

Clustering Large Datasets
and Visualizations of
Large Hierarchies and Pyramids
Symbolic Data Analysis Approach

Vladimir Batagelj, Enzo Pavleitič,

Matjaz Zaveršnik, Simona Korenjak-Černe

University of Ljubljana, Slovenia

PKDD'2000 – Workshop on Symbolic Data Analysis
September 12, 2000, Lyon, France

Contents

- Clustering large datasets;
- Hyperbolic representation;
- Flags.

Clustering large datasets

Advanced methods of data analysis are mostly appropriate only for datasets of moderate size. To apply them to (very) large datasets we usually first reduce the data to an acceptable size.

For this purpose clustering is usually used: a large dataset is clustered into a smaller, tractable number of clusters. Afterward the properties and representatives of the obtained clusters are determined. These representatives are used as units (symbolic objects (SO)) in the further analyses.

For clustering large datasets some variant of leaders method (for example k -means) is usually used. Most of these methods can deal only with units with variables measured in numerical scales.

Uniform description of mixed units

On IFCS'98 we proposed an approach for clustering large datasets of *mixed* (nonhomogeneous) units – units described by variables measured in different types of scales (numerical, ordinal, nominal). The basic idea of our approach is to describe all units and clusters in a uniform way by frequency distributions of values of their variables. Let $\mathbf{C} = \{C\}$ be a clustering of the basic set of units E , $C \neq \emptyset$, $C \subseteq E$. We partition the range of the variable V into $k(V)$ classes $\{V_1, V_2, V_3, \dots, V_{k-1}, V_k\}$. Then the distribution (p_i) for the cluster C and variable V is determined by the relative frequencies

$$p_i \equiv p(i, C; V) = \frac{q(i, C; V)}{n(C)} \quad \text{for } i = 1, \dots, k(V)$$

where $n(C) = \text{card } C$, $Q(i, C; V) = \{\mathbf{Y} \in C : V(\mathbf{Y}) \in V_i\}$, and $q(i, C; V) = \text{card } Q(i, C; V)$.

Properties

Such a description has two important properties:

- it requires a fixed space per variable;
- it is *compatible* with merging of clusters – knowing the description of two disjoint clusters, $C_1 \cap C_2 = \emptyset$, we can, without additional information, produce the description of their union.

$$q(i, C_1 \cup C_2; V) = \frac{n(C_1)q(i, C_1; V) + n(C_2)q(i, C_2; V)}{n(C_1) + n(C_2)}$$

In the implementations of methods based on this representation we store the *histograms*:

$$(n(C), q(1, C; V), q(2, C; V), q(3, C; V), \dots, q(k-1, C; V), q(k, C; V)).$$

On the basis of this representation, the adapted versions of leaders method, adding method, and hierarchical agglomerative method were developed and implemented.

Dissimilarity

Suppose that the descriptions of units consist of m variables. To develop a clustering procedure for their analysis we first define the dissimilarity between two descriptions \mathbf{X}_1 and \mathbf{X}_2 as the weighted sum of the dissimilarity between them on each variable V

$$d(\mathbf{X}_1, \mathbf{X}_2) = \sum_{j=1}^m \alpha_j d(\mathbf{X}_1, \mathbf{X}_2; V_j), \quad \sum_{j=1}^m \alpha_j = 1$$

where

$$d(\mathbf{X}_1, \mathbf{X}_2; V) = \frac{1}{2} \sum_{i=1}^{k(V)} |p(i, \mathbf{X}_1; V) - p(i, \mathbf{X}_2; V)|$$

or

$$d(\mathbf{X}_1, \mathbf{X}_2; V) = \frac{1}{2} \sum_{i=1}^{k(V)} (p(i, \mathbf{X}_1; V) - p(i, \mathbf{X}_2; V))^2.$$

Here, $\alpha_j \geq 0$ ($j = 1, \dots, m$) are weights, which can be equal for all variables, $\alpha_j = \frac{1}{m}$, or different if we have same information about their importance or interdependance. They could be also used for 'learning'. Because the units and clusters are represented in the same way we also use so defined dissimilarity for defining the dissimilarity between two clusters.

We get the description of a single unit \mathbf{X} as

$$p(i, \{\mathbf{X}\}; V) = \begin{cases} 1 & V(\mathbf{X}) \in V_i \\ 0 & \text{otherwise} \end{cases}$$

The clustering problem

Finally we define the *clustering problem* as an optimization problem:

Find a clustering \mathbf{C}^* for which

$$P(\mathbf{C}^*) = \min_{\mathbf{C} \in \Phi} P(\mathbf{C}).$$

For the criterion function measuring the goodness of a clustering \mathbf{C} we use

$$P(\mathbf{C}) = \sum_{C \in \mathbf{C}} \sum_{X \in C} d(X, L(C)),$$

where $L(C) = [s(i, L; V)]$ represents the leader (representative) of the cluster C . $L(C)$ is also a distribution.

The adapted leaders method

The proposed method for clustering very large datasets is a variant of the *dynamic clustering method* and can be shortly described with the following procedure:

determine an initial clustering

repeat

 determine leaders of the clusters;

 assign each unit to the nearest leader –

 producing a new clustering

until the process stabilizes.

The leaders

It can be proved that for the first selected criterion function the optimal leaders are determined with maximal frequencies

$$s(i, L; V) = \begin{cases} \frac{1}{t} & i \in M \\ 0 & \text{otherwise} \end{cases}$$

where $M = \{j : q(j, C; V) = \max_i q(i, C; V)\}$ and $t = \text{card } M$; and for the second with average distributions

$$s(i, L; V) = \frac{1}{n(C)} \sum_{\mathbf{X} \in C} p(i, \mathbf{X}; V)$$

This provides an easy interpretation of the clustering results.

