International Journal of Approximate Reasoning 51 (2010) 275–293

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

journal homepage: www.elsevier.com/locate/ijar

Visualization of network structure by the application of hypernodes

Jan Terje Bjørke [a,*,1], Stein Nilsen [a], Margaret Varga [b]

[a] *Norwegian Defence Research Establishment (FFI), P.O. Box 115, N-3191 Horten, Norway*
[b] *QinetiQ, St. Andrew's Road, Malvern, WR14 3PS, UK*

ARTICLE INFO

Article history:
Received 11 April 2008
Received in revised form 9 September 2009
Accepted 9 September 2009
Available online 22 November 2009

Keywords:
Generalization
Graphs
Weighted networks
Levels of abstraction
Geographical patterns

ABSTRACT

In the literature several authors describe methods to construct simplified models of networks. These methods are motivated by the need to gain insight into the main properties of medium sized or large networks. The present paper contributes to this research by setting focus on weighted networks, the geographical component of networks and introducing a class of functions to model how the weights propagate from one level of abstraction to the next. Hierarchies of network models can be constructed from reordering of the adjacency matrix of the network; this is how "hypernodes" are derived in the present paper. The hypernode algorithm is explored and it is shown how it can be formulated to handle weighted networks. Weighted networks enable handling the uncertainty or the strength of the components which make up the network. The hypernode algorithm can be run in an iterative manner so that a hierarchy of simplified models of the network can be derived. Some case studies demonstrate the hypernode algorithm. In the first case the algorithm is compared with a similar implementation described in the literature. In the second case an airline dataset is analysed. This study shows that when networks are embedded in the geographical space hypernodes may relate to clusters in the spatial domain. The selection of the visual variables to illustrate the strength of the edges and nodes in a weighted network is discussed.

1. Introduction

1.1. The problem domain

Graphs, subsequently also called networks, are often used in the modelling of real world phenomena or abstract concepts in many disciplines involving information sciences, geographical information systems, military applications and threat detection, social sciences, biology, mathematics, among others [1–8]. In this paper a real airline routes network will be used as a case study.

Börner et al. [9], which give a comprehensive introduction to network science, classify networks by their size as (1) small, less than 100 nodes, (2) medium sized, between 100 and 1000 nodes, and (3) large, more than 1000 nodes. Often in large networks neither all nodes and edges can be shown at once. Even for small networks there may be structures which are hard to detect in a visualization without generalization, i.e., reduction of the number of nodes and edges. In order to keep the important characteristics of the network structure, this reduction must follow rules that maintain the underlying structure of the network in the visualization. This kind of problem is well known in geographical information systems, for example,

[*] Corresponding author. Tel.: +47 33 03 38 37; fax: +47 33 04 78 34.
E-mail address: jan-terje.bjorke@ffi.no (J.T. Bjørke).
[1] Also at Department of Mathematical Sciences and Technology, Norwegian University of Life Sciences (UMB), P.O. Box 5003, 1432 Ås, Norway.

0888-613X/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.ijar.2009.09.003

when data are to be visualized as maps. The present paper addresses the problem stated, i.e., how to make generalized models of networks so that the analyst can get a holistic view of the network and superior in-depth understanding of its main properties.

The term hypernode will be used to denote a node which represents an aggregation of nodes. In order not to confuse with the established theory of hypergraphs [8,10], we will not use the terms hyperedge and hypegraph. In the forthcoming hypernodes will be defined, their application to network visualization explored and an algorithm to generate hypernodes will be described. When constructing the network of hypernodes, the strength of the nodes and the edges should be considered. Therefore, different weight propagation models will be defined. In some cases there may exist an implicit geographical component in the network, i.e., a geographical clustering which can be explored from the hypernodes. A case study will demonstrate this kind of geographical nearness.

Since the present paper considers weighted networks, also strategies to visualize the weights will be discussed, i.e., colour coding or grey value coding.

1.2. Related work

Our work is inspired by the research of NATO-group IST-059/RTG-025 [3]. This group has identified key issues of the visualization of networks. As a part of the work in this group Bjørke [4] did show how hypernodes enable networks to be visualized at higher levels of abstraction. From reordering of the adjacency matrix of the network the groups of nodes which form the hypernodes, can be derived.

The concept of reordering the adjacency matrix of a network was first introduced by Bertin [11], although Hartigan presented in 1972 a method to clustering a data matrix [12]. The method of Bertin starts with mapping the data to an image which is termed the reorderable matrix. Then the rows, or the columns, of this image are interchanged to generate different views of the data. In this way meaningful patterns in the data can be detected by visual interpretation of the reordered image, see for example [13]. Automatically reordering rows and columns of matrices try to optimize an objective criterion, but no polynomial time approximation exists for the exact solution. An overview of different approaches to this problem can be found in [9,14–17].

Flake et al. [18] introduce graph clustering methods based on minimum cuts within a graph and define an expansion of a cut. In a forthcoming section we will show how this index fits into a class of similar measures. Huang and Lai [19] construct hierarchical clusters of nodes in graphs by applying an adjacency matrix of node similarities. In a forthcoming section our implementation of node clustering will be compared with a case study derived by Huang and Lai.

The novelty of the present paper is related to: (1) the focus on the geographical component in the case studies, (2) the development of a class of indices to measure the strength of the edges between hypernodes, (3) the application of information theory to explain the difference between different visualization methods, and (4) the demonstration of hierarchical node clustering to analyze and visualize real world network data.

2. Selection of visual variables

When visualizing weighted networks, the question of how the strength of the edges and nodes can be visualized, must be answered. Since we in the forthcoming sections will present several figures of weighted networks, visualization issues will be elaborated upon before the hypernode algorithm is presented and discussed.

The visualization of a weighted network should map the weights to an appropriate visual variable. Bertin [11] argues that quantitative information should be mapped to visual variables that are able to reflect the ordering of the data. According to Bertin the appropriate visual variables for visualizing a weighted network are size and grey value, see Fig. 1. Here, the weights are divided into three classes and a grey value associated to each class.

Since hue is a qualitative property of colours, Bertin argues that hue cannot offer an intuitive relation to a quantitative information variable. However, colour hue is a very selective visual variable. Therefore, the traffic light symbology (red, yellow, green) offers well separation between three weight classes, see window D in Fig. 2. It is intuitive and offers a clear interpretation/understanding. An alternative to showing all weight classes in a single display, they can be shown in different display windows; one window for each class. This provides valuable information about the individual sub-network structures and how they relate to each other in the whole network, see windows A, B and C in Fig. 2.

The different visualization methods can be characterized in terms of information theory. Information theory computes the entropy $H(X)$ of an information source X as

$$H(X) = -\sum_{x \in X} p(x)\log_2(p(x)),$$

see [20], for example, for a discussion on how information theory can be applied to evaluation of visualizations. The entropy formula can be applied to compare the visualization in Figs. 1 and 2. There are 19 strong edges, 22 medium strong edges and 1 weak edge, totally 42 edges in the network. If all the edges are equally visible, the entropy formulae can be written as $H(X) = \log_2(N)$, where N is the number of events. If we assume that the colours red, yellow and green as applied in Fig. 2, have equal visibility, we can argue that the entropy of the perceived network in window D is $H(X) = \log_2(42) = 5.39$. For

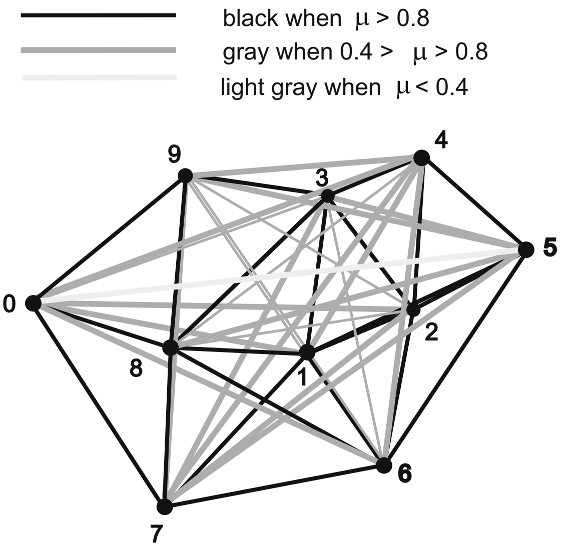

Fig. 1. Applying the visual variable grey scale to illustrate the strength of the edges in a network.

