Computer Science Journal of Moldova, vol.**19**, no.1(55), 2011

Multiset-based Tree Model for Membrane Computing

D. Singh C.M. Peter

Abstract

In this paper, we introduce a new paradigm - multiset-based tree model. We show that trees can be represented in the form of wellfounded multisets. We also show that the conventional approach for this representation is not injective from a set of trees to the class of multisets representing such trees. We establish a one-to-one correspondence between trees and suitable permutations of a wellfounded multiset, which we call *tree structures*. We give formal definitions of a *tree structure* and a *subtree structure* of a tree structure. Finally, we represent membrane structures in the form of tree structures – a form in which membrane structures can suitably be represented at programming level.

Keywords: wellfounded multiset, saw-like structure, multiset-based tree structure, membrane structure.

1 Introduction

A tree is an *acyclic* connected graph (having one source or trunk and several exits or leaves). It can also be defined as a partial order relation over a finite set with the smallest element. Note that if there is only one edge from a source node, then we have only one branch on the node.

Trees have served as handy tools in solving problems involving decisions and the flow of information. Thus they form a fundamental concept in graph theory. They are used in many disciplines such as Mathematics, Management, Economics, Commerce, Biology, Computer Science, Statistics and Probability, just to mention a few.

In Managerial accounting ([7], p. 502), a tree diagram in the form of an organizational chart was used to illustrate the organization of Aloha Hotels and Resorts. In Computer Science [1], the Huffman tree is used to compress bits so as to reduce the amount of storage that is necessary in a storage media. In Biology, ([12], p. 57) exploits a tree structure to represent the basic characteristics of meiosis involving one chromosome duplication followed by two nuclear and cell divisions. In Probability [8], the concept of *tree measure* is applied to represent a sequence of repeated throws of a coin with either side labelled head or tail. In English language ([6], p. 24), the classification of nouns is represented in the form of a tree. As some membranes can contain several other membranes [10], trees can also be suitably used to study membrane structures and hence in comprehending membrane computing.

Note that it may be relevant to count the number of nodes in an n-nary tree – a tree with ($n \geq 2$) number of branches emerging from each of the major branches after them (which is called a *tail* or a node) or a general tree, for that matter, with n varying from branch to branch.

In the recent years, the representation of a tree in the form of a wellfounded (cardinality–bounded) multiset has been arguably used. It is observed that the said representation is not *injective* from a set of trees to the class of multisets representing such trees. In order to achieve injection, we devise various permutations of the wellfounded multiset in consideration along with a suitable rule. We begin with a binary tree and generalize the approach to an n-nary tree as well as a *general* tree.

2 Aptness for the Use of Trees and a Multiset Environment

Unlike other graphs, trees are quite innovative especially in one of the recently researched areas of computer science – molecular and membrane computing. This is because among all other forms of graphs (e.g., a *loop* or a *multigraph*), only trees can suitably represent membrane structures (without intersections, loops or parallel edges). This singular but essential ability of trees is what has inspired us to study them and to see how they can help us in contributing to the improvement of membrane computing.

There is no provision for a diagrammatic or pictorial input of membrane structures at programming level. Thus, one aim of this paper is to demonstrate how membrane structures can be represented at programming level by devising a discrete approach for such a representation. It is in line with this that we have employed multiset as an environment in representing trees, and consequently membrane structures.

3 Some Basic Concepts

Definition i: Multiset.
A multiset (mset, for short) is a collection of objects in which, unlike a crisp (Cantorian) set, objects are allowed to repeat finitely in most of the application areas; although infinite multiplicities are also dealt with in a theoretical development (see [2] and [3] for details).

A multiset is represented in several ways. The use of square brackets to represent a multiset is quasi-general. Thus, a multiset containing one occurrence of a, two occurrences of b, and three occurrences of c is notationally written as $[[a, b, b, c, c, c]]$ or $[a, b, b, c, c, c]$ or $[a, b, c]_{1,2,3}$ or $[a, 2b, 3c]$ or $[a.1, b.2, c.3]$ or $[1/a, 2/b, 3/c]$ or $[a^1, b^2, c^3]$ or $[a^1b^2c^3]$. For convenience, the curly brackets are used in place of the square brackets. In fact, the last form of representation as a string, even without using any brackets, turns out to be the most compact one, especially in computational parlance. The following schematic representation of a multiset as a numeric-valued or count function abounds, particularly in the foundational development of multiset theory and its application:

A multiset is a mapping from some ground or generic set of objects into some set of numbers. For example, a multiset $\alpha = [x, y, z]_{1,2,3}$ is a mapping from a ground set D to $\mathbb{N}$, the set of non negative integers, defined by

$$\alpha(t) = \begin{cases} 1, & \text{if } t = x \\ 2, & \text{if } t = y \\ 3, & \text{if } t = z \\ 0, & \text{for all the remaining } t \in D. \end{cases}$$

In other words, a multiset α drawn from a ground set D can be represented by a cardinal-valued function $C_\alpha : D \to \mathbb{N}$.

In general terms, for a given ground set D and a numeric set T, we call a mapping $\alpha : D \to T$,

$$\begin{cases} \text{a set, if } T = \{0,\ 1\}\,; \\ \text{a multiset, if } T = \mathbb{N},\ \text{the set of natural numbers;} \\ \text{a signed multiset (hybrid or shadow set), if } T = \mathbb{Z}, \text{the set of integers;} \\ \text{a fuzzy (or hazy) set if } T = [0,1] \subseteq \mathbb{R}, \text{ a two}-\text{valued Boolean algebra.} \end{cases}$$

In view of the above definition, a multiset A can also be represented by a set of pairs as follows:

$A = \{\langle m_A(x_1), x_1\rangle, \ldots, \langle m_A(x_j), x_j\rangle, \ldots\}$ or

$A = \{m_A(x_1) \cdot x_1, \ldots, m_A(x_j) \cdot x_j, \ldots\}$ or

$A = \{n_1/x_1, \ldots, n_j/x_j, \ldots\}$, where $m_A(x_j) = n_j =$ the count or the multipliciy of x_j in A.

Note that there are other forms of representing a multiset (see [2], [3] and [11] in particular).

Definition ii: Submultiset.
Given a multiset M over a domain set D, a multiset A over D is called a submultiset of M written as $A \subseteq M$ or $M \supseteq A$ if $m_A(x) \leq m_M(x)$ for all $x \in D$, where $m_A(x)$ and $m_M(x)$ are the multiplicities of x in the multisets A and M respectively. Also if $A \subseteq M$ and $A \neq M$, then A is called a proper submultiset of M. A multiset is called the ancestor in relation to its submultiset (see [11], for details).

Definition iii: Dressed epsilon.
For any object x occurring as an element of a multiset A i.e., $m_A(x) > 0$, we write $x \in_+ A$, where $\in_+$ (dressed epsilon) is a binary predicate

intended to be 'belongs to at least once', as $\in$ is 'belongs to only once' in the case of sets. Also, $x \in_{+}^{k} A$ implies 'x belongs to A at least k times', while $x \in^{k} A$ means x belongs exactly k times to A. $x \notin A$ means 'x does not belong to A' ([11], for details).

Definition iv: Partial ordering.
A binary relation $\lessdot$ on a set X is called a *partial order* on X if $\lessdot$ satisfies the following axioms:

1. $x \lessdot x$ for all $x \in X$. (Reflexivity)
2. $x \lessdot y$ and $y \lessdot x \Rightarrow x = y$ for all $x, y \in X$. (Anti symmetry)
3. $x \lessdot y$ and $y \lessdot z \Rightarrow x \lessdot z$ for all $x, y, z \in X$. (Transitivity)

The set X is said to be *partially ordered* with respect to $\lessdot$. We note that for some pair of elements x, y in X, neither $x \lessdot y$ nor $y \lessdot x$ may hold. If $x \lessdot y$ or $y \lessdot x$ for all x, y in X then X is said to be *totally ordered* or linearly ordered or a chain (see [5] for details).

Definition v: The Dershowitz-Manna ordering on multisets.
We follow the Dershowitz-Manna ordering on multiset. Let $(S, >)$ be a partially ordered set and $M(S)$ be the set of all finite multisets with elements taken from the set S. Then a partial order $\gg$ on $M(S)$ can be defined as follows:

Let M, $M' \in M(S)$. Then $M \gg M'$ if for some multisets $X, Y \in M(S)$, with $\emptyset \neq X \subseteq M$, we have $M' = (M \backslash X) \cup Y$ and $(\forall y \in_{+} Y)(\exists x \in_{+} X) \quad x > y$.

For example, let $S = \mathbb{N}$, the set of natural numbers including 0 with the usual ordering $>$, then under the corresponding multiset ordering $\gg$ over $\mathbb{N}$, the multiset $\{3, 3, 4, 0\}$ is greater than each of the three multisets $\{3, 4\}$, $\{3, 2, 2, 1, 1, 1, 4, 0\}$ and $\{3, 3, 3, 3, 2, 2\}$. (see [4] for details).

Definition vi: Wellfounded multiset.
A wellfounded multiset is a multiset with an irreflexive and transitive

ordering defined on it, such that its every submultiset has a minimal element; in other words, no infinite descending chain occurs. We shall follow the Dershowitz-Manna ordering on multisets over a set of natural numbers which has been proved wellfounded (see for the proof in [4]).

Definition vii: An n-nary tree.
Given a non-negative integer n, an *n-nary tree* is a tree which has exactly n number of branching on each of its branches. In the case where the number of branching varies from branch to branch on the tree, we call such a tree a *general tree* or simply a tree.

Definition viii: A binary tree.
A *binary tree* is an n-nary tree for $n = 2$.

Definition ix: A subtree.
A *subtree* is a subgraph which is a tree.

See [4] for details of the aforesaid definitions.

4 The Binary Tree

Dershowitz and Manna ([4]) demonstrated the termination of a program to count the tips of a binary tree using a wellfounded multiset ordering. We describe in brief one of the examples they considered.

> *Consider a simple program to count the number of tips – terminal nodes (without descendents) – in a full binary tree. Each tree y that is not a tip has two subtrees, left(y) and right(y).*

Typically, a binary tree can be schematically represented as in Figure 1.

In Figure 1, the *tree trunk* is represented by the largest integer in the labelling of the tree. One of the branches on a y-shaped (or fan-shaped) subtree is called an *axis.* The part of the tree which continues from the base to a tip without a gap is called a *chain.* The y-shaped subtree at the bottom of the tree is called the *base* of the subtree.

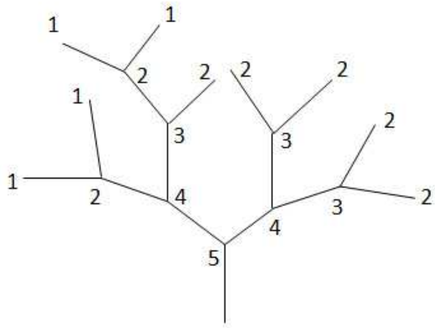


Figure 1. A binary tree

Notice that in the diagram above the integer label of a branch is less than the integer label of the node upon which the branch rests. One of the advantages of representing a tree in this way is reflected in membrane structures. Thus, the sizes of membranes in a membrane structure can be used in place of the integer labels, yet retaining the tree representation of membrane structures. This fact is vindicated in this paper when we shall be applying tree structures to membrane computing.

5 Conventional Approach to Representing a Tree by a Wellfounded Multiset

A *conventional method* of representing a tree by a wellfounded multiset seems to have first appeared in [4]. The conventional method entails that in a wellfounded multiset, any element-multiset representing a subtree which is built upon another element-multiset representing another subtree needs to be smaller than the one upon which it is built. Moreover, there is no permutation governing the arrangement of the element-multisets of the wellfounded multiset. Rather, one rule inherent in the representation is that the smaller and larger element-multisets have exactly one element in common. This common element is also the largest element in the smaller element-multiset and cannot be the largest element in the larger element-multiset. The action of picking an element-multiset to represent a subtree on the tree is done exhaustively. Though no definition of this method has been given in [4], we give the following formal definition to capture the concept.

Definition x.
Formally, given a multiset S over a domain set D, a multiset $T(S)$ whose elements are submultisets of S is a *conventional representation of a tree* if and only if it satisfies the following properties:

1. There exists $z \in^1 T(S)$ such that $z = max\{y : y \in_+ T(S)\}$,
2. For each $u \in_+ T(S)$ where $u \neq max\{y : y \in_+ T(S)\}, \exists w \in_+ T(S)$ with $w \gg u$ and $x_0 \in_+ S$ such that $x_0 \in^1 u \cap w$, $x_0 = max\{x : x \in_+ u\}$ and $x_0 \neq max\{x : x \in_+ w\}$.

In the above definition, the first condition is called *the base condition* and z is called the *base* of a *tree*. The second condition is called *the join condition* and x_0 is the *join* between two *subtrees*.

6 Wellfounded Multiset Representation of a Binary Tree (Use of the Conventional Approach)

A binary tree is represented in the form of a wellfounded multiset whose elements are multisets containing only three elements. For example, the wellfounded multiset $\{\{322\}, \{211\}\}$ represents the binary tree in Figure 2.

Each y-shaped subtree is represented by an element say a in the multiset; a itself is a multiset with three elements of the form $\{a_1, a_2, a_3\}$. We show that the *conventional* approach of the representation is not injective from a set of trees to the class of wellfounded multisets representing such trees.

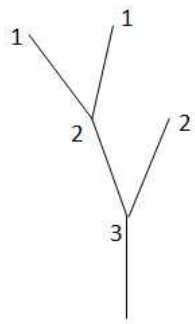


Figure 2. A two-element binary tree

Consider $N = \{\{544\}, \{433\}, \{432\}, \{322\}, \{322\}, \{322\}, \{211\}, \{211\}\}$. Each element of N has three elements and represents a y-shaped subtree of the tree in Figure 1.

It is easy to see that N is ordered according to the Dershowitz-Manna ordering on multisets.

The element $\{544\}$ represents the first y-shaped subtree with tail 5. The two 4's in $\{544\}$ show that we can locate two elements in N, each of which has 4 as its largest element. The two elements are $\{433\}$ and $\{432\}$ which give rise to two y-shaped subtrees with tails labelled 4 on the trunk labelled 5.

The three 3's in these two elements combined show that we can locate three other elements in N each of which has 3 as its largest element. These elements are three $\{322\}$'s, which give rise to three y-shaped subtrees with tails labelled 3 on the branch labelled 4.

Next are the six 2's in the three elements. But, since there are only two $\{211\}$'s in N each of which has 2 as its largest element, there can only be two y-shaped subtrees each of whose tail is labelled 2.

Also, the 2 in $\{432\}$ shows that we can locate an element in N which has 2 as its largest element. This element is $\{211\}$, and can give rise to a y-shaped subtree with tail labelled 2.

It can be observed that there arises a decidability problem as to how and on which of the two types of branches, one of which is labelled 4 and three of which are labelled 3, should the two subtrees with tail labelled 2 be built upon. If one of the two {211}'s is built on any one of the three {322}'s while the other {211} is built on {432}, we get the binary tree A in Figure 3 below. If on the other hand each of the two {211}'s is built on each of any two of the three {322}'s, we get the binary tree B in Figure 3.

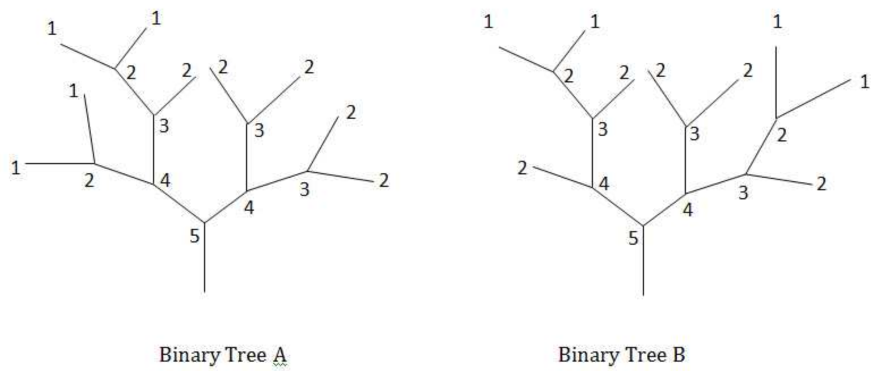


Figure 3. Two different binary trees generated by the multiset N

Thus, our ordered multiset N can yield two different binary trees and so the conventional method does not give room for a wellfounded multiset representation of a binary tree in a one-to-one manner.

The above assertion can be generalised in the case of an n-nary tree. It can similarly be shown for a tree with n varying from branch to branch on the tree, called a *general* tree.

7 The Saw Rule

We demonstrate below by considering a suitable permutation of a partially ordered multiset how to represent a rule in which each submultiset of elements is arranged in an *uphill* manner, while at the same time the multiset of all the elements in the system is arranged in a *downhill* manner, called a *saw-like permutation.*

An *uphill* multiset of elements is a permutation of a partially ordered multiset of elements which ends with the largest element in the multiset. A *downhill* multiset of elements is a permutation of a partially ordered multiset of elements which begins with the largest element in the multiset. Thus, all the elements in an uphill multiset of elements or in a downhill multiset of elements may neither be ascending nor descending.

This rule is called a *saw-like permutation* because it creates a resemblance of a wood saw blade when viewed pictorially using vertical bars to represent the elements of a permutation of a partially ordered multiset according to their sizes as in the following figure:

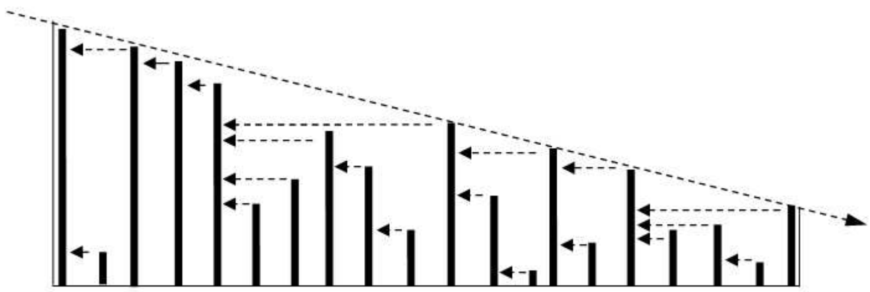


Figure 4. A saw-like structure illustrating a saw-like permutation

The scheme can be interpreted as follows: Each bar in a multiset of ascending bars is attached to the longest bar immediately before the multiset. In other words, any two bars in which one is attached to another must not have any bar longer than the attached bar in between them. The bars represent the element-multisets of a permutation of a partially ordered multiset. The left arrow pointing from one bar to another indicates attachment from the bar on its right to the bar on its left. In other words, each bar on the right side of an arrow represents a branch while the corresponding bar on the left represents a node on the tree. We shall often refer to the element-multisets of a wellfounded multiset as simply elements.

While going through the elements of the partially ordered multiset, we pick the first and largest element represented by the longest bar. This is followed by the smallest element that can be attached directly to the largest element. This in turn is followed by another element only larger than the smallest element, and can be attached directly to the largest element. Any element that must be attached to an element which has already been attached to the largest element must come after the element upon which it is to be attached, even if it happens to be smaller than the first element attached directly to the largest element.

We continue in this way until we have exhausted all the elements that can be attached to the largest element. In Figure 4 above, only two bars have been attached to the first (largest) bar. There is no doubt that these two bars are trivially in an uphill order according to their heights.

This gives us a submultiset of elements immediately following the largest element in an uphill order. Among all the submultisets in an uphill order which have been attached to an element, this one happens to be the largest. The last but not the least element (the third bar in Figure 4 above) in this submultiset turns out to be the second largest element in the partially ordered multiset, and it is the element upon which the second submultiset of elements in an uphill order will be attached starting with the smallest element directly attachable to it. In Figure 4, the second largest bar (which is the third bar) has only one bar attached to it. This one bar can be regarded as a submultiset in an uphill order containing a singleton multiset, though a very trivial case.

Next is the third largest submultiset of elements in an uphill order. We continue in this way until all the elements in the partially ordered multiset belong to a group of elements in an uphill order. Note that in Figure 4 above, the fifth bar has four bars attached to it, including the eighth bar, which in turn has two bars attached to it. These four bars are clearly in ascending order, whereas the six bars (the four bars with the two bars) make up the multiset of bars attached to the fifth bar in an uphill order. This is a non trivial example of bars arranged in an uphill order.

Every element on the saw-like permutation is called a *bar*; every bar which has an element attached to it is called a *column*; a column which is followed by a non-singleton submultiset of descending bars each of which is less than the column and at least one of which has a bar attached to it, is called a *pillar*. A submultiset of consecutive elements from the same object whose multiplicity is more than one is called a *platform*. The first bar on the saw-like permutation is called its *base*.

Having demonstrated how the *saw-like permutation* is used to arrange the elements of a partially ordered multiset in the above discussion, we now give a formal definition of the representation of a partially ordered multiset *in order of a saw-like permutation* in the following definition:

Definition xi.
A permutation $P(S)$ of a partially ordered multiset S of order n is said to be *in order of a saw-like permutation* if and only if $y_1 > y_i$ in $P(S)$, for all $i = 2, 3, \ldots, n$.

If such a permutation exists, we say that $P(S)$ is arranged in order of a saw-like permutation or $P(S)$ is a saw-like permutation of the elements of S. We shall see later in this paper that this construction is of immense help in defining a tree structure.

8 Representation of a Tree by a Saw-like Permutation of a Wellfounded Multiset (The Saw Rule)

To resolve the aforesaid issue of the representation not being injective from a set of trees to the class of multisets representing such trees as indicated in section six above, we demonstrate by considering *saw-like permutations*, how to represent a tree by a permutation of a wellfounded multiset. We called this *the saw rule*. We show below how to represent a binary tree using *the saw rule* and also how the injection is achieved using this rule. In this case, consider the following illustrations to bring our point home:

Let us consider the following two permutations N_1 and N_2 of N:

$N_1 = [\{544\}, \{423\}, \{322\}, \{211\}, \{433\}, \{322\}, \{322\}, \{211\}]$ and

$N_2 = [\{544\}, \{423\}, \{211\}, \{322\}, \{211\}, \{433\}, \{322\}, \{322\}]$

The element $\{544\}$ of N_1 represents the first y-shaped subtree (called the *base*) with tail 5. The two 4's in $\{544\}$ show that we can locate at most two elements in N_1 each of which has 4 as its largest element. The two elements are $\{423\}$ and $\{433\}$.

The element $\{423\}$ is smaller than $\{544\}$ and so can give rise to a y-shaped subtree with tail labelled 4.

The 3 in $\{423\}$ shows that we can locate at most an element in N_1 which has 3 as its largest element, and since the next element $\{322\}$ is smaller than $\{423\}$ it can give rise to a y-shaped subtree with tail labelled 3.

The two 2's in $\{322\}$ show that we can locate at most two elements each of which has 2 as its largest element. Since there is only one such element which has 2 as its largest element and smaller than $\{322\}$ then it can give rise to a subtree with tail labelled 2.

The next element $\{433\}$ is larger than $\{322\}$ and so no subtree representing $\{433\}$ can arise on $\{322\}$. However, $\{544\}$ is the smallest element larger than $\{433\}$, going backwards. Since only one of the two elements having 4 as their largest element has its subtree on $\{544\}$, there can arise another subtree with tail labelled 4 on $\{544\}$ (since $\{544\}$ has two 4's in it).

The two 3's in $\{433\}$ show that we can locate at most two elements each of which has 3 as its largest element. Since the next two elements satisfy this condition, and are smaller than $\{433\}$, then there can arise two subtrees with tails labelled 3.

The last two 2's in the last $\{322\}$ show that we can locate at most two elements each of which has 2 as its largest element. Since we have only one of such elements which is $\{211\}$ having 2 as its largest element, there can arise only one subtree with tail labelled 2 coinciding with only one of the axes labelled 2.

The permutation N_1 of N can be seen to have constructed the one and only tree A in Figure 3 above and no other. Similarly, N_2

determines only the binary tree B in Figure 3. We call N_1 and N_2 *tree structures.* The following figures are saw-like structures of N_1 and N_2 equivalent to the one in Figure 4, for a clear understanding of the concept.

