Clustering of environmental Data using a Multi-agent System

Christian Stern
Faculty of Geomatics, Karlsruhe University of Applied Sciences, 76133 KARLSRUHE, Germany
Christian.Stern@hs-karlsruhe.de

INTRODUCTION

In the recent years the importance of environmental data is increasing. This has been reflected by the European legislation, when the Directive to establish an Infrastructure for Spatial Information in the European Community (INSPIRE) was published in May 2007 and has been adopted for the national legislation in the member countries, e.g. the "Geodatenzugangsgesetz" (GeoZG) in February 2009 in Germany. (INSPIRE, 2007; GeoZG, 2009).

Environmental data is often captured in high resolution scales by the environmental agencies or similar institutions. In the German federal state of Baden-Württemberg water protection areas, biotopes or nature protection zones are collected as vector data in the scale 1 : 10 000 for example. But there is also a demand for this data in lower scales. From national mapping agencies all over Europe topographical data is available in various scales, sometimes compiled from scratch for each scale, or derived by using automated generalization processes. But yet there seems to be a lack for appropriate environmental data that can easily be used with existing topographical data in lower scales. Our overall research goal is to develop a process chain that provides the missing functionality to integrate environmental data into the existing topographical data by using a multi-agent system for this multi constrained problem composed of several tasks. The approach that we will propose in this paper is the first step of this process chain and will contribute to automate the process of generalization and integration of environmental data.

As it is widely accepted, prior to starting the process of generalization it is necessary to acquire knowledge about the data to be generalized and to find structures and patterns that have to be preserved or dealt in special manners. Additionally, for some generalization approaches it is important to find a way to divide the whole dataset into subsets and to perform the generalization process on each of them, before the whole result is compiled. Features that are defined to be invariant for the generalization process help to split the whole dataset into sections. In generalization process of topographical data the road or motorway network or administrative boundaries are used for this purpose. For environmental data it is far more difficult to find those invariants for segmentation. The river network or ridgelines could be taken as separators. Geographers divide regions into natural landscape units that could be used as basis. But not for all natural phenomena these features are suitable as divisors. Water protection areas for example cannot be clearly separated by any of these means. They can but not necessarily need to be bordered e.g. by roads, rivers, administrative boundaries. But they also might traverse them as well. So it is evident that for some types of environmental data it is necessary to run a preliminary clustering process to group similar features together so that following ones can rely on the results and make use of them. Of course there have been many projects and approaches that deal with clustering. For example, a comprehensive summary of clustering techniques in data mining can be found in Han et al. (2001) and a study concerning the advantages of using spatial clustering in support of the generalization process has been composed by Jiang (2004). But as far as we know there has been no attempt made to use a multi-agent system for clustering environmental data yet.

MULTI-AGENT SYSTEMS

Clustering is not only a very important research field in geoinformation science, but also in many other scientific and technical areas. In recent research works multi-agent systems have been used to perform this task. For example, Foner (1996) describes a matchmaking system to find people that share similar interests or Passadore et al. (2009) who explain their agent-based semantic search engine that improves document retrieval by finding and "clustering" texts which are semantically connected to what the user is searching for. Park & Oh (2006) report on intelligent clustering by unsupervised learning using a multi-agent system.

There has also been done some research work in the field of generalization with multi-agent systems. For example, Ruas & Duchêne (2007) presented a prototype generalization system based on the multi-agent system paradigm. Galanda (2003) had used agents to automatically generalize polygons, while Jabeur (2006) studied the "on-the-fly" generation of maps for the internet using agent technology.

In our approach, we also use the benefits of multi-agent systems. A multi-agent system is a method of distributed artificial intelligence. It is a system where multiple interacting, intelligent agents pursue their goals or perform specific tasks (Weiss, 1999) and that is concerned with finding a collaborative solution of problems by the help of a decentralized group of processes or agents (Torsun, 1995). It is a decentralized system with distributed control and asynchronous computation that provides a runtime environment and defines the infrastructure that is necessary for the agents' interaction and communication (Boccalatte et al., 2004).