The agglomerative hierarchical clustering method

After reducing, with the leaders method, the dataset to some hundreds of representatives of the obtained clusters we can produce a hierarchical clustering on them, $\mathbf{H} = \cup \mathbf{C}_k$, using the standard *agglomerative hierarchical clustering method*:

```
each unit is a cluster:  $\mathbf{C}_1 = \{\{\mathbf{X}\}: \mathbf{X} \in E\}$  ;  
they are at level 0:  $h(\{\mathbf{X}\}) = 0$ ,  $\mathbf{X} \in E$  ;  
for  $k := 1$  to  $n - 1$  do  
    determine the closest pair of clusters  
     $(p, q) = \operatorname{argmin}_{i,j:i \neq j} \{d(C_i, C_j) : C_i, C_j \in \mathbf{C}_k\}$  ;  
    join them  
     $\mathbf{C}_{k+1} = \mathbf{C}_k \setminus \{C_p, C_q\} \cup \{C_p \cup C_q\}$  ;  
     $h(C_p \cup C_q) = d(C_p, C_q)$   
endfor
```

We can also produce a pyramid on them. The pyramidal clustering is an extension of the hierarchical model. The main difference is that a cluster/unit in a pyramid can belong at most to two clusters. The graph of pyramidal structure is planar. For details see Diday, 1986. For clustering very large datasets into several hundreds of clusters and producing a hierarchy on them the adding clustering method can be applied.

Visualization of hierarchies and pyramids

There exist several solutions for visualization of hierarchies: Windows directory tree, Tree map, Pyramidal representation Cheops, Cone tree, Hyperbolic display in 2D and 3D.

Here we present two approaches to visualization of large hierarchies suitable for data analysis: *hyperbolic display* and *flags*.

Example

In the following we shall use as a (small) example dataset the data about 27 different types of food (see Table 1). They are described by 5 numeric variables: Food Energy, Protein, Fat, Calcium, and Iron. Values of variables were encoded according to six equidistant classes between their minimal and maximal values.

variable	unit	min	mean	max
Energy	cal	45	207.4	420
Protein	g	7	19.0	26
Fat	g	1	13.5	39
Calcium	mg	5	43.96	367
Iron	mg	0.5	2.38	6

Table 1: Types of Food with Encoding

No	Food	Engy	Prot	Fat	Calc	Iron	Codes
1	Beef, braised	340	20	28	9	2.6	5 5 5 1 3
2	Hamburger	245	21	17	9	2.7	4 5 3 1 3
3	Beef, roast	420	15	39	7	2.0	6 3 6 1 2
4	Beef, steak	375	19	32	9	2.6	6 4 5 1 3
5	Beef, canned	180	22	10	17	3.7	3 5 2 1 4
6	Chicken, broiled	115	20	3	8	1.4	2 5 1 1 1
7	Chicken, canned	170	25	7	12	1.5	2 6 1 1 2
8	Beef heart	160	26	5	14	5.9	2 6 1 1 6
9	Lamb leg, roast	265	20	20	9	2.6	4 5 3 1 3
10	Lamb shoulder, roast	300	18	25	9	2.3	5 4 4 1 2
11	Smoked ham	340	20	28	9	2.5	5 5 5 1 3
12	Pork, roast	340	19	29	9	2.5	5 4 5 1 3
13	Pork, simmered	355	19	30	9	2.4	5 4 5 1 3
14	Beef tongue	205	18	14	7	2.5	3 4 3 1 3
15	Veal cutlet	185	23	9	9	2.7	3 6 2 1 3
16	Bluefish, baked	135	22	4	25	0.6	2 5 1 1 1
17	Clams, raw	70	11	1	82	6.0	1 2 1 2 6
18	Clams, canned	45	7	1	74	5.4	1 1 1 2 6
19	Crabmeat, canned	90	14	2	38	0.8	1 3 1 1 1
20	Haddock, fried	135	16	5	15	0.5	2 3 1 1 1
21	Mackerel, broiled	200	19	13	5	1.0	3 4 2 1 1
22	Mackerel, canned	155	16	9	157	1.8	2 3 2 3 2
23	Perch, fried	195	16	11	14	1.3	3 3 2 1 1
24	Salmon, canned	120	17	5	159	0.7	2 4 1 3 1
25	Sardines, canned	180	22	9	367	2.5	3 5 2 6 3
26	Tuna, canned	170	25	7	7	1.2	2 6 1 1 1
27	Shrimp, canned	110	23	1	98	2.6	2 6 1 2 3

Hyperbolic display

The idea of hyperbolic display (Lamping, 1995) is applied for visualization of large pyramids and hierarchies and extended with some subject specific options. The hyperbolic display is an example of fish-eye displays that allow a closer look at the data in the selected neighborhood put into the context of the global view. It allows simultaneous view of complete pyramidal or hierarchical clustering and detailed inspection of selected region inside pyramid or hierarchy. It also represents a 'map' structure over clusters. Selected nodes (clusters) can be represented and compared using other representations for individual SO. Both, hierarchies and pyramids are represented by planar graphs.

The hyperbolic display is implemented in Java using 3D Master Suite – an extension of the Open Inventor for Windows.

In Figure 1 a (small) hierarchy over Food types is displayed.

