SYNERGETICS AND THEORY OF CHAOS

УДК 167.7, 517

Gritsak-Groener V. V.[1], Karpenko O. B.[2]

MATHEMATICAL METHODS OF IDENTIFICATIONS FOR VISUAL INFORMATION. 1.

[1] *University of Georgia, Georgia, USA*
e-mail: v_hrit3000@yahoo.com
[2] *International institute of Socionics,*
e-mail: olly.olga@gmail.com

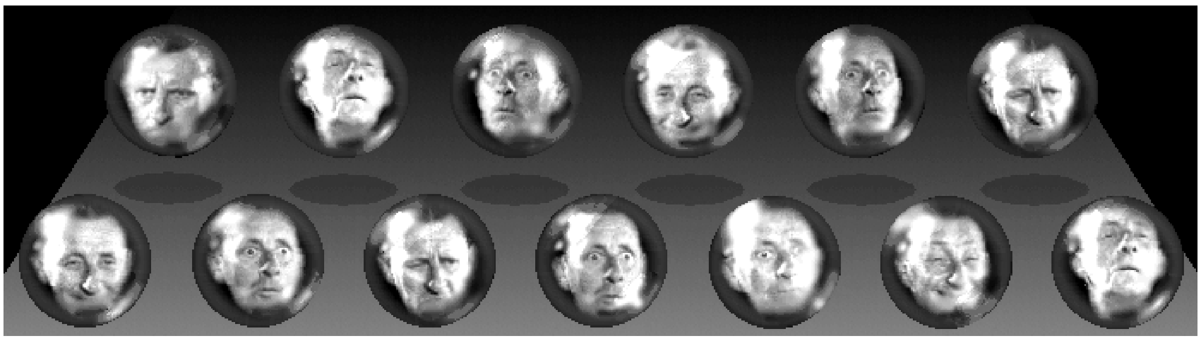
The article begins with a concise but adequate revision of basic graph-theoretic concepts which will be useful later on. It then introduces the central notion of G-shrubberies which serves as a general reference model in the rest of the work. We lead them to this notion on the basis of numerous examples, taken from sociology & socionics, author's identification of painting, mathematical psychology computer science, and mathematical life theory.

Key words: graph theory, pattern recognition, visual information.

1. Introduction

As a rule, Methods of Identification for Visual Information (MIVI) mean the recognition of the incomplete visual or graphic information, for example the recognition of the patterns in damaged photos or masked land objects, the problems of early medical diagnostics and so forth. Some of our publications have been devoted to that. We will not produce similar work in new cycle of papers, we have other purpose.

We will go further. We will consider good, accurate images, and, probably, the whole film from such images, and on their graphic characteristics, including colour, luminous and volume, we **will calculate** by algorithms, which we will place in our cycle of papers, their true, psychological, psychiatric, socionical invariants. For example, see pict. 1. It is represented the one person in 13 psychological conditions. It also is our program a minimum. "Sensitivity" of our algorithms should distinguish, a **minimum** 13 classical psychological conditions of a human face. Surprisingly, but running forward we will tell that our algorithms are much more sensitive, and we reflect, whether is the number discovered by us μ( ) psychological types a certain **universal** constant **of psychology**!



Pict. 1.

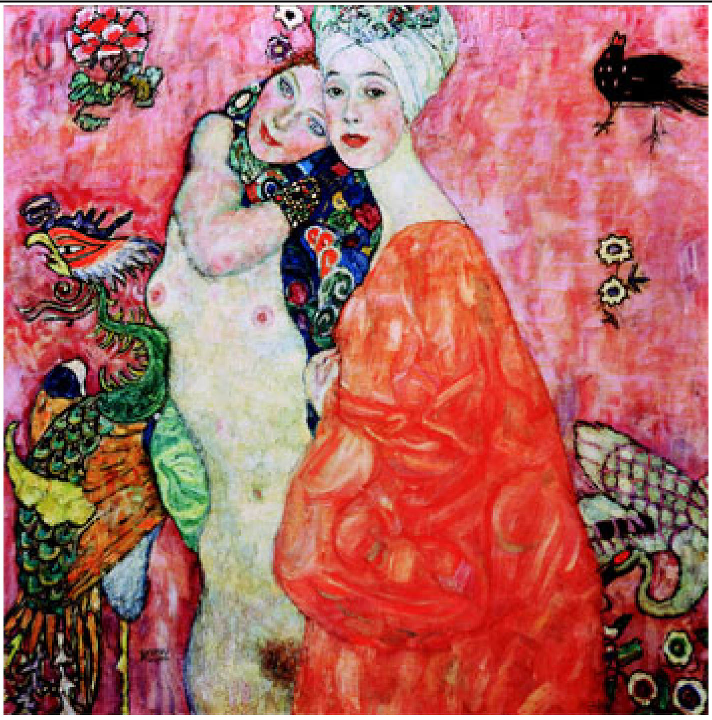
For painting, we will calculate a membership of pictures to their author **N**. It appears with each artist a certain interval of its creative individuality and as we place it not simply on a numerical axis, and in space of combinatorial random configurations then artist **N** will be especially individual, not less individual, than its fingerprint is connected .