Wooldrige (1996) defines an agent as a "computer system that is *situated* in some *environment*, and that is capable of *autonomous action* in this environment in order to meet its design objectives." By cooperating, negotiating and competing with each other, agents interact with each other to improve their performance. They act goal-directed and can take the initiative (pro-active), as well as reactive by responding to changes in their environment (Boccalatte et al., 2004; Wooldrigdge & Jennings, 1995).

Advantages of using a multi-agent system arise from the possibility to speed up computational time by having multiple agents performing parallel computation on different machines. Limitations imposed by time-bounded or space-bounded requirements can be overcome using the system's parallelism. If using redundant agents, one also can achieve a robust system if control and responsibility is distributed among them. If one agent fails, the system still is in a stable condition. From the programmer's point of view another benefit is the scalability of a multi-agent system. Since it is modular, it is easy to augment the system with new functionality by implementing and adding new agents. This also can lead to simpler programming. (See Stone & Veloso, 2000).

CLUSTERING APPROACH

In our approach we suppose that most of the environmental data is of planar type (polygons) and that similar features are grouped together. The challenge is to find these clusters. We want to detect them by comparing the geometry's shape of each environmental feature with neighboring features. As measurements of shape we want to take a polygon's area *A* and perimeter *P*, as well as the direction *Dir* of the main axis into account. Additional measures like the related bounding figure *RBF*, the area to perimeter *A2P* and length to width ratios *L2W* and the compactness *C* of the shape also help to find similar polygons. Details on how these measurements are defined and how they can be computed are well known and can be found in de Smith et al. (2007) or Peter (2001), for example. Together with the distance *dF* to the next feature in the cluster and the distance *dC* to the cluster's center point these shape measures have an influence on the so called "happiness" of each Cluster Agent (this agent type will be explained in the next paragraph). Calculating the happiness is a complex process and can only be explained briefly here: The happiness value h_m for almost all of the measures mentioned above (for distances slightly other rules apply) is determined by subtracting the ratio of the absolute value of the difference between the measure m and the cluster's corresponding average value m_a and the average value m_a from 1:

$$h_m = 1 - \left(\frac{|m - m_a|}{m_a}\right)$$

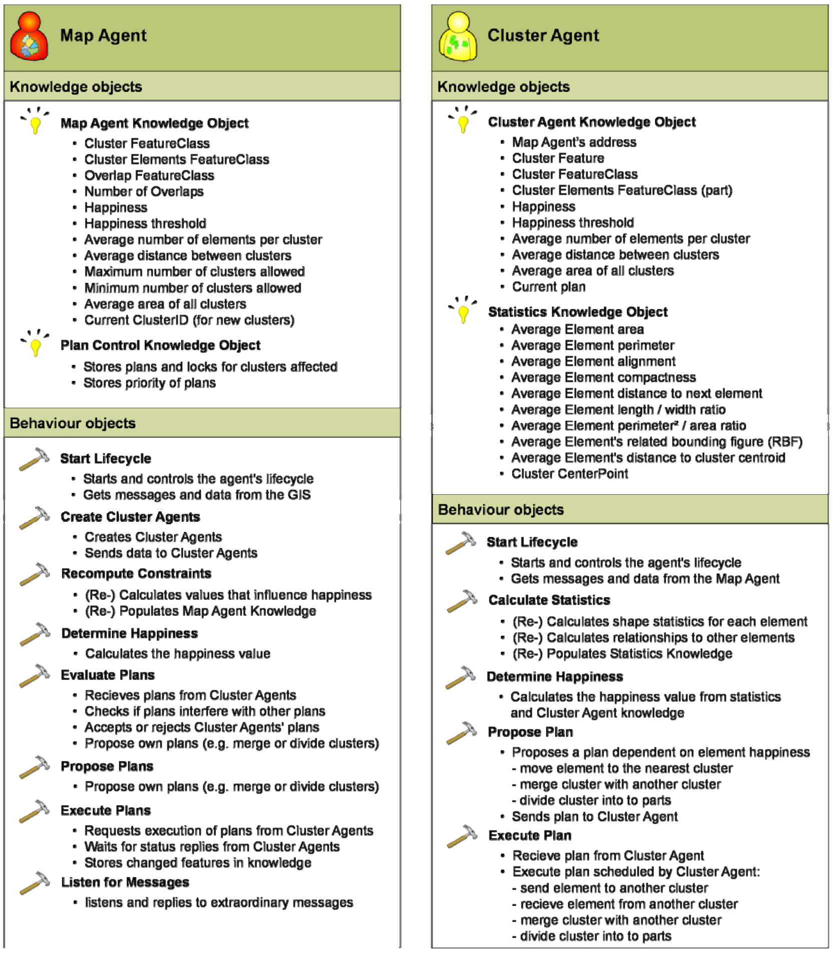
Then for each element in the cluster the happiness h_e can be calculated by building the weighted average of the happiness values determined before.

