Tracking Dynamics of Geospatial Phenomena in Distributed and Heterogeneous Environments Using Scientific Workflow and Web Services Technologies

Jianting Zhang
LTER Network Office, the University of New Mexico, Albuquerque, NM, 87131
Contact Email: jzhang@lternet.edu, Phone: 1-505-277-0666

Abstract: We propose to use W3C Web Services and recently emerging scientific workflow technologies for geospatial data processing in distributed and heterogeneous computation environments. The Web service technology provides platform independence and greater interoperability; the scientific workflow technology provides software reuse and visual programming. We demonstrate the feasibility of the proposed approach using storm dynamics tracking from weather radar data as an example. Technical details of algorithm development, Web services publishing, workflow composition and execution for the demonstrative example are discussed.

1 Introduction

Tracking dynamics of geospatial phenomena has wide applications in geosciences and environmental sciences, such as storm tracking and pollution diffusion monitoring. There have been considerable research efforts [1][2] in modeling geospatial dynamics in Geographical Information Systems (GIS). While there are several proposed methods and their prototype implementations (e.g. [3][4]), few of them are capable of being operational in distributed and heterogeneous computational environments. Building the prototypes on top of standalone commercial GIS systems (such as ESRI ArcGIS) might be one of the main reasons. First of all, data obtained from sensors might be stored in different locations and have to go through different preprocessing stages before they can be analyzed in GIS. The preprocessing modules could reside on different machines with different computation environments (such as operating systems and programming languages). Invoking the diverse preprocessing modules is beyond the scope of a single GIS system. However, existing prototypes generally assume that all the data are accessible to GIS system at the time of running tracking models. Second, fully automatic dynamics tracking is rare in real applications and domain experts' interventions are often required, such as changing algorithms/parameters, adding/removing visualization components. However, modifying the GIS scripts or programs of complex tracking models is often beyond the programming skills of domain users.

In this study, we propose a new approach to tracking geospatial dynamics in distributed and heterogeneous computation environments by using Grid/Web Services and Scientific Workflow technologies. First, both geospatial data and processing components are published as Web Services. W3C Web Services technology provides a published interface to data or programs by using Web Service description Language (WSDL, [5]). WSDL is an XML based language which is platform independent and can be understood by human and processed by machine. Second, Kepler scientific workflow system is used as the integrated environment to chain the tracking Web services and other related Web services into executable scientific workflows. Kepler allows user to drag-and-drop computation components (such as database queries and algorithms) into a canvas and link the interfaces of the components to compose a workflow, which essentially provides a visual programming environment. The proposed approach overcomes the problems of using a standalone GIS for tracking geospatial dynamics, i.e., the Web Service provides interoperability among distributed and heterogeneous computation environments; the scientific workflow allows to compose and execute the tracking pipeline in a convenient manor. In addition, our approach is friendly to software reuse. The implemented Web services can be used for other purposes, for examples, applications in [6], rather than solely for tracking. Finally, while powerful commercial GIS packages are expensive, we implement the Web services using open source software packages and Kepler scientific workflow system itself is open source. We demonstrate our approach by tracking storm dynamics from WSR-88D (Weather Surveillance Radar - 88 Doppler) images.

The rest of this paper is arranged as follows. Section 2 presents the three components of a contour tree-based tracking method, namely contour tracing, contour tree generation and contour tree matching. Section 3 provides technical details of publishing the components as Web services. Section 4 demonstrates the feasibility of the proposed approach using a running example. Section 5 briefly reviews related works and discusses the proposed approach. Finally section 6 is the summary and future work directions.

2 Contour Tree Based Dynamics Tracking Method

The most available data source for tracking dynamics of geospatial phenomena may be remotely sensed images, such as satellite data and Doppler radar data. From an image processing perspective, tracking dynamics of geospatial phenomena can be classified into three categories, namely region-based, pixel-based and block-based [7]. The region-based methods are more relevant to vector GIS that identifies groups of pixels with specific characteristics (regions) from a pair of images and then match these regions to track the dynamics.