For example, a picture 2 is immediately identified as a picture by our favourite Gustav Klimt.

Moreover, we calculate, for each picture **K** a certain psychological type. It is obvious that:

μ()> μ() and, unfortunately, we do not know true value. There is the following problem:

***Problem 1**. To discover value μ() and to describe all aspects of types.*


Pict. 2.

All enumerated problems we буде to solve in our cycle of papers **MIVI**.

So what we will consider in the first paper of cycle **MIVI**.

In this article we present a number of concepts which will be used throughout this them. After we have introduced a few elements of the terminology of graph theory, we will study shrubberies and describe the manner in which we propose to represent them in a computer (data-structure). Because of their importance in data-analysis and in sociology and socionic research we place particular emphasis on minimal shrubberies of a valued graph; a classical algorithm for their construction is studied. G-shrubberies, which constitute the "geometry" in which we work, are then introduced by means of several examples.

We then propose a program for selecting G-shrubberies at random: readers will thus be able to make up their own "data-games" with which they can amuse themselves at will. The notion of hierarchical classification is absolutely central in data-analysis and that is why we present it in conjunction with G-shrubberies. The part of this article is concluded by a set of algorithms and a program for drawing valued shrubberies (that is, shrubberies whose edges have lengths).

2. Graphs

Graphs allow one to represent the existence or absence of relations between objects. Formally, a ***graph*** is a pair

$$\mathbf{\Gamma} = (\mathbf{V}, \mathbf{E}), \mathbf{E} \subseteq \square \mathbf{V} \times \mathbf{V}, \qquad (1)$$

made up of a finite set **V** is (**t**, **u**, **v**,...) and a set **A** of two element subsets of **V**. The elements of **V** are called the ***vertices*** (or ***points***) of **Γ** and the elements of **E** are its ***edges*** (or ***arcs***). By convention, the edge $\boldsymbol{e}=(\boldsymbol{v}_1, \boldsymbol{v}_2)\in\mathbf{E}$ will simply be designated $\boldsymbol{v}_1\boldsymbol{v}_2$ and we say that $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ are ***adjacent*** or that they are ***end-points*** of the edge $\boldsymbol{v}_1\boldsymbol{v}_2$. If $\mathbf{e} = <v_1,v_2> \in\mathbf{E}$, **e** is said to be ***incident out*** of v_1 and ***incident into*** v_2. The edge $\boldsymbol{e}=(\boldsymbol{v}_1, \boldsymbol{v}_2)$ is called ***loop*** when $\boldsymbol{v}_1 = \boldsymbol{v}_2$. Edges $\mathbf{e}_1 = (v_1,v)$ and $\mathbf{e}_2 = (v,v_3)$ are called ***adjacent***.

The graph **Γ** is usually represented by a diagram: the vertices of **Γ** correspond to points in the plane and two vertices **u** and **v** are joined by a line segment if and only if they are adjacent (see fig. 1).

The set of vertices adjacent to a vertex **u** is called the ***neighborhood*** of **u** and is denoted by **deg(u)**. The number **st(u)** of elements of **deg(u)** is called the ***degree*** of the vertex **u**. For example, in the graph on Figure 1, the vertices $\mathbf{v}_6$ and $\mathbf{v}_7$ are of degree **1**; $\mathbf{v}_2$, $\mathbf{v}_3$, $\mathbf{v}_4$, $\mathbf{v}_5$ are of degree 4; $\mathbf{v}_1$ is of degree 6.

A vertex $\boldsymbol{v}_0\in\mathbf{V}$ is called **izie** of the graph **Γ** if $\mathbf{st}(\boldsymbol{v}_0) = \mathbf{0}$. The number n_Γ^E of vertices **Γ** is called ***amount*** of the graph **Γ**.

The number n_Γ^E of edges **Γ** is called ***capacity*** of the graph **Γ**. The graph **Γ** is said to be ***finite*** if $n_\Gamma^V < \infty$ and $n_\Gamma^E < \infty$. The graph **Γ** is said to be ***finite*** if **null-graph** if $n_\Gamma^V = 0$, and the graph **Γ** is called

discrete-graph if $n_{\Gamma}^{E} = 0$.