$$h_e = \frac{w_1 h_A + w_2 h_P + w_3 h_{Dir} + w_4 h_{RBF} + w_5 h_{A2P} + w_6 h_{L2W} + w_7 h_C + w_8 h_{dF} + w_9 h_{dC}}{\sum_{i=1}^{9} w_i}$$

Features that have been grouped together and share similar values will possess a high happiness value, whereas elements in a heterogeneous group or a "false" element in a homogeneous group will have low happiness values h_e. Now the happiness H_{ClAg} of the Cluster Agent can be determined by additionally taking the happiness values resulting from the distance to the next cluster h_{ClDist}, the cluster area h_{ClA} and the number of elements h_{NumEl} contained in the cluster compared to the average of all clusters into account. Once again weights w_i can be applied.

$$H_{ClAg} = \frac{w_1 h_e + w_2 h_{ClDist} + w_3 h_{ClA} + w_4 h_{NumEl}}{\sum_{i=1}^{4} w_i}$$

We work with two types of agents: the Map Agent and the Cluster Agent mentioned before. The Map Agent is in control of the complete clustering process and manages the whole life cycle of all agents, while the Cluster Agent represents a cluster of several single polygons that can be grouped together. Each agent possesses certain knowledge objects and behaviour objects that help him to reach his goals and to perform his tasks. The Map Agent has two knowledge objects. One is for keeping the data and constraints (Map Agent Knowledge Object), the second one for plans how to proceed with the clustering task. His behaviour objects represent all the necessary functions to run and control the clustering cycle with his Cluster Agents. While for each clustering task there is one Map Agent only, there exist multiple Cluster Agents, one for each cluster that has been found. Cluster Agents have two knowledge objects, too. The Cluster Agent Knowledge Object contains data and constraints, whereas the Statistics Knowledge Object basically stores the average of the cluster elements' shape measures. The behaviour objects assigned to the agent help him to perform the task requested by the Map Agent. (Illustrated by figure 1).



Map Agent

Knowledge objects

Map Agent Knowledge Object
- Cluster FeatureClass
- Cluster Elements FeatureClass
- Overlap FeatureClass
- Number of Overlaps
- Happiness
- Happiness threshold
- Average number of elements per cluster
- Average distance between clusters
- Maximum number of clusters allowed
- Minimum number of clusters allowed
- Average area of all clusters
- Current ClusterID (for new clusters)

Plan Control Knowledge Object
- Stores plans and locks for clusters affected
- Stores priority of plans

Behaviour objects

Start Lifecycle
- Starts and controls the agent's lifecycle
- Gets messages and data from the GIS

Create Cluster Agents
- Creates Cluster Agents
- Sends data to Cluster Agents

Recompute Constraints
- (Re-) Calculates values that influence happiness
- (Re-) Populates Map Agent Knowledge

