A Visualisation of a Hierarchical Structure in Geographical Metadata

Urška Demšar
Institute of Infrastructure, Royal Institute of Technology
Stockholm, Sweden
urska.demsar@geomatics.kth.se

SUMMARY

The paper presents a visualisation of a hierarchical structure discovered in a repository of geographical metadata by clustering. The hierarchical structure is visualised as a radial tree. The clusters at different levels of detail are represented by the vertices and the subset relationship between the clusters by the edges of the tree. The similarity between elements in different clusters is shown using a colour scheme for the vertices and the edges. The visualisation is integrated in a visual data mining tool for the exploration of the geographical metadata.

KEYWORDS: *visualisation, clustering, tree drawing, geographical metadata.*

INTRODUCTION

The paper describes a visualisation technique which detects and presents an underlying structure in geographical metadata to the user. The visualisation was developed as a part of one of the tools in the project INVISIP, "Information Visualisation for Site Planning" (a European Commission supported project, IST 2000-29640). One goal of the project was to support the search and selection of the appropriate geographical data for a site planning task. The tool described is designed for visual exploration of geographical metadata.

Metadata are data on data. They describe the characteristics and contents of an original document or piece of data (Milstead and Feldman, 1999). Geographical metadata describe geospatial data: maps, satellite images or other geographically referenced material. They usually conform to one of the two major standards: ISO 19115 standard for geographical metadata, (ISO, 2003) or Content Standard for Digital Spatial Metadata (CSDGM, 2003). For INVISIP geographical metadata the ISO 19115 standard was used. The ISO 19115 standard defines metadata elements and establishes a common set of metadata terminology, definitions and extension procedures. It provides information about the identification, the extent, the quality, the spatial and temporal schema, the spatial reference and the distribution of digital geographical data. The standard defines an extensive set of 409 metadata elements, out of which usually only a subset is used for describing a dataset. Metadata elements are grouped into metadata entities which are in turn grouped into fourteen main metadata packages. As it is essential that a basic minimum number of metadata elements are maintained for a dataset, the standard provides a minimal set of twenty-two core metadata elements that still adequately describe a geographical dataset (ISO, 2003).

Metadata that conform to the ISO 19115 standard have the following two characteristics: they are highly dimensional with several hundreds of attributes (dimensions) and the attributes are represented in various formats (numerical, nominal and ordinal data types as well as free text fields). These two characteristics present a problem for the success of automatic data mining algorithms when applied to geographical metadata (Podolak and Demšar, 2004).

Another difficulty when exploring a set of geographical metadata is its size. The user is faced with a large number of metadata, with which he might not be familiar. He might not know much about the characteristics and attributes of metadata, or he may have only a vague idea of what he is looking for.

This constrains the exploration and results in a potential loss of information. To show the user a structure in the metadata can therefore only be of help to him (Albertoni et al., 2003a).

We suggest a visual data mining approach to explore the geographical metadata, where a visualisation acts as a communication channel between the user and the computer. The process is based on the human ability to recognise unknown patterns and correlations from the visualisation. The metadata are presented to the user by an appropriate visualisation method, adapted to the various data types. There are five different visualisations in the tool: a histogram, a table, a pie chart, a parallel coordinates visualisation and finally a special clustering visualisation, which is described in this paper. The visualisations are connected using the brushing and linking principle: brushing is a simultaneous interactive selection process, while linking connects the selected metadata from the current visualisation to all other visualisations and to the metadatabase in the background (Albertoni et al., 2003b).

To structure the metadata, an automatic data mining technique, clustering, has been integrated with the visual data mining. Clustering is a method of dividing data into groups – clusters. Elements in a cluster share common features according to similarity criteria. These criteria are not known in advance, they are discovered during the process. To use such a statistical learning method is an advantage to the user, who might not know how to set such criteria if he would have to define them in the beginning of his exploration. An objective of clustering is to increase similarity of observations within clusters, and decrease it across clusters (Podolak and Demšar, 2004).