A graph $\mathbf{\Gamma_1}=(\mathbf{V_1}, \mathbf{E_1})$ is ***isomorphic to*** a graph $\mathbf{\Gamma_2}=(\mathbf{V_2}, \mathbf{\Gamma_2})$ if the following bijection φ: $\mathbf{V_1} \rightarrow \mathbf{V_2}$ conditions hold:

$$(v_1,v_2)\in E_1 \Leftrightarrow (\varphi(v_1), \varphi(v_2))\in E_2,$$

where $v_1,v_2\in\mathbf{V_1}$.

The simple graph $\mathbf{\Gamma}$ is called ***full*** if each pair of vertexes $\mathbf{\Gamma}$ is adjacent.

$$\mathbf{L} =: \{v_1\, e_1\, v_2\, e_1\, v_3 \ldots\, e_n\, v_{n+1}\},$$

where v_i are vertexes, and $\mathbf{e_i}$ are edges of graph $\mathbf{\Gamma}$, and vertexes $\boldsymbol{v_i}$ and $\boldsymbol{v_{i+1}}$ are final vertexes of an edge $\mathbf{e_i}$. The vertex$\boldsymbol{v_1}$ is ***input***, and top $\boldsymbol{v_{n+1}}$ is ***exit*** of foot-path $\mathbf{L}$.

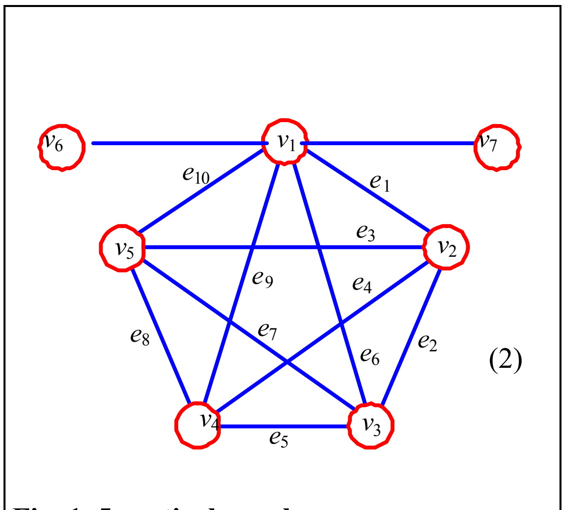


Fig. 1. 5-vertical graph $\wp$

Path $\mathbf{L}$ is a ***way*** (or a ***route***) if all its edges pairwise are different from each other. ***Length*** of path $\mathbf{L}$ is the amount of edges which enter into it (number $\mathbf{n}$ from (2)).

Path $\mathbf{L}$ (2) is ***opened***, if $v_1 \neq v_{n+1}$, and ***closed***, if $v_1 = v_{n+1}$. The closed tpath in length not less than 1 is a ***cycle*** when it is a way.

A graph $\mathbf{\Gamma_2}=(\mathbf{V_2}, \mathbf{E_2})$ is a ***subgraph*** of graph $\mathbf{\Gamma_1} = (\mathbf{V_1}, \mathbf{E_1})$ if it is fulfilled $\mathbf{V_2}\subseteq\mathbf{V_1}$ and $\mathbf{E_2}\subseteq\mathbf{E_1}$. Graph $\mathbf{\Gamma_2}$ is a ***skeleton*** (a ***spanning subgraph***) of graph $\mathbf{\Gamma_1}$ if $\mathbf{V_1} = \mathbf{V_2}$. The graph $\mathbf{\Phi}$ is a ***tree*** if it has no any subgraph which is a cycle. Spanning subgraph $\mathbf{\Gamma^*}$ of graph $\mathbf{\Gamma}$ is called as a ***spanning tree*** if it is a tree.

A ***walk*** joining the vertex v_1 to the vertex v_{n+1} in a graph $\mathbf{\Gamma}$ is an alternating sequence $\mathbf{L}$:

$$v_1\, e_1\, v_2\, e_2\, v_3 \ldots\, e_n\, v_{n+1} \qquad (3)$$

with $\mathbf{e_i}$ incident out of v_i and incident into v_{i+1}. A vertex v_2 is ***reachable*** from a vertex v_1 of $\mathbf{\Gamma}$ if there is a directed path $\mathbf{L}$ in $\mathbf{\Gamma}$ from v_1 to v_2. In addition, it is assumed that each edge $\mathbf{u_iu_{i+1}}$ only occurs once in $\boldsymbol{\phi}$. The integer $\mathbf{m}$ is called the ***length*** of the path $\boldsymbol{\phi}$. In particular, an edge is a path of length $\mathbf{1}$. We identify a single vertex with a path of length zero. For example, in the graph of Figure 1, $\mathbf{v_5v_1v_4v_3}$ is path $\boldsymbol{\phi^*}$ joining $\mathbf{v_5}$ and $\mathbf{v_3}$. $\boldsymbol{\phi^*}$ is of length $\mathbf{4}$. The vertices $\mathbf{u}$ and $\mathbf{u'}$ are called the ***endpoints*** of the path $\boldsymbol{\phi}$. A ***cycle*** is a path whose endpoints coincide. In the graph of Figure 1 $\mathbf{v_5v_1v_4v_5}$ is a cycle of length 3. The graph $\mathbf{\Gamma}$ is called ***connected*** when there exists at least one path joining every pair of distinct vertices of $\mathbf{\Gamma}$. The graph of Figure 1 is connected. On the other hand, the graph of Figure 2 is not.