Determine Happiness
- Calculates the happiness value

Evaluate Plans
- Recieves plans from Cluster Agents
- Checks if plans interfere with other plans
- Accepts or rejects Cluster Agents' plans
- Propose own plans (e.g. merge or divide clusters)

Propose Plans
- Propose own plans (e.g. merge or divide clusters)

Execute Plans
- Requests execution of plans from Cluster Agents
- Waits for status replies from Cluster Agents
- Stores changed features in knowledge

Listen for Messages
- listens and replies to extraordinary messages

Cluster Agent

Knowledge objects

Cluster Agent Knowledge Object
- Map Agent's address
- Cluster Feature
- Cluster FeatureClass
- Cluster Elements FeatureClass (part)
- Happiness
- Happiness threshold
- Average number of elements per cluster
- Average distance between clusters
- Average area of all clusters
- Current plan

Statistics Knowledge Object
- Average Element area
- Average Element perimeter
- Average Element alignment
- Average Element compactness
- Average Element distance to next element
- Average Element length / width ratio
- Average Element perimeter² / area ratio
- Average Element's related bounding figure (RBF)
- Average Element's distance to cluster centroid
- Cluster CenterPoint

Behaviour objects

Start Lifecycle
- Starts and controls the agent's lifecycle
- Gets messages and data from the Map Agent

Calculate Statistics
- (Re-) Calculates shape statistics for each element
- (Re-) Calculates relationships to other elements
- (Re-) Populates Statistics Knowledge

Determine Happiness
- Calculates the happiness value from statistics and Cluster Agent knowledge

Propose Plan
- Proposes a plan dependent on element happiness
  - move element to the nearest cluster
  - merge cluster with another cluster
  - divide cluster into to parts
- Sends plan to Cluster Agent

Execute Plan
- Recieve plan from Cluster Agent
- Execute plan scheduled by Cluster Agent:
  - send element to another cluster
  - recieve element from another cluster
  - merge cluster with another cluster
  - divide cluster into to parts

Figure 1: Knowledge objects and behaviour objects in Map Agent and Cluster Agent

The agents' life cycle represents the clustering process. In the beginning, constraints like the maximum or minimum number of clusters that shall be found and the happiness threshold have to be set. Additionally the clustering direction can be chosen. When choosing the "top-down" method, it is assumed that at the beginning there is one large cluster containing all geometries. This initial cluster is divided into reasonable parts until the desired number of clusters has been found. Following the "bottom-up" approach, each geometry is initially considered as a unique cluster. These clusters are merged in an expedient way until the threshold has been reached. Both "top-down" as "bottom-up" don't make much difference to the cycle, because all the necessary computational functions have to be implemented by the agent's behaviour (we will describe the "bottom-up" method). In the next step the input geometries are prepared for the successive tasks, e.g. a dataset for the new cluster geometries is created, new fields are added to store unique cluster IDs and the shape measures for each of the geometries is computed as attributes to it. Then the complete input data (geometries and attributes, constraints) are transferred to the multi-agent platform. This is the starting point of the agent's life cycle as it can be viewed in figure 2.