We have used a hierarchical clustering algorithm. This algorithm produces a hierarchy of clusters, organised in the form of a mathematical tree. The root of the tree represents the whole metadata set which is split into several subgroups. Each of these subgroups contains elements that are more similar to each other than to elements in the other subgroups. Each subgroup is represented by a child vertex of the root. This process of splitting the set of metadata is then recursively repeated on each child of the root. The hierarchical structure of clusters shows the nested groupings of patterns and the similarity levels at which the groupings change (Fisher, 1996, Jain et al., 1999). The metadata items in the resulting tree are presented as the leaves on the lowest level in the tree structure, while the internal vertices represent metadata clusters on different levels of similarity. When defining the similarity criteria for clustering the variety of data types in the ISO metadata standard was taken into account (Podolak and Demšar, 2004).

VISUALISATION OF THE CLUSTERED STRUCTURE

To integrate the automatic clustering into the visual data mining process a visualisation of the clustered hierarchical structure was needed.

The classical way to represent the hierarchical structure is to draw it as a rooted-tree, usually called a dendrogram, with the root anchored centrally on the top of the image and the children vertices drawn downwards using straight or bended lines (Kaufmann and Wagner, 2001). We list some of the attempts to use the traditional rooted-tree visualisation of the dendrogram adapted to particular needs of the authors' research. Seo and Shneiderman (2002) have connected the dendrogram with a two-dimensional scatterplot, which shows a correlation between two chosen attributes for the selected cluster in the dendrogram. The dendrogram can be projected on other surfaces rather than on a simple plane, for example on a hemisphere in the Magic Eye View by Kreuseler and Schumann (2002) or into a three-dimensional space in a ShapeVis, a cluster visualisation technique by the same authors. Another way to present the dendrogram is to overlay it by coloured shapes and contours, such as it is done in a visualisation technique called structure-based brushes (Fua et al., 2000).

While the above-mentioned visualisations all display the actual hierarchical tree, there are other approaches that use different graphical solutions to represent the structure. For example, Bederson et al. (2002) display the hierarchy using a structure of nested-boxes, a so-called treemap, where the screen is split into rectangles in alternating horizontal and vertical directions as one traverses down the levels in the hierarchy. In a treemap the groupings are represented using the colour and the area of the resulting

boxes. A similar approach is suggested by Wills (1998): a space-filling recursive division of a rectangular area that lets the user manipulate the number of clusters produced by the clustering method without requiring to redraw the whole clustering tree. A sunburst visualisation by Stasko and Zhang (2000) shows the hierarchical structure of the data in a radial layout. A small circle represents the root of the hierarchy. For each recursive level a ring is added to the outer rim of the circle. The rings are subdivided in sectors according to the number and size of the children on the respective level. Sprenger et al. (2000) present a visualisation of clusters mapped into a hierarchy of implicit surfaces, which are wrapped in a three-dimensional space around the points belonging to the same cluster.

Most of the visualisations of the dendrogram have certain disadvantages. The visualisations that draw the structure as a classical top-down rooted tree are difficult to read when a large amount of data is displayed, especially when it comes to recognising finer details of the structure on the leaf-node level. The patterns in such a drawing are difficult to recognise. Other visualisations which do not directly show the tree structure, but represent hierarchy using different graphical entities and relationships between them, are fairly complex and may be difficult to grasp for a non-expert user. The complexity in these approaches is a serious obstacle when trying to integrate such a visualisation into the visual data mining process.

In this paper we suggest a visualisation of the hierarchical tree which is suitable for integration with the interactive selection of a subset of metadata items that our implementation of the visual data mining is based on.