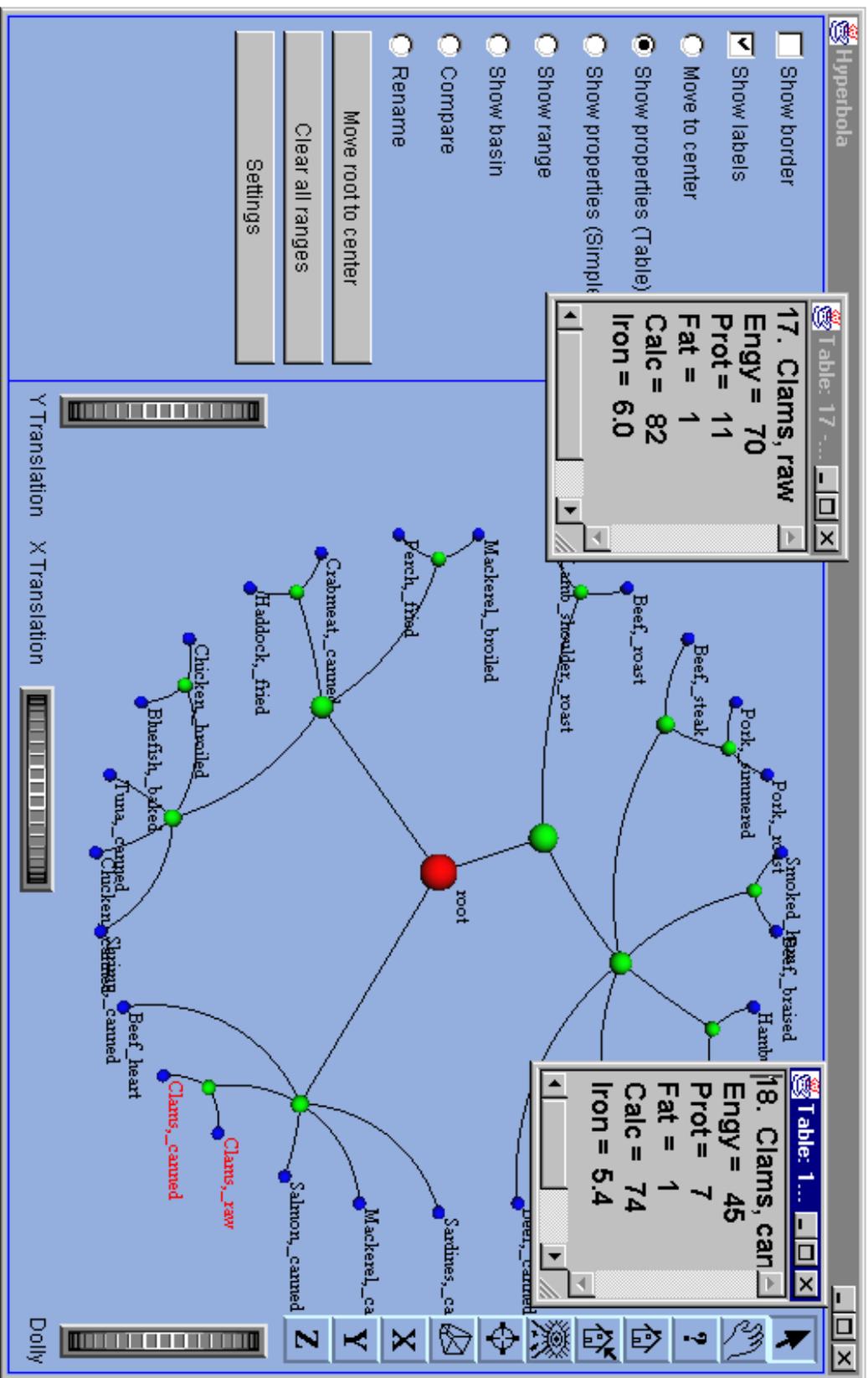


Figure 1: Small Hierarchy – Food Types

Besides the general functionalities (left-right / up-down / in-out moves, . . .) of the OpenInventor the left side of the window provides a control panel with the following options:

- **show border**: switches the display of the border – circle representing the hyperbolic plane;
- **show labels**: switches the display of labels of units;
- **move to center** (selects a state): click on a node will move the selected node to the center;
- **show properties** (selects a state): click on a node will display in a separate window the detailed information (table or simple star) about the selected node;
- **show range** (selects a state): click on a node will show, by an arc on the border, the range – set of leaves covered by the selected node in the given pyramid/hierarchy;

- **show basin** (selects a state): click on a node will show, by recoloring, the set of all nodes covered by the selected node;
- **compare** (selects a state): for the selected pair of nodes will compute and display their dissimilarity;
- **rename** (selects a state): allows to give names to internal nodes.

The next two buttons provide shortcuts to **move the root to the center** and to **clear all ranges**. The last button opens a special window in which we can change different **settings** of the hyperbolic display.

In Figure 1 we can also see an example of the tabular display of the properties of selected nodes (marked by labels of different color).

In Figure 2 a hyperbolic display of large pyramid with 1347 nodes (500 leaves) is presented. On it we can see also *ranges* of 3 selected nodes (the node and the corresponding arc colored with the same color) and the *basin* of selected node (black nodes).

In Figure 3 two different views of the pyramid obtained by moving selected nodes into the center are presented. We obtain a detailed view on the neighborhood of the selected node. The other possibility for detailed inspection of a part of pyramid is to zoom-in, as shown in Figure 4.