In this study, we propose a contour tree-based method to link the regions from an image pair. The proposed method is based on the observation that many geospatial phenomena are concentrically distributed, such as elevations, pollution diffusions and storms. These geospatial phenomena are often represented by contours in traditional cartographic maps. A contour tree represents the topological relationship among the contours. We use the notation proposed in [8] to represent a contour tree: an edge in a contour tree symbolizes a contour line while a node in a contour line symbolize the inter-contour regions. The idea behind the new proposed tracking method is that, while many geographic phenomena may change their locations and shapes dramatically over time, their topologies might be relatively static. Tracking dynamics based on topology consistency over time might provide an interesting alternative to traditional overlap/distance based tracking.

Fig. 1 shows a portion of a WSR-88 radar image, its contours at the contour values of 20, 30, 40 and 50 dbZ using VisAD [9] and the corresponding contour tree. To handle the split and merge cases, i.e., contour trees become a forest or vice versa, we use an artificial node *R* as the root for both a contour tree and a related forest as shown in Fig. 1. Conceptually, *R* represents the whole study area.

We omit the preprocessing (such as smoothing) of image data since it is application specific. The contour tree based dynamics tracking method can be outlined as three steps. First, contours of predefined value sets (or *iso-values*) are generated. We transform the closed contours into polygons (contour regions) while discard open contours since the open contours are located at the borders of the study area and the geospatial phenomena they represent are generally less interesting. Second, the polygons are assembled into a contour tree based on polygon containment. Finally the contour trees generated at two successive times are matched by using an unordered tree matching algorithm presented in [10]. Dynamics of corresponding contour regions can be computed using a variety of criteria, such as distance, velocity, direction. In this section, we present the algorithms used in the three steps respectively. The algorithms are implemented in Java or C++ and the implementations are published as Web services (which are discussed in Section 3). Third party packages are used when necessary.

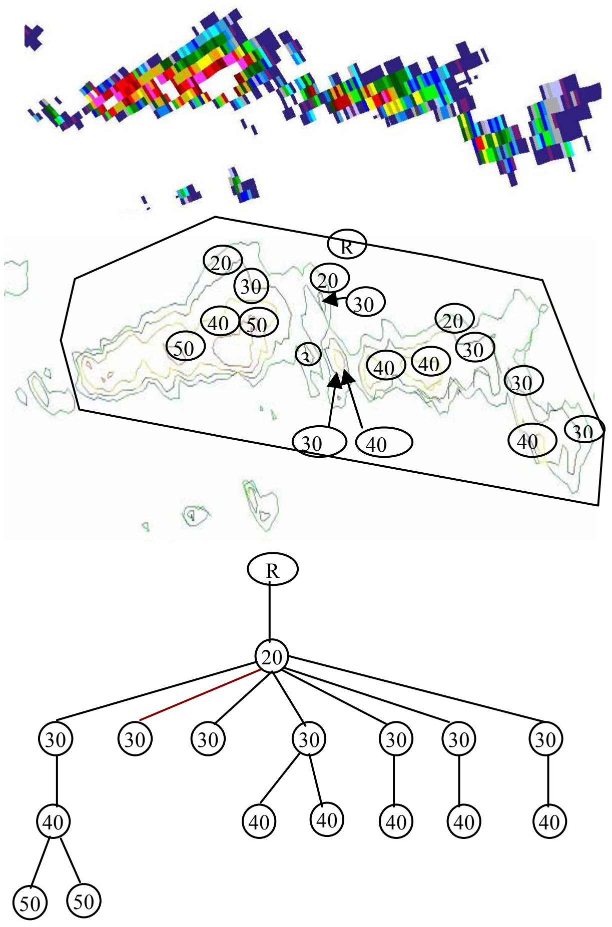


Fig. 1 Illustration of Contour Tree using WSR-88 Radar Data

2.1 Contour Tracing

Contour tracing is a well-studied field and several open source packages are available such as VisAD in Java (Visualization for Algorithm Development, http://www.ssec.wisc.edu/~billh/visad.html) and VTK in C++ (The Visualization ToolKit, http://public.kitware.com/VTK/). However, they are mostly for visualization purposes. The marching square algorithm or its variants used in these packages only generates contour line segments but does not put the line segments into a sequence that can be used to generate a contour region (polygon). In this study, we implement the algorithm presented in [11] to trace all the closed contours of predefined contour values and transform them into polygons. These polygons are then used to generate a contour tree.