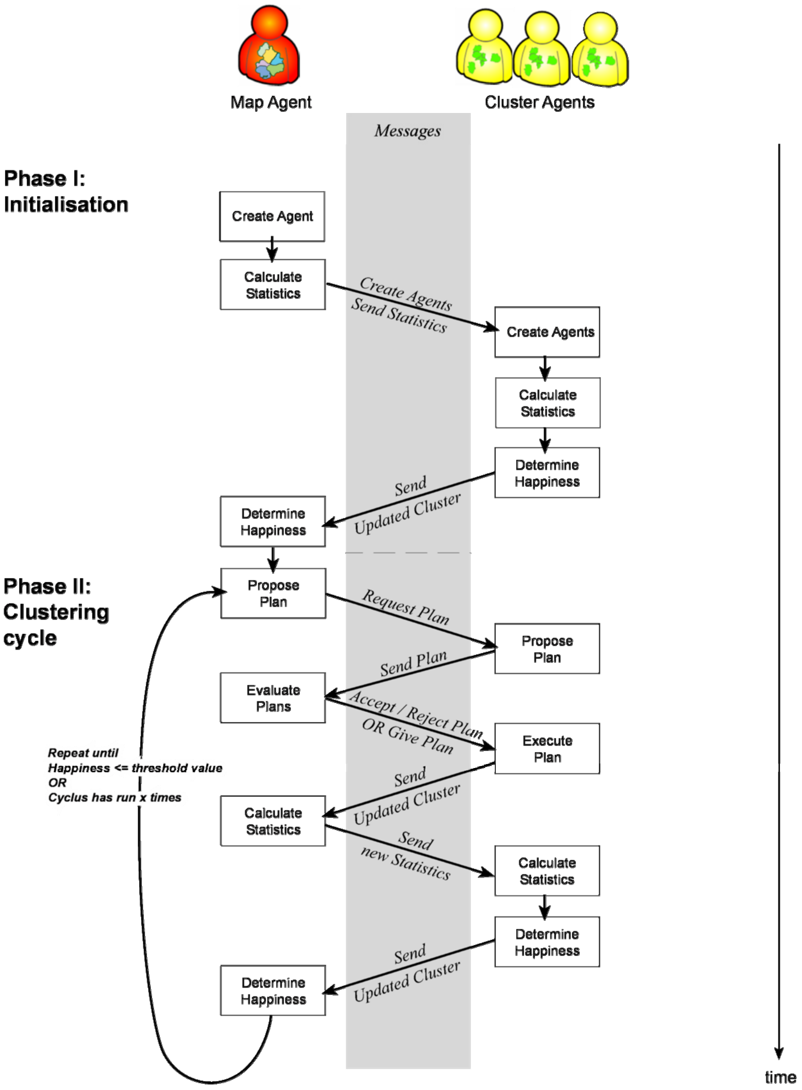


Figure 2: Clustering cycle run through by the agents

The process can be divided into two main phases. The first phase covers the initialization of the Map and Cluster Agents. The Map Agent is created and statistics describing the dataset are calculated. Following that, for each of the input geometries a Cluster Agent is instantiated. They themselves begin to calculate statistical values and can determine their happiness from them. By sending the information about their happiness to the Map Agent, he himself can compute his happiness. The Map Agent's happiness depends on the average Cluster Agent's happiness and the constraints that there exist no cluster overlaps and that the number of clusters is in the threshold limits. Now the second phase, the clustering cycle, is starting. At first, the Map Agent checks his constraints and then proposes plans to solve occurring restraints. This can be plans to merge or divide clusters. In case of overlapping or touching cluster geometries, or if the maximum number of clusters specified before has not been reached yet, clusters will be merged. A division occurs, if the desired minimum number of clusters has been under-run. In the next step, Cluster Agents are asked to submit their own plans.

They propose plans to move an element of the cluster to another cluster, if it appears that the element will be happier in the neighboring cluster. If there is no neighbor available, a plan to split off the element to a new cluster will be suggested. These plans are sent back to the Map Agent, where they are evaluated. Plans that promise a higher happiness increase will be accepted over ones with a lower increase if they concern the same cluster. Map Agent's plans are approved prior to Cluster Agent's plans. In the next step the plans are sent to the Cluster Agents and executed. This leads to changes in the cluster geometry that are sent back to the Map Agent to update his knowledge object. Then, the controlling agent calculates new statistical values and sends them to his subordinated agents. They can now calculate statistics themselves and determine their happiness value, which is sent back to the Map Agent. Finally, the Map Agent computes his happiness and the whole cycle is re-run if the threshold has not been reached or the number of clusters is not in the desired range. Summarizing the clustering process one can say that the Map Agent is in control of the process by deciding about the plans, while the "real clustering work" of merging and dividing clusters, adding and removing elements and calculating statistical values is performed by the Cluster Agents.