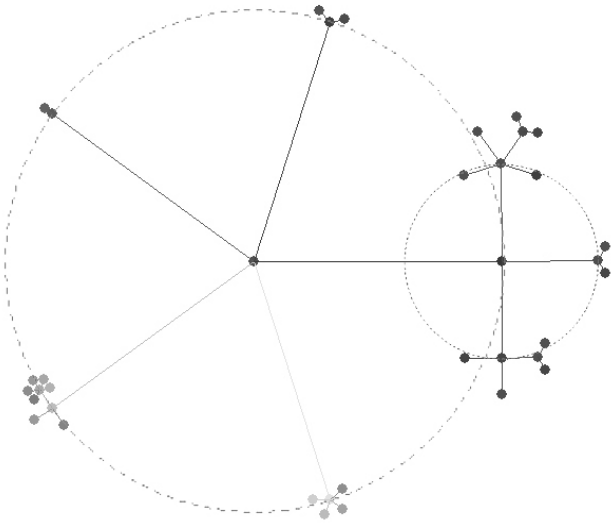

Figure 1: The recursive construction of a snowflake graph

We suggest to draw the dendrogram as a radial tree. In contrast to traditional top-down tree-drawing techniques, which anchor the root vertex of the tree centrally on the upper edge of the image, in a radial display the root of the tree is placed in the centre of the image. The root's outgoing edges are placed at equal angles around the root. The direct descendant vertices of the root are drawn on the circumference of the circle, which has its centre in the root vertex. This structure is recursively continued at each level: the current vertex is the centre of a circle and its incoming edge and outgoing edges are evenly

distributed in this circle, with the descendant vertices placed on the circle's circumference as shown in figure 1. We propose to call such a structure *a snowflake graph*.

The description of such a drawing algorithm sounds simple, but its implementation is quite complex. The geometrical positions of the vertices are calculated using first a recursive traversal of the structure to define the radii of each circle and then a second non-recursive traversal of the tree during which the absolute position of each vertex is calculated.

The following parameters are assigned to each vertex v:

(x,y) – the absolute position of the vertex v in the visualisation window,
r – the absolute radius of the circle on which the children vertices of the vertex v are placed,
d – the radius of the circle which the children's respective circles touch (figure 2),
r_j – the maximal possible radius for the circles of the children vertices if these are to be distributed around the parent vertex v at equal angles (figure 2),
α - the angle belonging to each of the children vertices v_i when they are equally distributed around the parent vertex v (figure 2),
φ - the absolute angle of the incoming edge of the vertex v, measured from x-axis in the counter-clockwise direction (figure 3).

The values r, r_j, d and α for each vertex are calculated during the recursive traversal of the tree. This calculation is adapted from the circular drawing algorithm for rooted trees by Melançon and Herman (1998). Here is the pseudocode for the recursive method setRadii that assigns all the above-mentioned values to vertex v:

```
method setRadii (vertex v) {
 if v=leaf {
 d=0;
 rj=const;
 }
 else {
 loop over children vi of v {
  setRadii(vi);
 }
 set d, alpha, rj;
 }
 r = rj + d;
}
```

If the vertex v is a leaf, we assign value zero to d and a constant value to r_j. This latter constant is the minimal possible edge length, which is in our implementation set to 15 pixels, since we draw vertices as full circles with a diameter of 10 pixels. If the vertex v is not a leaf, we call the method setRadii recursively on all its children and then assign the values to the d, α and r_j. The value of d is according to the heuristics set to the maximum value of the absolute radii of all children vertices of the vertex v, as proposed in the algorithm by Melançon and Herman (1998). The angle α equals one half of the angle that encloses the circle of each child vertex, i.e., it is one half of the $n+1^{th}$ part of the full angle 2π, where n is the number of children of the vertex v. We divide the full angle into $n+1$ parts, since we need to equally distribute not only the n children edges but also the incoming edge of the vertex v in the circle. The formula for obtaining the value r_j from d and α is derived from the geometric relationship which states that the tangent of the angle α is the quotient of r_j and r (as can be deduced from figure 2). To avoid division by zero when α equals $\pi/4$ (which happens if a vertex has three children, $n=3$), a special formula was derived for this case. To summarise, the three values are calculated as:

$$d = \max_{i=1,\ldots,n}(r_i),\ \alpha = \frac{\pi}{n+1},$$

where r_i is the absolute radius r of each children vertex v_i, $i=1, \ldots n$, n is the number of children of the vertex v and

$$r_j = \begin{cases} d \cdot \frac{\tan(\alpha)}{1-\tan(\alpha)} & ;\alpha \neq \frac{\pi}{4} \\ \frac{d}{\sqrt{2}-1} & ;\alpha = \frac{\pi}{4} \end{cases}.$$

Finally, the absolute radius r is the sum of r_j and d regardless if the vertex v is a leaf or not: $r=r_j+d$.

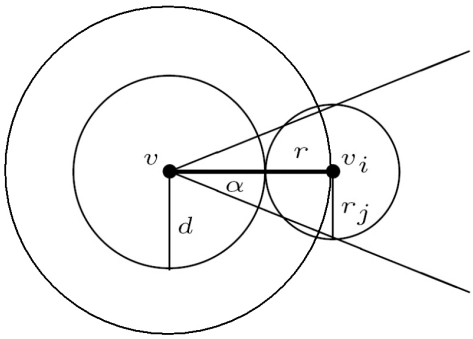

Figure 2: Definition of r, r_j, d and α for the vertex v with its child vertex v_i.

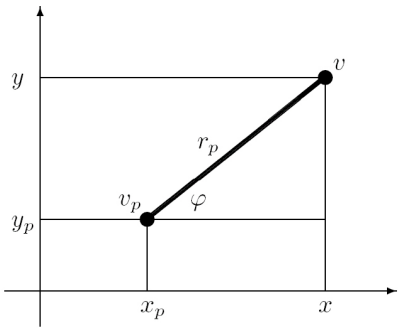

Figure 3: Absolute positioning of a vertex – transformation from radial to Cartesian coordinates

The second step in the geometric positioning algorithm is to traverse the tree structure once more (but this time non-recursively) and to calculate for each vertex its absolute position from the radius r and the angle φ and then assign the absolute angle φ to all its children. The coordinates and the angle of the root are set to:

$$x_{root} = \frac{w}{2},\ y_{root} = \frac{h}{2},\ \varphi_{root} = 0,$$

where *w* is the width and *h* the height of the visualisation window in which the snowflake graph is displayed.

We start traversing the tree at the root and perform a classical depth-first traversal. The first step at the vertex *v* is to assign the absolute angle φ_i to all children vertices v_i of the vertex *v*, *i=1,...,n*:

$$\varphi_i = \pi + \varphi_p + \frac{2\pi i}{n+1},\ i = 1, \ldots, n.$$

The angles φ_i are geometrically defined by equally distributing the *n+1* edges (edges of the *n* children and the incoming edge of the vertex *v*) into the full angle 2π with the apex in the vertex *v*. The angles are measured counter-clockwise from the incoming edge, therefore we need to add $\varphi+\pi$ to each of them.

For each vertex *v* (except the root, where the absolute coordinates are defined before the traversal) we calculate its absolute position from the absolute coordinates (x_p, y_p) of the parent vertex, the absolute radius r_p of the parent vertex and the angle φ, which was assigned to *v* when we traversed its parent vertex (figure 3). The absolute coordinates of the vertex *v*, *(x,y)*, are calculated as:

$$x = x_p + r_p \cdot \cos(\varphi),\ y = y_p + r_p \cdot \sin(\varphi).$$

Once all this is done and the absolute positions of all vertices are known, the tree is drawn on the screen using the colour scheme for the similarity relationship, which is described below.