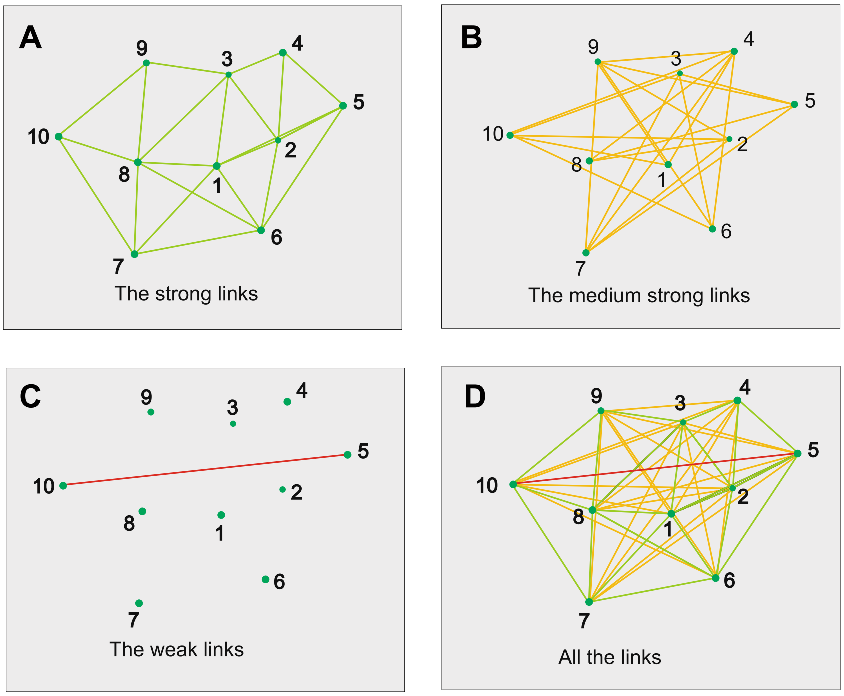

Fig. 2. A network is visualized in four windows; three windows for each of the weight classes and one window with the whole network.

windows A, B and C in the figure, the entropy is lower than in window D. This follows from $H(X) = \log_2(N)$, and that N is reduced when several windows are applied.

Alternatively, grey value can be used as visual variable, but since the dark lines get higher visibility than the light grey lines, grey value has the property of not equal visibility. In order to compute the entropy of the perceived network in Fig. 1, a model that reflects the visibility of the lines is to be set up. First, we will define the visibility of the different lines on the scale 0–1. The black line has maximum contrast to the white background and its visibility μ_1 can be set to 1. The grey value of the medium strong edge is more similar to the background than the black colour and we set its visibility $\mu_2 = 0.5$. For the last grey class we set $\mu_3 = 0.25$. In order to consider the limit case when all the edges have colour similar to the background colour, the background can be introduced as an element of the image, i.e., its visibility is set to 1. In the further computations we will not consider this limit case and only look at the main case, i.e., we assume there are visible edges in the image. The visibility measures introduced must be normalized in order to be used in the entropy formulae. Therefore, we derive the probabilities from

$$p(x) = \frac{\mu(x)}{\sum_{x \in X}\mu(x)}.$$

This gives the probability of detecting a black, a grey and a light grey edge as 0.0331, 0.0165, 0.0083, respectively. The entropy of the network in Fig. 1 is therefore 5.30, i.e., lower than in the traffic light case. Generally, the entropy has its maximum value when all the events are equally probable.

The contribution of the different grey value classes to the entropy of the image can be computed. The contribution from a black line b_i is $h(b_i) = -0.0331\log_2(0.0331) = 0.16$. For the group of black lines we get $H(b) = h(b_i)19 = 3.04$. For an edge g_i of medium grey colour $h(g_i) = 0.10$ and $H(g) = h(g_i)22 = 2.20$. Similarly, the light grey edges l_i give $h(l_i) = 0.06$ and $H(l) = h(l_i)1 = 0.06$. Since $h(b_i) \neq h(g_i) \neq h(l_i)$, from an information theoretic point of view we can argue that the visual variable grey value alters the visual importance of the edges in the network. Moreover, since $H(b) > H(g) > H(l)$ the black lines will dominate the entropy of the perceived image.

Lessons learned 1. *In a grey value visualization of a weighted network the different weight classes will have different visibility, i.e., the edges will be moved towards the front of the image according to the contrast between the background colour and the assigned grey value. Applying grey value to visualize a weighted network will reduce the entropy of the perceived image compared with applying the visual variable colour hue.*

Which method to select is a question about: (1) equal visibility, for example, the traffic light symbology; (2) mapping the weights into an ordered visual variable; (3) bringing some of the elements to the front of the image, for example, the visual variable grey value; (4) being able to study each weight class separately, i.e., apply several windows; or (5) getting a holistic view, i.e., map all information into one single window.

Thorough exploration of the link between information theory and the visualization problem considered, is outside the scope of the present paper, but a direction will be sketched. In information theory the difference between the entropy of an image and the amount of misinterpretations is computed as

$$R = H(Y) - H(Y|X),$$

where R is termed useful information, $H(Y)$ is the entropy of the interpreted image and $H(Y|X)$ is the degree of visual confusion in the interpreted image Y when source image X is presented. The computation of R requires knowledge of the conditional probabilities that the different lines visually are mixed. The degree of misinterpretation, i.e., $H(Y|X)$, of a network visualization depends on the number of presented nodes and edges. Therefore, a reduction of the complexity of the image will effect R. Since this reduction also changes the entropy of the image, the maximal value of R, i.e., the channel capacity, represents a balance between the complexity of the image and the degree of misinterpretations. Bjørke and Sæheim [21] show how the channel capacity of a map can be computed. The computation of the level of detail that corresponds to the channel capacity, can serve as a method to derive an optimal presentation of a network.

3. Basic definitions and notation

Consider a graph $G = (V, E)$ with $|V| = n$ nodes (vertices) and $|E| = m$ edges, where V is a set of nodes and E a set of edges between the nodes. An adjacency matrix of G is defined as $R = (w_{i,j})_{n\times n}$, where $w_{i,j}$ is a weight, i.e., a real valued function, associated to the edge going from node i to node j. The weight of a node is modelled as the self-connecting edge $w(i, i)$. Associating a weight to a node can be useful when we interpret the weights as uncertainties, for example.

Definition 1. A partition P of a graph is a disjoint collection of subsets of graph nodes whose union is the whole node set.

From the definition it follows that $P = \{V_1, V_2, \ldots, V_n\}$ define a partition of V when the subsets $V_1, V_2, \ldots, V_n$ of V satisfy $V_1 \cup V_2 \cup \cdots \cup V_n = V$, and $V_i \cap V_j = \emptyset$ for $i \neq j$.

Definition 2. Let P_1 and P_2 be partitions of a network G. If every element $\pi \in P_1$ is a subset of an element of $P_2,\ P_1$ is called a refinement of P_2. We say that P_1 is finer than P_2, and P_2 is coarser than P_1.

The set of partitions of a network is partially ordered under refinement.

For any graph G the partition P_0 consisting of singleton subsets, is called the *trivial partition*. The trivial partition is finer than every other partition of G. The *simple partition* is the partition with only one element, namely the whole node set. This is denoted by P_G. The simple partition is coarser than any other partition of G.

Definition 3. A sequence of partitions of a graph is called nested, or hierarchical, if every element of the sequence is coarser than the previous element.

Definition 4. The elements of the non-trivial partitions in a nested sequence are called hypernodes.

Definition 5. The set of edges between two subsets V_i and V_j of a partition of V is denoted as $E_{i,j}$ and defined from (p, q) as

$$E_{i,j} = \cup(p,q), \;\; p \in V_i \;\text{ and }\; q \in V_j.$$

A certain hypernode represents a collapse from a set of nodes into one single node. Since hypernodes correspond to a partition of V, and the sets in a partition are none overlapping, Definition 4 implies that hypernodes represent disjoint subsets of V. Huang and Lai [19] apply a similar definition, but they use the terms abstract node, supernode and metanode. For edges among abstract nodes they use the term abstract edge.

An alternative to our term hypernode might be abstract node, but the graph itself is an abstraction of some features in the real world. From this perspective the term abstract node is not appropriate. Metanode is another term we have evaluated, but in geographical information science (GIS) one talks about meta-information, i.e., information about information. Therefore, metanode can give false associations in the literature dealing with GIS. Supernode or aggregated node are probably the best alternatives, but the term hypernode is already used in the work of NATO-group IST-059/RTG-025 [3]. For the reasons stated, we keep the term hypernode, but in order not to confuse with the established theory of hypergraphs [8,10], we will not use the terms hypergraph, hypernetwork or hyperedge.

From the hypernodes a generalized graph, i.e., a coarser graph, G_1 of graph G_0 can be constructed. This procedure can be repeated in a recursive manner. In this way a hierarchy $\mathbf{T}$ of graphs can be derived as

$$\mathbf{T} = \{G_0, G_1, \ldots, G_k\} \;\text{ where }\; |G_i| > |G_{i+1}| \;\text{ and }\; |G_k| \geqslant 1.$$

Here, $|G|$ is the number of nodes of G, G_0 denotes the original graph and G_k the top level, i.e, the most generalized version of the graph.

Since a hypernode represents a generalized view of a set of nodes, information about the graph is lost during the mapping from one level in the hierarchy to the next. This is the price to be paid for the reduction of complexity. The question to be raised, is what type of information in the graph need to be maintained.

3.1. The weight of the edges between the hypernodes

In the forthcoming a class of views to the computation of the weight of the edges between hypernodes will be identified.

Assume two sets V_i and V_j of a partition P and their corresponding hypernodes h_i and h_j. If we regard all the edges between V_i and V_j to participate in the computation of the strength of the edge between h_i and h_j, the arithmetic mean value $w_1(\cdot)$ represents one view, i.e.,

$$w_1(h_i, h_j) = \frac{1}{|E_{i,j}|} \sum_{e \in E_{i,j}} w(e).$$

Flake et al. [18] measure the strength of the edge between abstract nodes as

$$w_f(h_i, h_j) = \frac{1}{\min(|V_i|, |V_j|)} \sum_{e \in E_{i,j}} w(e).$$

Here, the average is weighted as the inverse of number of nodes. The relation between the measure of Flake et al. and the arithmetic mean value, can be written as $w_f(\cdot) = \beta w_1(\cdot)$, where $\beta = |E_{i,j}| \,/\, \min(|V_i|, |V_j|)$. In the case that $|E_{i,j}| = \min(|V_i|, |V_j|)$, $w_f(\cdot) = w_1(\cdot)$.

