THE INTEGRATON OF GRAPH-BASED KNOWLEDGE DISCOVERY WITH IMAGE SEGMENTATION HIERARCHIES FOR DATA ANALYSIS, DATA MINING AND KNOWLEDGE DISCOVERY

James C. Tilton[*], *Diane J. Cook*[+], *and Nikhil Ketkar*[+]

[*]Code 606.3, NASA Goddard Space Flight Center, Greenbelt, MD 20771, USA
[+]Electrical Engineering and Computer Science, Washington State University, Pullman, WA 99164, USA

ABSTRACT

Currently available pixel-based image analysis techniques do not effectively extract the information content from the increasingly available high spatial resolution remotely sensed imagery data. We are exploring an approach to object-based image analysis in which hierarchical image segmentations provided by the Recursive Hierarchical Segmentation (RHSEG) software are analyzed by the Subdue graph-based knowledge-discovery system. In this paper we discuss our initial approach to representing the RHSEG-produced hierarchical image segmentations in a graphical form understandable by Subdue, and discuss results from real and simulated data.

Index Terms— Object-based image analysis, hierarchical image segmentation, graph-based analysis, knowledge discovery.

1. INTRODUCTION

Most available image analysis techniques do not effectively extract the information content from the increasingly available high spatial resolution remotely sensed imagery data. High spatial resolution imagery can resolve individual objects such as man-made structures and even individual large trees. However, several studies have shown that most currently available pixel-based analysis techniques do not perform well on this type of data. We are exploring an alternative object-based image analysis approach.

Object-based image analysis relies on image segmentation as a starting point. The quality of this initial image segmentation strongly influences the effectiveness of the ensuing object-based analysis. We have chosen to use the Recursive Hierarchical Segmentation (RHSEG) software [1] developed internally at NASA GSFC as our image segmentation approach. RHSEG is an excellent choice because of three key factors: (i) RHSEG produces image segmentations with high spatial fidelity, (ii) RHSEG automatically groups spatially connected region objects into region classes, and (iii) RHSEG automatically produces a hierarchical set of image segmentations.

An approach capable of discovering patterns in the segmented image data is needed for analyzing a hierarchical set of image segmentations, such as the Subdue graph-based knowledge-discovery software [2]. Subdue was developed by a team at Washington State University, and is designed to discover patterns in graph-based structural databases. Subdue has been successfully applied in a number of areas, including bioinformatics, web structure mining, counter-terrorism, social network analysis, aviation and geology [3-5]. We hypothesize that the capabilities of RHSEG and Subdue can be combined to provide insightful analysis of remotely sensed data.

We have made some initial steps in translating image segmentations into relational graphs for analysis by Subdue, and achieved some limited data analysis success. The grouping of region objects into regions classes, as provided by RHSEG, has proved important in this translation.

2. RHSEG

RHSEG is an approximation to the HSEG hierarchical image segmentation algorithm. HSEG is a hybrid of hierarchical step-wise optimization (HSWO) and constrained spectral clustering that produces a hierarchical set of image segmentations. HSWO is an iterative approach to region growing segmentation in which the optimal image segmentation is found at N_R region objects, given a segmentation at N_R+1 region objects [6].

HSWO produces hierarchical segmentations that are useful in many applications. However, with the addition of constrained spectral clustering, HSEG produces segmentations that capture with greater fidelity the spatial detail of the segmented images, while describing the image data compactly in terms of region classes, which are groups of region objects[†].

HSEG's addition of constrained spectral clustering makes it a computationally intensive algorithm for all but the smallest of images. To counteract this, a

[†] A region object is a set of spatially contiguous image pixels. A region class is a group of one or more spatially disjoint region objects.

computationally efficient recursive approximation of HSEG (called RHSEG) was devised. Further improvements in processing speed are obtained through a parallel implementation of RHSEG. The HSEG and RHSEG algorithms, along with a coarse-grained parallel implementation of RHSEG on MIMD computer clusters, are described in detail in [7,8].

HSEG controls the relative priority given to merges of spatially adjacent regions versus merges of spatially non-adjacent regions (spectral clustering) through the input parameter S_{wght}. This parameter, which can vary from 0.0 to 1.0, controls the relative importance of spatially adjacent and spatially non-adjacent region merges. When S_{wght} = 0.0, spatially non-adjacent region merges are not allowed and HSEG becomes equivalent to HSWO. With S_{wght} = 1.0, merges between spatially adjacent and spatially non-adjacent regions are given equal priority. For values of S_{wght} between 0.0 and 1.0, spatially adjacent merges are given priority over spatially non-adjacent merges by a factor of $1.0/S_{wght}$. Appropriate values for S_{wght} depend on the application, but usually range from 0.1 to 1.0.