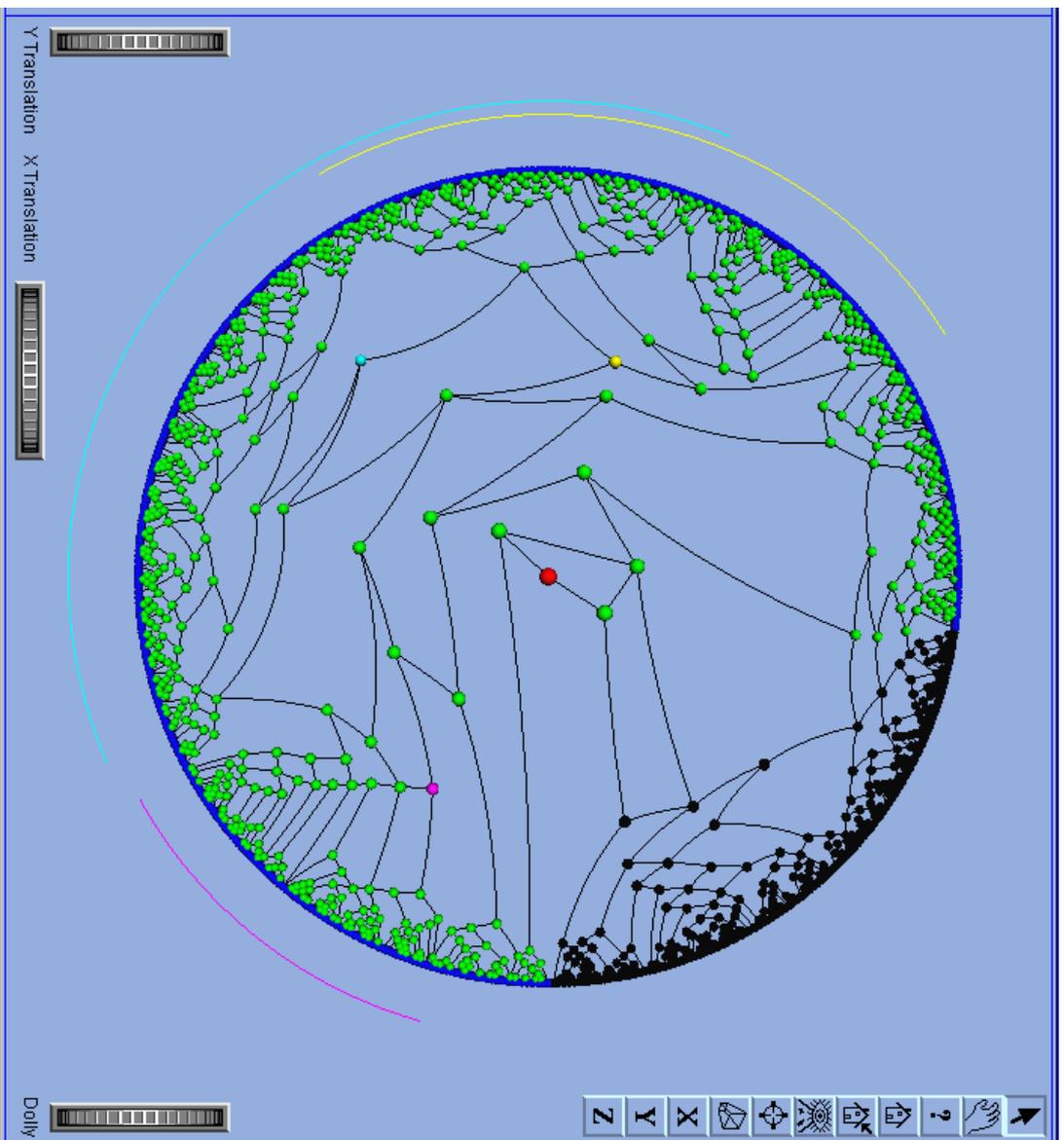


Figure 2: Large Pyramid – 1347 nodes / 500 leaves

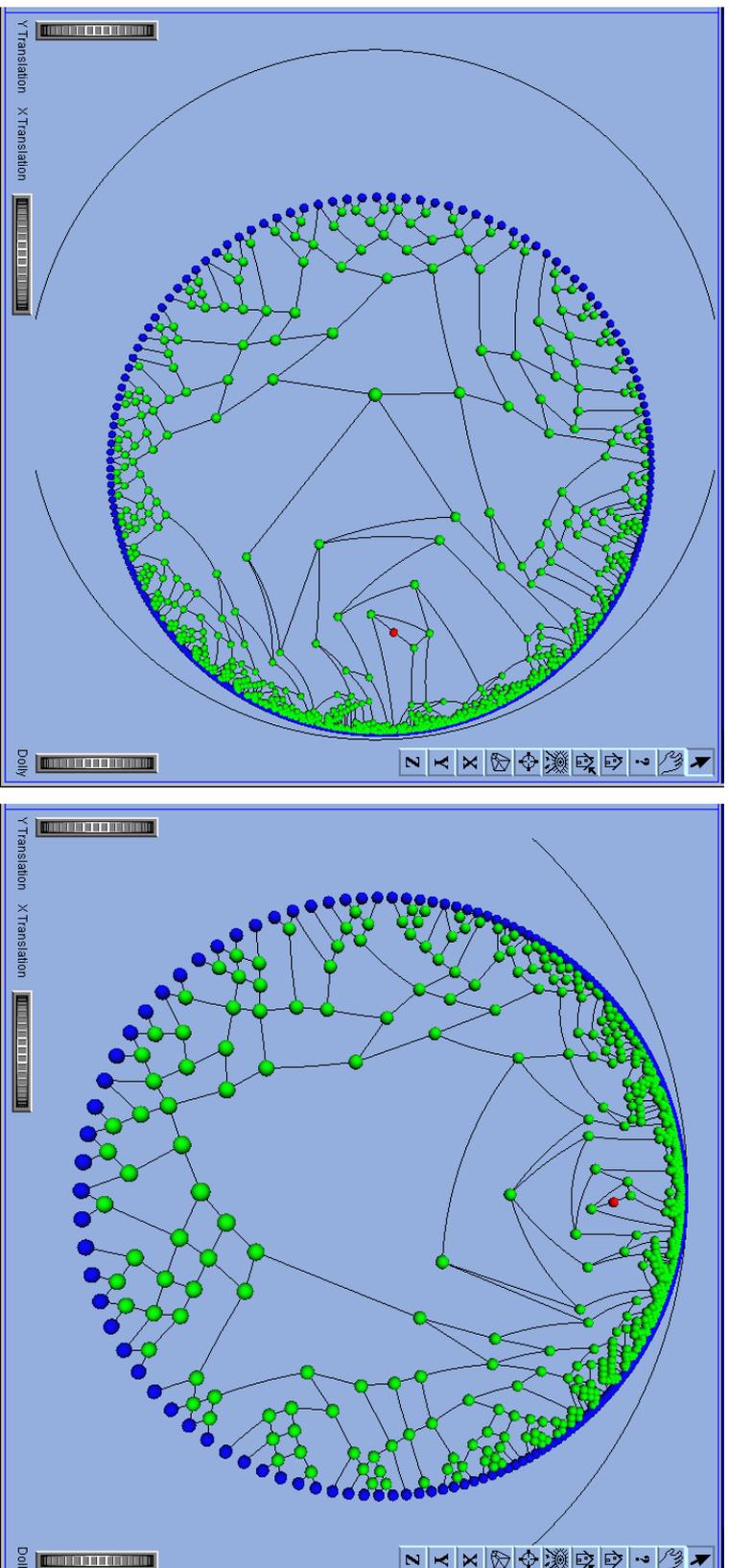


Figure 3: Large Pyramid – two moves

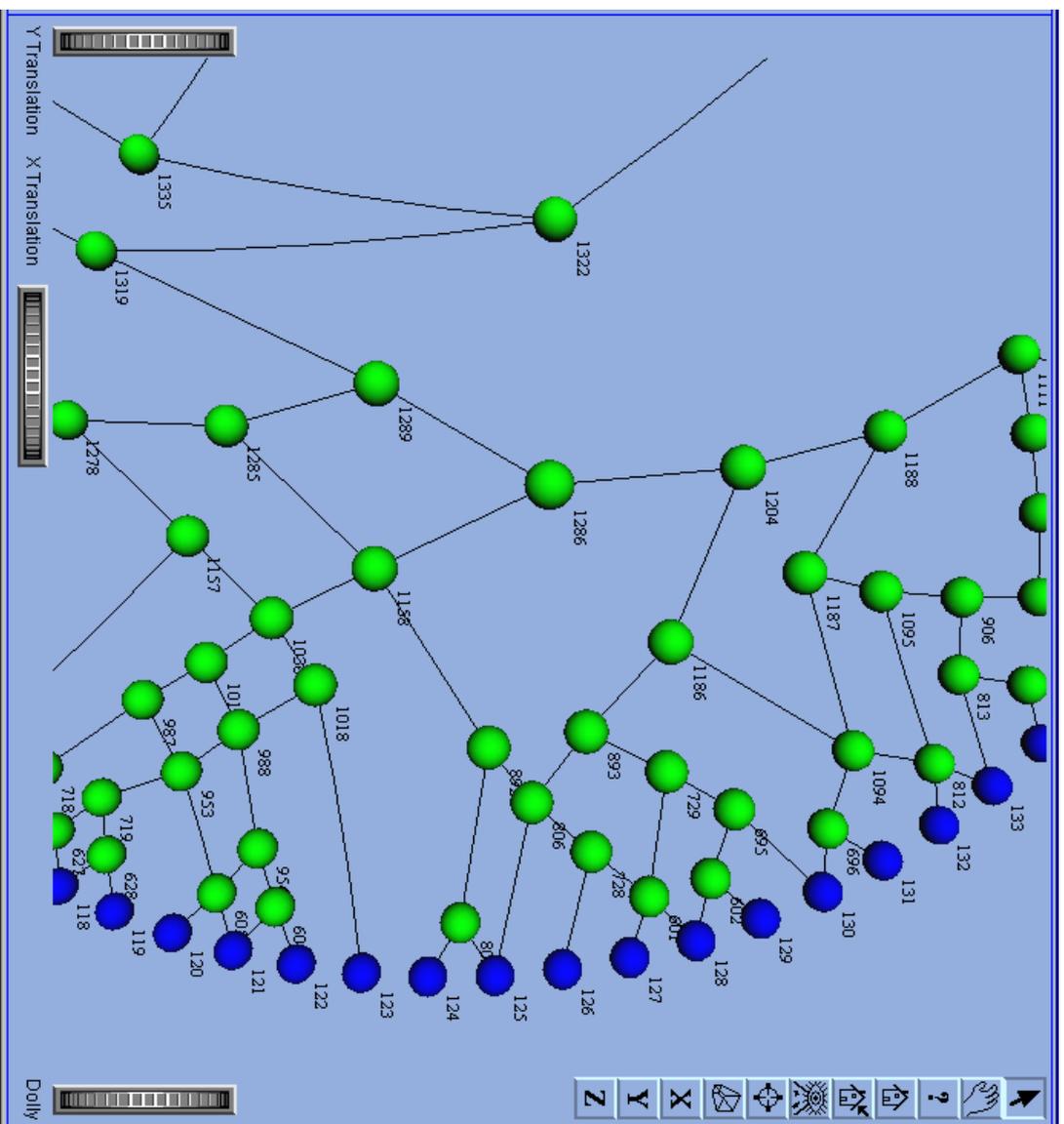
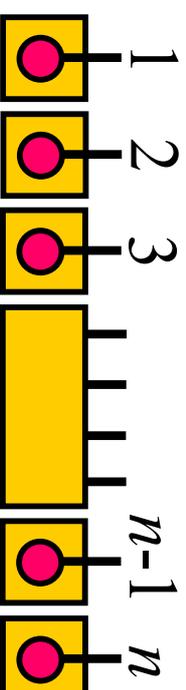


Figure 4: Large Pyramid – zoom in