2.2 Contour Tree Generation

We use the following Java class data structure to represent a contour tree:

```
class CTree
{
    String _label;
    int _lev;
    CTree[] _children;
    Geometry _g;

    public void AttachNode
            (int lev,Vector v, String label);
    public void GetNodes
            (int lev,Vector ret);
}
```

where Geometry is defined in the open source JTS (Java Topology Suit) package (http://www.vividsolutions.com/jts/JTSHome.htm) and used to represent a contour region. Vector *v* used in the two public methods (*AttachNode* and *GetNodes*) stands for the collections of contour regions for a particular contour value. Method *AttachNode* attaches the contour regions (stored in Vector *v*) to a contour tree node by creating child nodes based on polygon containment. Function *GetNodes* is used to retrieve all the nodes of a contour tree at a particular level (*lev*) and stored the result in a Vector (*ret*). The process of generating a contour tree is as follows. The algorithm first puts the contour regions into a set of vectors according to their contour values. The algorithm then creates the root node of a contour tree whose label is "R" and set its *_g* (Geometry) to *null*. The root node is used to represent the whole study area. For each of the contour values (also the level in the contour tree, i.e., *lev*), the algorithm retrieves all the nodes at the level. For each of the nodes (serve as parent nodes), the algorithm calls *AttachNode* and creates child nodes under the parent node if there are contour regions in the vector corresponding to the contour value are contained by the contour region of the parent node. Note that unlike conventional tree building algorithms which are often recursive, the algorithm is sequential and requires less computational resources. While polygon containment test is expensive, the efficiency can be improved by first testing whether the Minimum Bounding Rectangles (MBRs) of the two polygons overlap in JTS. Other optimization techniques such as using spatial indexing can be applied to further improve the efficiency but are beyond the scope of this study. A contour tree is built when the node attachment process finishes.

2.3 Contour Tree Matching

We use the ordered tree matching algorithm presented in [10] which is based on the concept of edit distance. The edit distance between two trees is defined by considering the minimum cost edit operations sequence that transform one tree to the other. There are three types of operations for ordered trees: inserting, deleting and re-labeling. Each of the operations is assigned a cost and the minimum cost to transform one tree to the other can be computed efficiently by adopting the dynamic programming strategy. The matching nodes in the two trees can be identified as a byproduct of computing tree distance. We modify the C code developed by the authors of [10] to find the corresponding nodes in a contour tree pair. An example will be given to illustrate the method in Section 4.

3 Publishing Tracking Method as Web Services

As discussed previously, the Web Services technology provides interoperability among distributed and heterogeneous computation environments. Each of the three components in the tracking methods are published as Web services. We also publish data preprocessing module by combining APIs in a weather radar data processing package called WDSS-II (http://www.wdssii.org/) as a Web service (see below). To be in accordance with the open source strategy, we use Apache Tomcat as Web server and Axis Java as Web service engine. For the modules written in Java, the publishing is relatively simple. What we need to do is to put the Java classes into specific directories under the Web server and Web service engine directories, compose a deployment file and register the file with Axis engine. For the modules written in languages other than Java (C/C++ in our prototype), we need to write a glue class to wrap the C/C++ modules using Java Native Invocation (JNI) technology.

The Warning Decision Support System - Integrated Information (WDSS-II, http://www.wdssii.org/) is the second generation of a suite of algorithms and displays for severe weather analysis, warnings and forecasting. WDSS-II is developed under Linux. Its algorithms and display modules are compiled as Linux shared libraries. The data preprocessing module used in our prototype utilizes only a small portion of WDSS-II functions. A shared library is created and used to delegate the requests to WDSS-II libraries and can be called in Java using JNI technology. To simplify the Web service interface, all the input parameters are combined into a string and the output is also a string representing the URL of the image after preprocessing. The interpretation of the input string is left to the Web service implementation. Parameters in the input string can include time, Area of Interests (AOI) and preprocessing methods. Default parameter values are allowed and will be translated into real values in runtime. An example is given in Section 4.