The implementation of the system has been realized using ArcGIS Desktop 9.3 as geoinformation component, whereas for the multi-agent part we used the AgentService framework that has been developed by a research team at the University of Genoa, Italy and has been published under the GNU Lesser General Public License. Vecchiola et al. (2008) describe the framework that is aimed to help programmers to develop, implement and deploy agent-oriented applications by providing a variety of tools and offering a flexible software infrastructure. The components can be easily customized and extended with new functionalities. The software has been written in C# .NET and relies on the Common Language Infrastructure that makes it portable to different implementations of this specification. This enabled us to integrate the framework into the programming environment of ArcGIS using C# .NET and ArcObjects.

RESULTS

For testing the clustering procedure a suitable vector dataset of water protection areas has been chosen. This dataset contains a mixture of small and large polygons (see figure 3). Some of them seem to be aligned; others seem to be grouped together a little bit. When manually trying to find clusters, one would probably find seven or eight of them (red ellipses). Some features are difficult to assign directly to one or another cluster, this can be seen where the red ellipses overlap.

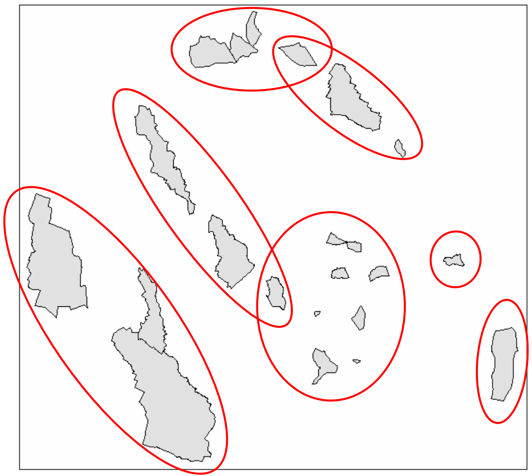


Figure 3: Test area with water protection zones. Red ellipses indicate potential clusters.

The clustering cycle has been started with constraints of seven clusters at maximum and three clusters at minimum. A happiness threshold of seventy percent should be reached, and the process was made to terminate after twenty iterations. The progress of the cycle can be seen in figure 4. Six interim results are shown that demonstrate the development of the clusters. The clustering process starts with building a cluster (convex hull) for each cluster element. At first, cluster elements that touch each other are merged together in one cluster (iterations 1 and 3). Then, the neighborhood of the clusters is checked if there are any similar features that could be integrated into the cluster. The Map Agent decides which clusters to merge (iterations 5 and 7). In iterations 9 and 11 the large clusters are formed to reduce the number of clusters to the desired value. After thirteen iterations the clustering process is stopped because the Map Agent has reached a happiness value of eighty-four percent, thus having a higher value than set by the constraint.