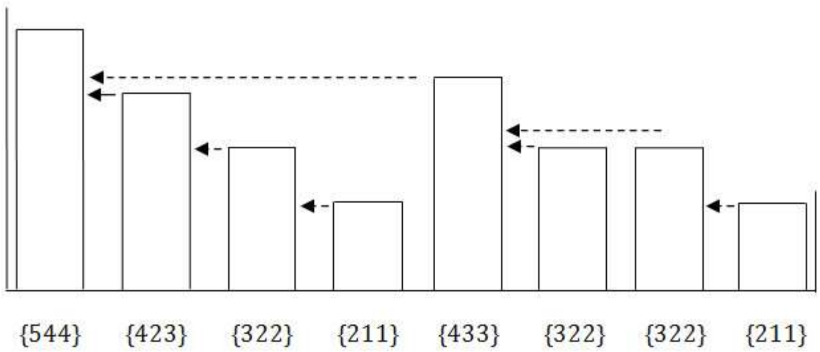


Figure 5. A saw-like structure illustrating N_1

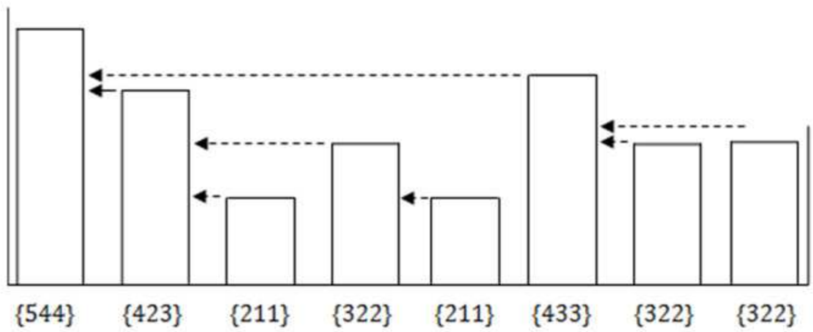


Figure 6. A saw-like structure illustrating N_2

Notice in Figures 5 and 6, that a bar (representing a subtree on the tree) can only be attached to a larger bar (representing a node on the tree) where the larger bar is a multiset containing the largest element of the smaller multiset representing the smaller bar. Such largest element must not be the largest in the larger bar. Notice also that a bar has not been attached to another bar of equal size (or height in this case).

Not all wellfounded multisets, even with the *conventional method*, can represent a tree. This is seen from the multiset $[\{544\}, \{432\}, \{211\}, \{432\}, \{211\}, \{211\}]$ since the supposed subtree structure $[\{211\}]$ of the last element $\{211\}$ does not have a branch to rest upon. The *saw rule* ensures, firstly, that a wellfounded multiset can suitably represent a tree; secondly, that every tree can be represented in the form of a wellfounded multiset (this is the only property that the conventional method ensures) and thirdly, that this representation is injective from a set of trees to the class of multisets representing such trees.

Remark.
It is important to observe that in order to construct a tree using a unique permutation of a given wellfounded multiset, the use of *the saw rule* in building an element upon a preceding element is not only that the subsequent element be smaller than the preceding one, but also that its largest element must belong to the preceding element, and that the multiplicity of such subsequent element in an uphill submultiset of elements must not be greater than the number of occurrences of such largest element in the preceding element.

For example, in $[\ldots\{543\}, \{211\}\ldots]$, the subtree representing $[\{211\}]$ cannot be built upon the subtree representing $[\{543\}]$ since the largest element 2 of $\{211\}$ does not belong to $\{543\}$. But in $[\ldots\{543\}, \{411\}\ldots]$, $[\{411\}]$ can be built upon $[\{543\}]$ since the largest element 4 of $\{411\}$ is in $\{543\}$. Again, we cannot build any subtree using $[\ldots\{543\}, \{411\}, \{411\}, \{411\}\ldots]$ since 4 does not appear up to three times in $\{543\}$. However, $[\ldots\{5444\}, \{411\}\ldots]$ represents a subtree. Such an element as 4 in this case is called a *join* in a *tree structure*. We use square brackets for a *tree structure* since a tree structure is a list – an ordered sequence of elements with repetitions allowed.

To avoid misinterpretation, we shall not use a tree and a tree structure interchangeably. As mentioned in the introduction, a tree is an **acyclic connected graph**. On the other hand, a *tree structure* is a

multiset representation of a tree by exploiting the *saw rule*, e.g., N_1 and N_2 above. Notwithstanding the fact that there is a one-to-one correspondence between the two, this distinction is useful especially when we shall be applying the concept of tree structures to membrane computing. The same argument goes for a subtree and a subtree structure.

Having discussed the representation of a binary tree in order of a saw-like permutation of a wellfounded multiset (called a *tree structure*), we consider it necessary to generalize our discussion to capture the concepts of an n-nary tree structure (corresponding to a tree with exactly n branches on each node of the tree) and a *general tree* (corresponding to a tree with n varying from branch to branch on the tree). Therefore, based on the discussions above, we now give the following formal definitions:

Definition xii.
Let S be a partially ordered multiset over a domain set D. A permutation $P(\tau(S))$ of a wellfounded multiset $\tau(S)$ of cardinality m, whose elements are submultisets of S is called an *n-nary tree structure* if and only if it satisfies the following properties:

1. For all $y \in_+ P(\tau(S))$ $\exists\, n \in \mathbb{N}$ such that $C(y) = n$, where $C(y)$ denotes the cardinality of y.

2. $y_1 \gg y_i$ in $P(\tau(S))$ for all $i = 2, 3, \ldots, m$.

3. For all $i, j \in \mathbb{N}$ with $y_i \gg y_j$ for $i < j$, if $\nexists\ k \in \mathbb{N}$ such that $y_i \gg y_k \gg y_j$ for $i < k < j$ then $\exists\ x_0 \in^1 y_i \cap y_j$ such that $x_0 = max\{x : x \in_+ y_j\}$ and $x_0 \neq max\{x : x \in_+ y_i\}$, where $y_i, y_k, y_j \in_+ P(\tau(S))$, $i = 1, 2, \ldots, m-1$, $k = 2, 3, \ldots, m-1$ and $j = 2, 3, \ldots, m$.

   In other words, it says that y_i *yields* y_j via x_0 or y_j is an immediate successor of y_i or y_i is an immediate predecessor of y_j, and we denote this by $y_i \overset{\{x_0\}}{\twoheadrightarrow} y_j$.

4. For each $z \in P(\tau(S))$ and for a given $x \in_+ S$ the multiset $Y = \{y : z \overset{\{x\}}{\twoheadrightarrow} y\}$ is such that $C(Y) \leq m_z(x)$, where $C(Y)$ is

the cardinality of Y and $m_z(x)$ is the multiplicity of x in z.

In the above definition, the first condition is known as *the equal cardinality condition*. The second condition is known as *the base condition* (or *the saw rule condition*) and y_1 is *the base* of the tree structure. The third condition is the *join condition* and x_0 is *the join* between two subtree structures. The fourth condition is *the parallelism condition*. The parallelism condition ensures that the multiplicity of a join in a node is greater than or equal to the number of subtree structures (or branches) joinable to the node using such join.

Definition xiii.
Let S be a partially ordered multiset over a domain set D, a permutation $P(\tau(S))$ of a wellfounded multiset $\tau(S)$ of cardinality m, whose elements are submultisets of S, is called a *general tree structure* if and only if it satisfies the following properties:

1. $y_1 \gg y_i$ in $P(\tau(S))$ for all $i = 2, 3, \ldots, m$.

2. For all $i, j \in \mathbb{N}$ with $y_i \gg y_j$ for $i < j$, if $\nexists\ k \in \mathbb{N}$ such that $y_i \gg y_k \gg y_j$ for $i < k < j$ then $\exists\ x_0 \in^1 y_i \cap y_j$ such that $x_0 = max\{x : x \in_+ y_j\}$ and $x_0 \neq max\{x : x \in_+ y_i\}$, where $y_i, y_k, y_j \in_+ P(\tau(S))$, $i = 1, 2, \ldots, m-1$, $k = 2, 3, \ldots, m-1$ and $j = 2, 3, \ldots, m$.

   In other words, it says that y_i *yields* y_j via x_0 or y_j is an immediate successor of y_i or y_i is an immediate predecessor of y_j, and we denote this by $y_i \overset{\{x_0\}}{\twoheadrightarrow} y_j$.

3. For each $z \in P(\tau(S))$ and for a given $x \in_+ S$ the multiset $Y = \{y : z \overset{\{x\}}{\twoheadrightarrow} y\}$ is such that $C(Y) \leq m_z(x)$, where $C(Y)$ is the cardinality of Y and $m_z(x)$ is the multiplicity of x in z.

The first condition is *the base condition*. The second condition is *the join condition* and the third condition is *the parallelism condition*.

In the above two definitions, the tree structure $P(\tau(S))$ is said to be built over the multiset S with D as its domain. We define the root set of the tree structure $P(\tau(S))$ as the set $R = \{x \in D : x \in_+ y \; \forall \; y \in_+ P(\tau(S))\}$. If there is no confusion about which multiset S is intended, we simply write τ for $P(\tau(S))$.

The process by which a subtree structure yields another subtree structure in a tree structure is called a *succession.*

That is, if $y_1 \overset{\{x_1\}}{\twoheadrightarrow} y_2$ and $y_2 \overset{\{x_2\}}{\twoheadrightarrow} y_3$ then y_3 is a successor (not an immediate successor) of y_1 and we write $y_1 \overset{\{x_1,x_2\}}{\twoheadrightarrow} y_3$ (y_1 yields y_3 via the set $\{x_1, x_2\}$).

There are immediate successions between y_1 and y_2 and between y_2 and y_3. There is also a succession between y_1 and y_3, however, this is not an *immediate succesion.*

Definition xiv.
Given a tree structure τ over a multiset S, a tree structure σ over S is called a *subtree structure* of τ if and only if $\forall \; y, z \in_+ \sigma$ and $x \in_+ S$ such that $z \overset{\{x\}}{\twoheadrightarrow} y, \exists \; a, b \in_+ \tau$ with $y \subseteq a$ and $z \subseteq b$ such that $b \overset{\{x\}}{\twoheadrightarrow} a$.

In other words, a tree structure σ is a subtree structure of a tree structure τ if and only if it *inherits* all its immediate successions from the tree structure τ.

The following are some immediate consequences of this definition: A subtree structure of a tree structure may not necessarily be a submultiset of the tree structure and vice versa. There are subtree structures of a tree structure whose members are not elements of the tree structure. For example, for the tree structure $\tau = [\{544\}, \{423\}, \{322\}, \{211\}, \{433\}, \{322\}, \{322\}, \{211\}]$, the subtree structure $[\{54\}, \{43\}, \{32\}]$ of τ is not a submultiset of τ since all of its elements do not belong to τ. The submultiset $[\{544\}, \{322\}, \{211\}]$ of τ does not represent any subtree structure of τ, since it does not form a tree structure. The multiset $[\{544\}, \{433\}, \{21\}]$ is neither a submultiset of τ nor a subtree structure of τ while the multiset $[\{544\}, \{423\}, \{322\}]$ is both a submultiset of τ and a subtree structure of τ.

9 Tree Structure Based Representation of Membrane Structures

In this section, we apply the aforesaid technique to represent membrane structures. Membrane structures can be represented in the form of a tree ([10], p. 8). Let us consider the schematic representation of a membrane structure as in Figure 7. It is customary to label the largest membrane with the number 1, the next larger membrane with the number 2 and so on.

In our example, in order to have an intuitively clearer representation, we shall identify the membranes by their membrane sizes. For instance, a membrane labelled 2 will be identified by the membrane size M_2.

The schematic representation of the membrane structure (μ) in Figure 7 can be discussed as follows: M_1 contains M_2, M_3, M_5 and M_8; M_2 contains M_4 and M_6; M_3 contains M_7 while M_8 is empty. Also the ordering relations $M_2 > M_3 > M_5$ and $M_6 > M_4$ hold.

Let $S = \{M_i : i = 1, 2, \ldots, 9\}$. The size of a membrane is defined as the sum of the multiplicities of all the objects in the membrane ([9], p. 6). The domain set of the multisets consisting of the sizes of membranes in a membrane structure as elements, is wellfounded with the usual ordering, being a subset of the set $\mathbb{N}$ of natural numbers. Thus, S is wellfounded and it follows that a multiset having elements of S as elements of its elements is also wellfounded ([4]).

We note that the membrane sizes may change during the process of *transition*. In particular, the above relations may not hold especially for elementary membranes. However, this does not change the fact that the tree structure representations still apply. Figure 8 is the tree representation of the membrane structure in Figure 7.

The tree structure representation of the tree in Figure 8 is denoted by μ.

$$\mu = [\{M_1 M_2 M_3 M_5 M_8\}\{M_3 M_7\}\{M_2 M_4 M_6\}].$$

We now present the above representation in greater details. The contents of the membranes are represented by a letter such as a_{ij} of

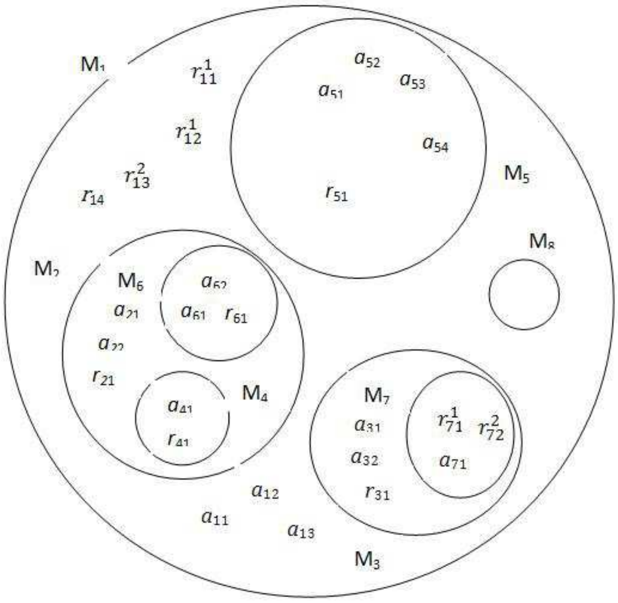


Figure 7. A membrane structure

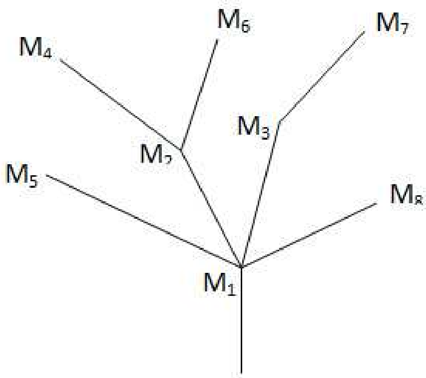


Figure 8. A tree representation of the membrane structure in Figure 7

the j^{th} element in the i^{th} membrane, while the m^{th} rule in the k^{th} membrane is represented by r^l_{km}, l being the rules' priorities if any. The membrane structure can further be expanded to show the contents of each membrane in the initial configuration of the system.

$$\begin{aligned}\mu \quad = \quad & [\{M_1[a_{11}a_{12}a_{13}r^1_{11}r^2_{12}r^3_{13}]M_2[a_{21}a_{22}r_{21}]M_3[a_{31}a_{32}r_{31}] \\ & M_5[a_{51}a_{52}a_{53}r_{51}]M_8[]\}\{M_3M_7[a_{71}r^1_{71}r^2_{72}]\} \\ & \{M_2M_4[a_{41}r_{41}]M_6[a_{61}a_{62}r_{61}]\}].\end{aligned}$$

The subscripts used for the labelling of the elements are just for illustration purpose and will not appear in the example we shall give below. The contents of membrane M_i have been grouped in the square brackets immediately following the membrane. If a membrane is contained in M_i, such containment follows the rule governing attachment in the tree structure. An empty membrane is denoted by an empty square bracket. Elementary membranes appear only once in the tree structure.

The first membrane is the skin membrane and is the only non-elementary membrane which is allowed to appear once in the tree structure. Any other membrane which is neither the skin membrane nor an elementary membrane will appear more than once, since it contains some other membranes.

10 Computation (An Example)

The following is the example, given in ([9], pp. 10-11), of a transition in a (cooperative) super-cell system. In this example we substitute *membrane structures* by *tree structures* (saw-like permutations of wellfounded multisets). Also, membranes are represented by subtree structures having both objects and rules (with associated rule priorities) as members. However, a membrane or a susbtree structure may exist having only rules as its members. The original tree structure representing the membrane structure is the initial configuration prior to the transitions.

Let us consider the following super-cell system of degree 4.

$\Pi = (V, \mu, M_1, \ldots, M_4, (R_1, \rho_1), \ldots, (R_4, \rho_4), 4)$,
$V = \{a, b, c, d\}$,
$\mu = [\{M_1 M_2 M_4\}\{M_2 M_3\}]$,
$M_1 = [aacr_{11}^1 r_{12}^1 r_{13}]$,
$M_2 = [ar_{21}]$,
$M_3 = [cdr_{31}]$,
$M_4 = [r_{41}]$,
$R_1 = \{r_{11} : c \to (c, in_4), r_{21} : c \to (b, in_4), r_{13} : a \to (a, in_2)b, dd \to (a, in_4)\}$,
$\rho_1 = \{r_{11} > r_{13}, r_{12} > r_{13}\}$,
$R_2 = \{r_{21} : a \to (a, in_3), ac \to \delta\}$,
$\rho_2 = \emptyset$,
$R_3 = \{r_{31} : a \to \delta\}$,
$\rho_3 = \emptyset$,
$R_4 = \{r_{41} : c \to (d, out), b \to b\}$,
$\rho_4 = \emptyset$.

$$C_0 : \mu = [\{M_1[aacr_{11}^1 r_{12}^1 r_{13}]M_2[ar_{21}]M_4[r_{41}]\}\{M_2 M_3[cdr_{31}]\}]$$

In the initial configuration C_0 we can apply a rule in membrane M_1 and one in membrane M_2. If we use the rule $c \to (b, in_4)$ in membrane M_1, the rule $b \to b$ can be applied non-stop and the computation will never end. Therefore, we will not use the rule $c \to (b, in_4)$, but the rule $c \to (c, in_4)$. Since both these rules can be applied and they have priorities over the rule $a \to (a, in_2)b$, this latter rule cannot be used. Hence, the object c is sent from membrane M_1 to membrane M_4 and at the same time the object a is sent from membrane M_2 to membrane M_3.

$$C_1 : \mu = [\{M_1[aar_{11}^1 r_{12}^1 r_{13}]M_2[r_{21}]M_4[cr_{41}]\}\{M_2 M_3[acdr_{31}]\}].$$

No rule can be applied on c in membrane M_1, hence the rule $a \to (a, in_2)b$ can be used. It will be used for both copies of a in membrane

M_1, and so two copies of a will be sent to membrane M_2 and two copies of b will remain in membrane M_1. At the same time, the rule $a \to \delta$ will be used in membrane M_3, dissolving it, and the rule $c \to (d, out)$ will be used in membrane M_4, sending a copy of d to membrane M_1. As a result of these operations, membrane M_1 will contain the string bbd, membrane M_2 will contain the string $aacd$, while membrane M_4 will contain no string; membrane M_3 no longer exists, therefore the rule $a \to (a, in_3)$ in membrane M_2 is useless for now.

$$C_2 : \mu = [\{M_1[bbdr^1_{11}r^1_{12}r_{13}]M_2[aacdr_{21}]M_4[r_{41}]\}].$$

The rule $ac \to \delta$ can be used in membrane M_2, dissolving it and releasing the remaining objects ad. Thus, membrane M_1 will contain the string $abbdd$.

$$C_3 : \mu = [\{M_1[abbddr^1_{11}r^1_{12}r_{13}]M_4[r_{41}]\}].$$

It is now possible for the first time to use the rule $dd \to (a, in_4)$ from membrane M_1. It consumes the two copies of d and sends a copy of a to membrane M_4. No further rule can be applied, and the "life" of the super-cell stops here.

$$C_4 : \mu = [\{M_1[abbr^1_{11}r^1_{12}r_{13}]M_4[ar_{41}]\}].$$

11 Conclusion

The paper is an attempt to indicate that a multiset-based tree model may prove useful in membrane computing and by extension to other computing devices, especially of biological orientation.

Moreover, the application of the saw-like permutation can be exploited in describing various algebraic properties of a *tree structure*, and hence, that of a *membrane structure*.

ACKNOWLEDGEMENT

We wish to acknowledge that most of the basic concepts are borrowed from the works of Dershowitz and Manna and that of Gh. Păun.

References

[1] D. A. Bailey, *Java Structure,* McGraw-Hill, 1999.

[2] W. Blizard, *Multiset theory*, Notre Dame Journal of formal logic, 1989, Vol. 30, pp. 36–66.

[3] W. Blizard, The development of multiset theory, *Modern logic*, 1991, Vol 1, pp. 319–352.

[4] N. Dershowitz, Z. Manna, *Proving Termination with Multiset Ordering*, Comm. ACM, 1975, Vol. 22, pp. 465–476.

[5] L. L. Dornhoff, F. E. Hohn, *Applied Modern Algebra*, Macmillan Publishing Co, 1978.

[6] J. Eyisi, *A Grammar of English, the Student's Companion,* Rex Charles publications, 2006.

[7] R. W. Hilton, *Managerial Accounting*, MacGraw-Hill/Irwin, 1996.

[8] J. G. Kemeny, *Introduction to finite Mathematics,* Prentice-Hall, 1957.

[9] Gh. Păun, *Computing with Membranes*, TUCS Report 208, Turku Centre for Computer Science, 1998.

[10] Gh. Păun, *Introduction to Membrane Computing*, Springer-Verlag, 2002.

[11] D. Singh, A. M. Ibrahim, Y. Tella, J. N. Singh, *Fundamentals of Multisets and their Applications*, Novi Sad J. Math, 2007, Vol. 37, pp. 73–92.

[12] D. J. Tailor, *Biological Science*, Cambridge, 1996.

[13] The P System web page: http://ppage.psystems.eu/.

D. Singh, C.M. Peter

Received October 30, 2010
Revised February 4, 2011

Department of Mathematics,
Ahmadu Bello University, Zaria, Nigeria

E–mails:
D. Singh: *mathdss@yahoo.com*,
C. M. Peter: *macpee3@yahoo.com*

Computer Science Journal of Moldova, vol.**19**, no.1(55), 2011

Maximal induced colorable subhypergraphs of all uncolorable $BSTS(15)$s

Jeremy Mathews

Abstract

A Bi-Steiner Triple System ($BSTS$) is a Steiner Triple System with vertices colored in such a way that the vertices of each block receive precisely two colors. When we consider all $BSTS(15)$s as mixed hypergraphs, we find that some are colorable while others are uncolorable. The criterion for colorability for a $BSTS(15)$ by Rosa is containing $BSTS(7)$ as a subsysytem. Of the 80 non-isomorphic $BSTS(15)$s, only 23 meet this criterion and are therefore colorable. The other 57 are uncolorable. The question arose of finding maximal induced colorable subhypergraphs of these 57 uncolorable $BSTS(15)$s. This paper gives feasible partitions of maximal induced colorable subhypergraphs of each uncolorable $BSTS(15)$.