A variety of region dissimilarity criteria may be used with HSWO and HSEG (see [1] for a complete list). In our work with Subdue, we found the "square root of band sum mean squared error" ($BSME^{1/2}$) criterion to be most appropriate. This criterion is defined as:

$$d^{1/2}_{BSMSE}(X_i, X_j) = \left[\frac{n_i n_j}{(n_i + n_j)} \sum_{b=1}^{B} (\mu_{ib} - \mu_{jb})^2 \right]^{1/2}, \quad (1)$$

where n_i (n_j) is the number of pixel in region X_i (X_j), and μ_{ib} (μ_{jb}) is the mean of region X_i (X_j) for spectral band b.

The region object classification provided by RHSEG is needed for input to Subdue because the region class provides a node label which Subdue needs to determine interesting associations between neighboring region objects.

3. SUBDUE

Numerous approaches have been developed for discovering concepts in linear, attribute-value databases [9]. Although much of the data collected today has an explicit or implicit structural component (e.g., spatial or temporal), only recently have discovery systems been designed to handle this type of data. Prevoius data mining research focused on algorithms to discover sets of attributes that discriminate data entities into classes, such as shopping trends for a particular demographic group. These approaches experience difficulty when key concepts involve relationships between the data points. In contrast, structural data mining techniques discover patterns consisting of complex relationships between entities.

In [2], a method was introduced for discovering substructures in structural databases implemented in the Subdue system. In contrast with alternative approaches, Subdue is devised for general-purpose automated discovery, concept learning, and hierarchical clustering. Hence, the method can be applied to many structural domains.

Subdue accepts as input directed or undirected graphs with labeled vertices (nodes) and edges (links), and outputs graphs representing the discovered pattern or learned concept. Formally, Subdue uses a labeled graph G = (V,E,L) as both input and output, where $V = \{v_1, v_2, \ldots, v_n\}$ is a set of vertices, $E = \{(v_i, v_j) \mid v_i, v_j \in V\}$ is a set of edges, and L is a set of labels that can appear on vertices and edges. The graph G can contain directed edges, undirected edges, self-edges, and multi-edges. The input to Subdue can consist of one large graph or a collection of individual graphs, and in the case of supervised learning, the individual graphs are classified as positive or negative examples.

As an unsupervised algorithm, Subdue searches for a substructure, or subgraph of the input graph, that best compresses the input graph. Subdue uses a variant of beam search for its main search algorithm. A substructure in Subdue consists of a subgraph definition and all its occurrences throughout the graph.

Subdue's discovery algorithm uses a polynomial-time beam search. The initial state of the search is the set of all uniquely-labeled vertices. Search progresses by applying an "ExtendSubstructure" operator to each substructure in the current state. As its name suggests, it extends a substructure in all possible ways by a single edge and a vertex, or by only a single edge if both vertices are already in the subgraph. The resulting new substructures are ordered based on their compression (or sometimes referred to as value) as calculated using the MDL principle described below, and the top substructures (as determined by the beam) remain on the queue for further expansion.

Search terminates upon reaching a limit on the number of substructures extended, or upon exhaustion of the search space. Once the search terminates and Subdue returns the list of best substructures, the graph can be compressed using the best substructure. The compression procedure replaces all instances of the substructure in the input graph by single vertices, which represent the substructure definition. Incoming and outgoing edges to and from the replaced instances will point to or originate from the new vertex that represents the instance. The Subdue algorithm can be iterated invoked again on this compressed graph.

Subdue's search is guided by the Minimum Description Length (MDL) principle formalized in (2), where $DL(S)$ is the description length of substructure S being evaluated, $DL(G|S)$ is the description length of the graph as compressed by the substructure, and $DL(G)$ is the description length of the original graph [10]. The best substructure is the one that minimizes this compression ratio:

$$Compression = \frac{DL(S) + DL(G \mid S)}{DL(G)} \quad (2)$$

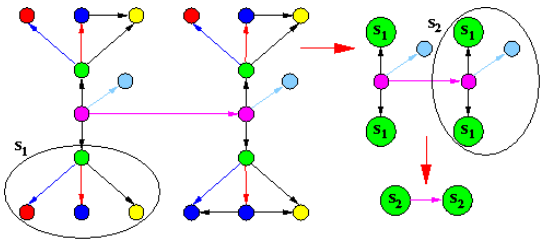

Fig. 1. An example of Subdue's substructure discovery capability. The figure shows the discovered pattern (S_1) from the original graph, the substructure found during the second iteration (S_2), and the final graph compressed using substructures S_1 and S_2.