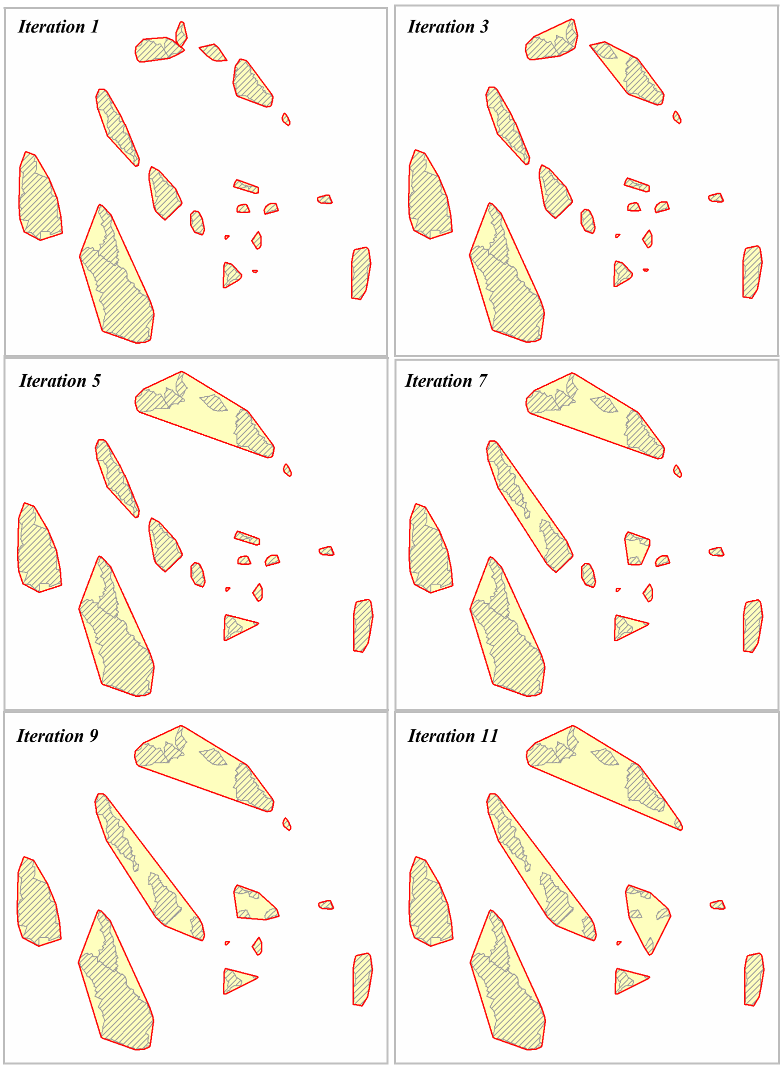


Figure 4: Progress of the clustering cycle. Current clusters are marked with a red border, while the cluster elements have a grey color.

The result is illustrated in figure 5. When comparing the outcome to the considerations made in the forefront of the clustering process, one can be quite happy. Most of the clusters expected have been found. The large territories in the left part of the image that share a similar alignment and size have been grouped correctly. The individual polygons in the right part of the image have been left lonely, while the small polygons in the center have been clustered in a right manner. The blue ellipses mark the results that might also have been expected but that not necessarily need to be a better result.

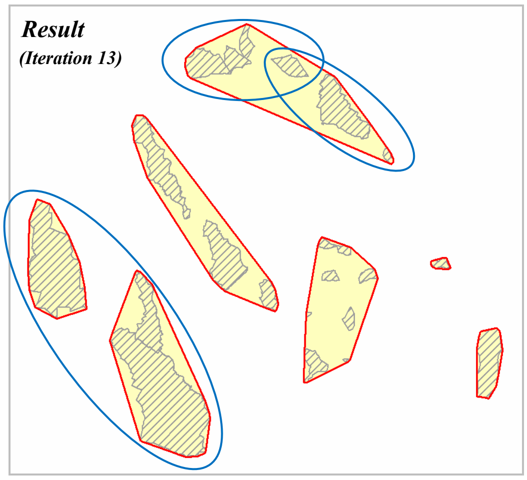


Figure 5: Result of the clustering process after thirteen iterations. The Map Agent has reached a happiness value of eighty-four percent. Elements that could have been grouped differently are marked in blue color.

CONCLUSION

It has been shown that our approach enables us to find groups of related polygons of environmental data that belong together. This has at least been true for our test area. We know that to finally judge our presented clustering process, there is a need of further tests with additional data that should vary in size and structure. In future work also semantic attributes of the polygons could be taken into consideration. Environmental features that consist of separated areas belonging together, e.g. biotopes on both sides of a river that share the same name or ID, could then be fixed together to make sure that they cannot be separated. However, it is planned to implement the “top-down” clustering method and the possibility to work on multiple distributed computer systems. This will point out one of the strengths of the multi-agent approach and hopefully improve performance further and reduce computational time.