Another approach is to select the minimum or maximum value, i.e.,

$$w_{-\infty}(h_i, h_j) = \min(w(e) \mid e \in E_{i,j})$$

or

$$w_{+\infty}(h_i, h_j) = \max(w(e) \mid e \in E_{i,j}).$$

In the case of computing the average value $w_1(\cdot)$, the question about the uncertainty of $w_1(\cdot)$ can be raised.

According to the Gaussian error propagation the variance $\hat{\sigma}^2$ of the average value of some uncorrelated observations $X = \{x_1, x_2, \ldots, x_n\}$ is computed as $\hat{\sigma}^2 = \sum_{i=1}^{n} \sigma_i^2 / n^2$.

The strength q, i.e., the weight of an observation, is computed as the inverse of its variance. This enables us to identify $1/\hat{q} = (\sum_{i=1}^{n} 1/q_i)/n^2$. By reorganizing we get

$$\hat{q} = \frac{n^2}{\sum_{i=1}^{n} 1/q_i} = n \frac{n}{\sum_{i=1}^{n} 1/q_i} = \beta \psi(Q),$$

where $\beta = n$, $\psi(Q)$ the harmonic mean of the weights and $Q = \{q_1, q_2, \ldots, q_n\}$. A Gaussian weight $w_g(\cdot)$ of the edges between the hypernodes can therefore be defined as

$$w_g(h_i, h_j) = \beta w_{-1}(e | e \in E_{i,j}), \quad \text{where } \beta = |E_{i,j}|,$$

and $w_{-1}(e)$ the harmonic mean of the weights of the edges.

The Gaussian model yields when the uncertainty of the average value is to be computed, but we also propose it as a metaphor for the computation of the weights of the edges between hypernodes in general. This enables us to identify a class of average operators defined by the generalized mean $w_\alpha(\cdot)$ as

$$w(h_i,h_j) = \beta w_\alpha(h_i,h_j) = \beta\left[\frac{1}{|E_{i,j}|}\sum_{e\in E_{i,j}} w(e)^\alpha\right]^{1/\alpha}, \quad w(e)>0 \tag{1}$$

where $\alpha \in \mathbb{R}$ and β is a modifier defined in Table 1.

When $\alpha \to -\infty,\ w_\alpha(\cdot)$ corresponds to the minimum operator, $\alpha = -1$ defines the harmonic mean, $\alpha \to 0$ in the limit case approaches the geometric mean, $\alpha = 1$ the arithmetic mean and $\alpha \to \infty$ the maximum operator. An example in Fig. 3 illustrates how the generalized mean relates to α.

The Gaussian model considers the number of edges between V_i and V_j. For example, if all the edges between V_i and V_j have weights equal to one, the strength of an edge between two hypernodes is equal to the number of edges connecting V_i and V_j, i.e., $w(h_i,h_j) = |E_{i,j}|$.

Eq. (1) does not define restrictions on the combinations of α and β. For example, $\alpha = -\infty$ and $\alpha = +\infty$ combined with $\beta = |E_{i,j}|$ can be said to represent a pessimistic and an optimistic view of the Gaussian model, respectively.

3.2. The weight of the hypernodes

The weight of the hypernodes can be computed in a similar way as the weight of the edges between the hypernodes as

$$w(h_i,h_i) = \beta w_\alpha(h_i,h_i) = \beta\left[\frac{1}{|V_i|}\sum_{e\in E_{i,i}} w(e)^\alpha\right]^{1/\alpha}, \ w(e)>0, \tag{2}$$

where $E_{i,i}$ denotes the self-connecting edges in V_i, i.e., $w(e|e \in E_{i,i})$ defines the weight of the nodes in V_i, $\alpha \in \mathbb{R}$ and β is a modifier defined in Table 2.

Table 1
Computation of the weight of the edges between hypernodes; values of β and their interpretation.

β	Interpretation
1	The number of edges or nodes is not considered; simple average model when $\alpha = 1$
$\lvert E_{i,j}\rvert\ /\ \min(\lvert V_i\rvert, \lvert V_j\rvert)$	The number of nodes in V_i and V_j is considered; index used by Flake et al. [18] when $\alpha = 1$
$\lvert E_{i,j}\rvert$	Consider the number of edges; Gaussian weight model when $\alpha = -1$

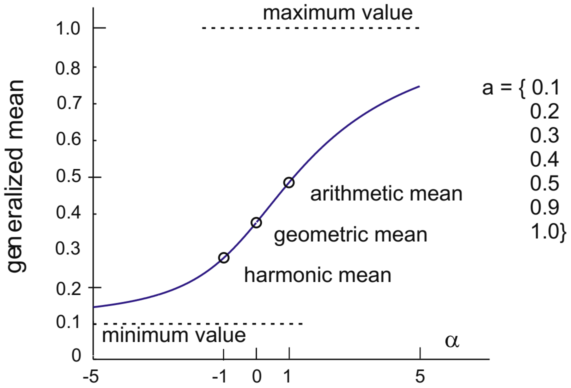

Fig. 3. Example on how the generalized mean relates to parameter α. The example is based on data vector $a = [0.1, 0.2, 0.3, 0.4, 0.5, 0.9, 1.0]$.

Table 2
Computation of the weight of the hypernodes; values of β and their interpretation.

β	Interpretation
1	The number of nodes is not considered; simple average model when $\alpha = 1$
$\lvert V_i\rvert$	Consider the number of nodes; Gaussian weight model when $\alpha = -1$

4. Algorithm to compute the hypernodes

Our algorithm to derive the hypernodes is based on the reordering of the adjacency matrix *R* of a weighted network *G*, see Fig. 4 for an illustration of a weighted network and its adjacency matrix. The reordering of *R* is the first step in the algorithm and the second step is partitioning the reordered matrix into similar groups, i.e., finding the hypernodes of the initial network. Huang and Lai [19] also base their algorithm on finding similar rows in a matrix representation of the network, but the matrix in their case is not an adjacency matrix, but a node similarity matrix. They measure the similarity between the nodes by the Jaccard coefficient. The Jaccard coefficient *J*(*A*,*B*) for two sets *A* and *B*, see [22] for example, is defined as the size of the intersection divided by the size of the union of two sets as

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}.$$

Since Huang and Lai only deal with unweighted graphs, the Jaccard coefficient can be applied.

In order to find the global best ordering of the rows or columns, all combinations of rows or columns should be investigated and global similarity measures introduced. The computational complexity of this algorithm can be reduced to polynomial time by searching for an approximation to the global best ordering. The *k*-means cluster technique can be applied to the reordering, but this method requires that seed nodes are defined, see for example [19]. A method which does not require a priori information about the clusters, but which is more time consuming than *k*-means clustering, is applied in our implementation.

In the first step the row vector with maximum length is searched. The length *l*(*p*) of a vector *p* with *n* elements is computed as

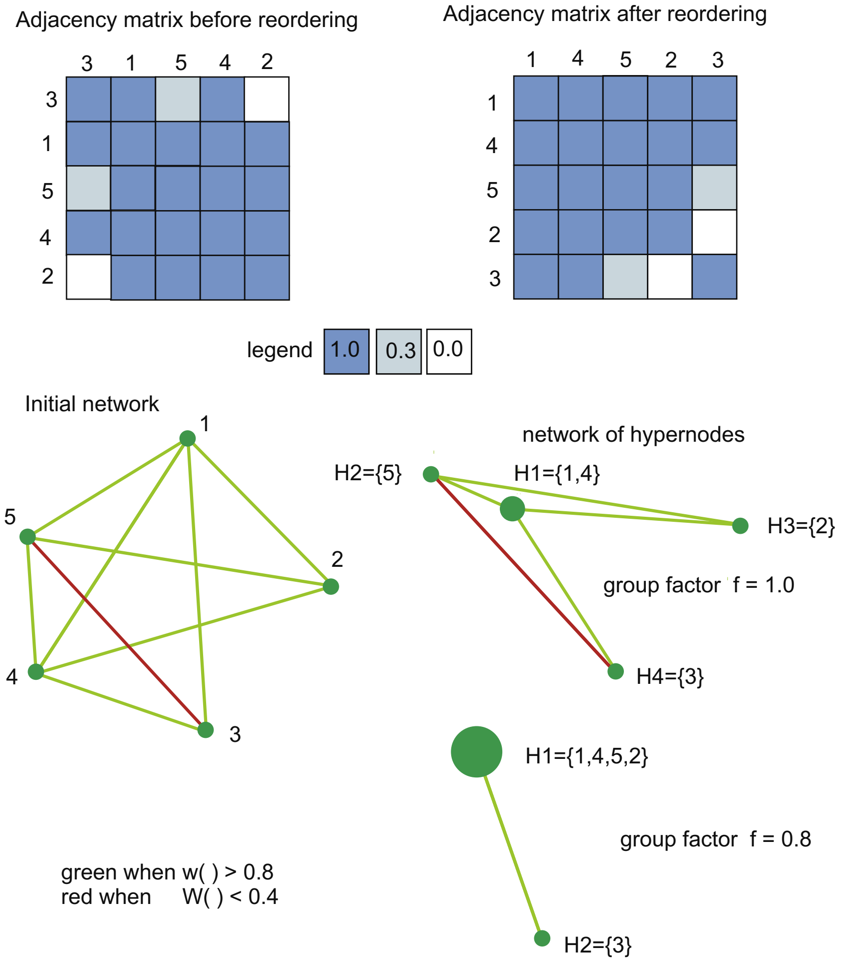

Fig. 4. A weighted network and its adjacency matrix. The strength of the connection between two nodes in the network is colour coded. The hypernodes are generated from two settings of the group factor, i.e., *f* = 1.0 and *f* = 0.8.

$$l(p) = \sum_{i=1}^{n} |p_i|. \tag{3}$$

Then the row with maximum $l(p)$ and row 1 are swapped. Thereafter, the Manhattan distance is used as a metric in the reordering. The Manhattan distance between two vectors p and q of length n is defined as

$$d(p,q) = \sum_{i=1}^{n} |p_i - q_i|. \tag{4}$$

An overview of similarity and distance measures is given in [22], for example.