Our algorithm is implemented in Java, which affects the geometric positioning. The algorithm was developed using the traditional right-handed Cartesian coordinate system, with the origin (0,0) in the lower left corner of the display and the positive values of x and y increasing towards the right and upwards respectively. Graphics in Java uses a left-handed coordinate system, where the origin is in the upper left corner and the positive directions are towards the right and downwards for x and y respectively. This has two effects on our algorithm: the vertices' absolute positions are mirrored over the x axis downwards and the angles are measured clockwise instead of counter-clockwise. The calculations remain the same and the direction of the coordinate system has no effect whatsoever on the calculated distances between vertices and the appearance of the snowflake graph.

The similarity of the metadata elements in this structure is represented in two ways. One is the natural hierarchical structure, i.e. metadata elements that have the same ancestral vertices at a near-by level are more similar to the metadata elements, whose ancestral branches meet higher up in the tree. To additionally emphasize the similarity we developed a colour scheme for the edges and vertices, where the similarity of the colours corresponds to the similarity of metadata elements.

We used the HSB (hue-saturation-brightness) colour model (Foley et al., 1990) for the definition of the colour scheme. This model represents colours in a cylindrical three-dimensional coordinate system. The subspace where the model is defined is a hexagonal pyramid. The pyramid is turned upside-down and has the apex in the origin of the coordinate system. One of the three dimensions, the brightness, is assigned to the central axis of the pyramid. The other axis, saturation, starts in the apex of the pyramid and is perpendicular to its central line. The remaining coordinate, hue, is represented by the angle on the regular hexagon (forming the base side of the pyramid) which is perpendicular to the brightness axis and cuts the axis at the brightness value 1. Brightness and saturation are both defined in the range [0,1]. The sides of the hexagon are of length 1 and the distance from the hexagon's central point towards its edge represents the saturation of the colour. The central point in the hexagon is white (with the values *h=0, s=0, b=1*) and the apex of the pyramid is black (*h=0, s=0, b=0*). Sometimes a unit circle is used instead of the hexagon and a cone instead of the hexagonal pyramid. The usual orientation of the hue hexagon or circle is as follows: the red colour is at 0° angle, the green at 120° and the blue at 240° in the counter-clockwise direction. Complementary colours in the hue hexagon or circle are 180° opposite one another.

The colour scheme for the edges and the vertices of the snowflake graph is defined in two steps. We colour the child vertices of the root and their respective incoming edges according to their position in a hue circle (figure 4). The hue circle is mirrored over the line at 0° (as illustrated in figure 4). This fact is due to Java, another effect of the left-handed coordinate system. As figure 4 shows, red remains in the same position as in the original HSB model, but green and blue colours swap places at the 120° and 240° respectively.

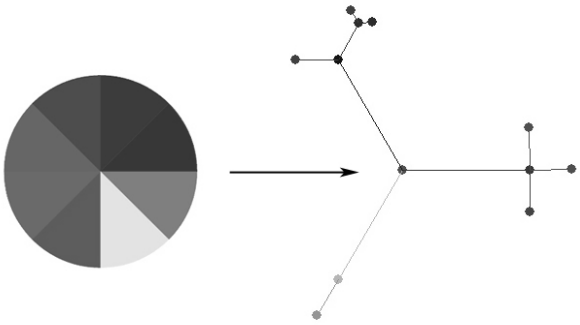

Figure 4: Colour-coding the child vertices of the root by their position in an inverted hue circle.

For all other vertices we assign to the children vertices a colour derived from the colour of the parent vertex in the following way: the hue of the children's colour remains the same as the hue of the parent vertex, whereas the saturation and brightness change linearly with the number of the each child. If the parent's hue, saturation and brightness are *(h, s, b)*, then each child's hue, saturation and brightness *(h_i, s_i, b_i)* equal to:

$$h_i = h,\ s_i = \frac{i}{2n} + \frac{1}{2},\ b_i = \frac{i}{2n} + \frac{1}{2},$$

for each child vertex v_i, where $i = 1, \ldots n$, and n is the number of children of the parent vertex. This principle is illustrated in figure 5.