1 Introduction

1.1 Mixed Hypergraphs

The concept of mixed hypergraphs was introduced in [4] in 1993 by V. Voloshin. A **mixed hypergraph** is a triple H=(X,C,D), where X is a finite vertex set, C is a family of subsets of X called C-edges, and D is a family of subsets of X called D-edges. If a mixed hypergraph has $C = D$, then it is called a **bi-hypergraph** and its edges are called **bi-edges**. For a proper coloring of a mixed hypergraph, C-edges must contain at least two vertices of the same color and D-edges must contain at least two vertices of different colors. If at least one proper coloring of a mixed hypergraph exists, then it is called **colorable**; otherwise, it is called **uncolorable**. The **strong deletion of a vertex** from a hypergraph is the removal of x from H along with all C-edges and D-edges containing x. An **induced subhypergraph** *H'=(X',C',D')* of a mixed hypergraph *H=(X,C,D)* is obtained through the strong deletion of vertices from *X*. If H is an uncolorable mixed hypergraph and after the deletion of some vertex (or vertices) the induced subhypergraph *H'* becomes colorable, then *H'* is called an **induced colorable subhypergraph** of *H*. It is called a **maximal induced colorable subhypergraph** of H if, with the addition of any of the deleted vertices from *H*, it becomes uncolorable. [5]

1.2 Steiner Triple Systems

A **Steiner System** is a block design of the form $S(t, k, v)$ where v is the total number of vertices, k is the number of vertices that are in each block and t is the number of distinct vertices that appear together in precisely λ blocks. When $\lambda = 1$, $k = 3$, $t = 2$, and $v \equiv 1$ or $3(mod6)$, it is called a **Steiner Triple System** or $STS(v)$ [6]. If a $STS(v)$ is considered as a bi-hypergraph *H=(X,B,B)* where X is a finite vertex set and $|X| = v$, B is the family of 3-element subsets of X known as blocks (which are bi-edges), and each pair of distinct elements of X appear together in precisely one block, then it is called a **Bi-Steiner Triple System** or $BSTS(v)$. Bi-Steiner Triple Systems are also known as bi-colorings of Steiner Triple Systems [1]. Since each block consists of bi-edges and each block contains exactly 3 vertices, the vertices of each block are colored with precisely 2 colors by the definition of a proper coloring of a mixed hypergraph.

2 Method

The number of blocks b in a $BSTS(v)$ is given by the following:

$$b = \frac{v(v-1)}{6}.$$

So in the case of a $BSTS(15)$, $b = 35$. Also, each vertex of a $BSTS(v)$ is contained in r blocks, where

$$r = \tfrac{v-1}{2}.$$

Therefore, for a $BSTS(15), r = 7$. If just one vertex is strongly deleted from X, then $|X'| = 15 - 1 = 14$ and $b = 35 - 7 = 28$. It follows that there exists a maximal induced colorable subhypergraph *H'=(X',B',B')* of any uncolorable $BSTS(15)$ *H=(X,B,B)* with $|X'| \leq 14$ and $|B'| \leq 28$. In this paper, the case where $|X'| = 14$ and $|B'| = 28$ is proved for every uncolorable $BSTS(15)$. In order to find any *H'* of *H*, we must strongly delete a vertex and test for colorability using any number of colors.

3 Theorem and Proof

Theorem 1. *Every uncolorable* $BSTS(15)$ *H=(X,B,B) has some maximal induced colorable subhypergraph H'=(X',B',B') obtained through the strong deletion of exactly one vertex from X.*

Proof. To show that all 57 uncolorable $BSTS(15)$s have a maximal induced colorable subhypergraph obtained by one vertex deletion, we delete vertex {15}, or as the vertices are labeled in [2] vertex {e}, and show a proper coloring by listing a feasible partition of *X'*. Note that obviously the partitions listed below are not all of the possible partitions for *H'*, nor is {15} (or {e}) the only vertex which can be deleted in order to obtain *H'* from *H*. Some systems can have any vertex from X deleted to obtain *H'* while others only have certain vertices that can be deleted to obtain *H'*. Also, many of the maximal induced colorable subhypergraphs of these uncolorable $BSTS(15)$s are colorable using 3 and 4 colors; however, here we only show a partition into 3 cells for 3 colors. These calculations were made with the aid of a computer program written by B. Tolbert and myself. The systems below are numbered as in [2] (Note that systems no. 1-22 and no.61 are the 23 colorable $BSTS(15)$s).

$BSTS(15)$ no. 23	$\{1,2,9,13,14\}\bigcup\{3,12\}\bigcup\{4,5,6,7,8,10,11\}$
$BSTS(15)$ no. 24	$\{1,2,4,7,8\}\bigcup\{3,5,6\}\bigcup\{9,10,11,12,13,14\}$
$BSTS(15)$ no. 25	$\{1,2,4\}\bigcup\{3,5,6,7\}\bigcup\{8,9,10,11,12,13,14\}$
$BSTS(15)$ no. 26	$\{1,2,4\}\bigcup\{3,5,6,7\}\bigcup\{8,9,10,11,12,13,14\}$
$BSTS(15)$ no. 27	$\{1,2,4,7,11\}\bigcup\{3,5,6\}\bigcup\{8,9,10,12,13,14\}$
$BSTS(15)$ no. 28	$\{1,2,4,7,8\}\bigcup\{3,5,6\}\bigcup\{9,10,11,12,13,14\}$
$BSTS(15)$ no. 29	$\{1,2,4,7,12\}\bigcup\{3,5,6\}\bigcup\{8,9,10,11,13,14\}$
$BSTS(15)$ no. 30	$\{1,2,4,7,11\}\bigcup\{3,5,6\}\bigcup\{8,9,10,12,13,14\}$
$BSTS(15)$ no. 31	$\{1,2,4,7,11\}\bigcup\{3,5,6\}\bigcup\{8,9,10,12,13,14\}$
$BSTS(15)$ no. 32	$\{1,2,4,7,11\}\bigcup\{3,5,6\}\bigcup\{8,9,10,12,13,14\}$
$BSTS(15)$ no. 33	$\{1,2,4,7,8\}\bigcup\{3,5,6\}\bigcup\{9,10,11,12,13,14\}$
$BSTS(15)$ no. 34	$\{1,2,4,7,8\}\bigcup\{3,5,6\}\bigcup\{9,10,11,12,13,14\}$
$BSTS(15)$ no. 35	$\{1,2,7,10,13,14\}\bigcup\{3,12\}\bigcup\{4,5,6,8,9,11\}$
$BSTS(15)$ no. 36	$\{1,3,5,7,8,12,14\}\bigcup\{2,10,11\}\bigcup\{4,6,9,13\}$
$BSTS(15)$ no. 37	$\{1,3,5,7,8,10\}\bigcup\{2,4\}\bigcup\{6,9,11,12,13,14\}$
$BSTS(15)$ no. 38	$\{1,2,7,9,10,13,14\}\bigcup\{3,4,5,6,11\}\bigcup\{8,12\}$
$BSTS(15)$ no. 39	$\{1,3,4,6,13,14\}\bigcup\{2,12\}\bigcup\{5,7,8,9,10,11\}$

$BSTS(15)$ no. 40	$\{1,2,7,10,13,14\}\bigcup\{3,12\}\bigcup\{4,5,6,8,9,11\}$
$BSTS(15)$ no. 41	$\{1,2,7,10,13,14\}\bigcup\{3,12\}\bigcup\{4,5,6,8,9,11\}$
$BSTS(15)$ no. 42	$\{1,3,5,7,8,10\}\bigcup\{2,4\}\bigcup\{6,9,11,12,13,14\}$
$BSTS(15)$ no. 43	$\{1,4,6,8,10,12,14\}\bigcup\{2,3,7\}\bigcup\{5,9,11,13\}$
$BSTS(15)$ no. 44	$\{1,2,5,6,13\}\bigcup\{3,8,9,10,11,12,14\}\bigcup\{4,7\}$
$BSTS(15)$ no. 45	$\{1,2\}\bigcup\{3,4,5,6,7\}\bigcup\{8,9,10,11,12,13,14\}$
$BSTS(15)$ no. 46	$\{1,3,5,7,8,10\}\bigcup\{2,4\}\bigcup\{6,9,11,12,13,14\}$
$BSTS(15)$ no. 47	$\{1,3,5,7,8,10\}\bigcup\{2,4\}\bigcup\{6,9,11,12,13,14\}$
$BSTS(15)$ no. 48	$\{1,5\}\bigcup\{2,3,6,7,10,11\}\bigcup\{4,8,9,12,13,14\}$
$BSTS(15)$ no. 49	$\{1,3,5,7,9,11\}\bigcup\{2,10\}\bigcup\{4,6,8,12,13,14\}$
$BSTS(15)$ no. 50	$\{1,2\}\bigcup\{3,4,5,6,7\}\bigcup\{8,9,10,11,12,13,14\}$
$BSTS(15)$ no. 51	$\{1,3,5,7,8,10\}\bigcup\{2,4\}\bigcup\{6,9,11,12,13,14\}$
$BSTS(15)$ no. 52	$\{1,3,4,6,11,13,14\}\bigcup\{2,5,8,9,12\}\bigcup\{7,10\}$
$BSTS(15)$ no. 53	$\{1,4,6,9,10,12\}\bigcup\{2,3,8,11,13,14\}\bigcup\{5,7\}$
$BSTS(15)$ no. 54	$\{1,5\}\bigcup\{2,3,6,7,10,11\}\bigcup\{4,8,9,12,13,14\}$
$BSTS(15)$ no. 55	$\{1,2,4,7,8,11\}\bigcup\{3,5,9\}\bigcup\{6,10,12,13,14\}$
$BSTS(15)$ no. 56	$\{1,3,4,6,11,13,14\}\bigcup\{2,5,8,9,12\}\bigcup\{7,10\}$

$BSTS(15)$ no. 57	$\{1,3,5,7,8,10\}\bigcup\{2,4\}\bigcup\{6,9,11,12,13,14\}$
$BSTS(15)$ no. 58	$\{1,4,6,9,10,12\}\bigcup\{2,3,8,11,13,14\}\bigcup\{5,7\}$
$BSTS(15)$ no. 59	$\{1,11\}\bigcup\{2,3,6,7,8,9\}\bigcup\{4,5,10,12,13,14\}$
$BSTS(15)$ no. 60	$\{1,2,4,9,10,13,14\}\bigcup\{3,8\}\bigcup\{5,6,7,11,12\}$
$BSTS(15)$ no. 62	$\{1,2,4,7,12\}\bigcup\{3,5,6\}\bigcup\{8,9,10,11,13,14\}$
$BSTS(15)$ no. 63	$\{1,2,4,7,11\}\bigcup\{3,5,6\}\bigcup\{8,9,10,12,13,14\}$
$BSTS(15)$ no. 64	$\{1,2,4,7,11\}\bigcup\{3,5,6\}\bigcup\{8,9,10,12,13,14\}$
$BSTS(15)$ no. 65	$\{1,12\}\bigcup\{2,3,8,9,13,14\}\bigcup\{4,5,6,7,10,11\}$
$BSTS(15)$ no. 66	$\{1,2,5,6,8,13,14\}\bigcup\{3,4,7,9,12\}\bigcup\{10,11\}$
$BSTS(15)$ no. 67	$\{1,8,10\}\bigcup\{2,3,4,5,9\}\bigcup\{6,7,11,12,13,14\}$
$BSTS(15)$ no. 68	$\{1,12\}\bigcup\{2,3,8,9,13,14\}\bigcup\{4,5,6,7,10,11\}$
$BSTS(15)$ no. 69	$\{1,8,10\}\bigcup\{2,3,4,5,9\}\bigcup\{6,7,11,12,13,14\}$
$BSTS(15)$ no. 70	$\{1,2,5,9,10,13,14\}\bigcup\{3,8\}\bigcup\{4,6,7,11,12\}$
$BSTS(15)$ no. 71	$\{1,2,5,9,10,13,14\}\bigcup\{3,8\}\bigcup\{4,6,7,11,12\}$
$BSTS(15)$ no. 72	$\{1,5,7,9,11,12\}\bigcup\{2,3,4,10,13,14\}\bigcup\{6,8\}$
$BSTS(15)$ no. 73	$\{1,3,4,7,11,13,14\}\bigcup\{2,5,6,10,12\}\bigcup\{8,9\}$
$BSTS(15)$ no. 74	$\{1,6,9,11,12\}\bigcup\{2,3,4,5,10,13,14\}\bigcup\{7,8\}$

$BSTS(15)$ no. 75	$\{1,2,5,6,10,12\} \bigcup \{3,7,11,13,14\} \bigcup \{4,8,9\}$
$BSTS(15)$ no. 76	$\{1,9\} \bigcup \{2,3,6,7,12,13,14\} \bigcup \{4,5,8,10,11\}$
$BSTS(15)$ no. 77	$\{1,4,6,8,10\} \bigcup \{2,3,9\} \bigcup \{5,7,11,12,13,14\}$
$BSTS(15)$ no. 78	$\{1,4,6,8,10,13\} \bigcup \{2,3,5\} \bigcup \{7,9,11,12,14\}$
$BSTS(15)$ no. 79	$\{1,3\} \bigcup \{2,6,7,12,13,14\} \bigcup \{4,5,8,9,10,11\}$
$BSTS(15)$ no. 80	$\{1,3\} \bigcup \{2,8,9,12,13,14\} \bigcup \{4,5,6,7,10,11\}$

Therefore, each uncolorable $BSTS(15)$ has a maximal induced colorable subhypergraph using 3 colors with the deletion of exactly one vertex.

□

4 How the Program Works

This program was written in the C++ language and contains several sub-programs and functions. We created incidence matrices for each colorable $BSTS(15)$ as text files in the source code. We added display functions for all relevant data to check our results and to double check the computer results. We started by creating files that would find partitions and collect them with different permutations of colors being collected as a single partition. The main program calls the incidence matrix that is specified in a subprogram and displays it along with each block of vertices. The program then prompts the user to enter the number of colors that are to be used and then the number of colorings the user wishes to find (first 10 or first 200 for example, or the user can enter -1 for all colorings). If the user wants to find all proper colorings, the program runs an exhaustive search of all possible colorings from a string of all 0s to a string of all 3s. If a coloring is proper, then the coloring is displayed and counted; and if it is not a proper coloring, then that coloring is skipped. Also, if the coloring is proper, then that feasible partition is stored. After all proper colorings have been found and displayed and counted, the monitor prompts the user to press any key to see the feasible partitions displayed and counted and the number of colorings of each partition. All of the different permutations of colors of the partitions that were stored from the proper colorings are grouped together by the computer and only the first permutation of colors is displayed. For example, 011222233333333 would be displayed and 122333300000000 would not be displayed because it is a permutation of the same partition where vertex 1 is mapped to one color, vertices $2, 3$ are mapped to one color, vertices $4 - 7$ are mapped to one color, and vertices $8 - 15$ are mapped to one color. By altering the incidence matrices to account for vertex deletions, we were able to use this program to test the colorability of these induced subhypergraphs of every uncolorable $BSTS(15)$. This enabled us to check the accuracy of our hypothesis and our results; and by displaying all of the relevant data on the monitor, we were able to check the accuracy of the computer results [3].

5 Concluding Remarks

This paper shows that the minimum number of vertex deletions required for a maximal induced colorable subhypergraph of each uncolorable $BSTS(15)$ is precisely one. Obviously with the deletion of vertex {15}, there are more partitions than those listed in the proof; however, they are not needed to show the existence of one *H'* from *H*. However, it remains to find a maximal partial colorable subhypergraph for each uncolorable $BSTS(15)$. The **weak removal of edges** is the deletion of edges without changing the vertex set *X*. A **partial subhypergraph** *H'=(X,B',B')* of an uncolorable $BSTS(15)$ *H=(X,B,B)* is obtained through the weak removal of C-edges and/or D-edges resulting in *B'*. It is called a **partial colorable subhypergraph** if with the weak removal of edges, *H'* becomes colorable. Given *H*, which is an uncolorable $BSTS(15)$, then *H'*, which is a partial colorable subhypergraph, is called a **maximal partial colorable subhypergraph** of *H* if adding any C-edge or D-edge of *H* to *H'* makes it uncolorable [5].

References

[1] C. J. Colbourn, J. Dinitz, A. Rosa. *Bi-coloring Steiner Triple Systems.* The Electronic Journal of Combinatorics 6, (1999), R25.

[2] C. J. Colbourn, A. Rosa. *Triple Systems.* (Oxford University Press, Inc.), New York, (1999).

[3] J. Mathews, B. Tolbert. *All proper colorings of every colorable* $BSTS(15)$. Computer Science Journal of Moldova, 1 (2010), 41–53.

[4] V. I. Voloshin. *The mixed hypergraphs.* Computer Science Journal of Moldova, 1 (1993), 49–52.

[5] V. I. Voloshin. *Introduction to Graph and Hypergraph Theory.* (Nova Science Publishers, Inc.), New York, (2009), 202–204.

[6] E. W. Weisstein, *Steiner Triple System*, Mathworld – A Wolfram Web Source, http://mathworld.wolfram.com/SteinerTripleSystem.html, (2009).

Jeremy Mathews

Received March 15, 2011

Troy University,
Troy, AL 36082
E–mail: *jmathews9106@gmail.com*

Computer Science Journal of Moldova, vol.**19**, no.1(55), 2011

Complex of abstract cubes and median problem

Sergiu Cataranciuc Petru Soltan

Abstract

In this paper a special complex $\mathcal{K}^n$ of abstract cubes [2, 3], which contains only n-dimensional cubes is examined. The border of this complex is an abstract $(n-1)$-dimensional sphere. It is proved that the abstract sphere contains at least one 0-dimensional cube, which belongs to exactly n cubes with dimension 1, if the complex is a homogeneous n-dimensional tree. This result allows to solve, in an efficient way, the problem of median for a skeleton of size 1 of the tree with weighted vertices and edges. The algorithm to calculate the median without using any metric is described. The proposed algorithm can be applied with some modifications, for arbitrary complex of abstract cubes.

Mathematics subject classification: 18F15, 32Q60, 68R10

Keyword and phrases: Abstract cube, complex, multidimensional homogeneous tree, abstract sphere, median

1 Introduction

Let (X, d) be a metric space, determined by a finite set X, with ordered elements $X = \{x_1, x_2, ..., x_n\}$, and $f(x) = \sum\limits_{i=1}^{n} d(x, x_i)p(x_i)$ be a defined function on X, where $p(x_i)$ is a positive real number, called weight of the element $x_i \in X$.

Definition 1.1. *Point $x^* \in X$, which satisfies the following equality*

$$f(x^*) = \min_{x \in X} \sum_{i=1}^{n} d(x, x_i)p(x_i) \tag{1.1}$$

is called ***median*** *of* X.

Median applications for solving applied problems, especially in service location problems, are well known. But calculating procedure of the median according to the formula (1.1), usually in some circumstances is a difficult issue. In the works [2], [3], [4], efficient algorithms for finding the median without using metric are proposed for some special complexes. This paper largely generalizes results presented in the papers [3] and [4]. However, if in [2] the median is calculated by effective way for weighted tree, then in this paper the problem is studied for an abstract homogeneous tree $\mathcal{K}^n$, $n \geq 1$, with dimension n. For the 1-dimensional skeleton (see below) of this tree an efficient algorithm for finding the median without using any metric is proposed.

In the paper [2] algorithm for calculating median for a finite tree with positive weights of the elements is described. It is an elegant algorithm, which has found many practical applications. The same situation we have in the case of a quadrilateral complex [3] with weighted edges especially for the Euclidean space E^2.

The problem, which is studied in this paper was formulated for the first time, and partially solved in the late '60s, at the Institute of Mathematics with Computer Center of Academy of Sciences of Moldova. At that time, the two-dimensional case of the problem was studied. For the greater dimensions any effective solutions weren't obtained. Last years researches have led to some theoretical results for dimensions greater than 2, using new constructions (such is the complex of abstract cubes). These complexes are studied through their groups of direct homologies [5-7].

2 Homogeneous complexes of abstract cubes

Let $r = \{Q^1_{i_1}, Q^1_{i_2}, ..., Q^1_{i_n}\}$, $n \geqslant 2$, be a landmark of 1-dimensional oriented cubes with a common vertex (a 0-dimensional cube), considered the origin of these cubes. Any one-dimensional and oriented cube can be an ordered pair of 0-dimensional cubes (vertices of 1-dimensional cube). Consider $Q^0_{i_0}$ – the origin of cubes which forms the landmark r

and $Q^1_{i_1} = (Q^0_{i_0}, Q^0_{i_1})$, $Q^1_{i_2} = (Q^0_{i_0}, Q^0_{i_2}), ..., Q^1_{i_n} = (Q^0_{i_0}, Q^0_{i_n})$. Next, we describe the landmark r by ordered tuple of indices $(i_{1,}i_{2,...,}i_n)$, which actually is a permutation of the tuple *(1, 2,..., n)*.

It is clear that any landmark r of n oriented cubes, with dimension 1, determines unequivocally an n-dimensional oriented cube Q^n and vice versa. This cube is also determined by ordered tuple of $(i_{1,}i_{2,...,}i_n)$ indices.

Definition 2.1. *If the number of inversions k of tuple $(i_{1,}i_{2,...,}i_n)$ is even (odd), then the n-dimensional cube Q^n, which is determined by the landmark r, is called* ***positively (negatively) oriented****.*

Corollary 2.1. *The same tuple of indices $(i_{0,}i_{1,...,}i_n)$ describes unequivocally an n-dimensional simplex. Therefore, we consider that the sign of an n-dimensional cube coincides with the sign of n-dimensional simplex, determined by the same landmark r of 1-dimensional cubes.*

Definition 2.2. *Two cubes with dimension 1 are called* ***abstract-convex*** *cubes, if their emptinesses [23] have only one common point $x \in M$ of intersection. A cube Q^m, $2 \leqslant m \leqslant n$, is called abstract-convex if in intersection with another abstract-convex cube Q^1 we obtain at most an abstract-convex cube with dimension 1.*

According to Boltyanski [5] and Hilton [7], any abstract and orientable manifold V^n_p without borders [1], [23], which is determined by a finite number of n-dimensional and abstract-convex cubes, can be oriented if it satisfies the following property: any two arbitrary n-dimensional cubes of V^n_p, which have an intersection of an $(n-1)$-dimensional cube, are both positively or negatively directed. In other words, for these two cubes exists an n-dimensional way [22], that unites them – the so-called **strong orientation**, as a connected and oriented graph [11] is.

Definition 2.3. *Let V^n_p be an abstract manifold, determined by abstract-convex cubes, and x', x'' are two points of infinite set M [22],* *which does not belong to* V_p^n*, and determines an abstract-convex 1-dimensional cube. Let the edge* (x', x'') *intersects* V_p^n *in an odd number of points from* M*. There are two points* x_1 *and* x_2 *in* V_p^n *which leads to a nonorientable one-dimensional cube (edge) of* V_p^n*, denoted by* (x_1, x_2)*. We consider that the intersection of edges* (x', x'') *and* (x_1, x_2) *is only a single point* $y \in M$*. Points* x' *and* x'' *belong to* V_p^n*. One of these two vacuums we consider as intern and will denote by* $\mathbf{int}V_p^n$*, and another – exterior, denoted by* $\mathbf{ext}V_p^n$ *(a situation which generalizes theorem of Jordan and Holder [14], [24]).*

The abstract and oriented cube is defined by abstract simplexes in inductive way [1], starting with an abstract oriented arc. A complex of abstract-convex cubes

$$\mathcal{K}^n = \left\{ Q_\lambda^p : 0 \leq p \leq n,\ \lambda \in \Lambda,\ \dim \Lambda < \infty \right\},$$

where $n = dim\mathcal{K}^n$, and his groups of direct homologies over the group $\mathbb{Z}$ of integers:

$$\Delta^0(\mathcal{K}^n, \mathbb{Z}),\ \Delta^1(\mathcal{K}^n, \mathbb{Z}), ..., \Delta^n(\mathcal{K}^n, \mathbb{Z}), \tag{2.1}$$

is defined as it is shown in [7] (case Euclidean space E^n), and as for the complex of abstract simplexes [16].

Definition 2.4. *If* K^n *and* K^p *are two complexes of abstract-convex cubes, so that* $K^p \subset K^n$*, then* K^p *is called subcomplex of* K^n*.*

Definition 2.5. *A set of all cubes of* K^n*, with dimensions* $p \geq 0$*, is called* **p** ***-dimensional skeleton*** *of* K^n *and is denoted by* $sk(p)K^n$*,* $0 \leq p \leq n-1$*.*

Obviously, $\mathbf{sk}(p)\mathcal{K}^n$ is a subcomplex of $\mathcal{K}^n$ and $\mathbf{sk}(1)\mathcal{K}^n$ is its oriented graph [11]. Next we will make the following denotations:

- $\mathbb{Q}^p$ – the family of all p-dimensional and abstract-convex cubes of $\mathcal{K}^n$, $0 \leq p \leq n$;

- $G = (X; U)$ – a graph which represents the 1-dimensional skeleton of $\mathcal{K}^n$, where $X = \mathbb{Q}^0$, and $U = \mathbb{Q}^1$. Hence $\mathbb{Q}^0$ and $\mathbb{Q}^1$ are the sets of vertices and arcs of G.

