediately act, afterwards, can ask for a more precise information, as P-SPARROW go on finding new points of clusters.

5. CONCLUSIONS

This paper presents the MOSÈ system that is capable of managing Geoprocessing applications on a Grid using content based mining techniques. The primary advantages of MOSÈ are the performance gain obtained using web/grid distributed resources and the support for the interoperability of data and resources. Furthermore, the P-Sparrow service is able to perform approximate clustering on distributed resources using a multi-agent based paradigm. Its incremental nature is particularly adapt for coping with emergency conditions.

6. REFERENCES

[1] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz. Swarm intelligence: From natural to artificial systems. *J. Artificial Societies and Social Simulation*, 4(1), 2001.

[2] Martin Ester, Hans-Peter Kriegel, Jorg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[3] Gianluigi Folino and Giandomenico Spezzano. Mose: a grid-enabled software platform to solve geoprocessing problems. *Il Nuovo Cimento*, 28(2), 2005.

[4] Gianluigi Folino and Giandomenico Spezzano. An autonomic tool for building self-organizing grid-enabled applications. *Future Generation Computer Systems*, 2007.

[5] P.P. Grass. *La Reconstruction du nid et les Coordinations Inter-Individuelles chez Beellicositermes Natalensis et Cubitermes sp. La Thorie de la Stigmergie : Essai d'interprtation du Comportement des Termites Constructeurs in Insect. Soc. 6.* Morgan Kaufmann, 1959.

[6] Foster I. and Kesselman C. *The Grid2: Blueprint for a New Computing Infrastructure.* Morgan Kaufmann, 2003.

[7] Cannataro M., Comito C., Congiusta G., Folino G., Mastroianni C., Pugliese A., Spezzano G., Talia D., and Veltri P. A general architecture for grid-based pse toolkits. In *Workshop on State-of-art in Scientific Computing (PARA'04)*, 2004.

[8] James Macgill. Using flocks to drive a geographical analysis engine. In M. A. Bedau J. S. McCaskill N. H. Packard S. Rasmussen, editor, *Artificial Life VII: Proceedings of the Seventh International Conference on Artificial Life.* The MIT Press, Cambridge, Massachusetts, 2000.

[9] Craig W. Reynolds. Flocks, herds and schools: A distributed behavioral model. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 25–34, New York, NY, USA, 1987. ACM Press.

[10] Michael Stonebraker, James Frew, Kenn Gardels, and Jeff Meredith. The sequoia 2000 benchmark. In Peter Buneman and Sushil Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., May 26-28, 1993*, pages 2–11. ACM Press, 1993.