The idea is to reorder R so that $d(p,q)$ is as small as possible for any two adjacent rows in R. When the reorderable matrix is regarded as a grey value image, reordering the rows, or the columns, can be regarded as minimizing the entropy of the image [13]. A pseudo code of the implementation is shown in Table 3. Since local measures are applied, the question about the robustness of the reordering can be raised. If it should happen that similar rows appear distant in the reordered matrix, they may be assigned different hypernodes; but nodes which are not aggregated at a certain level in the hierarchy of hypernodes, have the possibility to be aggregated at a higher level. Therefore, the hierarchy of hypernodes introduces some kind of robustness.

The time complexity of the algorithm is $O(n)^3$. Therefore, if the algorithm is to be applied to huge networks, for example if $n \gg 1000$, the computing time should be considered, i.e., implement methods to limit the exponential growth of the computing time. However, the problem of fast computing is outside the scope of the present paper.

In the second step clusters of similar rows in the reordered matrix is to be computed. Table 4 shows a pseudo code of the grouping algorithm. Since this grouping represents a critical step in the creation of the hypernodes, the analyst should be

Table 3
Pseudo code to reorder the adjacency matrix.

```
Reorder the rows (or the columns) of the adjacency matrix

Definitions:
              P is the set of rows of the initial adjacency matrix R
              n is the number of rows in P
              P(i:j) denotes the set of rows in P with index from i to j
              p(i) is row number i
              l(q) is the length of a vector computed from Eq. (3) Manhattan distance is defined by Eq. (4)
(1)           swap p(1) and the row p(q) in P with maximum length l(q)
(2)           for i=2 to n do
(3)             find the row p(q) in P(i:n) which has minimum Manhattan distance to p(i-1)
(4)             swap p(q) and p(i)
              end
```

Table 4
Pseudo code to define groups of rows in the reordered adjacency matrix.

```
Compute groups of rows (or columns) in the reordered adjacency matrix

Definitions:
              P is the set of rows of the initial adjacency matrix R
              n is the number of rows in P
              P(i:j) denotes the set of rows in P with index from i to j
              p(i) is row number i
              f is a user specified factor that defines the strength of the group membership
              Tanimoto coefficient is defined by Eq. (5)
(1)           j=1
              finished=false
(2)           repeat
(3)             for i=j+1 to n do
                    compute the Tanimoto coefficient T for
                    row p(j) and p(i)
(4)                 if T < f
                      create group P(j:i-1)
                      j=i
                      go to step (5)
                    end if
                end do
(5)             if j=n or i=n
                    create group P(j:n)
                    finished=true
                end if
(6)           until finished
```

able to control the grouping criteria. Therefore, an index that measures the distance between rows in the interval [0,1] is appropriate. When p and q are real valued vectors, the Tanimoto coefficient, which is an extended Jaccard coefficient, can be applied to compute the similarity between the vectors. The Tanimoto coefficient is defined as

$$T(p,q)=\frac{(p,q)}{\|p\|^2+\|q\|^2-(p,q)}, \tag{5}$$

where (p,q) is the standard inner product on $\mathbb{R}^n$, and $\|p\|^2=(p,p)$, see [22], for example.

The Tanimoto coefficient can also be written as

$$T(p,q)=\frac{(p,q)}{\|p-q\|^2+(p,q)}.$$

Some might react to the presence of negative elements in this formula, but this form simplifies the analysis of the coefficient. From the rewritten Tanimoto coefficient it is easily seen that $T(p,q)=1$ when $p = q$ and $T(p,q)=0$ when $(p,q)=0$. When the components are restricted to positive values only, T is bounded by zero and one. If the vector components can have negative values, it can be shown that T is bounded above by one and below by $-1/3$.

Fig. 4 demonstrates the algorithm. The initial network and its adjacency matrix are shown. After reordering of the matrix, the sequence of nodes is 1, 4, 5, 2, and 3. With the group factor $f = 1$, nodes 1 and 4 form a hypernode, i.e., hypernode H1. Nodes 5, 2, and 3 cannot be merged to any other node with $f = 1$. The similarity between nodes 1 and 5, 1 and 2, 1 and 3 is $T(1,5)=0.90$, $T(1,2) = 0.80$ and $T(1,3) = 0.69$. Therefore, if the group factor is lowered to 0.8, nodes 1, 4, 5, and 2 are merged into a single hypernode, see Fig. 4. This demonstrates the significance of the group factor to the creation of the hypernodes. The group factor is vital to the creation of the hypernode and therefore careful consideration must be made. Furthermore, the group factor is an application dependent parameter.

5. Demonstrations of the algorithm

An implementation of the present algorithm will be demonstrated in three case studies. The first two cases are gathered from the literature, the third is based on open source data about flight traffic in USA.

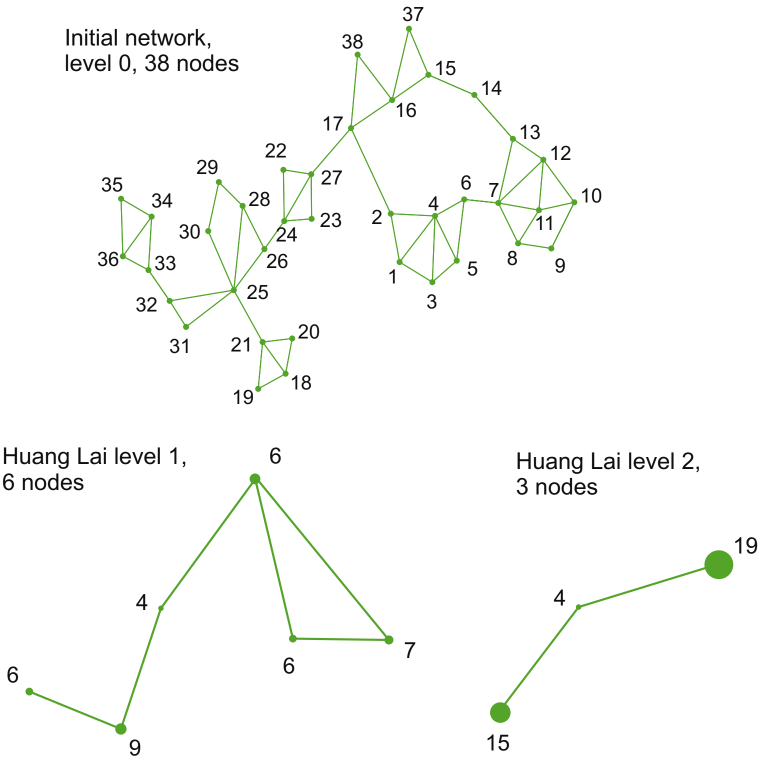

Fig. 5. Hierarchy generated by Huang and Lai [19, p. 239].

5.1. Network used by Huang and Lai

Our implementation is tested against an example given by Huang and Lai [19], see Fig. 5. The initial network is composed of 38 nodes. All the edges have the same weight. Huang and Lai show that their algorithm can generate 2 levels of hypernodes as shown in Fig. 5. Our algorithm handles the same network as depicted in Fig. 6. Here, the ordinary weight computation is used and the group factor is set to 0.5. The computed hierarchy consists of 5 levels. At the top level all the nodes are aggregated into one single hypernode.

Our level three with seven hypernodes is similar to Huang-Lai level one with six nodes, i.e, our model has one hypernode more. The Huang-Lai model level two has three nodes and is similar to our level four with four nodes. The most striking difference between the two hierarchies is that our hierarchy has more levels than Huang-Lai, i.e., our algorithm allows the network to grow more gradually from the initial network to the one hypernode at the top level of the hierarchy.

Fig. 7 shows how the selection of the Gaussian weight model effects to the hierarchy. At each level the weights are normalized, i.e., they are assigned a number in the interval $[0, 1]$. As seen from Fig. 7 the green edges have more subedges than the yellow and red edges. Therefore, the green edges have higher weights than the yellow and red edges. All the green, yellow and red edges at level 1 have three, two and one sudedges, respectively.

In the initial network all the edges have equal weights. Therefore, at level 1 the Gaussian model assigns weights proportional to the number of subedges between the hypernodes. At level 1 the hypernodes are identical in the two cases, simple and Gaussian weight model, since the effect of the weight model takes place at first from level 2.