Definition 2.6. *The complex K^n of abstract-convex cubes, satisfying the following conditions:*

a) if Q^p, $0 \le p \le n$ is an element of K^n, then every facet $Q^k \subset Q^p$, $0 \le k < p$, is an element of K^n;

b) for every two cubes Q^{p_1} and Q^{p_2} of K^n, the intersection $Q^{p_1} \cap Q^{p_2}$ is empty or is an element of K^n, where $0 \le p_1, p_2 \le n$;

c) any cube Q^k of K^n, $0 \le k \le n$, belongs at least to one n-dimensional cube Q^n of K^n;

d) for any two subcomplexes K_1^n and K_2^n of K^n, which satisfy conditions a)-c) and $K_1^n \cup K_2^n = K^n$, their intersection is a subcomplex $K^p \subset K^n$, with dimension $p = n - 1$;

e) the homology group of rank zero is isomorphic to the group of integers $\mathbb{Z}$, i.e.

$$\Delta^0(K^n, \mathbb{Z}) \cong \mathbb{Z}; \tag{2.2}$$

f) the homology groups of rank $1, 2, 3, ...n$ are isomorphic to zero, i.e.

$$\Delta^1(K^n, \mathbb{Z}) \cong \Delta^2(K^n, \mathbb{Z}) \cong ... \cong \Delta^n(K^n, \mathbb{Z}) \cong 0, \tag{2.3}$$

*is called **homogeneous** n-dimensional abstract complex and is denoted by K_A^n.*

Following the conditions e) and f) of this definition for the complex $\mathcal{K}_A^n$ means that it is **connected** and **acyclic** [5], [7].

Definition 2.7. *The set of all $(n-1)$ -dimensional cubes of the complex K_A^n, which belong exactly to one n-dimensional cube Q^n of K_A^n, will be called the border of this complex and will be denoted by*

bdK^n_A. The set of vacuums of all cubes of K^n_A, which do not belong to the border bdK^n_A will be called interior of this complex and will be denoted by $intK^n_A$.

Definition 2.8. *[22] An orientable variety V^n_p, which is determined by abstract cubes will be called abstract n-dimensional sphere, if V^n_p satisfies the conditions:*

$$\chi(V^n_p) = 2, \quad for\ n = 2m,$$
$$\chi(V^n_p) = 0, \quad for\ n = 2m - 1, \tag{2.4}$$

We will denote this sphere by $S^n = V^n_0$, where χ is the Euler characteristic:

$$\chi\left(V^n_p\right) = \sum_{i=0}^{n} (-1)^i \alpha_i, \tag{2.5}$$

and α_i is the number of i-dimensional cubes, $0 \leq i \leq n$ [5-7].

Theorem 2.1. *If K^n_A is a convex-abstract and homogeneous n-dimensional cubic complex, then its boundary bdK^n_A is an abstract $(n-1)$-dimensional sphere S^{n-1}.*

Proof: First we will show that the border $bd\mathcal{K}^n_A$ is an abstract $(n-1)$-dimensional orientable manifold V^{n-1}_p, with genus $p = 0$.

Suppose the contrary. Let $p > 0$. In these circumstances, taking into account the homogeneity of $\mathcal{K}^n_A$ and the assumption that $p > 0$, we immediately obtain a contradiction: the complex $\mathcal{K}^n_A$ is not acyclic, (contradicts the condition f) of the Definition (2.6)); the variety V^{n-1}_p can be *cut* by a variety V^{n-2}_q, $1 \leq q \leq p$, and in this case we will have at least $\Delta^{n-1}(V^{n-1}_p, \mathbb{Z}) \not\cong 0$ [22]. In this case, satisfying the equalities (2.3), we obtain that V^{n-1}_p is not a variety. Then p loses its meaning. According to the homogeneity of $\mathcal{K}^n_A$ it leads to the fact that this complex does not satisfy the condition d) of the Definition 2.6: there does not exist two subcomplexes $\mathcal{K}^n_{A(1)}$ and $\mathcal{K}^n_{A(2)}$, such that

$\mathcal{K}^n_{A(1)} \cap \mathcal{K}^n_{A(2)} = \mathcal{K}^p_{A(3)}$, where $0 \leq p \leq n-2$. Thus $V_p^{n-1} = V_0^{n-1} = \mathbb{S}^{n-1}$. ■

Theorem 2.2. *If V_p^n is an orientable (abstract) variety, determined by abstract-convex cubes, and any cycle of size k, $1 \leq k \leq n-1$, is homologous to* 0*, then V_p^n, is an abstract sphere.*

Proof: The variety V_p^n is connected [22]. Therefore $\Delta^0(V_p^n, \mathbb{Z}) \cong \mathbb{Z}$. Let C_0^k be an arbitrary cycle of variety V_p^n. If all k-dimensional cycles, $1 \leq k \leq n-1$, are homologous to zero, then in $\mathbf{sk}(k)V_p^n$, each such cycle is the boundary of some subcomplex V_p^K This leads to the fact that the subgroup Z_0^k of the group of cycles Z^k coincides with Z_0^k. Therefore, the factor-group does not contain cycles which are not homologous to 0. This means that $\Delta^k(V_p^n, Z) \cong 0$, $1 \leq k \leq n-1$. Thus we have:

$$\Delta^1(V_p^n, \mathbb{Z}) \cong \Delta^2(V_p^n, \mathbb{Z}) \cong \ldots \cong \Delta^{n-1}(V_p^n, \mathbb{Z}) \cong 0.$$

Considering the Euler-Poancare equality [22]:

$$\chi(V_p^n) = \sum_{i=0}^{n} (-i)^i \alpha_i = \sum_{i=0}^{n} (-1)^i r_i,$$

where α_i is the number of i-dimensional cubes of V_p^n, and r_i is the rank of the group $\Delta^i(V_p^n, \mathbb{Z})$, $0 \leq i \leq n$, we obtain:

$$\chi(V_p^n) = 1 + 0 + \ldots + (-1)^n 1 = \left\{ \begin{array}{ll} 2, & \text{for } n = 2m; \\ 0, & \text{for } n = 2m-1 \end{array} \right\}.$$

However, according to the Definition 2.8, the variety V_p^n is an abstract n-dimensional sphere $\mathbb{S}^n$.■

In the paper [9] it is defined the notion of emptiness *(vacuum)* of a p-dimensional cube, $1 \leq p \leq n$. For the 0-dimensional cube Q^0, we consider that it coincides with its emptiness, and it will be called 0-dimensional emptiness.

Corollary 2.2. *Let $Q^k \in \mathbb{Q}^k$ be a convex-abstract, k-dimensional cube, $0 \leq k \leq n$, of some homogeneous complex. Then the variety bdQ^k is an abstract $(k-1)$-dimensional sphere.*

The proof of this assertion follows immediately from the Theorem 2.1.

3 n-dimensional homogeneous tree

Let us first explain some auxiliary issues.

Theorem 3.1. *If $\mathbb{S}^{n-1}$ is an abstract sphere determined by the border $bd\mathcal{K}_A^n$ of a homogeneous complex $\mathcal{K}_A^n$ of abstract-convex cubes, then it has at least one cube (vertex) $Q^0 \in \mathbb{Q}^0$, that exactly belongs to n 1-dimensional cubes (edges).*

To prove this theorem some additional examinations are necessary.

Lemma 3.1. *There exists at least one homogeneous n -dimensional complex $\mathcal{K}_A^n$ with the property that every cube $Q^p \in \mathbb{Q}^p$ from $\mathbf{int}K_A^n$, which intersects the border $\mathbf{bd}\mathcal{K}_A^n$ at most through a cube of the dimension $(p-1)$, $1 \leq p \leq n$, is incident to a number not less than 2^{n-p} of n-dimensional cubes.*

Proof is done in a constructive way. Let Q_1^n be a cube. Let us stick the cube Q_2^n to the cube Q_1^n so that $Q_1^n \cap Q_2^n = Q^{n-1}$ and continue this process in a way not contrary to the Definition 2.2. It is enough to stop this process when we obtain **the complex** $\mathcal{K}_A^n$.

According to the Definition 2.5. we may consider that any n-dimensional abstract sphere $\mathbb{S}^n$ is determined by a complex of n-dimensional abstract cubes or at least of abstract simplexes. Obviously, for any abstract sphere $\mathbb{S}^n$ there always exists a complex of n-dimensional abstract cubes $\mathcal{K}^{n+1}$ (not necessary homogeneous), so that its border is $\mathbb{S}^n$. The interior of the complex $\mathcal{K}^{n+1}$ will be considered to be the interior of the n-dimensional sphere $\mathbb{S}^n$ and will be denoted by $int\mathbb{S}^n$. The union $\mathbb{S}^n \cup int\mathbb{S}^n$ is called an *n-dimensional disk***.**

To examine the possibility of using the homogeneous complex of abstract cubes, in solving some practical problems, some additional issues are necessary to be examined. First let define the notion of parallel edges class of a homogeneous complex $\mathcal{K}_A^n$. Let Q^n be an n-dimensional abstract cube and $\mathcal{F}_1^{n-1}$ and $\mathcal{F}_2^{n-1}$ two n-dimensional opposite **facets** of this cube. The cube Q^n contains 2^{n-1} edges between the facet vertices $\mathcal{F}_1^{n-1}$ and $\mathcal{F}_2^{n-1}$.

Definition 3.1. *Edges of the cube Q^n, that merge vertices of the facets F_1^{n-1} and F_2^{n-1} are called parallel edges of this cube. The set of all parallel edges between two $(n-1)$-dimensional facets we will denote by $C(Q^n)$.*

Obviously the set content is unequivocally determined by the pair of opposite facets $\mathcal{F}_1^{n-1}$ and $\mathcal{F}_2^{n-1}$. In the cube Q^n there exist n different sets of parallel edges.

Let us iteratively choose a special family of n-dimensional cubes. We denote by i the numbers of cubes of this family. Initially we will consider this family empty, i.e. $i = 0$. To make this kind of family we should follow the following 4 steps:

p.1. *Let us choose an n-dimensional cube Q^n of the complex K_A^n. So, we may consider that $i = 1$. Let us denote by $Q_T^n(1)$ the family of cubes and by $C(Q_T^n(1))$ – one of the parallel edges set of the chosen cube Q^n.*

p.2. *Suppose that some $i \geqslant 1$ n-dimensional cubes from K_A^n were selected. Thus we obtained family of cubes $Q_T^n(i)$, for which the parallel edges set $C(Q_T^n(i))$ is known.*

p.3. *Let us choose a new cube (if there exists one) $Q_*^n \in \mathbb{Q}^n \backslash \mathbb{Q}_T^n(i)$, which contains at least an edge from $\mathbb{C}(\mathbb{Q}_T^n(i))$. We denote by $\mathbb{C}(Q_*^n)$ the parallel edges set of this cube, that satisfies the following relation: $\mathbb{C}(\mathbb{Q}_T^n(i)) \cap \mathbb{C}(Q_*^n) \neq 0$, and forms new sets*

$$\mathbb{Q}_T^n(i+1) = \mathbb{Q}_T^n(i) \cup \{Q_*^n\}$$

$$\mathbb{C}(\mathbb{Q}_T^n(i+1)) = \mathbb{C}(\mathbb{Q}_T^n(i)) \cup \mathbb{C}(Q_*^n).$$

p.4. *Repeat step 3 until it is possible. Since only finite n-dimensional homogeneous complex is studied, at a certain point we will reach the situation when we cannot select an n-dimensional cube from K_A^n that satisfies the step 3. In this case we will consider the searched family of n-dimensional cube formed.*

Definition 3.2. *The family of cubes, constructed according to the steps p.1-p.4, will be called* ***n-dimensional transversal*** *of the complex $\mathcal{K}_A^n$. We will denote this family by T^n, and by $\mathbb{C}(T^n)$ – the respective class (set) of parallel edges.*

By the Definition 2.6. any n-dimensional homogeneous complex $\mathcal{K}_A^n$ contains $m \geq n$ sets of parallel edges, that we will denote by $\mathbb{C}_1, \mathbb{C}_2, ..., \mathbb{C}_m$. The equality $m = n$ is true only if $\mathcal{K}_A^n$ is formed from a single n-dimensional cube Q^n.

From those mentioned above follows that any class of parallel edges $\mathbb{C}_i, 1 \leq i \leq m$, determines an n-dimensional transversal and vice versa, any n-dimensional transversal determines a set of parallel edges. Also we will consider that any class of parallel edges $\mathbb{C}_i$, $1 \leq i \leq m$, generates unequivocally an n-dimensional homogeneous subcomplex of abstract cubes (see definition 2.6). This subcomplex is determined by the facets family, both own and unfit, of all n-dimensional cubes from transversal T_i^n. We will denote this subcomplex by $\mathcal{K}_i^n$.

Corollary 3.1. *The n-dimensional subcomplex $\mathcal{K}_i^n$, generated by class of parallel edges $\mathbb{C}_i$, $1 \leq i \leq m$, of the complex $\mathcal{K}_A^n$, is an n-dimensional subcomplex from $\mathcal{K}_A^n$.*

The border of the complex $\mathcal{K}_i^n$ contains exactly two maximal $(n-1)$-dimensional and acyclic subcomplexes, that don't contain an edge from the respective class of parallel edges $\mathbb{C}_i$. We denote these subcomplexes by $\mathcal{K}_{i(1)}^{n-1}$ and $\mathcal{K}_{i(2)}^{n-1}$. Prior let call $\mathcal{K}_{i(1)}^{n-1}$ "left" facet, and $\mathcal{K}_{i(2)}^{n-1}$ - "right" facet of the transversal T_i^n.

Definition 3.3. *The off-empty union of all abstract-convex cubes with dimension k, $1 \leq k \leq n$, that belong to the complex K_i^n, but do not belong to the left and right facets of transversal T_i^n, will be called vacuum of transversal T_i^n and will be denoted by $V(T_i^n)$.*

Corollary 3.2. *Any transversal T_i^n of an abstract and homogeneous complex $\mathcal{K}_A^n$ divides this complex, through its vacuum $V(T_i^n)$, in two connected complexes of abstract cubes. Each of these complexes is not necessary homogeneous, and has at least the dimension equal to $n-1$.*

Definition 3.4. *n-dimensional transversals $T_{i_1}^n, T_{i_2}^n, ..., T_{i_q}^n$ of an abstract and homogeneous complex K_A^n, determined by the classes of parallel edges $\mathbb{C}_{i_1}^n, \mathbb{C}_{i_2}^n, ..., \mathbb{C}_{i_q}^n$, $2 \leq q \leq m$, are called pairwise neighbors transversals if any two transversals $T_{i_r}^n, T_{i_s}^n$, $0 \leq i_r, i_s \leq q$, satisfy the following conditions:*

1) $V(T_{i_r}^n) \cap V(T_{i_s}^n) = \varnothing$;
2) *the classes of parallel edges $\mathbb{C}_{i_r}$ and $\mathbb{C}_{i_s}$ contain each at least one edge $Q_1' \in \mathbb{C}_i$ and $Q_1'' \in \mathbb{C}_j$, that at their turn have a common vertex.*

From the Definition 3.4. it follows: For a transversal $T_{i_k}^n$ we will denote the left and the right facets by $T_{i_k(1)}^n$ and $T_{i_k(2)}^n$. If transversals $T_{i_1}^n, T_{i_2}^n, ..., T_{i_q}^n$ are *pairwise neighbors, then there exists a forked transversal,* $T_{i_{1,q}}^n = \bigcup\limits_{j=1}^{q} T_{i_j}^n \cap T_{i_k}^n$, that divides $\mathcal{K}_A^n$ in many connected subcomplexes. Let us denote by $st\mathcal{K}_{i_k(1)}^n$ and $dr\mathcal{K}_{i_k(2)}^n$ the components respectively determined by the "left" and "right" facets of transversal $T_{i_k}^n$. For these components the relation $st\mathcal{K}_{i_k(1)}^n = \mathcal{K}_A^n \backslash dr\mathcal{K}$ is true.

Let T_i^n, $1 \leq i \leq m$, be any transversal of the complex $\mathcal{K}_A^n$ and $\mathcal{K}_{i(1)}^{n-1}$ be its "left" facet

Definition 3.5. *$(n-1)$-dimensional maximal connected subcomplex of the complex $\mathcal{K}_A^n$, with the following properties:*

1) *any subcomplex contains the facet $\mathcal{K}_{i(1)}^{n-1}$ of the transversal T_i^n;*

2) *any two $(n-1)$-dimensional cubes of this subcomplex do not belong to an n-dimensional cube from $\mathcal{K}_A^n$,*

is called an $(n-1)$-dimensional transversal, determined by n-dimensional transversal T_i^n, $1 \leq i \leq m$.

According to the Definition 3.5., any $n-$dimensional transversal of K_A^n determines exactly two $(n-1)$-dimensional transversals. Also an $(n-1)$-dimensional transversal could be determined by more than one n-dimensional transversal.

By analogy with the notation of n-dimensional transversal, we will denote the $(n-1)$-dimensional transversal by T^{n-1}.

$(n-1)$-Dimensional transversal, determined by the transversal T_i^n and containing the facet $\mathcal{K}_{i(1)}^{n-1}$ will be denoted by $T_{i(1)}^{n-1}$; and one that contains the facet $\mathcal{K}_{i(2)}^{n-1}$, will be denoted by $T_{i(2)}^{n-1}$.

Lemma 3.2. *If T_i^n is an $(n-1)$-dimensional transversal from K_A^n, and $T_{i(1)}^{n-1}$, $T_{i(2)}^{n-1}$ are the $(n-1)$-dimensional transversals of the K_A^n, determined by the transversal T_i^n, then $T_{i(1)}^{n-1} \cap T_{i(2)}^{n-1} = \varnothing$, $1 \leq i \leq m$.*

Proof. Let $T_{i(1)}^{n-1}$ and $T_{i(2)}^{n-1}$ be $(n-1)$-dimensional transversals, determined by an n-dimensional T_i^n. Their intersection is not empty. This intersection may coincide with a cube $Q^0 \in \mathbb{Q}^0$ or at least with a cube $Q^{n-2} \in \mathbb{Q}^{n-2}$. If this intersection is a cube $Q^0 \in \mathbb{Q}^0$, then two n-dimensional cubes from T_i^n and incident to Q^0 are degraded, what is excluded. The same situation is for the cube $Q^{n-2} \in \mathbb{Q}^{n-2}$: Q^{n-2} is the opposite facet with the dimension $n-1$ of the n-dimensional cube. Thus we have at least an n-dimensional degraded cube.

Proof of the Theorem 3.1.

Let $T_{i_k}^{n-1}$, $1 \le k \le m$ be an $(n-1)$-dimensional transversal. If $k \ne m$, then we will consider the transversal $T_{i_{k+1}}^{n-1}$. According to lemma 3.2. we obtain $T_{i_k}^{n-1} \cap T_{i_{k+1}}^{n-1} = \varnothing$. Let us consider the following $(n-1)$-dimensional transversals till $T_{i_m}^{n-1}$. We obtain again $T_{i_{m-1}}^{n-1} \cap T_{i_m}^{n-1} = \varnothing$. Let us consider such $n-1$ sections of n-dimensional and abstract-connected cubes from $\mathcal{K}_A^n$, that two neighbors are intersected through an $(n-1)$-dimensional facet, opposite to $(n-1)$-dimensional facet of the first and the second cube, which is at the border of T_m^n and intersects through the n-dimensional cube $Q^n \in \mathbb{Q}^n$. This cube is abstract-convex. Cube Q^n contains a vertex from circular border of $(n-1)$-dimensional cubes, that has a vertex $Q^0 \in \mathbb{Q}^0$ with $n+1$ edges incident to $st\mathbb{S}^{n-1}$.■

Let $\mathcal{K}_A^n$ be a homogeneous abstract complex (see Definition 2.6.), but non-oriented.

As for the Theorem 2.2, the border $\mathbf{bd}\mathcal{K}_A^n$ is an $(n-1)$-dimensional cubical sphere $\mathbb{S}^{n-1}$. Let $\mathbf{st}(Q^0)\mathbb{S}^{n-1}$ be the star of the vertex Q^0, calculated on sphere $\mathbb{S}^{n-1}$.

Definition 3.6. *An n-dimensional homogeneous undirected abstract complex $\mathcal{K}_A^n$, that satisfies the following conditions:*

1) *any cube $Q^k \in \mathbf{int}\mathcal{K}_A^n$ belongs at least to 2^{n-k} n-dimensional cube from $\mathcal{K}_A^n$, $0 \le k \le n$;*

2) *if Q^0 is a vertex from $\mathbf{bd}\mathcal{K}_A^n$, that has exactly n incident arcs of $\mathbf{bd}\mathcal{K}_A^n$, and the star $\mathbf{st}(Q^0)\mathbf{bd}\mathcal{K}_A^n$ contains exactly n cubes of $(n-1)$ dimension of $\mathbf{bd}\mathcal{K}_A^n$, then the n-dimensional cube Q^n determined by this star belongs to the complex K_A^n;*

is called ***n-dimensional homogeneous tree****. We will denote this tree by A^n.*

By the Definition 3.6. we exclude the situation from Figure 1 (Figure 1 is shown for a better understanding of the examined situation.)

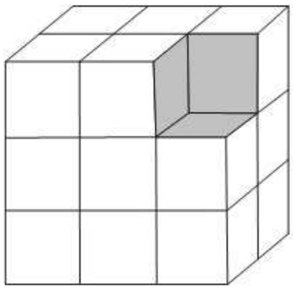


Figure 1. A complex, which does not satisfy the condition 2) of the Definition 3.6.

Note 3.1. *The first condition of the Definition 3.6. is assured by the Lemma 3.1.*

Let us next consider the following function:

$$P : \mathbb{Q}^0 \to R^+ \tag{3.1}$$

and the length of the edges from the classes of parallel edges $\mathbb{C}_1, \mathbb{C}_2, ..., \mathbb{C}_m$ equal to the numbers

$$d_1, d_2, ..., d_p, \tag{3.2}$$

where $d_i > 0$, for $1 \leq i \leq m$. Thus all edges that belongs to a class have the same length.

For any $Q^0 \in \mathbb{Q}^0$ number $p(Q^0)$ is the share of Q^0.

4 The representation of an n-dimensional tree in a normed space

Let $\mathbb{Q}^0 = \left\{Q^0_\lambda : \lambda \in \Lambda\right\}$ be the vertices set and $\mathbb{Q}^1 = \left\{Q^1_\mu : \mu \in \mathcal{M}\right\}$ be the edges set of the tree A^n. Let us fix an arbitrary chain $L^1 = (Q^1_{\mu_1}, Q^1_{\mu_2}, ..., Q^1_{\mu_k})$ with the origin $Q^0_{\lambda_i}$ and the extremity in $Q^0_{\lambda_j}$;

$\mu_1, \mu_2, ..., \mu_k \in \mathcal{M}$; $\lambda_i, \lambda_j \in \Lambda$. Next we will form an integer non-negative number (see conditions (3.1.) and (3.2.)).

$$d(Q^0_{\lambda_i}, Q^0_{\lambda_j}) = t_1 d_1 + t_2 d_2 + .. + t_k d_k, \tag{4.1}$$

where $t_i = 0$ (or $t_i = 1$), if L^1 passes an even (odd) number of times through the edges from $\mathbb{Q}$, that belongs to the class $\mathbb{C}_i$, $1 \leq i \leq p$.