We also apply the similar strategy in publishing the three components in the tracking method, i.e., both the inputs and outputs of the Web services are string type. For Contour Tracing Web service, the input is the URL of preprocessed image and the output is the contours in GML (Geographical Markup Language, [12]) format. For the Contour Tree Generation Web service, the input is a GML string of a collection of polygons and the output is the concatenation of blanks and labels of the contour tree nodes by pre-order traversal of the nodes. The Contour Tree Matching Web service accepts two strings representing two contour trees and output a string specifying the correspondence among the nodes. One advantage of using the string type in Web services is that string is a primitive XML type, it is widely supported and no special serialization/deserialization is required. In addition, the output of a Web service in string data type can be viewed in any text editor conveniently. However, a caution must be kept in mind that the strategy essentially shifts the responsibility of ensuring the compatibilities among the inputs and outputs of Web services to the Web service endpoints.

Once the preprocessing and tracking Web services are implemented and deployed, they can be used as the building blocks for tracking dynamics of geospatial phenomena. The scientific workflow technology provides a drag-and-drop environment to compose workflows visually and execute the composed workflow in a batch or interactive manor. We use Kepler scientific workflow system (Kepler (http://www.kepler-project.org/, [13]) in this study. The feasibility of the proposed approach is demonstrated through an example in the next section.

4 Experiment

4.1 Doppler Weather Radar Data and Preprocessing

The data we use to test the proposed approach is generated by KFWS WSR-88D located at Dallas/Fort Worth area, one of the ~160 radars in the US national radar network. The radar was operated under Volume Coverage Pattern (VCP) 21 operation mode which sweeps 9 elevation angles in 6 minutes (as shown in Fig. 2). The data was obtained in May 4th, 1995 around 12pm. One of the three Level II products called Reflectivity is used to track the storm dynamics. We use two images of the lowest elevation angle (0.5 degree) as an example to track the storm dynamics which are obtained at 12:11:02 and 12:16:52, respectively. The reflectivity data obtained at each time is stored as a NetCDF [14] file in compressed format to save storage space.

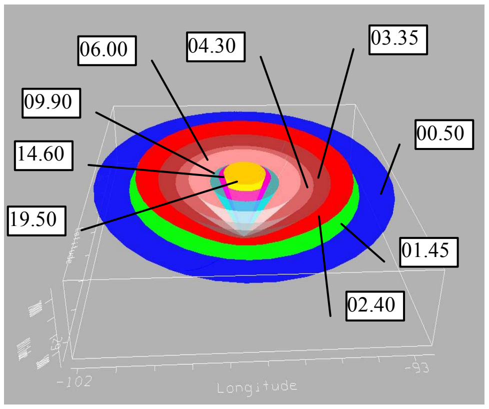


Fig. 2 Three Dimensional Illustration of WSR-88 Volume Scan under VCP 21 Mode

The preprocessing Web service accepts the following parameter string: "time=, x1=, y1=, x2=, y2=, filter =", where *time* is when a radar image is obtained, *x1, y1, x2, y2* are the coordinates in the image to obtain a subset image and *filter* is the filtering matrix to smooth the subset image. The final image after preprocessing is put in a data directory under Tomcat Web server's root directory where it can be mapped to a URL and accessed from outside. The preprocessing Web service returns the URL of an image after preprocessing.

4.2 Workflow Composition

Composing a workflow for storm tracking is straightforward using Kepler scientific workflow system once the related Web services are published and deployed. User can drag and drop the Web Services actors into Kepler graphic editor canvas as many times as necessary. After each of the Web Services actors is instantiated, users can connect the input/output ports of the actors intuitively (c.f. Fig. 3). In the experiment workflow, two copies of data preprocessing Web service, two copies of contour tracing Web service, two copies of contour tree generation Web service and one copy of contour tree matching Web service are used. For the Web service that multiple copies are used, each copy can be deployed in a separate machine to facilitate parallel computing and avoid single point failure problem as well.

Fig. 3 shows Kepler workflow composition environment and the contour tree based storm tracking workflow. Note that a diamond symbol in the workflow represents a relation. An output port can be connected to a relation and a relation can be connected to multiple input ports of different actors. In addition to using the output as the inputs to multiple subsequent actors, it allows to visualize the output of an actor in different formats by connecting it to different visualization actors. Currently we support viewing text information and graphically visualizing image and GML data. Visualizing contour trees and contour tree matching results are under development.