BIBLIOGRAPHY

Boccalatte, A., Gozzi, A. Grosso, A. & Ch. Vecchiola, 2004 AgentService. Proceedings of the 16th International Conference on Software Engineering and KnowledgeEngineering (SEKE04), Banff, Alberta, Canada.

de Smith, M, Goodchild, M. F. & P. A. Longley, 2007 Geospatial Analysis – A comprehensive Guide to Principles, Techniques and Software Tools, Matador, Leicester, UK. ISBN 1905886608.

Foner, L., 1996 A Multi-Agent Referral System for Matchmaking. PAAM '96 Proceedings, London, England, 1996.

Galanda, M., 2003 Automated Polygon Generalization in a Multi Agent System. Doctoral Thesis. Department of Geography, University of Zurich.

GeoZG, 2009 Gesetz über den Zugang zu digitalen Geodaten (Geodatenzugangsgesetz). In BGBl. I p. 278.

Han, J., Kamber M., Tung A. K. H., 2001 Spatial clustering methods in data mining. In Miller H. J. & Han J. (eds.). Geographic data mining and knowledge discovery. Taylor & Francis, pp. 188-217, 2001.

INSPIRE, 2007 Directive 2007/2/EC of the European Parliament and of the Council of 14 March 2007 establishing an Infrastructure for Spatial Information in the European Community (INSPIRE) 14.03.2007. In Office Journal of the European Union of 25 April 2007, L 108/1.

Jabeur, N., 2006 A Multi-Agent System for “On-the-Fly” Web Map Generation and Spatial Conflict Resolution. Thèse de doctorat, Faculté des Sciences et de Génie, Université Laval, Québec, Canada.

Jiang, B., 2004 Spatial Clustering for Mining Knowledge in Support of Generalization Processes in GIS. In Proceedings of the ICA Workshop Generalisation and Multiple Representation, Leicester, UK.

Park, J., Oh, K., 2006 Multi-Agent Systems for Intelligent Clustering. In Proceedings of World Academy of Science, Engineering and Technology, Volume 11, February 2006. ISSN 1307-6884.

Passadore A., Grosso A., & A. Boccalatte, 2009 AgentSeeker: an Ontology-based Enterprise SearchEngine, MALLOW-AWESOME 009, Agents, Web Services and Ontologies, Integrated Methodologies, Torino, September 2009.

Peter, B., 2001 Measures for the Generalization of Polygonal Maps with Categorical Data. In Proceedings of Fourth ICA Workshop on Progress in Automated Map Generalization, Bejing, 2-4 August 2001.

Ruas, A., Duchêne, C., 2007 A Prototype Generalisation System Based on the Multi-Agent System Paradigm. In W. A. Mackaness, A. Ruas & L. T. Sarjakoski (eds.). Generalisation of Geographic Information: Cartographic Modelling and Applications. Elsevier: 269-284, 2007.

Stone, P., Veloso, M., 2000 Multiagent Systems: A Survey from a Machine Learning Perspective. In Autonomous Robots 8, pp. 345-383, 2000. Springer Netherlands. ISSN 0929-5593.

Torsun, I. S., 1995 Foundations of Intelligent Knowledge-Based Systems. Academic Press. London. ISBN 0126960607.

Vecchiola, C., Grosso, A. & A. Boccalatte, 2008 AgentService: a framework to develop distributed multiagent systems. In International Journal of Agent-Oriented Software Engineering, vol. 2, no.3 pp. 290-323, 2008.

Weiss, G. (ed.) 1999 Multiagent Systems. A modern Approach to Distributed Artificial Intelligence. The MIT Press. Cambridge, Massachusetts. ISBN 0262232030.

Wooldrige, M., 1996 An Introduction to MultiAgent Systems. John Wiley & Sons Ltd. Chichester, England. ISBN 047149691X.

Wooldridge, M. & Jennings, N. R., 1995 Intelligent agents: Theory and practice. The Knowledge Engineering Review, 10 (2), pp. 115-152.