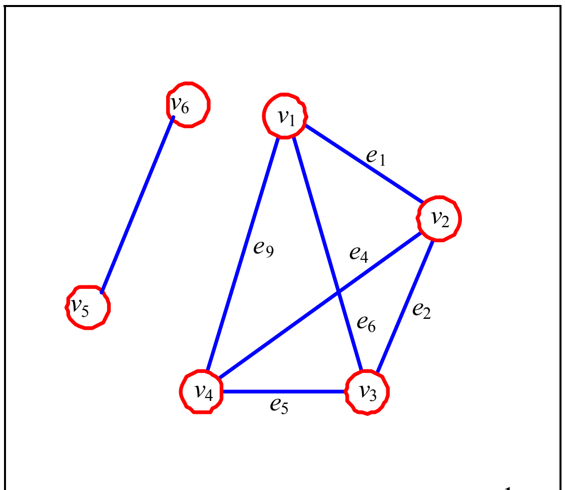


Fig. 2. 5-vertical disconnected graph $\wp^1$

Let $\mathbf{\Gamma^1}=(\mathbf{V^1}, \mathbf{E^1})$ and $\mathbf{\Gamma^2}=(\mathbf{V^2}, \mathbf{E^2})$ are graphs. A ***homomorphism*** from graph $\mathbf{\Gamma}^1$ in $\mathbf{\Gamma}^2$ is the pair of maps

$$\mathbf{F_1}: \mathbf{V^1} \rightarrow \mathbf{V^2} \text{ and } \mathbf{F_2}: \mathbf{E^1} \rightarrow \mathbf{E^2}$$

such, that the following condition is satisfied

$$(\mathbf{F_1}(\boldsymbol{v_1}), \mathbf{F_1}(\boldsymbol{v_2})) \in\mathbf{E^2} \text{ and } \mathbf{F_2}(\mathbf{e}) \in\mathbf{E^2},$$

for $\forall(\boldsymbol{v_1}, \boldsymbol{v_2}) \in\mathbf{E^1}$ and $\forall\mathbf{e}\in\mathbf{E^1}$.

One symbol $\mathbf{F} = \mathbf{F}\ (\mathbf{F_1}, \mathbf{F_2})$ for a label of pair maps is usually used. The set of all homomorphisms from the graph $\mathbf{\Gamma}$ in graph $\mathbf{G}$ is designated as $\mathbf{Hom}\ (\mathbf{\Gamma}, \mathbf{G})$.

Let $\mathbf{G} = (\mathbf{V}, \mathbf{E})$ will be the final graph with $\mathbf{k}$ verteces, which are designated by numbers 1,2,…k (the bijection φ: V↔[1…k] is set). The ***Incidence matrix*** $M_k^k\,|\,G\,|$ of the graph $\mathbf{G}$ is called the matrix **k** × **k** in which in a cell on intersection of a row with number **i** and a column with number **j** is noted the number of edges, that are incident to the number of edges which final verteces will be the verteces of the graph designated by numbers **i** and **j**, **1 ≤i, j ≤k**.

Example 1. The Incidence matrix $M_5^5\ G$, where G is the graph on fig. 3.

	Col.1	*Col.2*	*Col.3*	*Col.4*	*Col.5*
Row. 1	1	3	1	1	1
Row. 2	3	1	2	2	1
Row.3	1	2	1	1	1
Row.4	1	2	1	0	4
Row.5	1	1	1	4	0

It is clear that in any graph the sum of the degrees of all the vertices is equal to twice the number of edges:

$$\sum_{\forall u \in V} d(u) = 2\mu(E), \tag{3}$$

The ***subgraph*** of **Γ** ***induced*** by a subset $\mathbf{V}^*$ of **V** has $\mathbf{V}^*$ as its set of vertices and its set of edges consists of all the edges **Γ** of both of whose endpoints lie in $\mathbf{V}^*$. When every pair of vertices of **Γ** is joined by an edge, we say that **Γ** is a complete graph. A ***clique*** of an arbitrary graph **Γ** is a sub-graph of which is complete. A ***connected component*** of **Γ** is a subset $\mathbf{V}^*$ of **V** such that the subgraph $\mathbf{\Gamma}^*$ induced by $\mathbf{V}^*$ is maximal for this property. If $\mathbf{E}^*$ is a subset of the edges of the graph **Γ = (V , E)**, the graph $\mathbf{\Gamma}^* = (\mathbf{V}, \mathbf{E}^*)$, is called the ***partial graph*** of **Γ** induced by $\mathbf{E}^*$.