Kepler essentially provides a visual programming environment. Users can change the functions of the Web Services actors by changing the URLs of the endpoints. They can add as many data transformation and visualization actors to the workflow as possible provided that they are connected and compatible. Users can also change the parameters of the workflow (such as those labeled as "String constants" at the left part of Fig. 3) and watch the differences between the outputs. The workflow composition approach thus differs from traditional scripting or programming approach significantly. We believe the proposed approach is more efficient and effective to domain users with limited background in GIS and programming.

4.3 Workflow Execution

Once the workflow is composed, users can execute it in Kepler scientific workflow environment by hitting the triangle icon located at the top of Kepler window. Kepler allows users to execute a workflow in batch mode or interactive step-by-step mode. At any time during the execution of a workflow, users can stop and resume the execution and watch the intermediate results.

The images, contours, resulting contour trees and their matching result from the workflow execution are shown in Fig. 4. We can see that the tree matching algorithm correctly matches the corresponding contour regions. The algorithm also identifies three contour regions are disappeared from time T_1 (12:11:02) to time T_2 (12:16:52), one is contour region with contour value 20 dbZ located at the top of the study area and two contour regions with contour values 20 dbZ and 30 dbZ located at the lower-right of the study area.

Note that the 30dbZ contour region is actually nested in the 20dbZ contour region located at the lower-right of the study area and their disappearances could indicate a small sub-storm is vanishing. Another observation is that while the shapes of the contour regions of the main storm (located at the lower-left part of the study area) change a lot over time (especially the highest 40dbZ contour region), their topological relationships do not change. Tracking dynamics of geospatial phenomena based on topological consistency over time captures human perceptions of geospatial phenomena and can be used as one of the reliable criteria for tacking. On the other hand, the block-based matching methods [7], when applied to the 40 dbZ regions, may result in poor matching scores and fail to track the dynamics correctly since the distributions of pixel values in the regions change dramatically over time.

5 Related Work and Discussions

The NSF large Information Technology Research (ITR) project Linked Environment for Sharing Atmospheric Information (LEAD, http://lead.ou.edu/; [15][16]) is the most related work to ours since detecting weather event is also one of the functional categories in the project which is closely related to the example of tracking dynamics of geospatial phenomena used in this study. The project's proposed architecture shares many similarities with our approach regarding to service-oriented architecture and using workflow technology. While we share the similar vision on wrapping user applications as Web Services for building grid applications that allows domain scientists to access and run these applications without becoming experts on Grid technology [15], there are several differences in terms of technology adopted for chaining Web services into applications. To the best of our knowledge, currently no visual workflow composition and execution environment is provided in LEAD portal. LEAD either adopts Business Processing Execution Language (BPEL) or Open GCE Runtime Engine (OGRE) scripting language to specify a workflow [16][15]. Users need to either use a generic BPEL composition environment to specify a workflow or write an OGRE script in a text editor and then switch to a workflow execute environment to execute the workflow. In contrast, the integrated environment for workflow composition and execution provided by Kepler seems to be more convenient from a domain user's perspective.

The Open Geospatial Consortium (OGC) has published a discussion paper where the Web Services Architecture (WSA) has been proposed [17] and business workflow description languages are suggested for composing geospatial data processing workflows. Compared to business workflow, however, scientific workflows can be data intensive, computation-intensive and visualization intensive which distinguish them from business workflows that are generally transaction-oriented and event-driven [13]. The effectiveness of applying business workflow technologies to solve scientific problems is unclear. In the scientific workflow domain, there are several open source scientific workflow systems available, such as Taverna [18], Triana [19] and Kepler [13]. These systems support composing and execution of scientific workflows in distributed/Grid computing environments. However, we are not aware of previous work extending these systems for tracking geospatial dynamics. Finally we have proposed a framework for distributed geospatial data processing using scientific workflow techniques [20] and presented example studies in different domains, e.g., species distribution predictions [21].

With the emerging Web Services technology, we see a trend of paradigm shift of geospatial processing. Many traditional geospatial models that are built on specific GIS platforms can be decomposed into smaller components and then be transformed into Web services. These Web services can be used as the building blocks to visually compose scientific workflows in XML format that are interoperable among machines and are human readable as well. The Web services, as the building blocks of scientific workflows, can be implemented using any scripting/programming languages and commercial or open source GIS systems/spatial databases. The architecture provides much more flexibilities than relying on a single GIS system for geospatial modeling. This study is the first step towards this direction.