Theorem 4.1. *The relation (4.1) represents the Hamming metric [25].*

Proof. From the relation (4.1) and the condition (3.2) we have

1) $d(Q^0_{\lambda_i}, Q^0_{\lambda_j}) \geq 0$, and from the equality $d(Q^0_{\lambda_i}, Q^0_{\lambda_j}) = 0$ it follows that $t_1 = t_2 = ... = t_k = 0$, and $Q^0_{\lambda_i} = Q^0_{\lambda_j}$;

2) the equality $d(Q^0_{\lambda_i}, Q^0_{\lambda_j}) = d(Q^0_{\lambda_j}, Q^0_{\lambda_i})$ is obvious;

3) according to (4.1) for three different vertices $Q^0_{\lambda_i}$, $Q^0_{\lambda_j}$, $Q^0_{\lambda_k}$, it is easy to prove the equality

$$d(Q^0_{\lambda_i}, Q^0_{\lambda_j}) = d(Q^0_{\lambda_j}, Q^0_{\lambda_k}) + d(Q^0_{\lambda_k}, Q^0_{\lambda_j}).$$

The last equality proves that (4.1) is the Hamming metric. ∎

Suppose that the tree A^n has classes of parallel edges $\mathbb{C}_1, \mathbb{C}_2, ..., \mathbb{C}_m$ with the fixed length (3.2). Let us also consider space R^m_1 over real numbers set with norm $\|x\| = \sum\limits_{1=i}^{m} |x_i|$. We will fix on the coordinate axes $OY_1, OY_2, ..., OY_p$, from the origin $O \in R^m_1$, segments with lengths $d_1, d_2, ..., d_m$. Let us make unequivocally a parallelepiped $\mathbb{P}^m$ on these segments. The set of all k-dimensional facets of this parallelepiped forms a complex of k-dimensional parallelepipeds, $0 \leq k \leq m$. We will denote this complex by $\mathcal{P}^k = \{\mathbb{P}^k \subset \mathbb{P}^m, 0 \leq k \leq m\}$. For the case $k = 1$ we obtain the complex $\mathcal{P}^1$, that represents union of all 0- and 1-dimensional facets from $\mathbb{P}^m$. This complex $\mathcal{P}^1$ is a subcomplex of the $\mathcal{P}^m$. The complex $\mathcal{P}^1$ represents a connected, metric and undirected graph. This graph will be denoted by $H = (Y; V)$, where Y is the vertices set from $\mathcal{P}^1$, and V – the edges set from $\mathcal{P}^1$.

Theorem 4.2. *For the tree A^n there exists an unequivocal application $\alpha : A^n \to P^m$, that intersects A^n on a subcomplex from P^m, so that $\alpha : G \to P^1$ represents an isometry.*

Proof. The truth of the theorem follows from the construction method of the complex $\mathbb{P}^m$ and relation (4.1).

We will denote by $Y = \{y_1, y_2, ..., y_n\} \subset \alpha(A^n)$ the vertices set $\alpha(X)$ in the space R_1^m. Let us consider the following function:

$$f(y) = \sum_{i=1}^{n} p(y_i) \parallel y - y_i \parallel, \tag{4.1*}$$

where y_i is the image of $\alpha(x_i)$ and has the weight $p(y_i) = p(\alpha(x_i))$, $1 \leq i \leq n$.

We will prove that the point $y_* \in R_1^m$, that minimizes the function (4.1*), is **median** of the graph $H = (Y; V)$.

Note 4.1. The condition 2) from the Definition 3.6. is necessary in calculation of the median of skeleton $\mathbf{sk}(1)K_A^n$, that may not be on $\mathbf{sk}(0)K_A^n$. For example, let us consider the complex K_A^2 with $\mathbb{Q}^2 = \left\{Q_1^2, Q_2^2, Q_3^2\right\}$, $\mathbb{Q}^1 = \{(x_1, x_2), (x_1, x_4), (x_1, x_6), (x_2, x_3), (x_3, x_4), (x_4, x_5), (x_5, x_6), (x_6, x_7), (x_7, x_2)\}$ and $\mathbb{Q}^0 = \{(x_1, x_2, x_3, x_4, x_5, x_6, x_7\}$ (The geometric representation of this complex is given in Figure 2.)

For this complex we consider the edges length 1, and the vertices weight $p(x_1) = p(x_2) = p(x_4) = p(x_6) = 1$, $p(x_3) = p(x_5) = p(x_7) = 100$. According to the formula (1.1) the vertex x_8^* that does not belong to $\mathbb{Q}^0 = \mathbf{sk}(0)K_A^2$ is a median.

Thus we denote by y the vectors from R_1^m. So for the vector $y = (y^1, y^2, ..., y^m) \in R_1^m$ let us form the differences $y^i - y_1^i$, $y^i - y_2^i$, ..., $y^i - y_n^i$, $1 \leq i \leq m$, and the set of indices

$$\begin{aligned} I_i^+ &= j : y^i - y_j^i > 0, \\ I_i^0 &= j : y^i - y_j^i = 0, \\ I_i^- &= j : y^i - y_j^i < 0. \end{aligned} \tag{4.2}$$

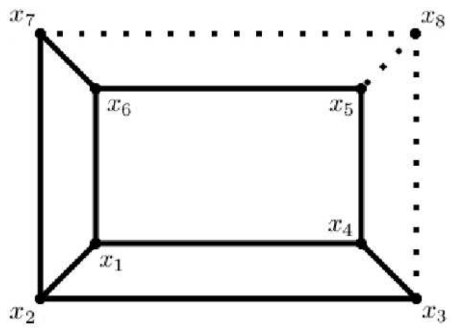


Figure 2. A complex, which does not contain the median.

The relations (4.2) are formed as in [4].

Theorem 4.3. *The vector $y_j \in R_1^m$ minimizes the function (4.1*) if and only if the following relations are satisfied:*

$$\sum_{j \in I_i^+ \cup I_i^0} p(y_j) \geq \sum_{j \in I_i^-} p(y_j),$$

$$\sum_{j \in I_i^+} p(y_j) \geq \sum_{j \in I_i^0 \cup I_i^-} p(y_j). \tag{4.3}$$

The Proof of the theorem is analogous to that of the Theorem 3.1. from [3].

The Theorem 4.3. permits us to state an important fact: *median of metric graphic H does not depend on the edges lengths $d_1, d_2, ..., d_m$ from V*. Thus, the parallelepiped $\mathbb{P}^m$ could be replaced by a unitary cube of the R_1^m, using the operation of the expansion or constraint for each edge with lengths d_i, $1 \leq i \leq m$.

So we obtain mapping

$$\beta : \mathbb{P}^m \to I^m, \tag{4.4}$$

where I^m is a *unitary* cube the vertices of which have the coordinates formed from 0 and 1 of the space R_1^m. As a result of this mapping the graph H passes in a metric graph $\beta(H)$ that we will denote by $\mathcal{G} = (Z; W) \subset I^m$. We have Hamming metric both on the cube I^m and on the graph G. The vertices set Z of this graph is $\beta\alpha(\mathbb{Q}^0)$, and the edges set W represents the edges set $\beta\alpha(\mathbb{Q}^1)$. The classes of parallel edges $\mathbb{C}_1, \mathbb{C}_2, ..., \ \mathbb{C}_m$ are transformed into the following classes of parallel edges from the cube I^m:

$$\mathbb{C}_1^1 = \beta\alpha(\mathbb{C}_1), \mathbb{C}_2^1 = \beta\alpha(\mathbb{C}_2), ..., \beta\alpha(\mathbb{C}_m^1) = \beta\alpha(\mathbb{C}_m).$$

The union of these classes does not necessary cover all the edges from unitary cube I^m. It could be a cover only if $A^n = \mathcal{P}^m$.

For the set of vertices $Z = \{z_1, z_2, ..., z_n\}$ the same weights are kept:

$$p(z_1), \quad p(z_2), ..., p(z_n), \tag{4.5}$$

where $p(z_i) = p(\beta\alpha(x_i), 1 \leq i \leq m$.

Thus we have an isometric mapping:

$$\beta\alpha : G \to \mathcal{G}.$$

Let the vertex $z_i \in Z$ of the cube I^m has the coordinates:

$$z_i = (z_i^1, z_i^2, ..., z_i^m),$$

where $z_i^j = 1$ or $z_i^j = 0$.

Because the median from R_1^m does not depend on metric, let consider that all lengths $d_1, d_2, ..., d_m$ are equal to 1. We will denote A^n by $A^n(1)$. Thereby the mapping $\beta\alpha(A^n(1)) \to I^m$ is also an isometry, where $\beta\alpha(A(1))$ is a subcomplex of the complex, formed from the facets of I^m.

Let us form the matrix as we did in [3]:

$$N = \begin{array}{cc} & \begin{array}{cccccc} C_1^1 & C_2^1 & ... & C_j^1 & ... & C_m^1 \\ \downarrow & \downarrow & & \downarrow & & \downarrow \end{array} \\ & \left(\begin{array}{cccccc} z_1^1 & z_1^2 & ... & z_1^j & ... & \varepsilon_1^m \\ z_2^1 & z_2^2 & ... & z_2^j & ... & z_2^m \\ ... & ... & ... & ... & ... & ... \\ z_i^1 & z_i^2 & ... & z_i^j & ... & z_i^m \\ ... & ... & ... & ... & ... & ... \\ z_n^1 & z_n^2 & ... & z_n^j & ... & z_n^m \end{array} \right) \end{array} \begin{array}{c} \\ \\ \leftarrow z_1 \\ \leftarrow z_2 \\ ... \\ \leftarrow z_i \\ ... \\ \leftarrow z_n \end{array}$$

For every column from the matrix N we calculate a new tuple $r = (r_1, r_2, ..., r_j, ..., r_m)$, considering $r_j = 1$ or $r_j = 0$:

a) if the scalar product $(z_1^j, z_2^j, ..., z_n^j)(p(z_1), p(z_2), ..., p(z_n)) = \sum_{i=1}^n z_i^j p(z_i)$ is a number bigger than $\frac{1}{2}\sum_{i=1}^n p(z_i)$, then $r_j = 1$, and if this product is less than the sum, then $r_j = 0$;

b) if the following equality:

$$\sum_{i=1}^n Z_i^j p(z_i) = \frac{1}{2}\sum_{i=1}^n p(z_i), \tag{4.6}$$

is true, then the value of r_j is chosen arbitrarily: 0 or 1.

Theorem 4.4. *Any vector $r = (r_1, r_2, ..., r_m)$ is a line of the matrix N.*

To prove this theorem some additional issues are necessary.

Let T_i^{n-1} be an $(n-1)$-dimensional transversal of the complex $A^n(1)$. This transversal divides $A^n(1)$ in two different parts $T_{i(1)}^n$ and $T_{i(2)}^n$, with a non-empty intersection T_i^{n-1}. These represent some distinct subcoplexes $A_1^n(j)$, $j = 1, 2$ of the A_1^n. We will call $T_{i(1)}^n$ and $T_{i(2)}^n$ n-dimensional subcomplexes, determined by T_i^{n-1} (let call them *left* and *right* subcomplexes).

Next let us denote by $\mathbb{Q}_{i(j)}^0$, $1 \le i \le B$, $j \in \{1, 2\}$, $\mathbb{Q}_{i(1)}^0$ and $\mathbb{Q}_{i(2)}^0$ the respective sets of vertices of $A^n(1)$, $T_{i(1)}^n$ and $T_{i(2)}^n$. Obviously the following relations are true:

$$\mathbb{Q}^0_{i(1)} \cap \mathbb{Q}^0_{i(2)} = \mathbb{Q}^0_{i(j)},$$
$$\mathbb{Q}^0_{i(1)} \cup \mathbb{Q}^0_{i(2)} = \mathbb{Q}^0, \tag{4.7}$$

If $\mathbb{Q}^0_{i(1)}$ and $\mathbb{Q}^0_{i(2)}$ are the sets of vertices of n-dimensional complexes $T^n_{i(1)}$ and $T^n_{i(2)}$, determined by the transversals $T^n_{i(j)}$, $j \in \{1, 2\}$, $1 \leq i \leq B$, where B is the number of all n-dimensional transversals, then we will denote by $p(\mathbb{Q}^0_{i(1)})$, $p(\mathbb{Q}^0_{i(2)})$ the weight sum of the respective left and right set of vertices, i.e.:

$$p(\mathbb{Q}^0_{i(1)}) = \sum_{x_i \in \mathbb{Q}^0_{i(1)}} p(x_i),$$
$$p(\mathbb{Q}^0_{i(2)}) = \sum_{x_i \in \mathbb{Q}^0_{i(2)}} p(x_i). \tag{4.8}$$

The numbers $p(\mathbb{Q}^0_{i(1)})$ and $p(\mathbb{Q}^0_{i(2)})$ will be called weights of the respective n-dimensional complexes.

Theorem 4.5. *A vertex x_* of the graph $G = (X; U) = sk(1)A^n(1)$ is a median of G if and only if this vertex represents in $A^n(1)$ the intersection of n transversals $T^n_{i(j)}$, $j \in \{1, 2\}$, $1 \leq i \leq B$, of distinct directions pairwise, for which the weight sum of the pair of n-dimensional complexes $A^n_{i_1(1)}$, $A^n_{i_1(2)}$,..., $A^n_{i_n(1)}$, $A^n_{i_n(2)}$, determined by the mentioned and accommodated transversals at the graph G, satisfies the following relations:*

$$\mathrm{p}(\mathrm{Q}^0_{i_1(1)}) = \mathrm{p}(\mathrm{Q}^0_{i_1(2)}),$$
$$p(\mathbb{Q}^0_{i_2(1)}) = p(\mathbb{Q}^0_{i_2(2)}), \tag{4.9}$$

....

$$p(\mathbb{Q}^0_{i_n(1)}) = p(\mathbb{Q}^0_{i_n(2)}).$$

Proof. Necessity. Let x_* be the median vertex of the graph $G = (X;U) \subset A^n(1)$. This vertex is the median of the tree $A^n(1)$ also. We observe first that according to the condition c) of the cubes complex, any its vertex is situated at the intersection of at least n $(n-1)$-dimensional transversals pairwise. As for the Theorem 4.2, it follows unequivocally that through $T^{n-1}_{i_1}, T^{n-1}_{i_2}, ..., T^{n-1}_{i_b}$ there exist some n transversals at the intersection of which there is the median vertex x_*, for which it is necessary to take place relations analogous to (4.3). Let us be more explicit. Let $A^n_{i_1(1)}$, $A^n_{i_1(2)}$, $A^n_{i_2(1)}$, $A^n_{i_2(2)}$,..., $A^n_{i_n(1)}$, $A^n_{i_n(2)}$ be the pairs of subcomplexes of the $A^n(1)$, determined by the mentioned transversals; $Q^0_{i_1(1)}$, $Q^0_{i_1(2)}$; $Q^0_{i_2(1)}$, $Q^0_{i_2(2)}$; $Q^0_{i_n(1)}$, $Q^0_{i_n(2)}$ – the sets of vertices of the respective subcomplexes; and $p(Q^0_{i_1(1)})$, $p(Q^0_{i_1(2)})$; $p(Q^0_{i_2(1)})$, $p(Q^0_{i_2(2)})$;..., $p(Q^0_{i_n(1)})$, $p(Q^0_{i_n(2)})$ – the pairs of sums from the theorem. Some of the mentioned subcomplexes may be even some transversals $T^{n-1}_{i_1}, T^{n-1}_{i_2}, ..., T^{n-1}_{i_n}$. Obviously the inequalities (4.6) determine the vertex x_*, i.e.

$$T^{n-1}_{i_1} \cap T^{n-1}_{i_2} \cap ... \cap T^{n-1}_{i_n} = x_*, \tag{4.10}$$

because the relations (4.9) are the adapted ones to the A^n from (4.3).

Let the relations (4.9) be again verified.

$$T^{n-1}_{i_1} \cap T^{n-1}_{i_2} \cap ... \cap T^{n-1}_{i_n} = \oslash. \tag{4.11}$$

Through a simple syllogism we get to a contradiction that the condition 2 from the Definition 3.1. is not satisfied.

According to the results from [4] it follows that the set of all medians of the graph $\mathcal{G}$= $(Z;W)$ represents a facet-cube of the I^m. So, if we would be interested in those medians of the graph $\mathcal{G}$, that are also the vertices of this graph, then these are the vertices of the respective facets. More than that, *their existence does not depend on the distances (3.2).*

Now let us return to the Theorem 4.4.

Proof. Suppose the opposite. That means that there exists a vertex $r = (r_1, r_2, ..., r_m)$ of the cube I^m, that does not belong to the graph $\beta\alpha(G) = \mathcal{G}$, where β is the mapping

$$\beta : \mathbb{P}^m \to I^m.$$

Let $r = z_* \in I^n \backslash \mathcal{G}$ minimize the function $f(z)$, that is analogous to (4.1). The point z_* has m facets of dimension $m-1$. These facets for the cubic complex of all facets from I^m, represent some $(n-1)$-dimensional transversals, that have the point (vector) z_* as intersection. Each of these facets contains an $(n-1)$-dimensional transversal of the complex $\beta\alpha(A_n)$, that is isometric with $A_n(1)$. Now let us mention that z_* satisfies the pair of relations (4.9), adapted to the complex $\beta\alpha(A_n(1))$. Also, in the complex $\beta\alpha(A_n(1))$ any m transversals $\beta\alpha(T_{i_1}^{n-1})$, $\beta\alpha(T_{i_2}^{n-1})$, ..., $\beta\alpha(T_{i_m}^{n-1})$ of $(n-1)$-dimension has an empty intersection, because the vector z_* does not belong to them. This is in contradiction with the Theorem 4.2.

5 The algorithm of median calculation for an n-dimensional tree

Let $A^n(1)$ be an n-dimensional tree. From those studied above follows that the median of the tree $A^n(1)$ could be determined by the median z_* calculated in the m-dimensional cube I^m for a special graph $\mathcal{G}$. This graph could be obtained as a result of two consecutive mappings α and β. Thus, if z_* is the median of the graph $\mathcal{G}$ in the cube I^m then median x^* of the tree $A^n(1)$ is determined by the relation

$$x^* = \alpha^{-1}\beta^{-1}(z^*).$$

The obtained results let us describe as an efficient algorithm of median calculation in $A^n(1)$ which does not depend on metric.

So the searched algorithm is as follows:

1) Let $X = \{x_1, x_2, ..., x_m\} = Q^0$

2) To find the classes of parallel edges in the tree $A^n(1)$. Suppose we have m classes of parallel edges

$$\mathbb{C}_1, \mathbb{C}_2, ..., \mathbb{C}_m;$$

3) To establish an arbitrary vertex from X, for example x_1, and put in correspondence the tuple $x_1 = (0, 0, ...0)$, formed from m zeros;

4) For any other vertex $x_i \in X$ we choose an arbitrary chain $L^1(x_1, x_i)$, that merge together two vertices x_1 and x_i;

5) The vertex x_i will have in correspondence the tuple

$$x_i = (\varepsilon_i^1, \varepsilon_i^2, ..., \varepsilon_i^m),$$

for $\varepsilon_i^j = 1$, if the chain $L^1(x_1, x_i)$ passes an odd number of times through the edges of the class $\mathbb{C}_j$, and for $\varepsilon_i^j = 0$, if this number is even, $i \in \overline{1, n}$.

From tuples we form a matrix M, the lines of which represent the tuples, proper to the vertices x_i, $1 \leq i \leq n$:

$$M = \begin{array}{c} \begin{array}{cccc} \mathbb{C}_1 & \mathbb{C}_2 & ... & \mathbb{C}_m \\ \downarrow & \downarrow & & \downarrow \end{array} \\ \left(\begin{array}{cccc} \varepsilon_1^1 & \varepsilon_1^2 & ... & \varepsilon_1^m \\ \varepsilon_2^1 & \varepsilon_2^2 & ... & \varepsilon_2^m \\ ... & ... & ... & ... \\ \varepsilon_n^1 & \varepsilon_n^2 & ... & \varepsilon_n^m \end{array} \right) \end{array} .$$

6) To calculate a new tuple $r^* = (r_1^*, r_2^*, ..., r_m^*)$ using the matrix M elements and the vertices weight $p(x_i)$ from X, $1 \leq i \leq n$, according to the rules:

$$r_j = \begin{cases} 1, \text{ if } \sum\limits_{i=1}^{n} \varepsilon_i^j p(x_i) > \frac{1}{2} \sum\limits_{i=1}^{n} p(x_i), \\ 0, \text{ if } \sum\limits_{i=1}^{n} \varepsilon_i^j p(x_i) < \frac{1}{2} \sum\limits_{i=1}^{n} p(x_i), \\ 0 \text{ or } 1 \text{ (unconcerned), if } \sum\limits_{i=1}^{n} \varepsilon_i^j p(x_i) = \frac{1}{2} \sum\limits_{i=1}^{n} p(x_i). \end{cases}$$

7) As for the the Theorem 4.4. the tuple r^* belongs to the matrix M, and the the vertex x^* which corresponds to this tuple is the median A^n.

References

[1] Bujac M. *The Multidimensional Directed Euler Tour of Cubic Manifold*, Buletinul Academiei de Ştiinţe a Republicii Moldova. Seria Matematica, **Nr. 1(50)**, Chişinău, 2006, pp. 15–22.

[2] Soltan P. S., Zambitki D. K. *About one Extremal Problem on Tree,* Mathematical methods to solving economic problems, **Nr. 1,** Moscow, "Nauka", 1969, pp.102–107 (in Russian).

[3] Soltan P. S., Zambitki D. K., Prisacaru K. F. *Extremal problems on graphs and algorithms of their solving*, Kishinev, 1974 (in Russian).

[4] Soltan P. S., Prisacaru K. F. *The Steiner Problem on Graph*, Dokl. Acad. Nauk SSSR, **198(1971), Nr. 1**, p. 46–49 (in Russian).

[5] Boltyanski V. *The homotopic theory of continuous images and vector fields*, (Trudy Matematicheskogo instituta "V.A.Steklov", AN SSSR, XLVII), Moscow, 1955, p.199 (in Russian).

[6] Pontryagin A. S. *Basis of Combinatorial Topology*, Moscow, 1976 (in Russian).

[7] Hilton P. J., Wylie Sh. *Homology Theory: an introduction to Algebraic topology*, Cambridge University Press, 1960.

[8] Cataranciuc S., Soltan P. *Hypergraphs and their homology*, Trends in the Development of the Information and Communication Technology in Education and Management. International Conference, March 20-21, 2003, Chisinau, pp. 294–300 (in Romanian).

[9] Cataranciuc S., Bujac M., Soltan P. *On the Division of Abstract Manifolds in Cubes*, Buletinul Academiei de Ştiinţe a Republicii

Moldova. Seria Matematica, **Nr. 2(51)**, Chişinău, 2006, pp. 29–34.

[10] Cataranciuc S., Scripnic M., Soltan P. *About the median that does not depend on the space metric*, The 30-th Annual Congres of the American Romanian Academy of Arts and Sciences (ARA), Proceedings, Chişinău, July 5-10, 2005, pp. 58–61.

[11] Berge C. *Graphs and hypergraphs*, North-Holland, Amsterdam, 1973.

[12] Berge G. *Hypergraphs: combinatorics of finite sets*, North Holland, Amsterdam, 1989.

[13] Teleman S. *Elements of topology and differential varieties*, Bucharest, 1964 (in Romanian).

[14] *The Mathematical Encyclopedia*, **3**, 1982 (in Russian).

[15] Cataranciuc S., Scripnic M., Soltan P. *A metric space with independent median*, Annals of The Tiberiu Popoviciu Seminar of Functional Equations, Approximation and Convexity, vol. 3, Cluj-Napoca (România), 2005, pp. 3–12.

[16] Soltan P. *On the Homologies of Multy-ary relations and Oriented Hipergraphs*, Studii in metode de analiza numerica si optimizare, vol. 2, nr. 1(3), Chisinau, 2000, pp. 60–81.

[17] Cataranciuc S. *$\mathcal{G}$-Complex of Multu-ary Relations*, Analele stiintifice ale USM, Seria "Stiinte fizico-matematice", Chisinau, 2006, p. 119–122. (in Romanian)

[18] Cataranciuc S., Soltan P. *The generalized complex of multi-ary relation and its homologies*, Annals of The Tiberiu Popoviciu Seminar of functional equations, approximation and convexitz, vol. 6, Cluj-Napoca (România), 2008, pp. 17–44.