When one wishes to take account not merely of the existence of the relations re-presented by a graph but also of their intensity, one make use of the concept of valued graph.

A ***valued graph*** is a pair

$$(\Gamma, \mathscr{J}) \tag{4}$$

consisting of a graph **Γ = (V , E)** and a real-valued function

$$\mathscr{J}: \mathrm{E} \longrightarrow \mathbb{R}^+$$

defined on **E** taking strictly positive values. $\mathscr{J}$**(uv)** is called the ***length*** of the edge **uv**.

Example 2.

The graph **Γ** is a model of a road network (the vertices of **Γ** are the towns and an edge represents the existence of a road directly connecting two towns) one could value it by taking the length of an edge to be the length of the corresponding road, or the cost of transportation along it, or the length of time required to traverse it, etc .

In a valued graph **(Γ , $\mathscr{J}$)**, the ***$\mathscr{J}$-length*** (or simply ***length***, when the reference to $\mathscr{J}$ is clear) $\mathscr{J}$**(c)** of a path ɔı is the sum of the lengths of its edges.

Graphs, called ***shrubbery***, also correspond to a particularly simple and economical type of configuration: they are connected and have no cycles, or equivalently, there is one and only one path connecting any two distinct vertices, or again, they are connected graphs with the smallest possible number of edges.

A ***shrubbery*** is graph with no cycles. A ***tree*** is a connected shrubbery.

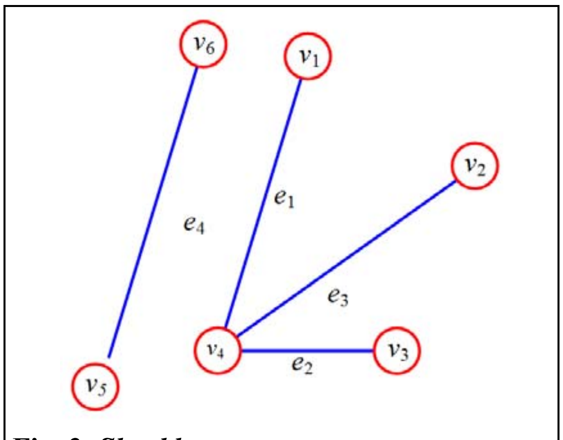


Fig. 3. ***Shrubbery***

Comment: Further, for study of sectional structures it is enough to consider to study only their coherent components trees and then, it is consecutive to each component of connectivity of a bush to apply the received outcomes.

At last, we will look on an example how to convert the information from a picture cloth (a photo of the person, a photo of district and any visual image) on columns. Let, in drawing 4 the slice of a cloth of a picture (a fragment from a picture on a case history 2) is represented.

Further we operate according to following algorithm.

Algorithm 1. Drawing representation in an aspect of the graph with the weighed vertexes.

1.1. To present drawing in the form of a population of the painted cells:

- To each cell to put in correspondence a vertex of a graph G_0. To paint vertexes in corresponding colours.
- To connect the vertexes of the graph G_0 by edge if corresponding cells have the joint side.

1.2. On the graph with unweighted vertexes G_0 will construct a graph G_1 with weighed vertexes:

- To suppose $G_1 = G_0$. To each vertexes of the graph G_1 to assign weight 1.
- If in the graph G_1 there are adjacent vertexes with the same colour, to tighten an edge connecting these vertexes. To establish weight of the received vertex equal to the sum of scales of strapped vertexes. To repeat this operation until in the graph G_1 consists the adjacent vertexes with the same colours.

1	2	3	4	5	1	2	3	4	5
6	7	8	1	9	6	7	8	9	1
1	2	10	11	12	10	11	12	13	2
13	14	15	16	17	14	15	16	17	3
1	2	3	4	5	1	2	3	4	5
6	7	8	9	10	6	7	8	9	10
11	12	13	14	15	11	12	13	14	15
16	17	18	19	20	16	17	18	19	20
21	22	23	24	25	1	1	2	3	1
26	27	28	29	30	2	4	5	6	2

Fig. 4.