The main difference between the two models is that the Gaussian model considers the number of subdedges, whereas the simple model does not. This can be seen at level 3 in Fig. 7. Here, the nodes 18, 19, 20 and 21 in the initial network are grouped into a hypernode which have only one subedge that connects it to the rest of the network, see the yellow edge from hypernode A at level 3. In the Gaussian model this weak connection is detected and therefore this group of nodes must wait to be merged to the rest of the network until a higher level in the hierarchy, i.e., at level 5. This demonstrates how the Gaussian model considers the number of subedges when the hypernodes are generated.

The Gaussian weight model gives a hierarchy that matches the Huang-Lai computation slightly better than the simple weight model. Level 5 in the Gaussian model is identical to level 2 in the Huang-Lai model, whereas our level 4 model is almost identical to Huang-Lai level 1. The two models deviate with respect to the clustering of our hypernode A, see level 3. The Gaussian model has considered the weak edge from hypernode A and not aggregated this node at level 4.

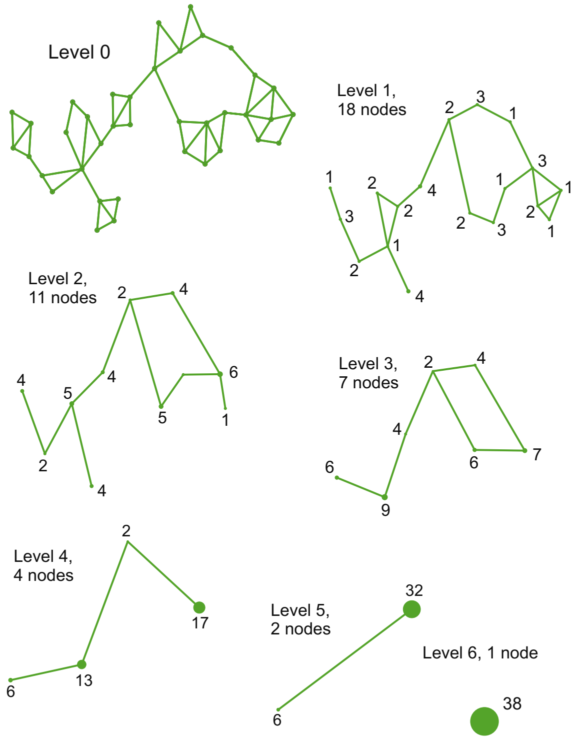

Fig. 6. Hierarchy of hypernodes based on ordinary weight computation, i.e., $\alpha = 1$ and $\beta = 1$. The group factor is set to 0.5. The number of subnodes in each hypernode is given.

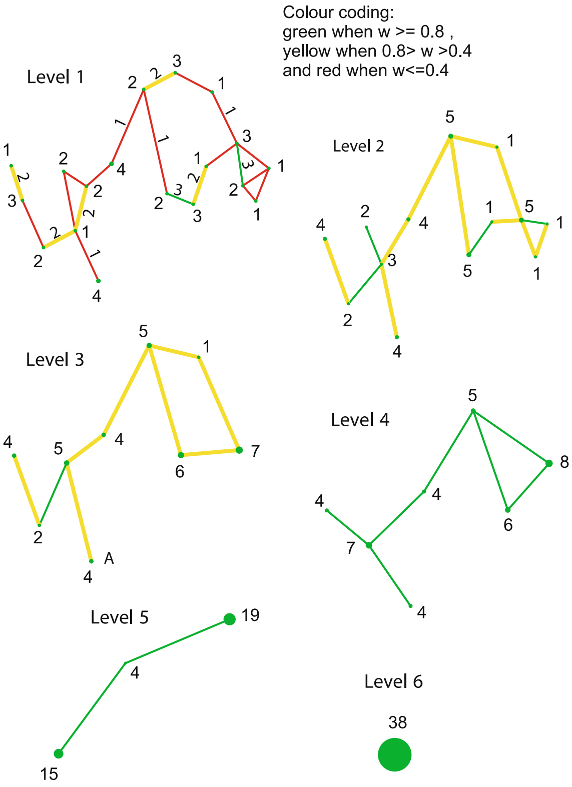

Fig. 7. Hierarchy of hypernodes derived from the Gaussian weight model for the edges, i.e., $\alpha = -1$ and $\beta = |E_{ij}|$, and the ordinary weight computation for the nodes. The group factor is set to 0.5. The number of subnodes are given. For the edges between the hypernodes their number of subedges is given at level 1. Due to the visual clarity of the figure some numbers are left out, but the green, yellow and red edges at level 1 have three, two and one subedges, respectively.

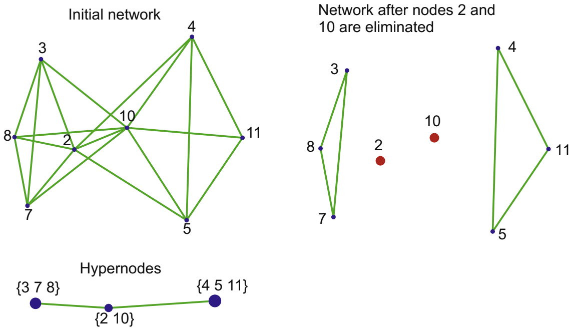

Fig. 8. Hypernodes generated from a network used by Estrada and Rodríguez-Velázquez [5]. Group factor is set to 0.8.

5.2. Network used by Estrada and Rodríguez-Velázquez

Estrada and Rodríguez-Velázquez [5] define centrality indices on hypergraphs which they demonstrate on the network in Fig. 8. According to their index $SC(i)_H$ ([5], p. 588) nodes 10 and 2 are ranked at the top, i.e., the two nodes play an important role for the connectivity of the network. At a certain level of the hierarchy of hypernodes three nodes are generated where one of them is composed of subnodes 10 and 2 in the original network.

From Fig. 8 it is easily seen that at a certain level of abstraction the network can be regarded as a chain composed of three nodes. Therefore, if the hypernode in the middle of the chain is eliminated, the network will be separated into two disjoint sub-networks, see Fig. 8. To arrive at the hypernodes shown in the case considered, an appropriate selection of the group factor is necessary. In our case the factor 0.8 is selected. The strength of the hypernode method is its ability to visualize the network structure at higher levels of abstraction. Centrality measures are used to rank the nodes, but lack a tight connection to visualization of the network structure. In closing, the hypernode approach provides additional salient information about networks.

5.3. Flight traffic in USA

Data from open source offered by the US Bureau of Transportation Statistics [23] is used. From this database traffic information about 468 airports in USA for a certain period was gathered. In this example data the number of traffic routes, i.e., the edges, in the network is 16292. Fig. 9 illustrates the problem of visualizing a network with that amount of edges. The airport with the highest number of edges in the network, i.e., ATL, with its 195 edges is selected for the illustration. Fig. 10 shows

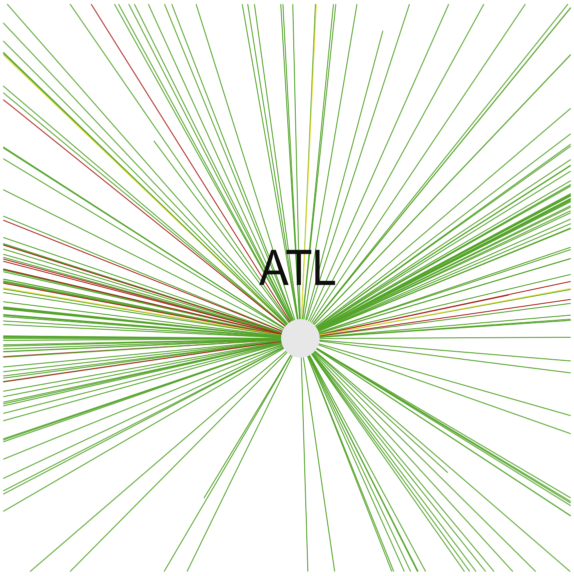

Fig. 9. The 195 edges connected to airport ATL.

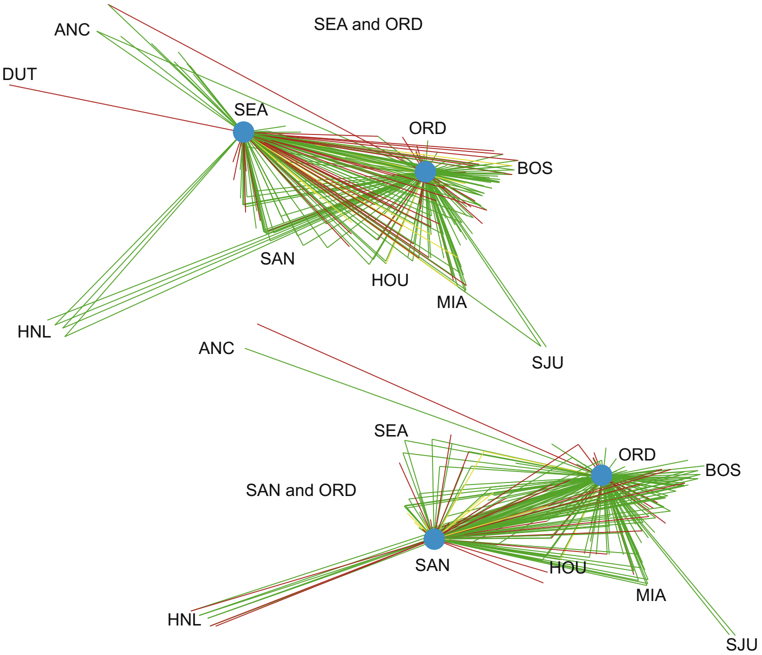

Fig. 10. Illustration of the density of edges in the network. The three airports SEA, ORD and SAN are selected.

three typical airports, SEA, ORD and SAN, where a large number of edges is observed. From the examples it is clear that simplified models of the network must be applied in order to understand the networks.