As an example, Fig. 1 shows patterns that Subdue discovers in an example input graph and a compressed version of the graph.

To allow slight variations between instances of a discovered pattern (as is the case in Fig. 1), Subdue applies an inexact graph match between the substructure definition and potential instances. Because instances of a substructure can appear in different forms throughout the database, Subdue computes the graph edit distance between two graphs and considers the substructure instance to be a match if the distance is less than a pre-defined threshold (0 for exact matches) [10].

4. CONVERSION OF RHSEG OUTPUT TO GRAPH FORMAT

Before significant experiments could be performed on image data sets of any significant size, a computer program had to be devised to convert outputs from RHSEG to the graphical representation required by Subdue. An existing RHSEG utility program, called "feature_extract", was augmented to add an option to output a Subdue-compatible input graph. Eventually this program was further augmented to order the RHSEG region objects by size (largest to smallest) and drop out from consideration region objects smaller than a specified size (number of pixels). The only information conveyed from the RHSEG segmentation output to the Subdue input graph was the region class label for each region object and whether or not a region object was spatially adjacent (linked) to another region object. This was done for just one selected level from the segmentation hierarchy. The Subdue input graph consisted of a list of graph vertices (region objects) labeled by the region class label and a list of undirected edges specifying which region objects were linked to what other region object.

5. INITIAL RESULTS

Initial experiments combining RHSEG and Subdue that were performed on Ikonos and SAR data produced results that were tantalizing but inconclusive. Because of this we decided to put together a simulated segmentation result to test the behavior of Subdue run with different parameter settings. The simulated segmentation, displayed in Fig. 2a, combines idealized segmentations of a residential area (most of the lower left quadrant), an apartment complex (most of the upper left quadrant), an industrial park (the upper right quadrant) and recreational parks (inserted in the apartment complex and residential quadrants) with a section of an actual segmentation of SAR data (lower right quadrant).

We performed our initial run with Subdue with the same parameters used in our previous runs on Ikonos and SAR data. In this run, the most significant subgraph found by Subdue covered portions of three land use classes: apartment complex, residential and parks. The second most significant subgraph covered most of the SAR quadrant. The third most significant subgraph covered portions of two land use classes: residential and parks.

What we noticed about these most significant subgraphs was that they were all two-vertex subgraphs. The first was a grass-roof subgraph, the second was a subgraph linking the two SAR classes, and the third was a grass-trees subgraph. The second thing we noticed was that each region object participated in only one subgraph instance. Thus, for each one grass region object only one adjacent roof object is included. This is also why all SAR region objects were not covered by the second most significant subgraph. The third thing we noticed was that the two apparent SAR subregions (upper left vs. lower right) were not distinguished. We note that one SAR subregion (upper left) consists primarily of a large orange region object containing many small purple region objects. Conversely, the other SAR subregion (lower right) consists primarily of a large purple region object containing many small orange region objects.

Realizing that the concepts of apartment complex, residential area, industrial park and recreational park involve more than a two-vertex graph, we decided to rerun Subdue on the same input graph with the requirement that only subgraphs with at least five vertices be considered. We also turned on the Subdue option to report all subgraph instances instead of limiting a region object to participate in only one subgraph instance. The results from this run of Subdue are displayed in Fig. 2.

The most significant subgraph discovered by Subdue in this second analysis of the simulated data is a grass region object linked to a roof region object, and three bare soil region objects. The area covered by all instances of this subgraph is the center of both parks (Fig. 2b). Since all subgraph instances are taken into account, all three roof region objects in these park sections are blacked out instead of just one.

The second most significant subgraph discovered by Subdue in this second analysis (Fig. 2c) is a grass region object linked to an asphalt region object and also linked to a tree region object and two roof region objects. This subgraph pattern occurs both in the residential area and in

the lower right portion of the parks area. The area blacked out also extends to the asphalt areas in the apartment complex and industrial park because the asphalt area is just one large region object. Again, all related roof region objects are blacked out since all subgraph instances are taken into account.