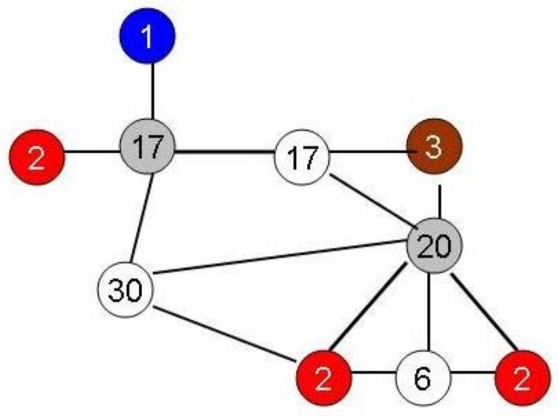


Fig. 5

Let's receive the following weighed graph (fig. 5).

In a tree, every vertex of degree 1 is called a ***leaf***. The other vertexes are called ***nodes***. A ***ternary shrubbery*** is a tree all of whose nodes are of degree **3**. And a ***n-ary shrubbery*** is a tree, all of whose nodes are of degree ***n***.

Theorem 1. A tree **T = (V, E)** with at least two vertices has at least two leaves.

Proof. Let **T** be a tree with **0** or **1** leaves. Construct a path in **T** as follows: in the first case start from an arbitrary vertex and in the second case start from the unique leaf. Proceed to an adjacent vertex and continue in this manner following the rule that one should never leave a vertex along the edge by which one entered it. Let $\Im_m$ be the sequence $\mathbf{u_0u_1u_2...u_m}$_up of vertexes which have been visited by the **m**th-stage. Since **T** has no cycles the vertexes in $\Im_m$ are pairwise distinct. On the other hand up has degree at least **2**, so that one can leave this vertex along an edge different from $\mathbf{u_{m-1}\ u_m}$. One therefore reaches a vertex $\mathbf{u_{m+1}}$ disjoint from $\Im_m$ (again, because **T** has no cycle).

Theorem 2. Let **Γ = (V, E)** be a graph. The following assertions are equivalent:

$\mathbf{P_1}$: **Γ** is a tree.

$\mathbf{P_2}$: **Γ** has no cycle and **V** = |**E** | + 1.

$\mathbf{P_3}$: **Γ** is connected and **V** = |**E** | + 1.

$\mathbf{P_4}$: Any two distinct vertexes of **Γ** are always connected by one and only one path.

$\mathbf{P_5}$: **Γ** is connected and, for any edge **uv** of **Γ**, the partial graph of **Γ** induced by **E/uv** is not connected.

Proof. We show that $\mathbf{P_1}$ implies $\mathbf{P_2}$ by means of induction on the number n of vertexes of **Γ**. The result is clear for **n = 1** or **n = 2**. Suppose it is true for **n- 1** and consider a tree **Γ** with **n** vertices. By Theorem 1, **Γ** has at least one leaf **u**. Let **uv** be the unique edge incident with u and consider the subgraph **H** of **Γ** induced by **V/u** . Trivially, **H** is connected and has no cycle. One can therefore apply the induction hypothesis: **H** has **n – 1** vertices and **n – 2** edges. But **Γ** is obtained from **H** by the addition of the vertex **u** and the edge **uv**. Hence, **Γ** has **n** vertexes and **n – 1** edges.

The preceding construction can also be used to show by induction that $\mathbf{P_2}$ is equivalent to $\mathbf{P_3}$: the graph **H** has no cycle (resp. is connected) and has **n – 1** vertices and **n – 2** edges. The induction hypothesis then shows that **H** is connected (resp. has no cycle). It follows immediately that **Γ** is connected (resp. has no cycle). We have just seen that $\mathbf{P_3}$ implies $\mathbf{P_2}$. Given any two distinct vertices of **Γ**, there is always a path joining them since **Γ** is connected, and this path is necessarily unique since **Γ** is acyclic. Thus $\mathbf{P_3}$ implies $\mathbf{P_4}$. Suppose **Γ** satisfies $\mathbf{P_4}$ and consider an edge **uv**. If the partial graph of **Γ** induced by **E/uv** was connected there would exist a path from **u** to **v** distinct from **uv**; this would contradict $\mathbf{P_4}$. Thus $\mathbf{P_4}$ implies $\mathbf{P_3}$. We show that $\mathbf{P_5}$ implies $\mathbf{P_1}$ by remarking that if **Γ** contained a cycle, it would remain connected if one removed an edge from this cycle.

Problem 2. How many tops and edges have a n-ary tree whith m leaves?