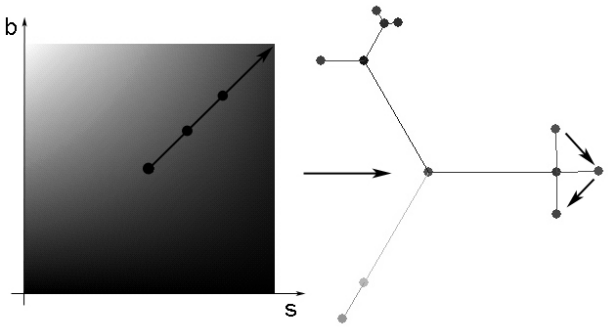

Figure 5: Colour-coding the vertices on a deeper level in a saturation (s) and brightness (b) graph.

The snowflake graph is easily integrated into the process of brushing and linking, which is the interactive selection process of our visual data mining implementation, described by Albertoni et al. (2003b). The interactive selection procedure is based on the user's selection of one or more graphical

entities that represent metadata items in each visualisation. Such an entity in the snowflake graph is a leaf vertex of the tree together with the corresponding path from the leaf to the root. If the user wishes to select only one metadata item he can click on the leaf that represents this item. At the same time the path from the leaf to the root is highlighted. If he wishes to select multiple metadata items which form clusters at different levels of detail, he has to select the internal vertex that represents the cluster in question. The path from the cluster vertex to the root and the subtree belonging to the respective cluster is highlighted during selection. Such selection of two metadata items is shown in figure 6.

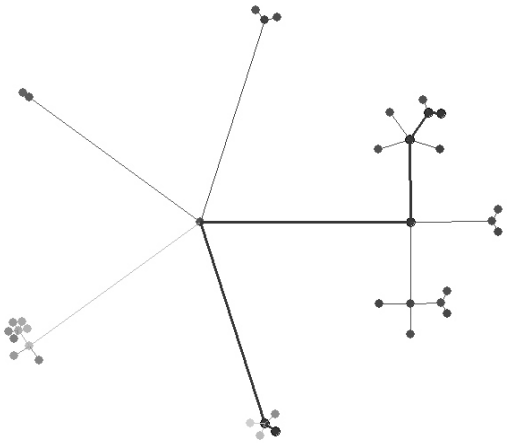

Figure 6: A selection of two leaves and their respective leaf-to-root paths in the snowflake graph.

CONCLUSIONS

A combination of automatic and visual data mining exploration techniques presents a good solution to the problems the user encounters while exploring geographical metadata, i.e. attributes of various data types and a large amount of available metadata. Pre-structuring the metadata and visualising this structure in an easy understandable way facilitates the user's exploration. Since a hierarchy is added to the structure, the user can apply either a bottom-up or a top-down exploration of the tree, going either from the metadata items towards the clusters of similar elements at various levels of detail or starting from the all available metadata and limiting his search by advancing deeper and deeper in the tree.

Drawing the hierarchical structure in the form of a radial tree as a snowflake graph solves some of the difficulties with readability in the usual presentations of the hierarchical structure, since the drawing space available is more optimally used than in the classical top-down drawing of a rooted tree. This visualisation is very intuitive and not difficult to grasp even for the novice user. It is not difficult to understand the meaning of the internal vertices and leaves, the hierarchy expressed with the edges and the similarity presented by the colour of the vertices and edges. Once the user grasps this, the structure becomes easy to navigate which leads to discovery of correlations among metadata elements that would otherwise stay unnoticed.

ACKNOWLEDGMENTS

The author would like to acknowledge the support of the European Commission and cooperation of the other members of the VDM development team: H. Hauska (Royal Institute of Technology, Stockholm, Sweden), K. Hemzaczek, A. Malczewska, I. Podolak, G. Surowiec (Agricultural University of Krakow, Krakow, Poland), R. Albertoni, A. Bertone and M. De Martino (Institute for Applied Mathematics and Information Technologies, Genoa, Italy).