In the demonstrative example, while all the components are based on open source packages, if desired, we can replace the contour tracing Web service by implementing a Web service on top of ArcGIS Spatial Analyst (http://www.esri.com/software/arcgis/extensions/spatialanalyst/index.html) and the workflow should still work without any problem. On the other hand, it can be very difficult if not impossible to implement the contour tree generation and matching algorithms using scripting languages within ArcGIS. Although invoking remote processes is partially supported in ArcGIS through RPC or using third party libraries, the Web Services technology is more interoperable, easier to use and widely accepted.

The capabilities of the Web Services and the Scientific Workflow technologies that enable distributed computing in heterogeneous environment do not come without prices. The Web Service is essentially another layer on top of existing functional components which means extra communication and computation costs, such as data serialization/deserialization costs in SOAP protocol stack, invocation cost between the Web server and the functional component at local machines. There are also costs for workflow scheduling and mapping data tokens between ports of workflow components and ports of Web services. Nevertheless, the costs are well paid off by the effectiveness and convenience that the technologies have brought to domain users in terms of workflow design, revision and execution.

6 Summary and Future Work Directions

In this study we propose to use Web Services and scientific workflow technologies to track dynamics of geospatial phenomena in distributed and heterogeneous computation environments. We use tracking storm dynamics from weather radar images to demonstrate the feasibility of the proposed approach. A novel contour-tree matching based method is proposed to track the concentric-distributed geospatial phenomena which consists three components: contour tracing, contour tree generation and contour tree matching. The three components are implemented using open source packages and published as Web services using Apache Axis Java. Kepler scientific workflow is used to compose the services into an executive scientific workflow and execute the workflow either in a batch or interactive mode. The results demonstrate the feasibility of the proposed approach. While the contour tree matching based method for geospatial dynamics only works for concentric-distributed geospatial phenomena, the framework allows developing other domain-specific tracking methods and publishing them as Web services as the building blocks for scientific workflows.

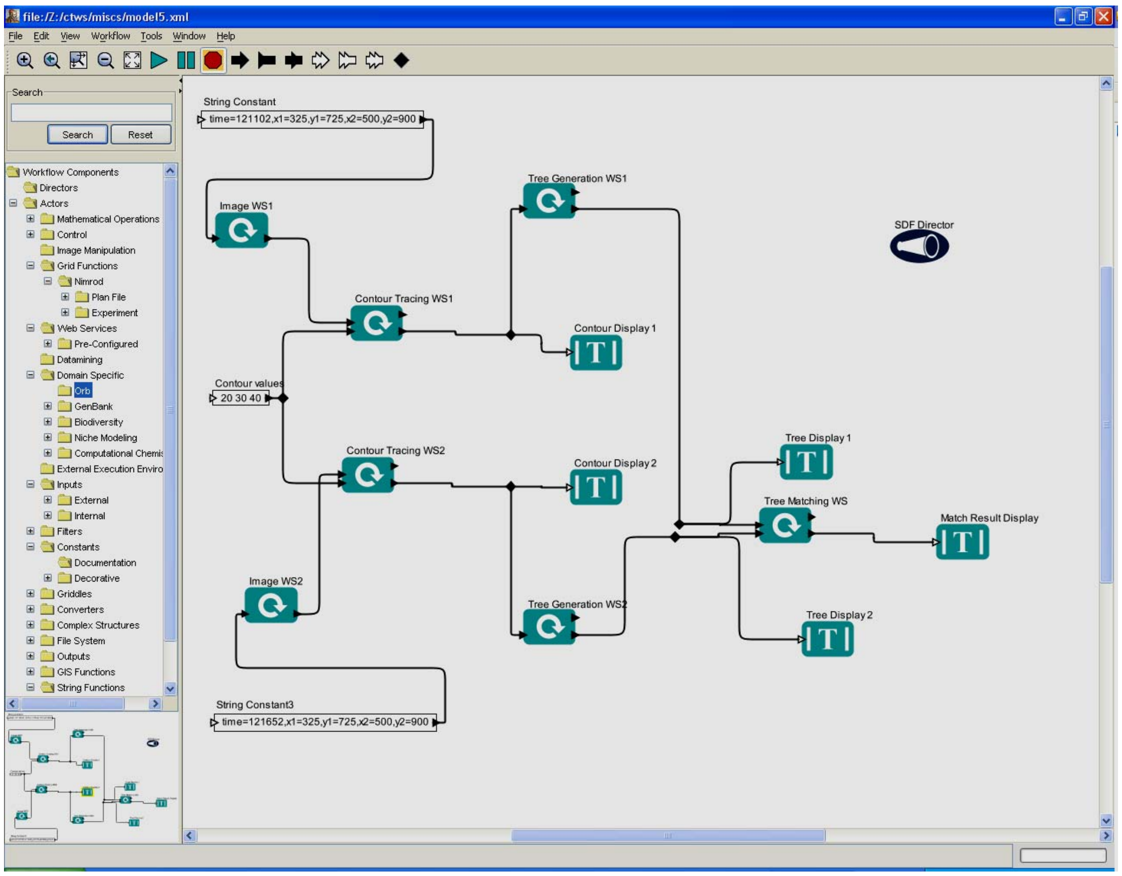
Currently the demonstrative example only allows finding the correspondences among the contour regions at two time points. While it is the core part of tracking dynamics over a period, there are many technical issues remain to be solved, such as adjusting the matching using past matching information and combining geometric matching with the topological matching. Furthermore, while we have developed several visualization actors to visualize raster and vector GIS data, we plan to develop actors to visualize contour trees and tree matching results.