A tree **D** which has only one node **z** is called a ***star***. In this case we call **z** the centre of **D**. If the star H has p vertices, **z** has degree p–1. The concept of a rooted tree, which we introduce in the following, is very useful for the coding and representation of trees. A ***rooted tree*** is a pair (**D**, **k**) consisting of a tree **D** and a node **k** of **D**, called its ***root***. Let (**D**, **k**) be a rooted tree and let $\mathbf{v_1}$, $\mathbf{v_2}$ be two vertices of **D**. We write:

$$v_1 \leq v_2 \tag{5}$$

iff $\mathbf{v_2}$ lies on the path joining $\mathbf{v_1}$ and **k**. It is clear that the relation ≤ is:

— **reflexive**: for all **v**, **v ≤ v**;

— **antisymmetric**: for all $\mathbf{v_1}$ and $\mathbf{v_2}$, $\mathbf{v_1 \leq v_2}$ and $\mathbf{v_2 \leq v_1}$ imply that $\mathbf{v_1 = v_2}$;

— **transitive**: for all $\mathbf{v_1}$, $\mathbf{v_2}$, $\mathbf{v_3}$, $\mathbf{v_1 \leq v_2}$ and $\mathbf{v_2 \leq v_3}$ imply that $\mathbf{v_1 \leq v_3}$.

The relation ≤ is therefore an ***order relation*** on the set of vertices of the tree **D**. For this order **k** is the largest element: for all vertices **v**, **v ≤ k**. Moreover, the leaves of **D** are its minimal elements: **u** is a leaf if and iff there is no vertex **v** different from **u** such that **v ≤ u**. In a rooted tree (**D**, **k**) we say that the edge **uv** ***originates*** from the vertex **u** when **v ≤ u**. In this case, by analogy with genealogical trees, we say that **u** is the ***father*** (or ***predecessor***) of **v** and that **v** is the ***son*** (or ***successor***) of **u**. Two sons of the same father are called ***brothers***.

3. Coding

A tree will always be denoted by integers (the nodes being numbered after the leaves) and the names of variables in programs will be indicated by capitals. **Θ** designates the number of vertices of the tree **T** which is given by a list of edges, each specified by its initial and final vertex, although no notion of orientation is intended by these terms. One works with two arrays $\mathbf{A_1}$ and $\mathbf{A_2}$ each of dimension **Θ** – 1 and the **i**th edge is denoted by $\mathbf{A_1(i)} – \mathbf{A_2(i)}$. The tree is coded using three arrays of dimension **Θ**. We emphasize first of all that a choice of a particular vertex in a tree allows us to give all the edges an orientation pointing towards this vertex which is called the root. One then has a coding of the rooted tree in the form of an array $\mathbf{A_p}$ of predecessors of each vertex. The vertex which is chosen as the root has no predecessor; it is given the value **0** which designates the root. This gives a tree-coding with **Θ** numbers which allows a linear algorithm to be created for finding a path to the root.

This coding is satisfactory for paths directed towards the root, but it is insufficient for paths in the opposite direction for, to obtain the successors of a vertex **S** one has to scan the array $\mathbf{A_p}$ to find the vertices **K** such that $\mathbf{A_p}$ (**K**) = **S**. We therefore define a data-structure which allows us to descend the tree. In the **i**th place of an array **SU** we store one of the successors of the vertex I and we create an array of pointers **FR** such that **FR(I)** is a vertex with the same predecessor as **I**. We can then obtain the successors of vertex **S** by starting with **SU(S)** and then obtaining

$$\mathbf{FR(SU(S)), FR(FR(SU(S))),} \tag{6}$$

etc, until we obtain 0 meaning that there are no more brothers.

A data-structure of this type allows us to traverse a tree without scanning whole arrays. To obtain this coding in three arrays $\mathbf{A_p}$, **SU**, **FR** from our initial array of edges it is necessary to choose a root. We initially determine a “centre” of the tree to serve as the root, namely one of the midpoints of the longest path. To do this, starting with an array **DG** of vertex degrees, we apply the algorithm for determining a centre which consists of erasing the edges which lead to leaves. We scan the array of edges; if one of the vertices **U** has degree 1, the other end **V** must be its predecessor. If **V** does not have a successor, we designate **U** to be it, otherwise we scan the brothers of this successor until we point to a free space, into which **U** is inserted. The array **M** serves to mark the edges which are removed and **D** contains the quantities whose degrees must be lowered after each scanning of the edge array. Later on we will be able to choose the root to be any arbitrary tree vertex, including a leaf, and the coding will be modified, starting, of course, from the initial coding.