Generating random pyramids

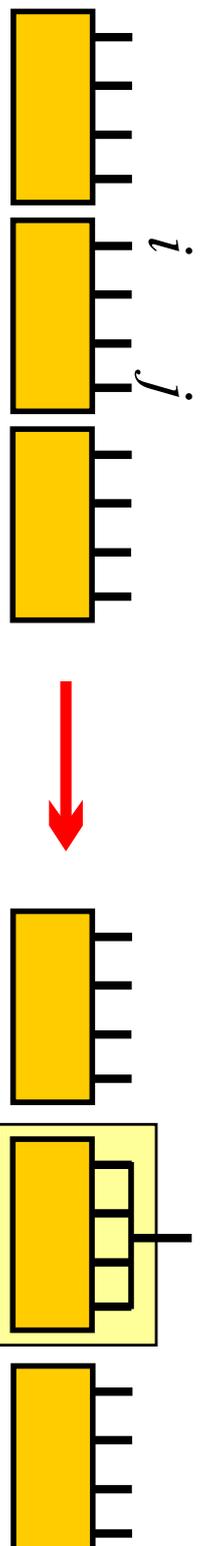
For testing the hyperbolic display we developed a special program for generating random pyramids / hierarchies. It is based on the random joining of *hooks* of partial pyramids.

We start with a list of n units (leaves) – each having a hook.