BIBLIOGRAPHY

Albertoni R., Bertone A., De Martino M., Demšar U. and Hauska H., Visual and Automatic Data Mining for Exploration of Geographical Metadata, Proceedings of the 6th AGILE Conference on Geographic Information Science, Lyon, France, pp. 479-488, 2003a.

Albertoni R., Bertone A., Demšar U., De Martino M. and Hauska H., Knowledge Extraction by Visual Data Mining of Metadata in Site Planning, Proceedings of the 9th Scandinavian Research Conference on Geographic Information Science, ScanGIS'2003, Espoo, Finland, pp. 119-130, 2003b.

Bederson B. B., Shneiderman, B. and Wattenberg, M., Ordered and Quantum Treemaps: Making Effective Use of 2D Space to Display Hierarchies, ACM Transactions on Graphics (TOG), Vol. 21, No. 4, pp. 833-854, 2002.

CSDGM, Content Standard for Digital Geospatial Metadata, Federal Geographic Data Committee, National Spatial Data Infrastructure, 1998.

Fisher D. H., Iterative optimisation and simplification of hierarchical clusterings, Journal of Artificial Intelligence Research, Vol. 4, pp. 147-179, 1996.

Foley J. D., van Dam A., Feiner S. K. and Hughes J. F., Computer Graphics: Principles and Practice, Addison-Wesley, Reading, Massachusetts, 1990.

Fua Y. H., Ward M. O. and Rundensteiner E. A., Structure-Based Brushes: A Mechanism for Navigating Hierarchically Organized Data and Information Spaces, IEEE Transactions on visualization and computer graphics, Vol. 6, No. 2, pp. 150-159, 2000.

ISO 19115, Geographic information – Metadata, International Standard Organisation, http://www.iso.org, 2003.

Jain A. K., Murty M. N. and Flynn P. J., Data Clustering: A Review, ACM Computing Surveys, Vol. 31., No. 2., pp. 264-323, 1999.

Kaufmann M. and Wagner D. (eds.), Drawing Graphs – Methods and Models, Lecture Notes in Computer Science 2025, Springer-Verlag, Berlin Heidelberg, 2001

Kreuseler M. and Schumann H., A Flexible Approach for Visual Data Mining, IEEE Transactions on Visualization and Computer Graphics, Vol. 8, No. 1., pp. 39-51, 2002.

Melançon G. and Herman I., Circular Drawings of Rooted Trees, Reports of the Centre for Mathematics and Computer Sciences, INS-9817, 1998.

Milstead J. and Feldman S., Cataloguing by Any Other Name..., Online, Vol. 23., No.1, Information Today Inc., 1999.

Podolak I. and Demšar U., Discovering a Structure in a Repository of Geographical Metadata, to appear in: Proceedings of Geoinformatics 2004, 12th International Conference on Geoinformatics, Gävle, Sweden, June 2004.

Seo J. and Shneiderman B., Interactively Exploring Hierarchical Clustering Results, IEEE Computer, Vol. 35, No. 7, pp.80-86, 2002.

Sprenger T. C., Brunella R. and Gross M. H., H-BLOB: A hierarchical visual clustering method using implicit surfaces, IEEE Visualization 2000, Salt Lake City, Utah, USA, Proceedings. IEEE Computer Society and ACM, pp. 61-68, 2000.

Stasko J. and Zhang E., Focus+Context Display and Navigation Techniques for Enhancing Radial, Space-Filling Hierarchy Visualizations, Proceedings of InfoVis2000, IEEE Symposium on Information Visualization, Salt Lake City, USA, pp. 57-68, 2000.

Wills G. J., An Interactive View for Hierarchical Clustering, Proceedings of InfoVis1998, IEEE Symposium on Information Visualization, North Carolina, USA, pp.26-34, 1998.