After the end nodes are moved to their mother node, the network is reduced from 468 to 420 nodes. Fig. 11 shows some examples of the end nodes, i.e., the blind alleys in the network. The reduced network serves as the starting point for creating the hierarchy of hypernodes.

For each route e_{ij} between any two airports i and j we have information about the number n_{ij} of passengers. From this information a weight $w(e_{ij})$ is derived and associated to the edge. The weights are normalized, i.e., given a number in the interval $[0,1]$. A median filter is applied so that a passenger number greater than the median is transformed to weight 1, else a linear relationship is applied as

$$w(e_{ij}) = \begin{cases} \frac{n_{ij}}{m} & \text{for } n_{ij} < m \\ 1 & \text{else}, \end{cases}$$

where m is the median of n_{ij} for all edges in the network. In the forthcoming examples the Gaussian model is applied for the edges between the hypernodes, and the simple weight computation is used for the hypernodes, i.e, $\alpha = -1$ and $\alpha = 1$, respectively.

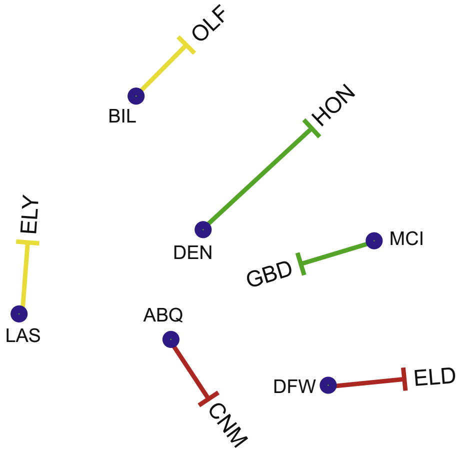

Fig. 11. Totally there are 68 blind alleys in the network. The figure shows some of them.

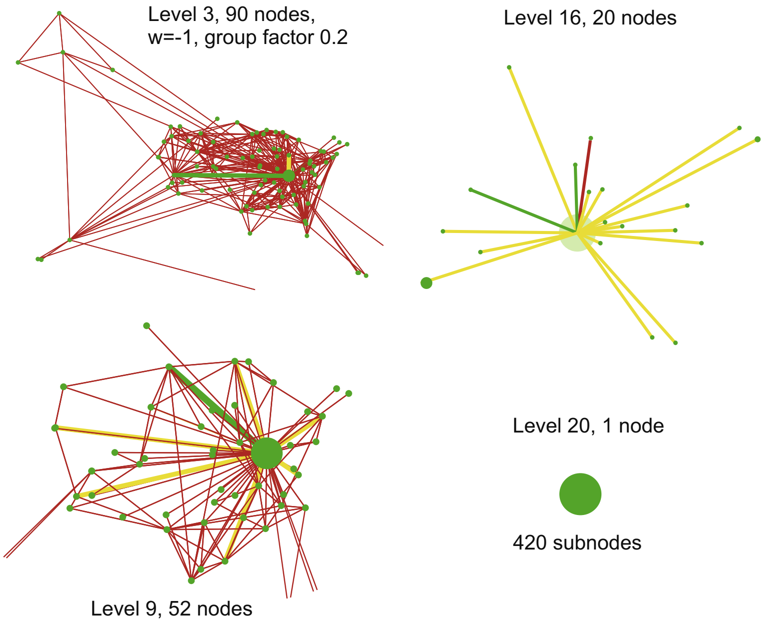

Fig. 12. Four levels of the hierarchy of hypernodes. At level 20 all the nodes are clustered into one hypernode.

When the group factor is set to 0.2, the hierarchy of hypernodes in Fig. 12 is derived. Four levels in the hierarchy are selected for the visualization. At the top level all the subnodes are merged into one hypernode. This shows that the network is not composed of disconnected groups, i.e., there is a path from one node to any other node in the network.

Since the network is geographically located, it is of interest to investigate how the hypernodes relate to the geography, see Fig. 13. In order to maintain the clarity of the figure, only the four most dominating hypernodes are selected for visualization at each level of the hierarchy. All the other airports are shown by the yellow + symbol.

The geographical grouping of the nodes is conspicuous. At level 3 a group of black circles can bee seen in the north. Southwest of this group there is a blue group and in west a red group stands out. The green group grows gradually from level to

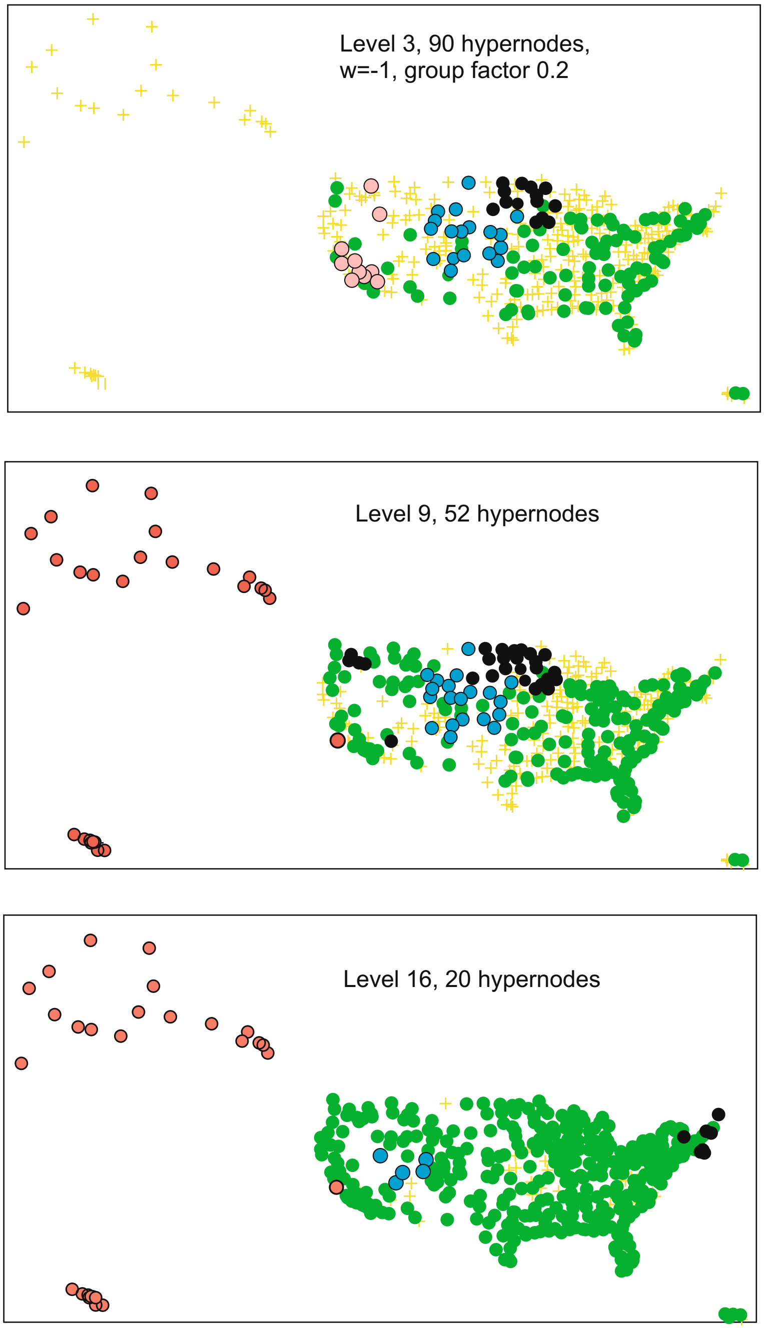

Fig. 13. The hierarchy of the hypernodes illustrated by colour coding the subnodes of the four largest hypernodes. The geographical clustering of the groups is striking.

level in the hierarchy. This group starts at level 3 with a dissemination mainly in the east, but there are also some members of the group in the west.

Fig. 14 zooms into the four most dominating hypernodes at level 3. Here, airports like BOS, EWR, BUF, ORD, ATL, MIA, HOU, DEN, LAS, SAN and SEA belong to the green group, i.e., airports close to the large cities in the US. The black group in the north has members like DVL, TVF and EAU. The blue group in the middle of US is made up of airports in the neighbourhood of SAF, LBF, GCC and ISN. In the south-west we find that SBA, BOI and GEG are members of the red group. The black, blue and red groups characterize small cities in the US. At level 9 a new distinct geographical group stands out, i.e., the red group (mainly Alaska and Hawaii).

Not all levels of the hierarchy are shown in Fig. 14, for example the lowest level of hypernodes. This level differs from level 3 in the way that the group of green nodes at level 3 are separated into four groups (one large and smaller groups). Despite this separation the geographical clustering is evident.

Lessons learned 2. *The lesson learned from the airport case so far is that for networks embedded in the geographical space, the spatial concentration of the subnodes of the hypernodes should be offered attention when tools for network visualization is developed.*

The two parameters group factor and level in the hierarchy can be used to answer questions about which nodes are most similar or which nodes are different from the main group. In both cases we can talk about anomalies. For threat detection, for example, anomalies can represent important information. Some examples will clarify the use of the two parameters considered.