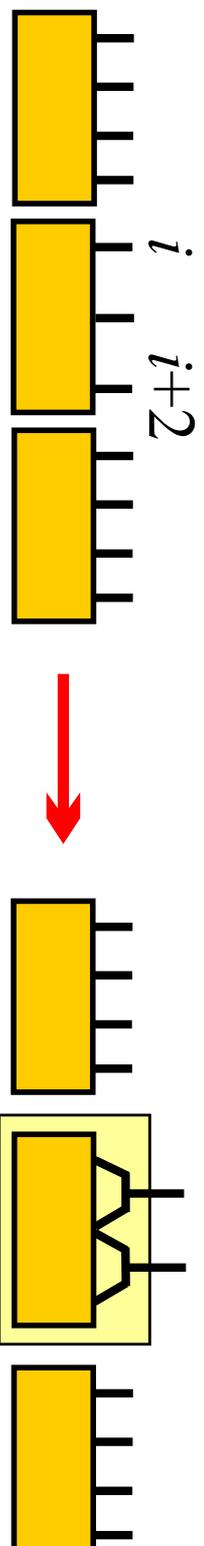


Afterward we repeat until the pyramid is built, the following:

- with probability p select the first (else select the second) of the two rules
the **tree rule**:



and the **pyramid rule**:



- determine the random position i in the current list of hooks (and, in the case of non binary tree rule, also the number k of hooks to be joined, $j = i + k - 1$) and apply the selected rule.

The pyramid displayed in Figure 2 was obtained by this procedure.

Flags

This representation was initially developed for visualization of partitions of large datasets obtained by leader's method described in the first section (based on the uniform representation of a cluster by histograms). Later the concept was extended to hierarchies and to some other types of descriptions of distributions of values of variables that have the property that the description of the union of two disjoint clusters can be obtained from their descriptions. Besides histogram, such descriptions are, for example, mean value and mean value with range.

Mean value: $(n(C), m(C))$:

$$m(C) = \frac{1}{n(C)} \sum_{X \in C} V(X), \quad n(C) = |C|, \quad n(C_1 \cup C_2) = n(C_1) + n(C_2)$$

$$m(C_1 \cup C_2) = \frac{n(C_1)m(C_1) + n(C_2)m(C_2)}{n(C_1 \cup C_2)}$$

Mean value with range: $(n(C), \min(C), m(C), \max(C))$

$$\min(C_1 \cup C_2) = \min(\min(C_1), \min(C_2))$$

$$\max(C_1 \cup C_2) = \max(\max(C_1), \max(C_2))$$

Flags are essentially a graphical display of the encoded table where each cell is colored with a color(s) (or level(s) of gray) assigned to the cell value(s). They are based on visual comparison of units (table rows). A more informative display can be obtained if we reorder the units according to some clustering.

In the case of large datasets such representation is not manageable any more. But we can reduce its size by replacing some clusters with their representatives. The things become even more interesting if a hierarchy over a given dataset is known. So extended flags combine global view with detailed local view. By interactively drilling into the hierarchy we can expose selected units in their contexts.

In Figure 5 a representation with flags of part of Food types is presented.

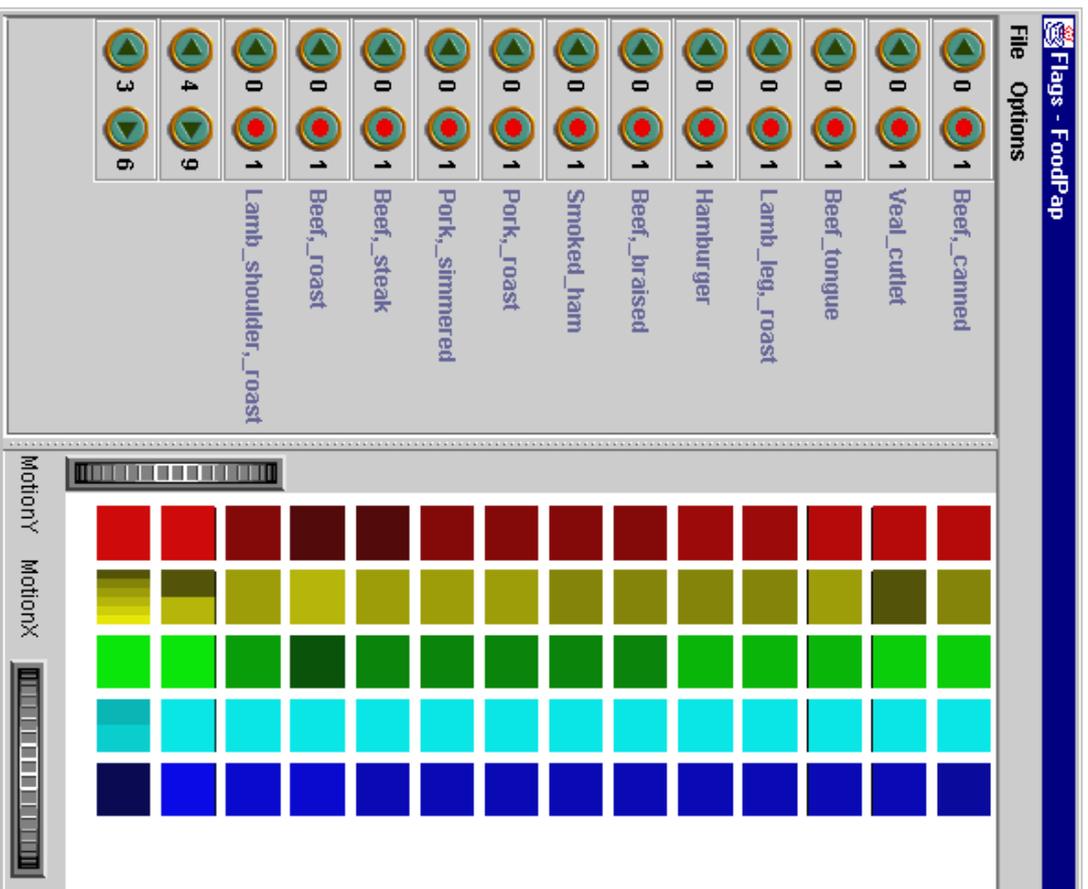


Figure 5: Flags – Food types

Flags are also implemented in Java using 3D Master Suite. They provide an interactive system for producing and manipulating hierarchical flags over large datasets. If no hierarchy is given, the trivial partition (a one level hierarchy) – each unit is a cluster, is assumed.