[19] Cataranciuc S. *The complex of parallelepipeds and the 3-dimensional tree*, The 33-rd Annual Congres of the American Romanian Academy of Arts and Sciences (ARA), Proceedings, Sibiu, Romania, July 02-07, 2009, **Vol. II**, pp. 293–296.

[20] Erdos P. *On extremal problems on graphs and generalized graphs*, Israel J. Math., **2**, 1964, pp. 183–190.

[21] Teleman S. *Elements of topology and differential varieties*, Bucharest, 1964 (in Romanian).

[22] Cataranciuc S., Soltan P. *Abstract complexes, their homologies and applications*, Buletinul Academiei de Ştiinţe a Republicii Moldova. Seria Matematica, **Nr. 2(63)**, Chişinău, 2010, pp. 31–58.

[23] Bujac M., Cataranciuc S., Soltan P. *On the Division in Cubes of Abstract Manifolds,* Buletinul Academiei de Ştiinţe a Republicii Moldova. Seria Matematica, **Nr. 2(51)**, Chişinău, 2006, pp. 29–34.

[24] *The Mathematical Encyclopedia*, **1**, 1977 (in Russian).

[25] Hamming Richard W. *Error detecting and error correcting codes*, Bell. System Technical Journal. **Nr. 29(2)**, 1950, pp. 147–160.

S. Cataranciuc, P. Soltan
Received April 4, 2011

Moldova State University
E–mail: *s.cataranciuc@gmail.com*,
psoltan@usm.md

Computer Science Journal of Moldova, vol.**19**, no.1(55), 2011

About one algorithm of bidimensional interpolation using splines

Igor Verlan

Abstract

In the paper an explicit algorithm for the problem of two-dimensional spline interpolation on a rectangular grid is proposed.

MSC2000: 41A05, 41A15, 65D05, 65D07.

Keywords: Two-dimensional interpolation, splines, explicit algorithm.

1 Introduction

Let's assume that the mesh $\Delta = \Delta_x \times \Delta_y$ is given on the domain $\Omega = [a, b] \times [c, d]$, where $\Delta_x = a = x_0 < x_1 < \ldots < x_n = b$ and $\Delta_y = c = y_0 < y_1 < \ldots < y_m = d$. Let us suppose that values $f(x_i, y_j) = f_{ij}$, $i = \overline{0, n}$, $j = \overline{0, m}$, are known at the knots of mesh Δ. The interpolant $S(x, y)$ is to be constructed such that $S(x_i, y_j) = f_{ij}$. In order to solve this problem, bilinear splines, which are local ones, but derivatives are not continuous, are widely used. If cubic splines of two variables [1] are used, we get an interpolating surface $S(x, y) \in C^{2,2}(\Omega)$, where the class of functions $f(x, y)$, which are continuous together with their derivatives $\frac{\partial^{k+l} f(x,y)}{\partial^k x \partial^l y}$, $k = \overline{0, 2}$, $l = \overline{0, 2}$, on Ω, is denoted by $C^{2,2}(\Omega)$. But in this case you have to solve at least $n + 2m$ or $m + 2n$ (in the case of periodic end conditions) systems of equations for determining unknown coefficients of the spline. This fact may become critical in the case of large set of data from computational point of view, therefore the problem of elaboration of new algorithms which possess the locality property, still remains actual. Below there is an algorithm for explicit two-dimensional spline interpolation for rectangular grid, which is a generalisation of the algorithm for one-dimensional case presented in [2]. Two-dimensional spline is constructed as a tensor product of one-dimensional splines.

2 Definition of splines

Let us introduce splines as follows: on $\Omega_{ij} = [x_i, x_{i+1}] \times [y_j, y_{j+1}]$

$$S(x, y) = \varphi_i(t) F_{ij} \phi_j(u), \tag{1}$$

where notations $h_i = x_{i+1} - x_i$, $t = (x - x_i)/h_i$, $l_j = y_{j+1} - y_j$ and $u = (y - y_j)/l_j$ are used.

In (1) the matrix F_{ij} is a matrix represented in the following form:

$$F_{ij} = \begin{pmatrix} F_{ij}^{(0,0)}, F_{ij}^{(0,1)}, F_{ij}^{(0,2)} \\ F_{ij}^{(1,0)}, F_{ij}^{(1,1)}, F_{ij}^{(1,2)} \\ F_{ij}^{(2,0)}, F_{ij}^{(2,1)}, F_{ij}^{(2,2)} \end{pmatrix},$$

where the submatrix

$$F_{ij}^{(0,0)} = \begin{pmatrix} f_{ij}, f_{ij+1} \\ f_{i+1j}, f_{i+1j+1} \end{pmatrix},$$

contains given data at the knots of the mesh and submatrices

$$F_{ij}^{(k,l)} = \begin{pmatrix} m_{ij}^{(k,l)}, m_{ij+1}^{(k,l)} \\ m_{i+1j}^{(k,l)}, m_{i+1j+1}^{(k,l)} \end{pmatrix},$$

where $k = \overline{0,2}$, $l = \overline{0,2}$ and $k + l \geq 1$ contain unknown coefficients of the spline $m_{ij}^{(k,l)} = \frac{\partial^{k+l} S}{\partial^k x \partial^l y}(x_i, y_j)$. Vector-functions $\varphi_i(t)$ and $\phi_j(u)$ are defined as follows:

$$\begin{aligned} \varphi_i(t) = \; & (1 - \nu(t), \nu(t), h_i(t^4 - 2t^3 + 2t - \nu(t))/2, \\ & h_i(2t^3 - t^4 - \nu(t))/2, h_i^2(3t^4 - 8t^3 + 6t^2 - \nu(t))/12, \\ & h_i^2(3t^4 - 4t^3 + \nu(t))/12) \end{aligned}$$

and

$$\begin{aligned}\phi_j(u) = \quad & (1-\nu(u), \nu(u), l_j(u^4-2u^3+2u-\nu(u))/2, \\ & l_j(2u^3-u^4-\nu(u))/2, l_j^2(3u^4-8u^3+6u^2-\nu(u))/12, \\ & l_j^2(3u^4-4u^3+\nu(u))/12)^T.\end{aligned}$$

A function ν, called generating function for the spline (1), has to satisfy the following conditions:

$$\begin{gathered}\nu(1)=1,\ \nu(0)=\nu'(0)=\nu'(1)=\nu''(0)=\nu''(1)=0, \\ \nu^{(3)}(0)=\nu^{(3)}(1)=24 \text{ and } \nu\in C^3[0,1]. \qquad (2)\end{gathered}$$

Some examples of generating functions ν (see [2]) are presented below. Conditions (2) are held by functions

$$\nu(t)=t^3(4+15t-48t^2+42t^3-12t^4),$$

$$\nu(t)=-48+120t-84t^2+106t^3-75t^4+30t^5-48t/(2-t)+48(1-t)/(1+t)$$

or

$$\nu(t)=\begin{cases}4t^3+6t^4-12t^5,\ t\in[0,1/2] \\ 1-4(1-t)^3-6(1-t)^4+12(1-t)^5,\ t\in[1/2,1].\end{cases}$$

Taking into account (2) we have that $\varphi(0)=\phi^T(0)=(1,0,0,0,0,0)$ and $\varphi(1)=\phi^T(1)=(0,1,0,0,0,0)$, therefore from (1) it immediately follows that

$$\begin{aligned}S(x_i,y_j)&=\varphi_i(0)F_{ij}\phi_j(0)=f_{ij}, \\ S(x_{i+1},y_j)&=\varphi_i(1)F_{ij}\phi_j(0)=f_{i+1j}, \\ S(x_i,y_{j+1})&=\varphi_i(0)F_{ij}\phi_j(1)=f_{ij+1}, \\ S(x_{i+1},y_{j+1})&=\varphi_i(1)F_{ij}\phi_j(1)=f_{i+1j+1},\end{aligned}$$

i.e. interpolation conditions are fulfilled.

Due to the fact that $\varphi'(0)=(\phi'(0))^T=(0,0,1,0,0,0)$, $\varphi'(1)=(\phi'(1))^T=(0,0,0,1,0,0)$, $\varphi''(0)=(\phi''(0))^T=(0,0,0,0,1,0)$ and $\varphi''(1)=(\phi''(1))^T=(0,0,0,0,0,1)$ it follows that the following equalities

$$\varphi_{i-1}^{(k)}(1)F_{i-1j}=\varphi_i^{(k)}(0)F_{ij}, \quad k=\overline{0,2},$$

$$F_{ij-1}\phi_{j-1}^{(l)}(1) = F_{ij}\phi_j^{(l)}(0), \quad l = \overline{0,2}$$

are valid. As a result it can be concluded that the function $S(x, y)$ and its derivatives $\frac{\partial^{k+l}S}{\partial^k x \partial^l y}(x, y)$, where $k = \overline{0,2}$, $l = \overline{0,2}$, $k + l \geq 1$, are the continuous ones.

3 Computing formulae for unknown coefficients

In this section we'll present computing formulae for unknown coefficients of the spline (1). Components of submatrix $F_{ij}^{(1,0)}$ are computed using formula

$$m_{ij}^{(1,0)} = \alpha_{1i}\delta_{i-2j}^{(1,0)} + \alpha_{2i}\delta_{i-1j}^{(1,0)} + \alpha_{3i}\delta_{ij}^{(1,0)} + \alpha_{4i}\delta_{i+1j}^{(1,0)}, \; i = \overline{2, n-2}, \; j = \overline{0, m},$$

where $\delta_{ij}^{(1,0)} = (f_{i+1j} - f_{ij})/h_i$, $i = \overline{0, n-1}$, $j = \overline{0, m}$ and

$$\alpha_{1i} = -\frac{h_{i-1}h_i(h_i + h_{i+1})}{(h_{i-2} + h_{i-1})(h_{i-2} + h_{i-1} + h_i)(h_{i-2} + h_{i-1} + h_i + h_{i+1})}$$

$$\alpha_{4i} = \frac{(h_{i-2} + h_{i-1})^2(h_{i-2} + h_{i-1} + h_i)\alpha_{1i}}{(h_{i-1} + h_i + h_{i+1})(h_i + h_{i+1})^2}$$

$$\alpha_{3i} = \frac{h_{i-1} + (h_{i-2} + h_{i-1})\alpha_{1i} - (h_{i-1} + 2h_i + h_{i+1})\alpha_{4i}}{h_{i-1} + h_i}$$

$$\alpha_{2i} = 1 - \alpha_{1i} - \alpha_{3i} - \alpha_{4i}.$$

For components of submatrix $F_{ij}^{(0,1)}$ we have formula of the similar form:

$$m_{ij}^{(0,1)} = \beta_{1j}\delta_{ij-2}^{(0,1)} + \beta_{2j}\delta_{ij-1}^{(0,1)} + \beta_{3j}\delta_{ij}^{(0,1)} + \beta_{4j}\delta_{ij+1}^{(0,1)}, \; i = \overline{0, n}, \; j = \overline{2, m-2}$$

where, respectively, $\delta_{ij}^{(0,1)} = (f_{ij+1} - f_{ij})/l_j$, $i = \overline{0, n}$, $j = \overline{0, m-1}$ and

$$\beta_{1j} = -\frac{l_{j-1}l_j(l_j + l_{j+1})}{(l_{j-2} + l_{j-1})(l_{j-2} + l_{j-1} + l_j)(l_{j-2} + l_{j-1} + l_j + l_{j+1})},$$

$$\beta_{4j} = \frac{(l_{j-2} + l_{j-1})^2(l_{j-2} + l_{j-1} + l_j)\beta_{1j}}{(l_{j-1} + l_j + l_{j+1})(l_j + l_{j+1})^2},$$

$$\beta_{3j} = \frac{l_{j-1} + (l_{j-2} + l_{j-1})\beta_{1j} - (l_{j-1} + 2l_j + l_{j+1})\beta_{4j}}{l_{j-1} + l_j},$$

$$\beta_{2j} = 1 - \beta_{1j} - \beta_{3j} - \beta_{4j}.$$

In order to compute $m_{ij}^{(1,1)}$ we can use values of $m_{ij}^{(1,0)}$ or values of $m_{ij}^{(0,1)}$. In the first case we get

$$m_{ij}^{(1,1)} = \beta_{1j}\delta_y m_{ij-2}^{(1,0)} + \beta_{2j}\delta_y m_{ij-1}^{(1,0)} + \beta_{3j}\delta_y m_{ij}^{(1,0)} + \beta_{4j}\delta_y m_{ij+1}^{(1,0)},$$

$$i = \overline{2, n-2}, \quad j = \overline{2, m-2},$$

where $\delta_y m_{ij}^{(1,0)} = (m_{ij+1}^{(1,0)} - m_{ij}^{(1,0)})/l_j$.

In the second case the computational formula has the form

$$m_{ij}^{(1,1)} = \alpha_{1i}\delta_x m_{i-2j}^{(0,1)} + \alpha_{2i}\delta_x m_{i-1j}^{(0,1)} + \alpha_{3i}\delta_x m_{ij}^{(0,1)} + \alpha_{4i}\delta_x m_{i+1j}^{(0,1)},$$

$$i = \overline{2, n-2}, \quad j = \overline{2, m-2},$$

where $\delta_x m_{ij}^{(0,1)} = (m_{i+1j}^{(0,1)} - m_{ij}^{(0,1)})/h_i$.

Let us require the continuity of $S^{(3,0)}(x,y)$ along $x = x_i$. This requirement has the form

$$\varphi_{i-1}^{(3)}(1)F_{i-1j}\phi_j(u) = \varphi_i^{(3)}(0)F_{ij}\phi_j(u)$$

or

$$\varphi_{i-1}^{(3)}(1)F_{i-1j} = \varphi_i^{(3)}(0)F_{ij}. \tag{3}$$

Taking into account that

$$\varphi_{i-1}^{(3)}(1) = (-24/h_{i-1}^3, 24/h_{i-1}^3, -6/h_{i-1}^2, -18/h_{i-1}^2, 0, 6/h_{i-1})),$$

$$\varphi_i^{(3)}(0) = (-24/h_i^3, 24/h_i^3, -18/h_i^2, -6/h_i^2, 6/h_i, 0)$$

from (3) we get

$$\begin{aligned} m_{ij}^{(2,0)} &= 4\left(\frac{\mu_i\delta_{ij}^{(1,0)}}{h_i} - \frac{\lambda_i\delta_{i-1j}^{(1,0)}}{h_{i-1}}\right) + \frac{\lambda_i(m_{i-1j}^{(1,0)} + 3m_{ij}^{(1,0)})}{h_{i-1}} - \\ &\quad - \frac{\mu_i(3m_{ij}^{(1,0)} + m_{i+1j}^{(1,0)})}{h_i}, \; i = \overline{3, n-3}, \; j = \overline{0, m}, \end{aligned}$$

$$
\begin{aligned}
m_{ij}^{(2,1)} &= 4\left(\frac{\mu_i \delta_x m_{ij}^{(0,1)}}{h_i} - \frac{\lambda_i \delta_x m_{i-1j}^{(0,1)}}{h_{i-1}}\right) + \frac{\lambda_i (m_{i-1j}^{(1,1)} + 3m_{ij}^{(1,1)})}{h_{i-1}} - \\
&\quad - \frac{\mu_i (3m_{ij}^{(1,1)} + m_{i+1j}^{(1,1)})}{h_i}, \ i = \overline{3, n-3}, \ j = \overline{2, m-2},
\end{aligned}
$$

$$
\begin{aligned}
m_{ij}^{(2,2)} &= 4\left(\frac{\mu_i \delta_x m_{ij}^{(0,2)}}{h_i} - \frac{\lambda_i \delta_x m_{i-1j}^{(0,2)}}{h_{i-1}}\right) + \frac{\lambda_i (m_{i-1j}^{(1,2)} + 3m_{ij}^{(1,2)})}{h_{i-1}} - \\
&\quad - \frac{\mu_i (3m_{ij}^{(1,2)} + m_{i+1j}^{(1,2)})}{h_i}, \ i = \overline{3, n-3}, \ j = \overline{3, m-3}. \qquad (4)
\end{aligned}
$$

The notations $\lambda_i = h_i/(h_{i-1} + h_i)$, $\mu_i = 1 - \lambda_i$ are used above.

Similarly, from the requirement of continuity of $S^{(0,3)}(x, y)$ along $y = y_j$, which has the form

$$F_{ij-1}\phi_{j-1}^{(3)}(1) = F_{ij}\phi_j^{(3)}(0),$$

where

$$\phi_{j-1}^{(3)}(1) = (-24/l_{j-1}^3, 24/l_{j-1}^3, -6/l_{j-1}^2, -18/l_{j-1}^2, 0, 6/l_{j-1}),$$

$$\phi_j^{(3)}(0) = (-24/l_j^3, 24/l_j^3, -18/l_j^2, -6/l_j^2, 6/l_j, 0)$$

the following formulae are derived:

$$
\begin{aligned}
m_{ij}^{(0,2)} &= 4\left(\frac{\mu'_j \delta_{ij}^{(0,1)}}{l_j} - \frac{\lambda'_j \delta_{ij-1}^{(0,1)}}{l_{j-1}}\right) + \frac{\lambda'_j (m_{ij-1}^{(0,1)} + 3m_{ij}^{(0,1)})}{l_{j-1}} - \\
&\quad - \frac{\mu'_j (3m_{ij}^{(0,1)} + m_{ij+1}^{(0,1)})}{l_j}, \ i = \overline{0, n}, \ j = \overline{3, m-3},
\end{aligned}
$$

$$
\begin{aligned}
m_{ij}^{(1,2)} &= 4\left(\frac{\mu'_j \delta_y m_{ij}^{(1,0)}}{l_j} - \frac{\lambda'_j \delta_y m_{ij-1}^{(1,0)}}{l_{j-1}}\right) + \frac{\lambda'_j (m_{ij-1}^{(1,1)} + 3m_{ij}^{(1,1)})}{l_{j-1}} - \\
&\quad - \frac{\mu'_j (3m_{ij}^{(1,1)} + m_{ij+1}^{(1,1)})}{l_j}, \ i = \overline{2, n-2}, \ j = \overline{3, m-3},
\end{aligned}
$$

$$\begin{aligned} m_{ij}^{(2,2)} &= 4(\frac{\mu'_j\delta_y m_{ij}^{(2,0)}}{l_j} - \frac{\lambda'_j\delta_y m_{ij-1}^{(2,0)}}{l_{j-1}}) + \frac{\lambda'_j(m_{ij-1}^{(2,1)} + 3m_{ij}^{(2,1)})}{l_{j-1}} - \\ &\quad - \frac{\mu'_j(3m_{ij}^{(2,1)} + m_{ij+1}^{(2,1)})}{l_j}, \ i = \overline{3, n-3}, \ j = \overline{3, m-3}, \qquad (5) \end{aligned}$$

where $\lambda'_j = l_j/(l_{j-1} + l_j)$, $\mu'_j = 1 - \lambda'_j$.

In order to compute values of $m_{ij}^{(2,2)}$ formula (4) or formula (5) can be used.

Taking into account values which indices take in the formulae presented above, we can conclude that we have an explicit algorithm for the subdomain $\Omega' = [x_3, x_{n-3}] \times [y_3, y_{m-3}]$.

4 Case of uniform mesh

In the case when the mesh is the uniform one, i.e. $h_i = h, \forall i$ and $l_j = l, \forall j$, the computational formulae are more simple. Taking into account that $\lambda_i = \mu_i = \lambda'_j = \mu'_j = 1/2$ and $\alpha_{1i} = \alpha_{4i} = \beta_{1j} = \beta_{4j} = 1/12$, $\alpha_{2i} = \alpha_{3i} = \beta_{2j} = \beta_{3j} = 5/12$ from the previous section we get

$$m_{ij}^{(1,0)} = (-f_{i-2j} - 4f_{i-1j} + 4f_{i+1j} + f_{i+2j})/12h,$$

$$m_{ij}^{(0,1)} = (-f_{ij-2} - 4f_{ij-1} + 4f_{ij+1} + f_{ij+2})/12l,$$

$$m_{ij}^{(1,1)} = (-m_{i-2j}^{(0,1)} - 4m_{i-1j}^{(0,1)} + 4m_{i+1j}^{(0,1)} + m_{i+2j}^{(0,1)})/12h \text{ or}$$

$$m_{ij}^{(1,1)} = (-m_{ij-2}^{(1,0)} - 4m_{ij-1}^{(1,0)} + 4m_{ij+1}^{(1,0)} + m_{ij+2}^{(1,0)})/12l,$$

$$m_{ij}^{(2,0)} = 2(f_{i-1j} - 2f_{ij} + f_{i+1j})/h^2 + (m_{i-1j}^{(1,0)} - m_{i+1j}^{(1,0)})/2h,$$

$$m_{ij}^{(0,2)} = 2(f_{ij-1} - 2f_{ij} + f_{ij+1})/l^2 + (m_{ij-1}^{(0,1)} - m_{ij+1}^{(0,1)})/2l,$$

$$m_{ij}^{(2,1)} = 2(m_{i-1j}^{(0,1)} - 2m_{ij}^{(0,1)} + m_{i+1j}^{(0,1)})/h^2 + (m_{i-1j}^{(1,1)} - m_{i+1j}^{(1,1)})/2h,$$

$$m_{ij}^{(1,2)} = 2(m_{ij-1}^{(1,0)} - 2m_{ij}^{(1,0)} + m_{ij+1}^{(1,0)})/l^2 + (m_{ij-1}^{(1,1)} - m_{ij+1}^{(1,1)})/2l,$$

$$m_{ij}^{(2,2)} = 2(m_{i-1j}^{(0,2)} - 2m_{ij}^{(0,2)} + m_{i+1j}^{(0,2)})/h^2 + (m_{i-1j}^{(1,2)} - m_{i+1j}^{(1,2)})/2h \text{ or}$$

$$m_{ij}^{(2,2)} = 2(m_{ij-1}^{(2,0)} - 2m_{ij}^{(2,0)} + m_{ij+1}^{(2,0)})/l^2 + (m_{ij-1}^{(2,1)} - m_{ij+1}^{(2,1)})/2l.$$

Obviously, for each of the above formulae the indices take the same values as for the corresponding formulae from the previous section.

5 Conclusions

In this paper we restricted ourselves to the interpolation problem on subdomain Ω'. If you want to construct an interpolation surface on the domain Ω you have to use boundary conditions in order to determine coefficients of the spline which remain unknown.

References

[1] Iu.S. Zavialov, B.I.Kvasov, V.L.Miroshnichenko. *Methods of spline functions.* Moscow, Nauka, 1980 (in Russian).

[2] I.I.Verlan. *An explicit of C^3 interpolation using splines.* Revue D'Analyse Numerique et de Theorie de L'Approximation, tome 23, n.1, 1994, pp.103–115.

I.Verlan

Received May 24, 2011

Department of Applied Mathematics, Moldova State University
60 Mateevich str. Chisinau,
MD2009, Moldova
E–mail: *i_verlan@yahoo.com*

Computer Science Journal of Moldova, vol.**19**, no.1(55), 2011

Optimal Correction of Infeasible Systems in the Second Order Conic Linear Setting*

Maziar Salahi

Abstract

In this paper we consider correcting infeasibility in a second order conic linear inequality by minimal changes in the problem data. Under certain conditions, it is proved that the minimal correction can be done by solving a lower dimensional convex problem. Finally, several examples are presented to show the efficiency of the new approach.

Keywords: Second Order Cone Program, Infeasibility, Interior Point Methods.

1 Introduction

Correcting infeasibility by minimal changes in problem data is a well studied problem and various approaches have been developed to do this task [2, 5, 6]. The aim of this paper is to consider the optimal correction of infeasible linear inequalities in the second order conic setting. Thus let us first introduce second order cone program that has been widely used in modeling many real world problems [1, 4].