At first, the level in the hierarchy will be elaborated upon. Fig. 15 shows details of the hypernodes at level 16. The nodes that still are separated at a this level, are very different from the main group. If this were not the case, the nodes would have been merged at a lower level. The network is at level 16 reduced from 420 to 20 nodes and can be described as a star network.

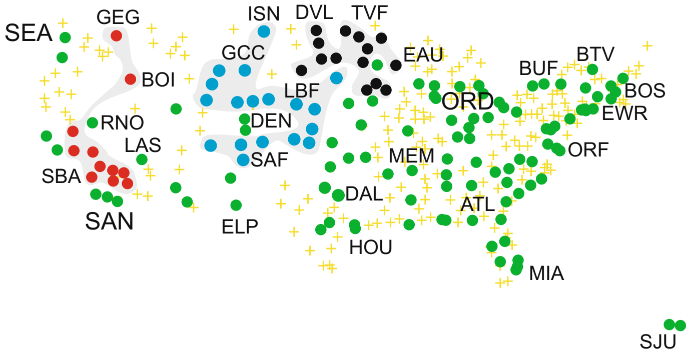

Fig. 14. Zoom into the four most dominating hypernodes at level 3, group factor 0.2.

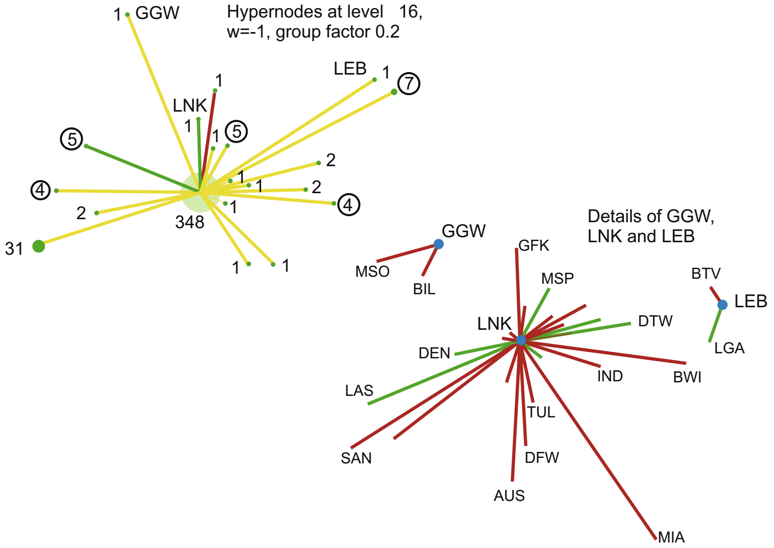

Fig. 15. Details of the simplified network at level 16 in the hierarchy. The number of subnodes connected to the hypernodes is shown. The five numbers marked with a black circle will be used for further references. The three nodes GGW, LNK and LEB are selected for detailing.

The hypernode with 31 subnodes corresponds to the red group at level 16 in Fig. 13, i.e., the Alaska/Hawaii group. Since this group is not merged to the main group at a high level in the hierarchy, the Alaska/Hawaii group is very different from the main group. Details of the tree nodes GGW, LNK and LEB are shown in Fig. 15. This illustrates that the hypernodes of the star are loosely connected to the node in the center. For LNK we can count 22 edges, but compared with the 195 edges of ATL, for example, LNK represents a node with few edges. Therefore, LNK is moved to the central hypernode first at the top level of the hierarchy.

Some of the hypernodes of the star considered, are also illustrated in Fig. 16. The details of the group marked with the character A, i.e., the group with four subnodes close to MRY in the south-west, shows how the nodes of the group are internally connected and how they are loosely connected to the surrounding nodes.

There is also a geographical perspective here. From Fig. 16 it can be derived that the hypernodes form spatial clusters. This observation confirms the previous proposition that for networks embedded in the geographical space, tools to visualize spatial properties of the hypernodes are useful in developing knowledge about the network structure. From the study about anomalies, the following proposition can be set out.

Lessons learned 3. *Nodes in the US flight network which are very different from the main group, may define geographical clusters.*

The next question we will ask is about which nodes are most similar. To answer this question a high similarity factor is selected. When factor 0.8 is used, the hierarchy reaches its maximum hight at level 3, i.e., only a few number of very similar nodes can be merged into hypernodes with the selected group factor. Fig. 17 shows the hypernodes in this case. The model consists of 15 hypernodes with at least two subnodes. Due to graphical limitations only ten of the hypernodes are shown in the figure.

The hypernodes are composed of mostly two subnodes, but two of them have four subnodes, i.e, {EUG MFR ACV RDD} and {DAL AUS SAT HOU}.

The traffic pattern of two of the airports in the last group is illustrated in Fig. 17. The visual impression verifies the high similarity of the two nodes. Among the hypernodes with two subnodes the following hypernodes can be mentioned: {LGA EWR}, {TPA FLL}, {YAK CDC}, {BON PSE}, and {ROC SYR}. The geographical nearness of the subnodes of the hypernodes is evident.

Lessons learned 4. *Airports in the US flight network which have very similar traffic patterns, are located close to each other. This verifies the importance of considering the spatial nearness when analyzing networks which are embedded in a geographical space.*

6. Discussion

Three parameters play an important role in controlling and guiding the creation of the hypernodes: (1) the group factor *f*, (2) the level in the hierarchy, and (3) the weight model used to compute the strength of the edges and the nodes.

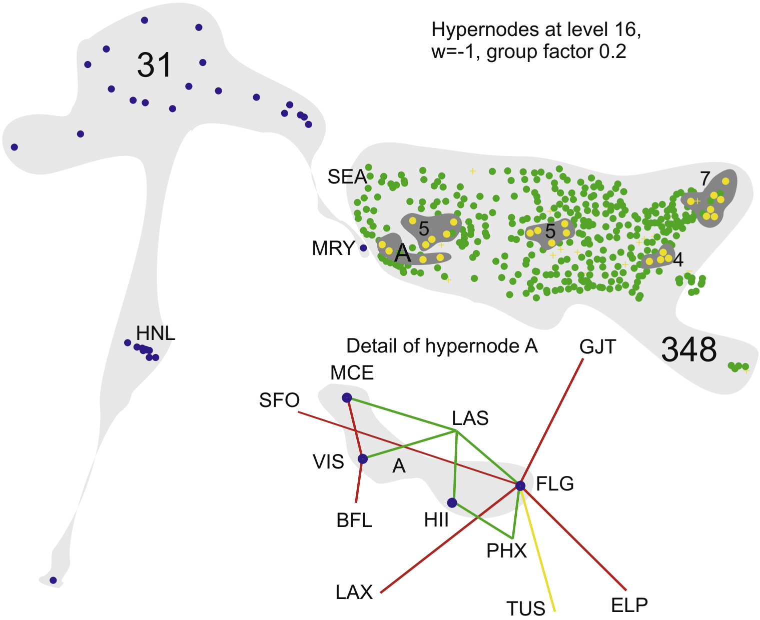

Fig. 16. Some selected hypernodes at level 16. The five small clusters correspond to the five hypernodes in Fig. 15 which are marked with a black circle.

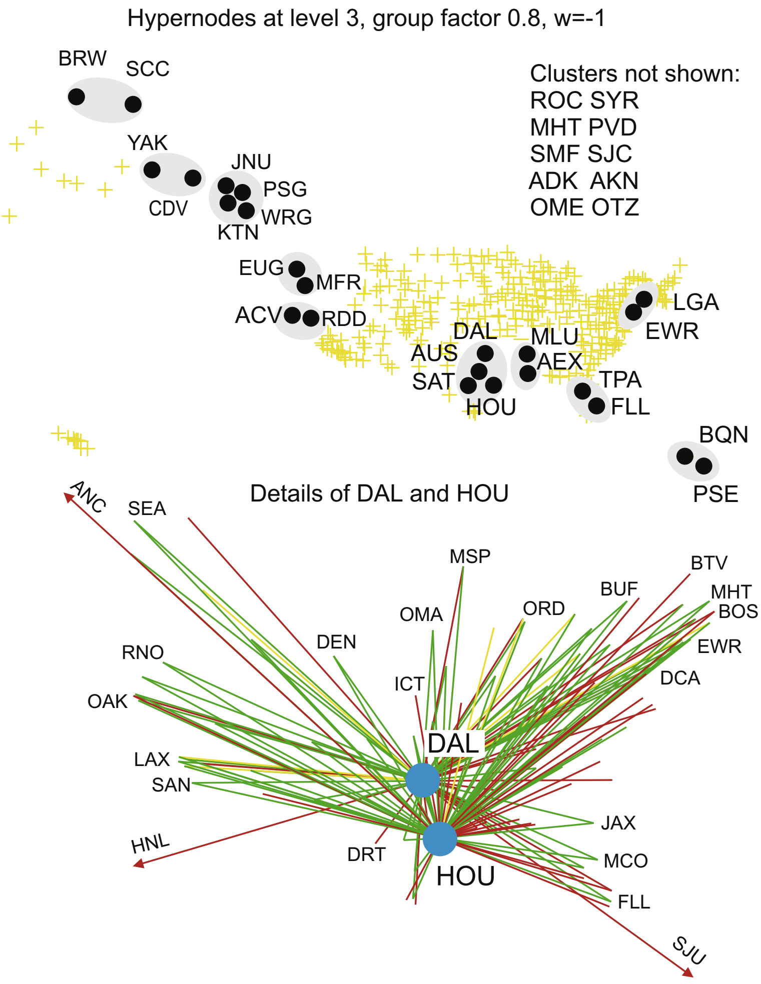

Fig. 17. Nodes with similar traffic patterns, group factor 0.8. The hypernodes are marked with the grey coloured ellipses. Out of the fifteen hypernodes ten of them are marked in the map. Details of two nodes, i.e., DAL and HOU, which belong to the same hypernode are shown.