The hierarchical flags display consists, besides the table with names of rows (clusters/units) and columns (variables), of additional information about the position of rows in the hierarchy:

- level of the cluster/unit in the hierarchy starting from units (level 0) (first number in the panel);
- number of units in the cluster (second number in the panel); and
- type – unit, cluster, context.

Context is the complement of selected elements – union of non selected units/clusters.

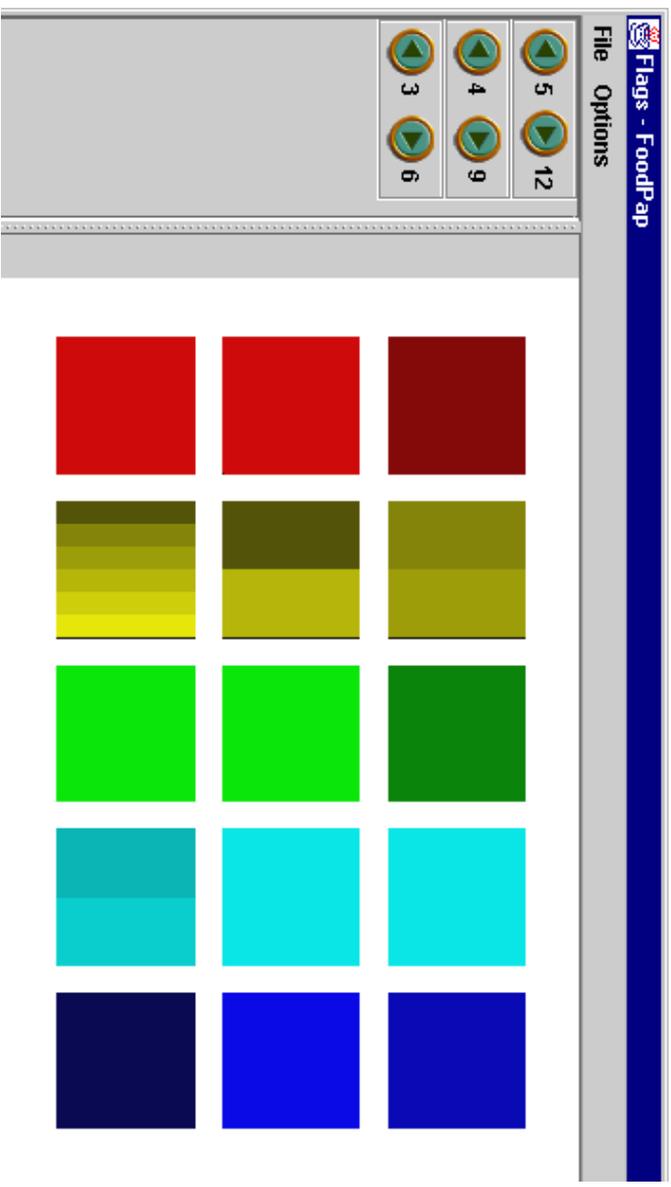


Figure 6: Flags – Three clusters

There are three groups of options:

Settings: general: palettes, fonts, sizes.

Column options:

- include variable;
- exclude variable;
- move variable (reorder);
- info about variable;
- change variable settings (encoding, palette, ...);
- display variable (separate display: histogram, text, ...).

Row options:

- open/enter cluster (all new clusters/units have type context);
- close/exit cluster;
- change type (context, selected/show);
- reduce context;
- rename cluster;
- move row (reorder respecting hierarchy);
- display cluster (separate display: star, text, ...).

The option reduce context collects all clusters/units marked with context into a single contextual cluster. It can be done level by level or iteratively over several levels.

Additional options: The hierarchical flags can be used also to perform '*visual clustering*' – semi-manual clustering. We start with trivial partition and analyzing the data we manually build the hierarchy. To support this some additional options are needed:

- show only / hide units/clusters with values of variable in the set;
- order (inside clusters) on the values of the variable;
- join selected two units/clusters into a new cluster;
- move unit/cluster into cluster;
- flatten selected cluster;
- compute the dissimilarity between selected clusters.

For non binary hierarchies the option open cluster can increase the size of the picture considerably. To remain in control we can assign to each open a 'window' (with slider or rotation) of some maximal size for the display of the son-rows.

In Figure 7 the use of the third dimension for presenting the relative frequencies of cell values is presented.

In the current implementation of flags contexts and additional options are not supported yet.