Definition 1. *A second order cone in R^n is defined as*

$$Q_n = \{x \in R^n | \ \ ||\bar{x}|| \leq x_1\}, \text{ where } \bar{x} = (x_2, \cdots, x_n)^T.$$

It has the following fundamental properties that enables one to extend interior point algorithms from linear program (LP) to second order cone program (SOCP) [1, 4]:

* This work was supported by the University of Guilan, Rasht, Iran.

- It is convex and closed.
- It is self-dual.
- It is pointed and has nonempty interior.

It is worth to note that $x \in Q_n$ is usually denoted by $x \succeq_{Q_n} 0$. An SOCP in the standard primal form similar to primal LP is given by

$$\begin{array}{ll} \min & c_1^T x_1 + \cdots + c_r^T x_r \\ & A_1 x_1 + \cdots + A_r x_r = b, \\ & x_i \succeq_{Q_{n_i}} 0, \quad i = 1, \cdots, r, \end{array}$$

where $A_i \in R^{m \times n_i}, b \in R^m,\ c \in R^{n_i}$ and its dual is given by

$$\begin{array}{ll} \max & b^T y \\ & A_i^T y + s_i = c_i, \quad i = 1, \cdots, r, \\ & s_i \succeq_{Q_{n_i}} 0,\ i = 1, \cdots, r. \end{array}$$

This dual pair in the compact form is written as

$$\begin{array}{ll} \min & c^T x \\ & Ax = b, \\ & x \succeq_Q 0, \end{array} \tag{1}$$

where $A = [A_1, \cdots, A_r]$, $x = (x_1^T, \cdots, x_r^T)^T$ and $Q = Q_{n_1} \times \cdots \times Q_{n_r}$ and

$$\begin{array}{ll} \max & b^T y \\ & A^T y + s = c, \\ & s \succeq_Q 0, \end{array} \tag{2}$$

where $s = (s_1^T, \cdots, s_r^T)^T$ and $c = (c_1^T, \cdots, c_r^T)^T$. Moreover, dual without slack variable can be written as

$$\begin{array}{ll} \max & b^T y \\ & c - A^T y \succeq_Q 0. \end{array} \tag{3}$$

It is worth to note that weak duality theorem holds for this dual pair analogous to the LP case, but the strong duality theorem requires stronger assumptions as follows:

- **Assumption 1:** A_i's $i = 1, \cdots, r$ are linearly independent.
- **Assumption 2:** Both primal and dual problems are strictly feasible i.e., there exist a primal feasible vector $x_1, \cdots, x_r$ such that $x_i \succ_{Q_{n_i}} 0$ for $i = 1, \cdots, r$ and there exist a dual feasible vector y and $s_1, \cdots, s_r$ such that $s_i \succ_{Q_{n_i}} 0$ for $i = 1, \cdots, r$.

Now under these two assumptions the strong duality theorem holds for SOCP [4].

2 Optimal Correction of an Infeasible Conic Linear Inequality

Suppose we have the following infeasible conic linear inequality

$$Ax - b \succeq_{Q_m} 0, \quad x \in R^n. \tag{4}$$

To correct this infeasible system to a feasible one by minimal changes in the vector b, it is sufficient to solve

$$\begin{aligned} \min_{x,r} \quad & ||r|| \\ & Ax - b - r \succeq_{Q_m} 0. \end{aligned} \tag{5}$$

This is obviously equivalent to

$$\begin{aligned} \max_{x,r,t} \quad & -t \\ & Ax - b - r \succeq_{Q_m} 0, \\ & ||r|| \leq t, \end{aligned}$$

which further can be written in the following dual SOCP form (3):

$$\begin{aligned} & \max \quad -t \\ & \begin{pmatrix} -b \\ 0 \\ 0_{m\times 1} \end{pmatrix} - \begin{pmatrix} 0_{m\times 1} & I_m & -A \\ -1 & 0^T_{m\times 1} & 0^T_{n\times 1} \\ 0_{m\times 1} & -I_m & 0_{m\times n} \end{pmatrix} \begin{pmatrix} t \\ r \\ x \end{pmatrix} \succeq_{Q_m \times Q_{m+1}} 0, \end{aligned} \tag{6}$$

which can be solved efficiently using any interior point based software packages for SOCP, like Mosek or SeDuMi [3, 7].

Now let us see whether it would be possible to have the optimal r value by solving a lower dimensional convex problem as in the nonnegative orthant case. In the following theorem we discuss this question.

Theorem 1. *The optimal r value in (5) is either given by*

$$r = \begin{pmatrix} \frac{(A(1,:)x^*-b(1))-||\bar{A}x^*-\bar{b}||}{2} \\ \frac{\bar{A}x^*-\bar{b}}{2} - \frac{(A(1,:)x^*-b(1))(\bar{A}x^*-\bar{b})}{2||\bar{A}x^*-\bar{b}||} \end{pmatrix},$$

where x^* is the optimal solution of

$$\begin{aligned} \min \quad & \frac{1}{\sqrt{2}}(||\bar{A}x - \bar{b}|| - (A(1,:)x - b(1))), \\ & |A(1,:)x - b(1)| \leq ||\bar{A}x - \bar{b}|| \end{aligned} \tag{7}$$

with $A(1,:)$ and $b(1)$ denoting the first row of A and the first element of b respectively and $\bar{A} = A(2:m,:)$ and $\bar{b} = b(2:m)$, or

$$r = Ax^* - b, \tag{8}$$

where x^* is an optimal solution of

$$\begin{aligned} \min \quad & ||Ax - b|| \\ & -A(1,:)x + b(1) \geq ||\bar{A}x - \bar{b}||. \end{aligned} \tag{9}$$

Proof. Problem (5) can be written as

$$\begin{aligned} \min_x \quad & \min_r ||r|| \\ & Ax - b - r \succeq_{Q_m} 0. \end{aligned}$$

Now let us first consider the inner minimization problem. It is equivalent to

$$\begin{aligned} \min_{t,r} \quad & t \\ & Ax - b - r \succeq_{Q_m} 0 \\ & ||r|| \leq t \end{aligned}$$

or the following dual SOCP, since x is a constant vector for the inner minimization problem:

$$\begin{array}{ll} \max & -t \\ \multicolumn{2}{l}{\begin{pmatrix} Ax-b \\ 0 \\ 0_{m\times 1} \end{pmatrix} - \begin{pmatrix} 0_{m\times 1} & I_m \\ -1 & 0_{1\times m} \\ 0_{m\times 1} & -I_m \end{pmatrix} \begin{pmatrix} t \\ r \end{pmatrix} \succeq_{Q_m\times Q_{m+1}} 0} \end{array} \tag{10}$$

and its corresponding primal problem is given by

$$\begin{array}{ll} \min & (Ax-b)^T y_1 \\ & \begin{pmatrix} 0_{m\times 1}^T & -1 & 0_{m\times 1}^T \\ I_m & 0_{1\times m}^T & -I_m \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} -1 \\ 0_{m\times 1} \end{pmatrix}, \\ & y_1 \in Q_m, \quad (y_2, y_3^T)^T \in Q_{m+1}. \end{array} \tag{11}$$

Now if for the given vector $x \in R^n$, we have $|A(1,:)x-b(1)| < ||\bar{A}x-\bar{b}||$, then the optimal solutions of (10) and (11) are given by

$$r = \begin{pmatrix} \frac{(A(1,:)x-b(1))-||\bar{A}x-\bar{b}||}{2} \\ \frac{\bar{A}x-\bar{b}}{2} - \frac{(A(1,:)x-b(1))(\bar{A}x-\bar{b})}{2||\bar{A}x-\bar{b}||} \end{pmatrix},$$

$$t = ||r||$$

and

$$y_2 = 1, \quad y_1 = y_3 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -\frac{\bar{A}x-\bar{b}}{||\bar{A}x-\bar{b}||} \end{pmatrix},$$

since they are both feasible and having equal objective values. It is easy to check that

$$||r|| = \frac{1}{\sqrt{2}}(||\bar{A}x-\bar{b}|| - (A(1,:)x-b(1))).$$

Thus in this case to have the optimal r value in (5), it is sufficient to solve (7). However, if $-A(1,:)x+b(1)| \geq ||\bar{A}x-\bar{b}||$, then the optimal solution for (10) and (11) are given by

$$r = Ax-b, \qquad t = ||r||$$

and $y_2 = 1, \quad y_1 = y_3 = -\frac{Ax-b}{||Ax-b||}$, since they are both feasible and having equal objective values. Thus in this case to find optimal r value in (5), it is sufficient to solve (9). $\square$

As we see, for the optimal correction of (4), unlike linear inequalities in the nonnegative orthant, we can not necessarily find the optimal r value by solving a lower dimensional convex problem. However under certain conditions it would be possible. These conditions are given in the next corollary.

Corollary 1. *If for all $x \in R^n, A(1,:)x - b(1) > 0$ or $|A(1,:)x - b(1)| \leq ||\bar{A}x - \bar{b}||$, then the optimal r value in (5) is given by*

$$r = \begin{pmatrix} \frac{(A(1,:)x^* - b(1)) - ||\bar{A}x^* - \bar{b}||}{2} \\ \frac{\bar{A}x^* - \bar{b}}{2} - \frac{(A(1,:)x^* - b(1))(\bar{A}x^* - \bar{b})}{2||\bar{A}x^* - \bar{b}||} \end{pmatrix},$$

where x^* is the optimal solution of

$$\min \qquad \frac{1}{\sqrt{2}}(||\bar{A}x - \bar{b}|| - (A(1,:)x - b(1))). \tag{12}$$

One can see that (12) is equivalent to the following dual SOCP:

$$\begin{aligned} &\max \qquad -\frac{1}{\sqrt{2}}z \\ &-b - \begin{pmatrix} -A(1,:) & -1 \\ -\bar{A} & 0_{(m-1)\times 1} \end{pmatrix} \begin{pmatrix} x \\ z \end{pmatrix} \succeq_{Q_m} 0. \end{aligned} \tag{13}$$

Remark 1. *Obviously the dimension of problem (13) is lower than the dimension of (6). Therefore doing the minimal correction by solving (13) should be much faster than (6), as it is verified by our numerical experiments.*

3 Illustrative Examples

In Table 1 we have listed the results of several randomly generated test problems with different dimensions using MATLAB version 7.2. For all problems, matrices are generated randomly and we set their first row equal to zero. Then we consider the vector b with all coordinates equal to one of appropriate dimension. Obviously $Ax - b \succeq_{Q_m} 0$ is infeasible since its first element is negative. To solve (6) and (13), which are exactly in dual form SOCP, we have used SeDuMi version 1.05 [7], which is an interior point methods based software package. SeDuMi's input format can be either (1) or (3), which in our case is (3). For all test problems we report the norm of r and the time taken to find it. As we see, by increasing the dimension of the problems, finding r by using the lower dimensional model (13) is extremely faster than (6).

Table 1. Comparison of problems (6) and (13)

m, n	Problem (6) (time(sec), $\|\|r\|\|$)	Problem (13) (time(sec), $\|\|r\|\|$)
50,30	$(0.2, 4.0697)$	$(0.1, 4.0697)$
100,80	$(0.3, 3.6762)$	$(0.1, 3.6762)$
300,200	$(1.9, 7.0938)$	$(0.5, 7.0938)$
500,300	$(5.9, 10.9243)$	$(1.2, 10.9243)$
1000,700	$(132.1, 12.9153)$	$(10.1, 12.9153)$

4 Conclusions

In this paper, we have considered correcting infeasible systems in second order conic linear setting by minimal changes in the vector b. It is proved that under certain conditions, the minimal correction can be done by solving a lower dimensional convex problem. Numerical examples show that the new approach is extremely faster than the original model, especially on large scale problems.

5 Acknowledgement

The author would like to thank the referee for reading the paper very carefully. The financial support of the University of Guilan is appreciated.

References

[1] F. Alizadeh, D. Goldfarb. *Second order cone programming,* Math. Program., Ser. B **95** 2003, 3–51.

[2] P. Amaral, P. Barahona. *A framework for optimal correction of inconsistent linear constraint,* Constraints, **10(1)** 2005, 67–86.

[3] E.D. Andersen, K.D. Andersen. *The MOSEK interior point optimizer for linear programming: an implementation of the homogeneous algorithm,* In: H. Frenk, K. Roos, T. Terlaky, and S. Zhang, editors, High Performance Optimization, Kluwer Academic Publishers, 2000, 197–232.

[4] A. Ben-Tal, A. Nemirovski. *Lectures on Modern Convex Optimization,* SIAM, Philadelphia, PA, 2001.

[5] Y. Censor, A. Ben-Israel, Y. Xiao, J.M. Galvin. *On linear infeasibility arsing in intensity-modulated radiation teraphy inverse planning.* Linear Algebra and Its Applications, **428** 2008, 1406–1420.

[6] M. Salahi, S. Ketabchi. *Correcting an inconsistent set of linear inequalities by the generalized Newton method.* Optimization Methods and Software, **25(3)** 2010, 457–465.

[7] J.F. Sturm. *Using SeDuMi 1.02, A MATLAB toolbox for optimization over symmetric cones,* Optimization Methods and Software, 11/12, pp. 625–653. F. Potra, C. Roos, and T. Terlaky, editors, Special issue on Interior Point Methods (CD supplement with software), 1999.

Maziar Salahi,

Received November 3, 2010
Revised February 15, 2011

Department of Applied Mathematics,
Faculty of Mathematical Sciences,
University of Guilan, Rasht, Iran.
E–mail: *salahim@guilan.ac.ir*, *salahi.maziar@gmail.com*

Computer Science Journal of Moldova, vol.**19**, no.1(55), 2011

Blind Collective Signature Protocol*

Nikolay A. Moldovyan

Abstract

Using the digital signature (DS) scheme specified by Belarusian DS standard there are designed the collective and blind collective DS protocols. Signature formation is performed simultaneously by all of the assigned signers, therefore the proposed protocols can be used also as protocols for simultaneous signing a contract. The proposed blind collective DS protocol represents a particular implementation of the blind multisignature schemes that is a novel type of the signature schemes. The proposed protocols are the first implementations of the multisignature schemes based on Belarusian signature standard.

Keywords. Digital signature, collective digital signature, discrete logarithm problem, blind signature, blind collective signature.

1 Introduction

The digital signatures (DS) are widely used in practical informatics to solve different problems connected with electronic documents authentication. There is proposed a variety of the DS protocols in the literature [11, 7]. Some type of the DS schemes, called multi-signature protocols, provide computing the single DS shared by several signers [1, 8]. A particular type of the multi-signature protocols, called collective DS, has been recently designed [9]. That variant of the multi-signature protocols is based on using the difficulty of finding large prime roots modulo 1024-bit prime p possessing the structure $p = Nk^z + 1$, where

* The work is supported by Russian Foundation for Basic Research grant # 10-07-90403-Ukr_a

$z \geq 2$, N is an even number, and k is a 160-bit prime. That protocol produces a fixed size collective DS for arbitrary number of signers, however the DS length is sufficiently large, actually, 1184 bits.

Using the general design of the collective DS scheme by [9] and DS algorithm specified by Belarusian DS standard, in this paper there is designed the collective DS protocol based on difficulty of finding discrete logarithm. The proposed protocol produces a 320-bit collective DS. Then the proposed collective DS protocol has been used to design the blind collective DS protocol that represents a new type of the multi-signature schemes. The blind collective signature protocol can be applied, for example, in the electronic voting systems and in the electronic money systems.

2 Collective signature protocol based on difficulty of discrete logarithm

2.1 Belarusian signature standard

Belarusian signature standard STB 1176.2−9 [6] is based on difficulty of finding the discrete logarithm in the finite group, order of which contains large prime factor q. The size of the factor q should be equal to $h \geq 160$ bits. The standard specifies the finite group as follows. Select prime p such that its size is $l \geq 1024$ bits. The group includes all numbers of the set $\{1, 2, \ldots, p-1\}$. The group operation is defined by the following formula:

$$a \circ b = abR^{-1} \bmod p,$$

where a and b are the group elements and $R = 2^{l+2}$. The standard specifies ten security levels corresponding to balanced pairs of the values h and l (see Table 1). The exponentiation operation is denoted as follows:

$$a^{(k)} = a \circ a \circ \ldots \circ a \bmod p \quad (k \text{ times}).$$

In the STB 1176.2−9 signature scheme the public key is computed using the following formula:

$$y = g^{(x)},$$

where g is the q order element of the group and x is the secret key $(1 < x < q)$. The signature generation procedure includes the following steps:

1. Generate a random number k $(1 < k < q)$ and compute $T = g^{(k)}$.
2. Concatenate the value T and message M to be signed: $M' = T\|M$.
3. Using the specified hash function F_H compute the hash value from M': $e = F_H(M') = F_H(T\|M)$, where $\|$ is the concatenation operation.
4. Compute the value $s = (k - xe) \bmod q$.

The pair of numbers (e, s) is the signature to message M. The signature verification is performed as follows:

1. If $1 < s < q$ and $0 < e < q$, then go to step 2. Otherwise the signature is false.
2. Compute value $T^* = g^{(s)} \circ y^{(e)}$.
3. Compute value $e^* = F_H(T^*\|M)$.
4. If $e^* = e$ the signature is valid, otherwise the signature is false.

2.2 Collective signature scheme

Suppose that m users should sign the given message M. The collective DS protocol works as follows:

1. Each of the users generates his individual random value k_i and computes $T_i = g^{(k_i)}$.

Table 1. Ten security levels of the STB 1176.2−9 standard

Security level	h, bits	l bits	Security level	h bits	l bits
1	143	638	6	208	1534
2	154	766	7	222	1790
3	175	1022	8	235	2046
4	182	1118	9	249	2334
5	195	1310	10	257	2462

2. It is computed the common randomization parameter as the product $T = T_1 \circ T_2 \circ \ldots \circ T_m$.

3. Using the common randomization parameter T and the specified hash function F_H it is computed the first element e of the collective DS: $e = F_H(T\|M)$.

4. Each of the users computes his share s_i in the second element of the collective DS

   $$s_i = k_i - x_i e \bmod q, \quad i = 1, 2, .., m.$$

5. The second element s of the collective DS (r, s) is computed as follows $s = \sum_{i=1}^{m} s_i \bmod q$.

Size of the value s is equal to h, since it is computed modulo prime q. The total size of the signature (e, s) is $h + h'$, where h' is the bit size of the specified hash function.

The signature verification is performed exactly as it is described in Section 2.1 except the collective DS verification uses the collective public key computed as follows:

$$y = y_1 \circ y_2 \circ \ldots \circ y_m.$$

The presented collective DS protocol works correctly. Indeed,

$$T^* = y^{(e)} \circ g^{(s)} = y^{(e)} \circ g^{\left(\sum_{i=1}^{m}(k_i - x_i e)\right)} =$$

$$= y^{(e)} \circ g^{\left(\sum_{i=1}^{m} k_i\right)} \circ g^{\left(-e\sum_{i=1}^{m} x_i\right)} = y^{(e)} \circ g^{\left(\sum_{i=1}^{m} k_i\right)} \circ y^{(-e)} =$$

$$= g^{(k_1)} \circ g^{(k_2)} \circ \ldots \circ g^{(k_m)} = T_1 \circ T_2 \circ \ldots \circ T_m = T \Rightarrow$$

$$\Rightarrow E^* = F_H(M, R^*) = F_H(M, R) = E.$$

Since the equality $E^* = E$ holds, the collective signature produced with the protocol satisfies the verification procedure, i.e. the described collective signature protocol is correct.

2.3 Attacks on the collective DS protocol

The participants of the collective DS protocol have significantly more possibilities to attack the protocol than outsiders. They can try to forge a collective signature (the first type of the attacks) and to compute the secret key of one of the signers that shares a collective DS.

The first attack. Suppose it is given a message M and $m-1$ signers attempt to create a collective DS corresponding to m signers owning the collective public key $y = y' \circ y_m$, where $y' = \prod_{i=1}^{m-1} y_i$, i.e. $m-1$ users unite their efforts to generate a pair of numbers (e^*, s^*) such that $T^* = y^{(e^*)} \circ g^{(s^*)}$ and $e^* = F_H(T^* \| M)$. Suppose that they are able to do this, i.e. the collective forger (i.e. the considered $m-1$ signers) is able to calculate a valid signature (e^*, s^*) corresponding to collective public key $y = y_1 \circ y_2 \circ \ldots \circ y_m$. The collective DS satisfies the following relation

$$T^* = y^{(e^*)} \circ g^{(s^*)} = \left(y' y_m\right)^{(e^*)} g^{(s^*)} =$$

$$= y'^{(e^*)} \circ y_m^{(e^*)} \circ g^{(s^*)} = g^{\left(e^* \sum_i^{m-1} x_i\right)} \circ y_m^{(e^*)} \circ g^{(s^*)} =$$

$$= y_m^{(e^*)} \circ g^{\left(s^* + e^* \sum_i^{m-1} x_i\right)} \quad \Rightarrow \quad T^* = y_m^{(e^*)} \circ g^{(s^{**})},$$

where $s^{**} = s^* - E^* \sum_i^{m-1} x_i \bmod q$. The collective forgery have computed the signature (e^*, s^{**}) which is a valid signature (to message M) of the mth signer, since e^* is equal to $F_H(M \| R^*)$ and the pair of numbers (e^*, s^{**}) satisfies the verification procedure of the underlying DS scheme. Thus, any successful attack breaking the collective DS protocol also breaks the underlying DS standard. Since the STB 1176.2−9 standard specifies secure DS scheme the proposed protocol is also secure. Otherwise two or more persons would be able to forge a signature of the STB 1176.2−9 standard.

The second attack. Suppose that $m-1$ signers that share some collective DS (e,s) with the mth signer are attackers trying to calculate the secret key of the mth signer. The attackers know the values T_m and s_m generated by the mth signer. This values satisfy the equation $T_m = y_m^{(e)} g^{(s_m)}$, where the values T_m and e are out of the attackers' control, since the value $T_m = g^{(k_m)}$, where k_m is a random number generated by the mth signer, and e is the output of the hash function algorithm. It is supposed that the standard uses secure hash function, therefore the attackers are not able to select the value T producing some specially chosen value e. This means that, like in the case of underlying DS algorithm, computing the secret key requires solving the discrete logarithm problem, i.e. i) to find $k_m = \log T_m$ and then compute $x_m = e^{-1}(k_m - s_m) \bmod q$ or ii) to compute $x_m = \log y_m$.

3 Blind collective signature protocol based on Belarusian DS standard

3.1 Blind signatures

Blind signature schemes [2] represent a particular type of the cryptographic protocols that are especially interesting for application in the electronic money systems and in the electronic voting systems. For practical applications it is interesting to use the blind signature schemes based on the DS algorithms specified by the DS standards. Belarusian DS standard STB 1176.2−9 suites well to be used as the underlying DS scheme of the blind signature protocols.

The properties of the blind signatures are [11]:

i) the signer can't read the document during process of signature generation;

ii) the signer can't correlate the signed document with the act of signing.

Usually in the DS algorithms the signature is calculated using the hash function from the document to be signed, therefore the first property can be easily provided. It is sufficiently to present the hash function to the signer keeping the document secret. The problem of providing the second property is known as anonymity (or untraceability) problem. To solve this problem there are used specially designed DS algorithms. There are known blind signature schemes based on difficulty of the factorization problem [3] and on difficulty of finding the discrete logarithm [10].

To provide the anonymity of the signature there are used so called *blinding factors*. Prior to submit a hash function value (or message M) for signing, the user U computes the hash function value H and multiplies H (or M) by a random number (blinding factor). Then the user submits the blinded hash function value (or blinded document) for signing. The signer signs the blinded value H (or M) producing the blinded signature that is delivered to user U. The user divides out the blinding factor producing the valid signature to the original hash function value (or directly to the original document).

The blind DS protocol based on Belarusian signature standard can be constructed using the blinding factors y^{τ} and g^{ϵ} applied earlier to construct a blind signature scheme based on Schnorr's DS scheme [10, 12]. The designed protocol works as follows.

The blind signature generation procedure includes the following steps:

1. The signer generates a random number k $(1 < k < q)$, computes $T = g^{(k)}$, and sends the value T to the user U.

2. The user U generates random values τ and ϵ, computes $T' = Ty^{(\tau)}g^{(\epsilon)}$, $e' = F_H(T'\|M)$, where M is document to be signed, and $e = e' - \tau \bmod q$. Then the user sends the value e to the signer.