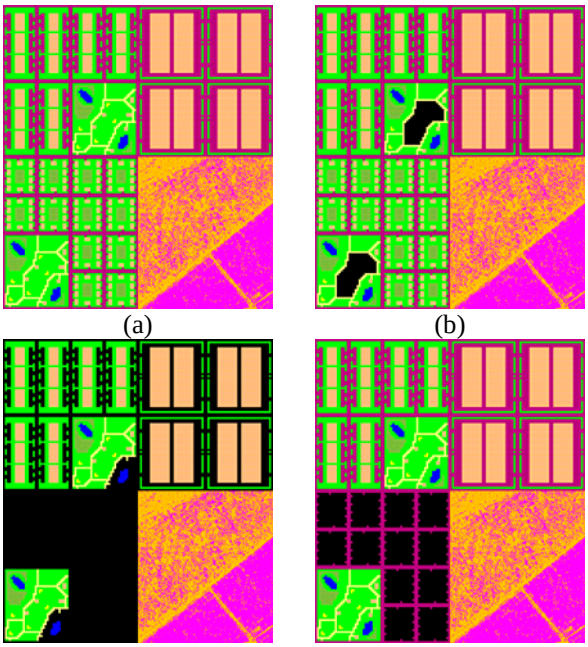

Fig. 2. (a) A simulated image segmentation. (b) Most significant sub-graph, (c) second most significant subgraph, and (d) third most significant subgraph from Subdue with at least five vertices. The image area covered by the region objects participating in instances of these subgraphs is blacked out.

The third most significant subgraph discovered by Subdue in this second analysis is a grass region object linked to a tree region object and also linked to three roof objects. The areas blacked out in Fig. 2d correspond precisely to the residential area sans the road/driveway network. Again, all roof objects in the residential area are blacked out because all subgraph instances are considered.

6. PRELIMINARY CONCLUSIONS

Much work remains to be done to fully exploit the possible capabilities of an RHSEG/Subdue combination. The most significant result reported here is that multiple vertices graphs can provide interesting results. However, the information provided to Subdue from the RHSEG generated image segmentations in our initial tests was very limited compared to what could potentially be provided. Two additional pieces of information that we plan to experiment with providing to Subdue are the size of the region objects and the nature of the region object neighbor relationship. In addition, we intend to investigate an approach that would enable Subdue to utilize the RHSEG segmentation hierarchy information.

7. ACKNOWLEDGMENT

This work was supported by NASA's Applied Information Systems Research program.

8. REFERENCES

[1] J.C. Tilton, *RHSEG, HSEGViewer and HSEGReader User's Manual, Version 1.40*, available from http://ipp.gsfc.nasa.gov/RHSEG, July 28, 2008.

[2] D.J. Cook, and L. Holder, "Graph-based data mining," *IEEE Intelligent Systems*, 15(2), pp. 32-41, 2000.

[3] L. Holder, D.J. Cook, J. Coble, and M. Mukherjee, "Graph-based relational learning with application to security," *Fundamenta Informaticae Special Issue on Mining Graphs, Trees and Sequences*, 66(1-2), pp. 83-101, 2005.

[4] A. Rakhshan, L. Holder, and D.J. Cook, "Structural web search engine," *International Journal on Artificial Intelligence Tools*, 31(1), pp. 27-44, 2004.

[5] I. Jonyer, D.J. Cook, and L. Holder, "Discovery and evaluation of graph-based hierarchical conceptual clustering, *Journal of Machine Learning Research*, 2, pp. 19-43, 2001.

[6] J.-M. Beaulieu, and M. Goldberg, "Hierarchy in picture segmentation: A stepwise optimal approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(2), pp. 150-163, 1989.

[7] J.C. Tilton, "Parallel Implementation of the Recursive Approximation of an Unsupervised Hierarchical Segmentation Algorithm," *High Performance Computing in Remote Sensing*, (C.-I. Chang and A. J. Plaza, editors), Chapman & Hall/CRC, Boca Raton, FL, pp. 97-107, 2007.

[8] J.C. Tilton, "A Split-remerge Method for Eliminating Processing Window Artifacts in Recursive Hierarchical Segmentation," *Disclosure of Invention and New Technology: NASA Case No. 14,994-1*, 2005.

[9] F. Frawley, G. Piatetsky-Shapiro, and C. Matheus, "Knowledge discovery in databases: An overview," *AI Magazine*, pp. 213-228, 1992.

[10] D. J. Cook, and L. B. Holder, "Substructure discovery using minimum description length and background knowledge." *Journal of Artificial Intelligence Research*, 1, pp. 231-255, 1994.