Acknowledgements

This work is supported in part by DARPA grant #N00017-03-1-0090. Part of the work was done when the author was with CIMMS at the University of Oklahoma and NSSL/NOAA. Kepler includes contributors from SEEK, SDM Center, Ptolemy II and Geon, supported by NSF ITRs 022567 (SEEK), 0225673 (GEON), DOE DE-FC02-01ER25486 (SciDAC/SDM), and DARPA F33615-00-C-1703 (Ptolemy).

References

1. Egenhofer, M. J., Golledge, R.G., eds, 1998, Spatial and temporal reasoning in geographic information systems, Oxford University Press, New York.
2. Hornsby, K., Egenhofer, M. J., 2000, Identity-Based Change: A Foundation for Spatio-Temporal Knowledge Representation, International Journal of Geographical Information Science, 14 (3), 207-224.
3. Cheng, T., Molenaar, M., 1999, Diachronic Analysis of Fuzzy Objects, GeoInformatica 3(4), 337-355
4. Yuan, M., 2001, Representing Complex Geographic Phenomena in GIS, Cartography and Geographic Information Science, 28(2), 83-96.
5. Christensen, E., Curbera, F., Meredith, G., Weerawarana., S., 2001, Web Services Description Language (WSDL) 1.1, http://www.w3.org/TR/wsdl.
6. Xie, H.; Zhou, X., Vivoni, E. R., Hendrickx, J. M. H., Small, E. E., 2005, GIS-based NEXRAD Stage III precipitation database: automated approaches for data processing and visualization, Computers & 31(1), 65-76.
7. Feidas, H., Cartalis, C., 2005, Application of an automated cloud-tracking algorithm on satellite imagery for tracking and monitoring small mesoscale convective cloud systems, International Journal of Remote Sensing, 26(8), 1677-1698.
8. Wu, C., 1993, Automated Identification Of Scanned Contour Information For Digital Elevation Models, Ph.D. Dissertation, University Of Maine.
9. Hibbard, W., Rueden, C., Emmerson, S., Rink, T., Glowacki, D., Whittaker, T., Murray, D., Fulker, D., and Anderson, J., 2005, Java distributed components for numerical visualization in VisAD. Communication of the ACM, 48(3), 98-104.
10. Shasha, D., Zhang, K. , 1997, Approximate Tree Pattern Matching, in Apostolico, A., Galil Z., editors, Pattern Matching Algorithms, chapter 14, Oxford University Press, New York.
11. Sutcliffe, D.C., 1980, Contouring over Rectangular and Skewed Rectangular Grids - an Introduction, in Brodlie, K. W. editor, Mathematical Methods in Computer Graphics and Design, Academic Press, New York, 147 pp.
12. Lake, R., Burggraf, D., Trninic, D. Rae, L., Geography Mark-Up Language: Foundation for the Geo-Web, Wiley, 406 pp.
13. Ludäscher, B., et al, 2005, Scientific Workflow Management and the Kepler System, Concurrency and Computation: Practice & Experience, Special Issue on Scientific Workflows, to appear.
14. Rew, R. K., Davis, G. P., Emmerson, S., Davies, H., 1997, NetCDF User's Guide for C, An Interface for Data Access, Version 3.
15. Gannon, D., Alameda, J., et al., 2005, Building grid portal applications from a web service component architecture, Proceedings of the IEEE, 93(3), 551-563.
16. Plale, B., Gannon, D., et al., 2005, Towards Dynamically Adaptive Weather Analysis and Forecasting in LEAD, International Conference on Computational Science (2) 2005: 624-631.
17. Lieberman, J., 2003, OpenGIS Web Service Architecture(WSA), OGC Discussion Paper, http://www.opengeospatial.org/docs/03-025.pdf.
18. Oinn, T., Addis, M., Ferris, J., Marvin, D., Senger, M., Greenwood, M., Carver T., Glover, K., Pocock, M. R., Wipat, A., Li., P., 2004, Taverna: A tool for the composition and enactment of bioinformatics workflows. Bioinformatics Journal, 20(17), 3045-3054.
19. Taylor, I., Wang, I., Shields, M., and Majithia, S., 2005, Distributed computing with Triana on the Grid, Concurr. Comput. : Pract. Exper. 17(9), 1197-1214.
20. Jaeger, E., Altintas, I., Zhang, J., Ludäscher, B., Pennington, D. & Michener, W. (2005). A Scientific Workflow Approach to Distributed Geospatial Data Processing using Web Services. SSDBM 2005, 87-90.
21. Zhang, J., Pennington, D. D. & Michener, W. K. (2005). Using web services and scientific workflow for species distribution prediction modeling. Advances in Web-Age Information Management, Proceedings, 3739, 610-617.