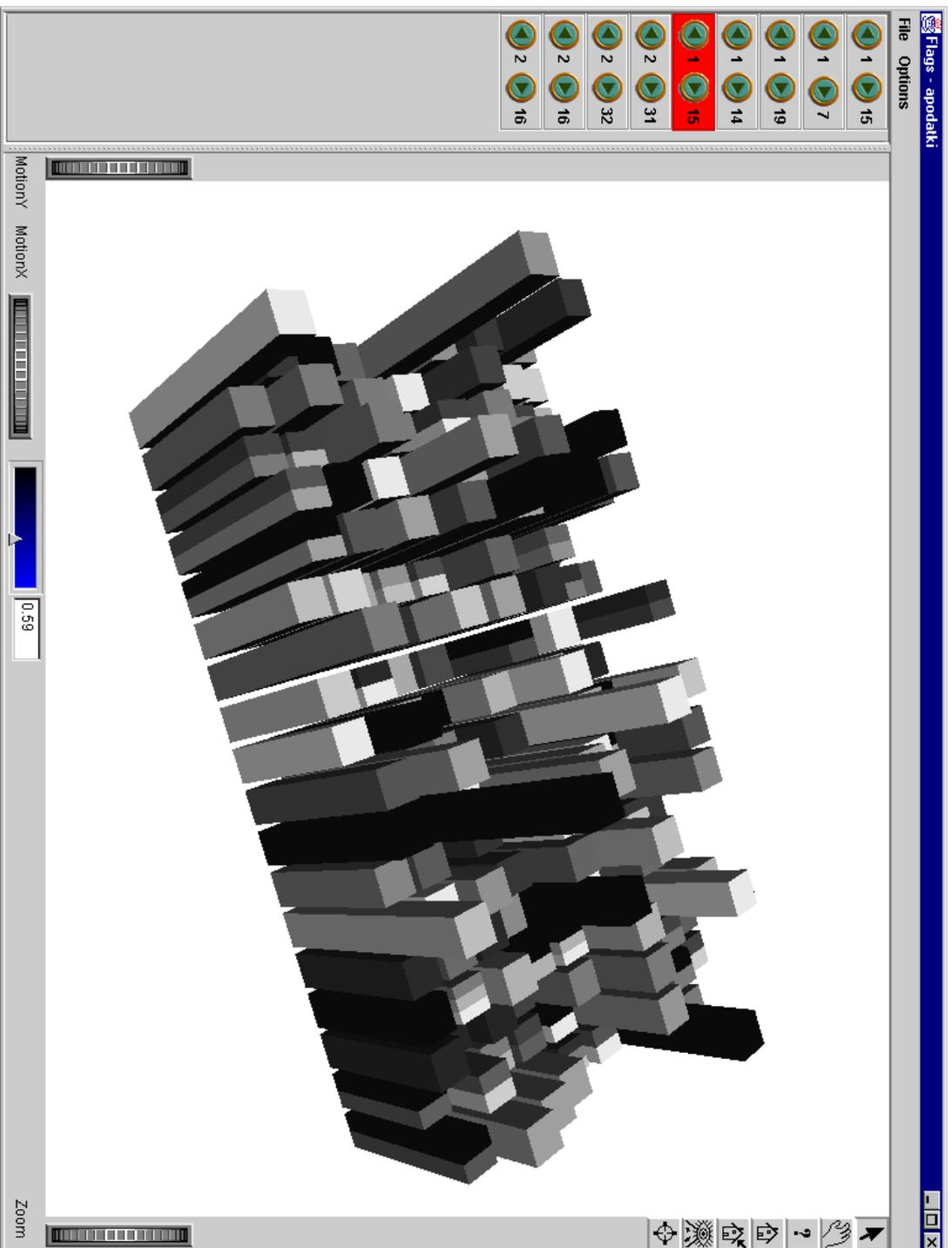


Figure 7: Flags – 3D

Acknowledgments

This work was partially supported by the Ministry of Science and Technology of Slovenia, Project J1-8532. Both types of visualization were developed for the ISO-3D project, ESPRIT Project N. 28953.

These slides are available at

<http://educa.fmf.uni-lj.si/datana/pub/>

References

- [1] Aude J.C.: Pyramids: A pyramidal analysis tool for sequence clustering.
<http://genome.genetique.uvsq.fr/Pyramids/index.html>
- [2] Beaudoin L., Parent M-A., Vroomen L.C.: Cheops: A Compact Explorer For Complex Hierarchies. Visualization'96, San-Francisco 1996.
<http://www.crim.ca/hci/cheops/index1.html>
- [3] Brito P., Diday E.: *Pyramidal representation of symbolic objects*. NATO ASI Series, Vol. F 61. Proc. Knowledge Data and computer-assisted Decisions. Schader and Gaul edit. Springer-Verlag, 1991.
- [4] Bulatov V.: HyperProf(v.1.3) - Java profile browser.
<http://www.physics.orst.edu/~bulatov/HyperProf/>
- [5] Diday E.: *Optimisation en classification automatique*, Tome 1.,2.. INRIA, Rocquencourt, 1979, (in French).
- [6] Diday E.: Orders and overlapping clusters by pyramids. in Multidimensional Data Analysis Proc., ed. J.De Leeuw etal. DSWO Press, Leiden 1986.

- [7] Diday E.: *The symbolic approach in clustering and related methods of Data Analysis*. in "Classification and Related Methods of Data Analysis", Proc. IFCS, Aachen, Germany. H. Bock ed. North-Holland, 1987.
- [8] Hartigan J.A.: *Clustering Algorithms*. Wiley, New York, 1975.
- [9] Inxight: Welcome to the Hyperbolic Tree Walk ThroughTour. http://www.inxight.com/Demos/HT_Tour/HT_Tour_Intro.html
- [10] Johnson B., Shneiderman B.: Tree-maps: A space-filling approach to the visualization of hierarchical information structures, Proc. IEEE Visualization '91, IEEE, Piscataway, NJ 1991, 284-291.
- [11] Korenjak-Černe S., Batagelj V.: Clustering Large Datasets of Mixed Units. Advances in Data Science and Classification (A. Rizzi, M. Vichi, H.-H. Bock, eds.), Proceedings of IFCS'98, Rome, 21-24. July 1998. Springer, Berlin 1998, p. 41-48. <http://www.educa.fmf.uni-lj.si/datana/clamix/largeset.pdf>
- [12] Korenjak-Černe S., Batagelj V.: Adding Methods for Clustering Large Datasets of Mixed Units. Manuscript, 1999; presented at IFCS'2000.

- [13] Lamping J., Rao R., Pirolli P.: A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies.
http://info.sigchi.acm.org/sigchi/ch95/Electronic/documents/papers/jl_bdy.htm
- [14] Munzner T.: Papers. <http://graphics.stanford.EDU/~munzner/papers.html>
- [15] OLIVE, On-line Library of Information Visualization Environments.
<http://otal.umd.edu/olive/>
- [16] Robertson G., Mackinlay J., Card S.: Cone Trees: Animated 3D Visualizations of Hierarchical Information. SIGCHI '91, pgs. 189-194.
- [17] TGS: 3D-MasterSuite for Java. <http://www.tgs.com/3DMS/index-java.html>
- [18] Warncke J.: *The Inventor Mentor*. Addison-Wesley, Reading, MA, 1994.