Therefore, in an implementation of the system tools should be provided so that the user can compare the different settings of the parameters. In the airline case, for example, different settings of *f* was used to find anomalies in the network, i.e., very similar nodes or groups of nodes that are very unlike the other nodes in the network. User guides that hints the consequence of the different settings can be helpful.

The reordering proposed in Table 3 is based on local optimization. Since global measures are not applied, it may happen that nodes with similar topologies are not merged at a certain level in the hierarchy, but at a higher the level the two nodes may be aggregated into the same hypernode.

A hypernode does not provide detailed information about its subnodes and their connections. Therefore, there is a need to implement the panning and zooming functionalities to allow the global and detail view of the network, i.e., tools to visualize information about the underlying subnodes and their connections.

When the network is embedded in a geographical space, the possible geographical grouping of the nodes should be considered. Tools to visualize the geographical nearness of the subnodes in the hypernodes are therefore of interest in exploration and hypothesis generation about the geographical component of the network.

The mapping of the network to a position in the plane is not obvious when the network is embedded in the geographical space, i.e., the hypernodes have no natural point location since they are aggregates of nodes and therefore are related to a geographical area. The question is therefore: Where to position the hypernodes? One solution to this problem is to compute the position of a hypernode as the average position of its subnodes. This method was used in the airline case.

Fabrikant et al. [24] use the term spatialization when mapping non spatial data to an information display. Information spatialization is inspired by the analogy that the strength of relatedness in the data space should be mapped to neighbourhood in the information display, such that semantically similar nodes are placed closer to one other than less similar ones. An empirical study suggests that the distance-similarity metaphor applies to network spatializations by equating metric distance along network lines to similarity. They also find that line size, colour value and hue, modify the distance-similarity metaphor in subtle yet logical ways. An implementation of the spatialization principle is not straight forward, since the mapping of a weighted network to a two-dimensional plane cannot allays guarantee that the strength of the edges is mapped to neighbourhood in the plane.

The time complexity of the proposed implementation is $O(n^3)$, where n is the number of nodes in the network. Therefore, when the size of the network passes a certain threshold, for example 1000 nodes, strategies to reduce the computing time should be considered. The scalability of the algorithm is highly related to whether the adjacency matrix is sparse or not. A matrix of size $n \times n$ takes $O(n^2)$ time to traverse. If the number of edges from the nodes is small, i.e., the adjacency matrix is sparse, the node-list representation of the matrix will reduce the storage space of the matrix as well as the computing time to traverse the matrix. Huang and Lai [19] claim that their algorithm has computing time $O(n^2)$, but this seems to presume that the number of hypernodes is small compared with the number of nodes in the initial network, i.e., few clusters and that the adjacency matrix is sparse.

The interpretation of the hypernodes is a domain specific task. Therefore, to associate meaning to the hypernodes and their edges the context and weights given to the edges and nodes in the network should be considered.

7. Conclusions

The hypernode algorithm on weighted networks is described and demonstrated on real world airline data. The airline case demonstrates that the clusters of nodes in the network have a geographical assosiation. Therefore, hypernodes can be a useful concept to detect geographical patterns.

The algorithm allows networks to be studied at different levels of abstraction. In that way a high level understanding of the network can be obtained. Therefore, the hierarchical graph clustering presented has applications in many areas outside networks embedded in the geographical space.

A class of indices to measure the strength or uncertainty of the edges between hypernodes is developed. This enables the analyst to select different weight models by manipulating a few parameters.

Hypernodes can be utilized to study the effect on the network when groups of nodes or groups of edges are eliminated from the network. For example, one can ask what happens to a network when a certain hypernode is destroyed. In that way information about the vulnerability of the network can be studied.

A topic for further research is to apply the algorithm to real situations of massive data sets. This requires that the time complexity of the algorithm is considered, i.e., eventual sparsity of the adjacency matrix must be utilized.

Acknowledgements

This research is supported by the FFI-projects 112901 and 110101. We are grateful to the members of the NATO research group IST-059/RTG-025 [3] for their fruitful discussions on the visualization of networks. Our special thanks goes to Joanne Treurniet, DRDC-Ottawa, for assisting us in the literature search and for the feedback on the implementation of the hypernode algorithm. The authors would like to thank the three anonymous reviewers for their comments and suggestions which helped to improve this paper significantly.

References

[1] A. Poulovassilis, A nested-graph model for the representation and manipulation of complex objects, ACM Transactions on Information Systems 12 (1) (1994) 35–68.
[2] R. Angles, C. Gutierrez, Survey of graph database models, ACM Computing Surveys 40 (1) (2008) 1–39.
[3] IST-059/RTG-025, A framework for network visualisation, NATO, RTO Technical Report (Working draft, paper in preparation 2009). <http://www.rta.nato.int/>.
[4] J.T. Bjørke, Reduction of complexity: an aspect of network visualization, in: RTO-MP-IST-063 Visualising Network Information, NATO Conference, Copenhagen, Denmark, NATO, 2006, pp. 9–1 to 9–10, ISBNs 92-837-1156-4 or 978-92-837-1156-8.
[5] E. Estrada, J.A. Rodríguez-Velázquez, Subgraph centrality and clustering in complex hyper-networks, Physica A 364 (2006) 581–594.
[6] M. Baitaluk, X. Quian, S. Godbole, A. Raval, A. Ray, A. Gupta, PathSys: integrating molecular interaction graphs for systems biology, BMC Bioinformatics 7 (55) (2006) 1–13.
[7] J.Y. Chen, A.Y. Sivachenko, Data mining in protein interactomics, IEEE Engineering in Medicine and Biology 24 (3) (2005) 95–102.
[8] C. Berge, Graphs and Hypergraphs, North-Holland, Amsterdam, London, 1973.
[9] K. Börner, S. Sanyal, A. Vespignani, Network science, in: B. Cronin (Ed.), Annual Review of Information Science and Technology, vol. 41, Information Today, Inc./American Society for Information Science and Technology, Medford, NJ, 2007, pp. 537–607 (Chapter 12).
[10] C. Berge, Hypergraphs: Combinatorics in Finite Sets, Elsevier, New York, 1989.
[11] J. Bertin, Graphics and Graphic Information Processing, Walter de Gruyter, Berlin, New York, 1981. p. 273.
[12] J.A. Hartigan, Direct clustering of a data matrix, Journal of the American Statistical Association 67 (337) (1972) 123–129.
[13] J.T. Bjørke, B. Smith, Seriation: An implementation and case study, Computers, Environments and Urban Systems 20 (6) (1996) 427–438.
[14] N. Henry, J.-D. Fekete, Matrixexplorer: a dual-representation system to explore social networks, in: IEEE Transactions on Visualization and Computer Graphics 12(5).
[15] C.-H. Chen, H. Hai-Gwo, J. Wen-Jung, K. Chiun-How, T. Yin-Jing, T. Sheng Li, W. Han-Ming, Matrix visualization and information mining, in: Proceedings in Computational Statistics 2004 (Compstat 2004), Physika-Verlag, 2004, pp. 85–100.
[16] P. Doreian, V. Batagelj, A. Ferligoj, Generalized Blockmodeling, Cambridge University Press, 2005.
[17] H. Siirtola, E. Mäkinen, Constructing and reconstructing the reorderable matrix, Information Visualization 4 (1) (2005) 32–48.
[18] G.W. Flake, R.E. Tarjan, K. Tsioutsiouliklis, Graph clustering and minimum cut trees, Internet Mathematics 1 (4) (2004) 385–408.
[19] X. Huang, W. Lai, Clustering graphs for visualization via node similarities, Journal of Visual Language and Computing 17 (2006) 225–253.

[20] J.T. Bjørke, Framework for entropy-based map evaluation, Cartography and Geographic Information Systems 23 (2) (1996) 78–95.

[21] J.T. Bjørke, K. Sæheim, Investigation of the channel capacity of seafloor maps with coloured depth intervals, in: J. Bjørke, H. Tveite (Eds.), Proceedings of ScanGIS'2007, 11th Scandinavian Research Conference on Geographical Information Science, Ås, Norway, 2007, pp. 61–73. <http://statisk.umb.no/conf/scangis2007/papers/>.

[22] T. Kohonen, Self-organizing maps, third ed., Springer Series in Information Sciences, Springer, Berlin, 2001.

[23] RITA, Research and Innovative Technology Administration, Bureau of Transportation Statistics (2009). <http://www.transtats.bts.gov/DL_SelectFields.asp?Table_ID=258&DB_Short_Name=Air20Carriers>.

[24] S.I. Fabrikant, D.R. Montello, M. Ruocco, R.S. Middleton, The distance-similarity metaphor in network-display spatializations, Cartography and Geographic Information Science 31 (4) (2004) 237–252.