String Constant: time=121102,x1=325,y1=725,x2=500,y2=900

Contour values: 20 30 40

String Constant3: time=121652,x1=325,y1=725,x2=500,y2=900

Fig. 3 Workflow Composition in Kepler for Tracking Strom Dynamics over Time

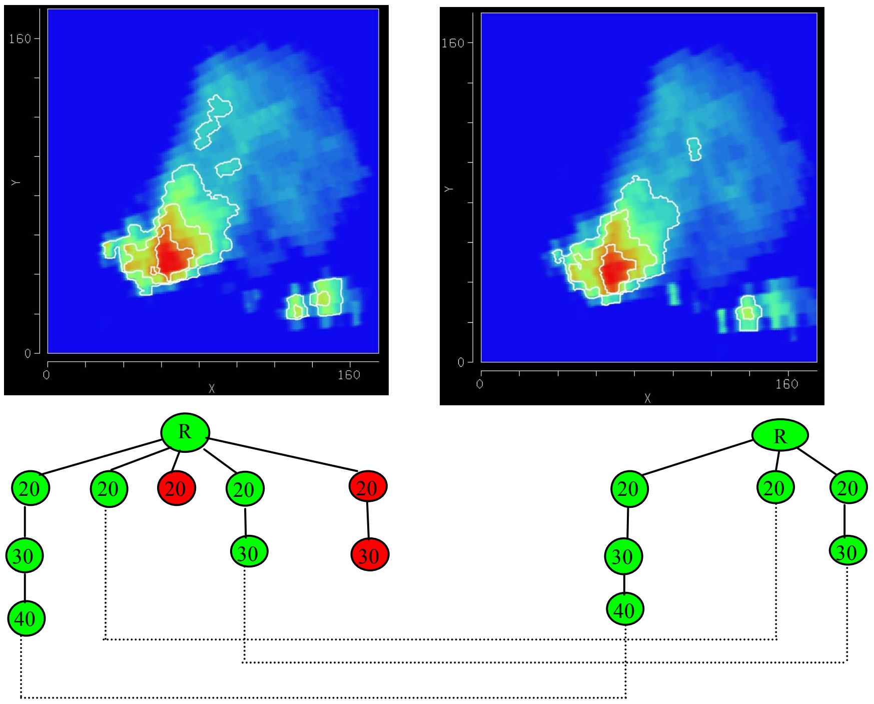


Fig. 4 Images, Contour Regions and Contour Tree Matching Result for the Example