4. Algorithm 2

This program enables us to read a tree file, with up to **N** vertices, such as is created in all the programs of the following chapters which determine a valued tree from a list of dissimilarities. The files contain the number of vertices, and then for each edge the numbers of the initial and final vertices and the length of the edge. In our coding this length is indexed by the number corresponding to its initial vertex. Since the choice of a new root could change the orientation of an edge, we use an array **AN** to permute these lengths.

```
10 REM Coding of a tree SD % — SF % in form PR %, SU %, FR %
20 DIM PR % (N), SU % (N), FR % (N), □G % (N), D % (N)
30 DIM SD % (N), SF % (N), M % (N), LG (N), AN (N)
40 INPUT "Name of tree file"; NA $: N □ = NA $ + ". ARB.
50 OPEN "I", □I, N $: INPUT #I,NS: NA=NS-1
60 FOR I=1 TO NA: INPUT #1, SD % (I): INPUT #I,SF% (I)
70 NEXT I: CLOSE #1

100 REM Tree rooted at its centre
110 GOSUB 3200: GOSUB 2400

200 INPUT "Tree rooted at another vertex (Y/N)"; CS: IF CS <> "Y" THEN END
220 INPUT "Number of this vertex"; S
230 GOSUB 3500: GOSUB 2400: GOTO 200

2400 "Tree rooted at"; R: PRINT
2410 FOR = 1 TO NS: PRINT USING "####"; SD % (I); SF % (I);

2420 ": PRINT USI□3" ### "; I;: PRINT":;
2430 PRINT USING "####"; PR % (I); SU % (I); FR % (I);
2440 PRINT ":;: PRINT LG (I): NEXT I: PRINT: RETURN

3200 REM Coding a tree
3210 FOR I=l TO NA: SD = SD % (I): SF = SF % (I)
3220 DG % (SD) = DG % (SD) +1: DG % (SF) = DG % (SF) +1
3230 SOM=SOM+2: NEXT I

3300 FOR I=l TO NS: D % (I) = 0: NEXT □ I
3310 FOR I =1 TO NA: IF M % (I) = 1 THEN 3380
3320 U = SD % (I): IF DG % (U) = 1 THEN V = SF % (I): GOTO 3350
3330 U = SF % (I): IF DG % (U) =1 THEN V = SD % (I): GOTO 3350
3340 GOTO 3380
3350 PR % (U) = V: M % (I) = l: D % (U) = D % (U) +1: AN (U) = LG (I)
3360 K = SU % (V): IF K=0 THEN SU % (V) = U: GOTO 3380
3370 IF FR % (K)> 0 THEN K = FR % (K): GOTO 3370 ELSE FR % (K) = U
3380 NEXT I
```

```
3390 FOR I = 1 TO NS: DG % (I) = DG % (I) — D % (I): SOM=SOM-D % (I) NEXT I
3400 IF SOM> 0 THEN 3300
3410 FOR I = 1 TO NS: LG (I) = AN (I): NEXT I
3420 R = V: RETURN

3500 REM Change of root
3510 NR = S: U = S: W = 0: LG (R) = LG (U)
3520 V = PR % (U): PR % (U) = W: W = U: IF V = 0 THEN 3590
3530 SU = SU % (U): IF SU = 0 THEN SU % (U) = V: GOTO 3550
3540 FR = FR % (SU): IF FR> 0 THEN SU = FR: GOTO 3540 ELSE FR % (SU) = V
3550 SV = SU % (V): IF SV=U THEN SU % (V) =FR % (U): FR % (U) =0: GOTO 3580
3560 FR=FR % (SV): IF FR> 0 AND FR <> U THEN SV=FR: GOTO 3560
3570 FR % (SV) = FR % (U): FR % (U) = 0
3580 U = V: IF U> 0 THEN SWAP LG (R), LG (V): GOTO 3520
3590 R = NR: LG (R) = 0: RETURN
```

References:

1. *Gritsak–Groener V. V., Gritsak-Groener J.* Arts combinatoria. — NTU «HPI», Charkiv, 2006.
2. *Gritsak–Groener V. V.* Fundamental of Mathematical Cybernetics. Vol. 2. — MIT PRESS, 2006.
3. *Groener W.* (*Gritsak–Groener V. V.*). Fundamental of Mathematical Cybernetics. Vol. 1. — 2004.
4. *Gritsak V. V. (Gritsak–Groener V. V.).* Pattern Recognition Algorithms of Information in Hereditarely Alphabets Languages. Algorithms an Software. — Kyjiv, 1985.
5. *Gritsak–Groener V. V.* A Course in Modern Analysis. — Bastil Verlag, 2006.

Статья поступила в редакцию 28.10.2010 г.

Гритсак-Грёнер В. В., Карпенко О. Б.

Математические методы идентификации визуальной информации. 1.

Статья начинается с краткого, но адекватного пересмотра основных концепций теории графов, которые будут использованы позже. Вводится центральное понятие кустарников, которое служит основной референтной моделью в дальнейшей части работы. К этому понятию подходим на основе многочисленных примеров, взятых из социологии и соционики, математической психологии, информатики и математической теории жизни, а также связанных с идентификацией авторства живописи.
Ключевые слова: граф, распознавание образов, визуальная информация.