3. The signer computes the blinded signature $s = (k - xe) \bmod q$ and sends the value e to the user U.

4. The user U computes the signature $s' = s+\epsilon$. The pair of numbers (e', s') is the valid signature to the message M.

Correctness of the described blind signature protocol is proved as follows. Computing the value T^* (see signature verification procedure in subsection 2.1) gives

$$T^* = g^{(s')} \circ y^{(e')} = g^{(k-xe+\epsilon)} \circ y^{(e+\tau)} =$$
$$= g^{(k)} \circ g^{(-xe)} \circ g^{(\epsilon)} \circ y^{(e)} \circ y^{(\tau)} = g^{(k)} \circ y^{(-e)} \circ g^{(\epsilon)} \circ y^{(e)} \circ y^{(\tau)} =$$
$$= g^{(k)} \circ y^{(\tau)} \circ g^{(\epsilon)} = T \circ y^{(\tau)} \circ g^{(\epsilon)} = T' \Rightarrow$$
$$\Rightarrow e^* = F_H(T^* \| M) = F_H(T' \| M) = e'.$$

Thus, the signature (e', s') satisfies the equations of the STB 1176.2−9 standard verification procedure.

3.2 Blind collective signature

Belarusian standard suits also to be used as underlying DS scheme of the blind collective DS scheme. Suppose some user U is intended to get a collective DS (corresponding to message M) of some set of m signers using a blind signature generation procedure. To solve this problem the user can apply the following protocol:

1. Each signer generates a random value $k_i < q$ and computes $T_i = g^{(k_i)}$, and presents the value T_i to each of the signers.

2. It is computed a common randomization parameter R as the product $T = T_1 \circ T_2 \circ \ldots \circ T_m$.

3. The value T is send to the user U.

4. The user U generates random values $\tau < q$ and $\epsilon < q$ and computes the values $T' = Ty^{(\tau)}g^{(\epsilon)}$ and $e' = F_H(T' \| M)$. The value e' is the first element of the collective DS.

5. The user U calculates the value $e = e' - \tau \bmod q$ and presents the value e to the signers.

6. Each signer, using his individual value k_i and his secret key x_i, computes his share in the blind collective DS: $s_i = k_i - x_i e \bmod q$.

7. It is computed the second part s of the blind collective DS: $s = \sum_{i=1}^{m} s_i \bmod q$.

8. The user U computes the second parameter of the collective DS: $s' = s + \epsilon \bmod q$.

The signature verification procedure is exactly the same as described in the case of collective DS based on Belarusian standard (see subsection 2.2). The signature (e', s') is a valid collective DS corresponding to the message M. Indeed, using the collective public key

$$y = y_1 \circ y_2 \circ \ldots \circ y_m = g^{\left(\sum_{i=1}^{m} x_i\right)}$$

we get

$$T^* = y^{(e')} \circ g^{(s')} = y^{(e+\tau)} \circ g^{(s+\epsilon)} = y^{(e)} \circ y^{\tau} \circ g^{(s)} \circ g^{(\epsilon)} =$$

$$= g^{\left(e\sum_{i=1}^{m} x_i\right)} \circ y^{(\tau)} \circ g^{\left(\sum_{i=1}^{m}(k_i - x_i e)\right)} \circ g^{(\epsilon)} = g^{\left(\sum_{i=1}^{m} k_i\right)} \circ y^{(\tau)} g^{(\epsilon)} =$$

$$= T \circ y^{(\tau)} \circ g^{(\epsilon)} = T' \Rightarrow e^* = F_H(T^* \| M) = F_H(T' \| M) = e'.$$

Thus, the protocol yields a valid collective DS (e', s') that is known to the user U and unknown to each of the signers. The protocol provides anonymity of the user in the case when the message M and collective signature (e', s') will be presented to the signers. Anonymity means that the signers are not able to correlate the disclosed signature with only one act of the blind signing, if the signers have participated in two or more procedures of blind signing. Indeed, suppose the signers save in a data base all triples (e, s, T) that are produced while performing the protocol.

Accordingly to the blind collective DS protocol the elements of each triple satisfy the expression:

$$T = y^{(e)} \circ g^{(s)}. \tag{1}$$

The signature (e', s') satisfies the expression:

$$T' = y^{(e')} \circ g^{(s')}. \tag{2}$$

From formulas (1) and (2) we get

$$T' \circ T^{-1} = y^{(e'-e)} \circ g^{(s'-s)} \Rightarrow T' = Ty^{(\tau)} \circ g^{(\epsilon)},$$

where $\tau = e' - e \bmod q$ and $\epsilon = s' - s \bmod q$. Since the values τ and ϵ are generated at random while performing the protocol, each of the triples has equal rights to be associated with the given disclosed signature.

3.3 Application as a protocol for simultaneous signing a contract

Due to the fact, that individual shares of the collective DS formed with the protocols described in subsections 2.2 and 3.2 are valid only in the frame of the given set of m signers, the mentioned protocols can be used to solve efficiently the problem of simultaneous signing a contract. The collective signature protocols solve the problem of signing simultaneously a contract being free of any trusted party. A scenario of practical application of the blind simultaneous signing some electronic messages can be attributed to the electronic money systems in which the electronic banknotes are issued by several banks.

4 Conclusion

Belarusian DS standard is recommended for practical application in information technologies connected with exchange and processing electronic documents accompanied by the usual-type digital signatures. The results of this paper show that the signature generation and signature verification procedures specified by Belarusian DS standard can be additionally used as underlying algorithms in the following protocols:

i) blind signature,

ii) collective signature;

iii) blind collective signature.

Besides, the collective DS protocols can be efficiently used as protocols for signing simultaneously a contract.

It is interesting to study possibility to implement such protocols using other official DS standards. Our preliminary investigation of this problem has shown that Ukrainian and Russian [4] DS standards provide such possibility, however American signature standards DSA and ECDSA [5] do not suite to this purpose. More detailed investigation of the proposed problem represents a subject of independent research.

References

[1] Boldyreva A., *Efficient Threshold Signature, Multisignature and Blind Signature Shemes Based on the Gap-Diffi-Hellman-Group Signature Sheme*, Springer-Verlag Lecture Notes in Computer Science, vol. 2139, pp. 31–46, 2003.

[2] Chaum D., *Blind Signature Systems*, U.S. Patent # 4,759,063. 19 July 1988.

[3] Chaum D., *Security without identification: Transaction systems to make big brother obsolete*, Communications of the AMS, vol. 28, no 10, pp. 1030–1044, 1985.

[4] GOST R 34.10-2001. Russian Federation Standard. Information Technology. Cryptographic data Security. Produce and check procedures of Electronic Digital Signature. Government Committee of the Russia for Standards, 2001 (in Russian).

[5] International Standard ISO/IEC 14888-3:2006(E). Information technology – Security techniques – Digital Signatures with appendix – Part 3: Discrete logarithm based mechanisms.

[6] Kharin Yu.S., Bernik V.I., Matveev G.V., Aguievich S.V. *Mathematic and computer foundations of cryptology*, Novoe znanie, Minsk, 2003. 381 p. (in Russian).

[7] Menezes A.J., Van Oorschot P.C., and Vanstone S.A., *Handbook of Applied Cryptography*, CRC Press, Boca Raton, FL, 1997. 780 p.

[8] Min-Shiang Hwang and Cheng-Chi Lee, *Research Issues and Challenges for Multiple Digital Signatures*, International Journal of Network Security, vol. 1. no 1, pp. 1–7, 2005.

[9] Moldovyan N.A., *Digital Signature Scheme Based on a New Hard Problem*, Computer Science Journal of Moldova, vol. 16, no 2(47), pp. 163–182, 2008.

[10] Pointcheval D. and Stern J., *Security Arguments for Digital Signatures and Blind Signatures*, Journal of Cryptology, vol. 13, p. 361–396, 2000.

[11] Schneier B., *Applied Cryptography*, Second Edition, John Wiley & Sons, Inc. New York, 1996. 758 p.

[12] C.P. Schnorr, *Efficient signature generation by smart cards*, Journal of Cryptology, vol. 4, pp. 161–174, 1991.

Nikolay A. Moldovyan

Received September 2, 2009

St. Petersburg Institute for Informatics and Automation of
Russian Academy of Sciences
14 Liniya, 39, St. Petersburg 199178, Russia
E–mail: *nmold@mail.ru*

Computer Science Journal of Moldova, vol.**19**, no.1(55), 2011

Architecting software concurrency

Dumitru Ciorbă, Victor Beşliu

Abstract

Nowadays, the majority of software systems are inherently concurrent. Anyway, internal and external concurrent activities increase the complexity of systems' behavior. Adequate architecting can significantly decrease implementation errors. This work is motivated by the desire to understand how concurrency can constrain or influence software architecting. As a result, in the paper a methodological architecting framework applied for systems with "concurrency-intensive architecture" is described. This special term is defined to emphasize architectures, in which concurrent interactions are crucial. Also in the paper potential models for each phase of architecting framework are indicated.

Keywords: software architecture, concurrency, concurrency-intensive architecture, architecting framework, concurrency model, formalization, specification, CSP#

1 Key challenges in managing concurrency

1.1 Benefits and costs of concurrency

Software development undoubtedly passes a revolutionary period. Desired performance can be achieved not only by increasing processor frequencies. This outlines a future that is determined by multi-core/multiprocessor architectures and multi-threading programs [1].

The benefits of concurrency reside in the full exploitation of advantages offered by multi-core/multiprocessor architectures, primarily, through the possibility to represent naturally and separately concurrent activities. This leads directly or indirectly to the key characteristics of modern software such as: availability, protection, scalability, performance.

However, concurrency has its costs for organizing inter-thread synchronization and communication. Non-determinism, inherent in concurrent systems, requires taking into account a number of properties that must satisfy programs (mutual exclusion, absence of deadlock, livelock, starvation and race conditions, etc.) [2]. Thus the coordination of activities is complicated, and multithreading related errors are intermittent and difficult to reproduce. Moreover, synchronization and communication worsen effects of code scattering and code tangling [3]; involve reuse difficulties due to conflicts between basic functionalities and synchronization functionalities. These conflicts have been intensively studied and we find in the literature by the name *the inheritance anomaly.*

1.2 Inheritance anomalies

Several researchers [4,5,6] have attempted to classify and formalize the anomalies that make unreasonable or even impossible to reuse the base class by inheritance, since the redefinition involves an excessive number of methods. In [7] authors state that any popular modern programming languages do not exclude yet the occurrence of these anomalies. The authors propose three generic classes for anomalies where the role of inheritance as a form of reuse is greatly diminished: history-sensitive anomaly; partitioning of states; modification of acceptable states.

In [8] it is proposed unified treatment of phenomena in a general designation - *reuse anomalies,* arguing that in a concurrency context, the above mentioned effects can also occur in the case of aggregation and association relations. This fact is easily claimed in other works. An example is the paper [9] which states that adverse phenomena may occur in cases other than inheritance, referring to the composition anomaly.

In order to investigate this phenomenon, which appears in code reuse in the context of concurrency, formalization was used in [5]. The results of formal analysis presented in it have surprised the scientific community. It can be easily emphasized that the following statements stand out from a more general content of the research:

- Inheritance anomalies are common to other paradigms, not only to object-oriented programming: e.g. agent-oriented programming (based on the Actor model [10]);

- If anomaly is present in the implementation, it does not necessarily cause practical problems;

- Inheritance anomaly problem in one form or another still cannot be resolved, but rather may be reduced adverse effects induced by anomalies.

Making the concurrency explicit and isolating it into a concurrent component, seems to be reasonable solution [11].

1.3 Concurrency "isolation" techniques

Elimination of adverse consequences (by defining separate and explicit orthogonal functionalities) can be accomplished in many ways at all levels of abstraction of software development (Figure 1): mix-in classes [12] and aspects [3,13,14] can be used in order to compose functionalities at implementation phases; design patterns allow us localization and integration of orthogonal functionalities in analysis and design stages [15,16]; and programming frameworks can help us improving components reuse [17,18,19].

The technological possibility of locating the synchronization code is not the solution by itself. It happens because the problem of inheritance anomalies cannot be fully solved [5]. Moreover in [20] the authors state that *aspectization* (localization) of concurrency in some cases may even be dangerous. Thus we have to diminish bad effects of anomalies with appropriate methods of concurrency management in the early stages of software development.

1.4 Concurrency management

Concurrency management has to provide a proactive strategy that will allow concurrency organization and management throughout the software life cycle. A successful strategy will be determined by the following characteristics:

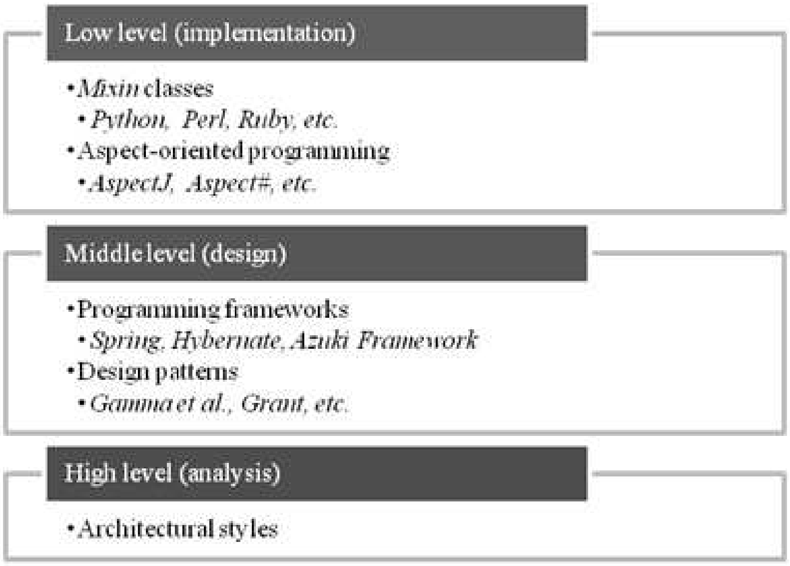


Figure 1. Abstraction levels of software development

- Process development centered: the strategy will require integration with existing processes and applications;
- Centrally managed: all development activities must comply with adopted policies;
- Heterogeneous: the strategy must take into account different organizational forms of concurrency.

One of the early activities of the strategy should include the classification of developed systems in accordance of the degree of concurrency. Some information systems have no concurrency whilst others become more concurrent, in order to maximize efficiency determined by a number of factors (e.g. cheap multiprocessor). In this context, there can be defined a special term: *concurrency-intensive architecture*. It is necessary for emphasizing architecture, where concurrency influences essentially the architecting. Concurrency-intensive architecture thus needs a special and distinct architecting approach.

2 Architecting and concurrency

Architecting is a process where the outcome is stakeholder's satisfaction towards architectural requirements. In the context of software architecting, *abstraction* is one of the main principles. It captures through *encapsulation* the essential properties of a system.

There are numerous definitions of architecture. An interesting idea is mentioned in [21] according to which "*a system may be composed of many levels of abstraction and many phases of operation, each with its own software architecture*". Anyway the entire concept, according to some researchers, is ambiguous in relation to information systems research and practice [22]. Even though, most of them define architecture analogically: as an abstract concept, which provides a certain perspective on information system [23], it was still necessary to standardize the architecting process. Finally, IEEE in 2000 [24], and ISO in 2007 [25] have standardized this conception: "*the activities of the creation, analysis and sustainment of architectures of software-intensive systems, and the recording of such architectures in terms of architectural descriptions*". The main result of this architecting standardization is a clear and comprehensive documentation of the architecture representations of information systems in various views.

This approach can be traced from the earliest major research directions and frameworks for software development. According to [26], architecture can be described in five main views: *logical*, *process*, *physical*, *development* and *scenarios*; the last of them is essentially redundant, but it represents the interaction of the other four (Figure 2).

In the architectural model presented above, planned active processing entities, communication structure, integrity, and other architecturally significant concerns of the control flow management, synchronization and concurrency are described in the *process view.*

It is important to note that the view, according to definitions from [24,25,26], is not yet the localization of implemented functionalities;

Figure 2. The "4+1" View Model of Software Architecture [26]

this is the representation of the entire system from some viewpoint of interrelated requirements (aspects).

Now the following question arises: is it possible to separate synchronization and concurrency concerns from the functional ones and localize them in a separate *structural unit*? The answer is simple: yes, it is possible and necessary [27,28]; but in order to integrate separate functionalities, developer has to either use non-traditional aspect-oriented programming [29], or create applications with loosely coupled architecture.

In loosely coupled architectures the separation, localization and composition of functionalities can be achieved by applying adequate design patterns [30], asynchronous messaging architectures [31], and/or event-driven architectures [32].

These days, most information systems are inherently concurrent, since they handle activities that can happen simultaneously in the world external to the system's world [33]. It is likely then, that concurrency is considered as a critical property of systems and that it must be considered in the early development stages – architecting stages.

3 Concurrency-intensive software architecting

According to IBM *Rational Unified Process*^{TM}, architecture is defined during the inception and elaboration phases [34]. This popular software development process is architecture centric. It means that the system's architecture is a primary artifact for system's development [35]. Thus the importance of an accurate architecting must be sustained by a distinct process.

In this context *The Visual Architecting Process*^{TM} (VAP) can be mentioned. It is promoted by Bredemeyer Consulting. According to it, the architecture specification phase consists of iterative five sub-phases: *Meta-Architecture*, *Conceptual Architecture*, *Logical Architecture*, *Execution Architecture* and *Architecture Guidelines* [36]. Concurrency issues are tightly related with *Execution Architecture*, where analysis focuses on *Process* and *Deployment* views.

Execution Architecture phase is the forth one. This allows us to confirm that concurrency, as a critical property of modern systems, is considered too late. This can be argued by means of structuralism [37], according to which "*a structure may be defined as a network of relationships between elements or elementary process... A structure thus manifests itself by means of relationships; a system manifests itself by means of communications of the relevant elements. A function within a system may be seen as a communicative relationship.*" Thus communication is necessary to "transport" data; but it is the means of communication by which *control information* is being transported as well. This particular communication form is well known as synchronization, which constraints event ordering and controls processing unit interference. It is usual then, that the concurrency influences systems via the communication style, hence concurrency must be an important factor for structural and functional analysis of architecture.

There are numerous architecting methodologies where concurrency is one of the key factors. Here it's case to mention *Nick Rozanski* and *Eoin Woods' work* [38], where concurrency concerns are described from the point of view of *Concurrency Viewpoint.*

A viewpoint is a way of looking at the system, and does not capture architecting focused on concurrency concerns. In this context a generic framework is proposed below. It will permit us to analyze concurrency-intensive architecture from the perspective of evolution (Figure 3).

The architecting process, represented in Figure 3, is a waterfall

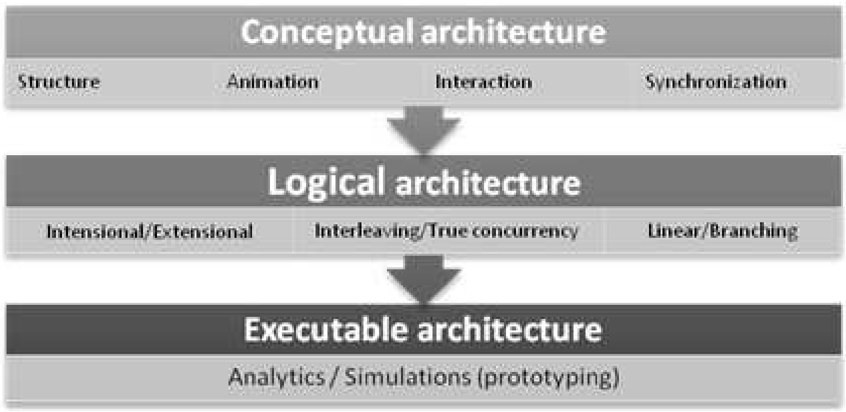


Figure 3. Concurrency-intensive software architecting

process with three phases. Each phase defines an architecture model, which may also include a number of views, where each view is related to a particular domain of the phase.

Conceptual architecture defines entities, their relationships and conceptual constraints. The structure view can show configurations in terms of components, which are units of runtime computation or data-storage, and connectors, which in their turn are the interaction mechanisms between components [39].

In order to facilitate structuring, architectural patterns can be used. A coherent set of related patterns makes up a pattern language. An interesting pattern language is presented in [40].

The last three views have been inspired by a survey of concurrency issues presented in [6]. The survey is organized of taxonomy of the features of concurrent object-oriented languages. In spite of the generalization of the described models, they allow us to use them in our architecting process as views.

Animation view shows the relationship between objects and active entities (process/thread/task). The treatment of threads and objects as independent or dependent concepts, defines two alternatives of activity organization: unrelated and related models (Figure 4).

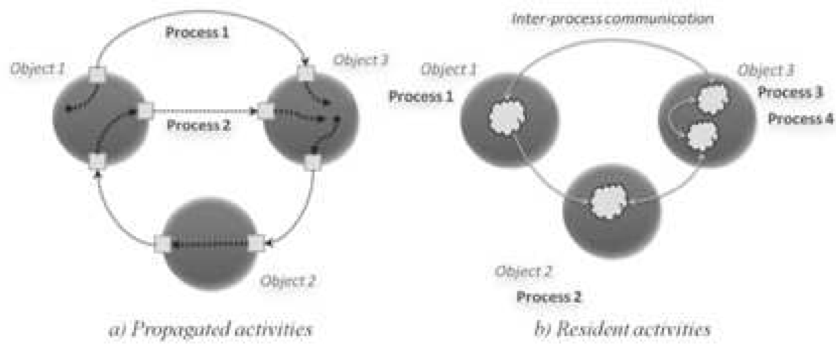


Figure 4. Unrelated (a) and related (b) models of Animation view

The interaction view depicts interactions between objects initiated by the client's invocation, which may be either synchronous or asynchronous. Semantics of returns is defined in this view as well.

Concepts, related to Synchronization view, specify concurrent invocation management. It is important to mention that *Conceptual architecture* phase defines rules to synchronize, select and accept operations, and these rules define control constraints used at the next Logical architecture phase.

Logical architecture conforms to the principles and rules of the conceptual architecture. This phase involves a variety of structures, which have the nature of mathematical formalisms. Logical architecture thus is represented by a generalized formal structure that determines *logic of specification,* which helps describing and reasoning about behavior of concurrent architecture.

Numerous formal models have been studied over the past 20-25 years. Formal semantics, provided by these models, can be classified by partitioning criteria of the following dichotomies [41,42]:

- *Intensional* and *Extensional* semantics,
- *Interleaving* and *True concurrency* semantics,

- *Branching time* and *Linear time* semantics.

Specifying systems as “machines”, determined by states (and possible state changes), obtains *intensional* models. *Extensional* (*also known as* behavior) models focus on occurrence patterns of actions over time. Concurrency is an implicit property in “true concurrency” models. Yet interleaving models reduce it to nondeterministic interleaving representations. Last dichotomy splits models into nondeterministic branching models and linear time setting models. In the former case, models describe concurrency in terms of the sets of their possible (partial) executions.

Formal relationships between models have been analyzed by many researchers. Here should be mentioned the work [43], where translation between models have been studied in terms of category theory. Eight models, more precisely model classes, have been obtained by varying, in all possible ways, the aforementioned criteria (Figure 5).

Model	Semantics		
	Intensional (I) Extensional (E)	Interleaving (Int) True concurrency(TC)	Branching time(BT) Linear time (LT)
Hoare languages (HL)	E	Int	LT
Synchronization trees (ST)	E	Int	BT
Deterministic labelled event structures (dLES)	E	TC	LT
Labelled event structures (LES)	E	TC	BT
Deterministic transition systems (dTS)	I	Int	LT
Transition systems (TS)	I	Int	BT
Deterministic transition systems with independence (dTSI)	I	TC	LT
Transition systems with independence (dTSI)	I	TC	BT

Figure 5. Concurrency models

Positioning of models from Figure 5 in a three-dimensional space relative to dichotomies is represented in Figure 6.

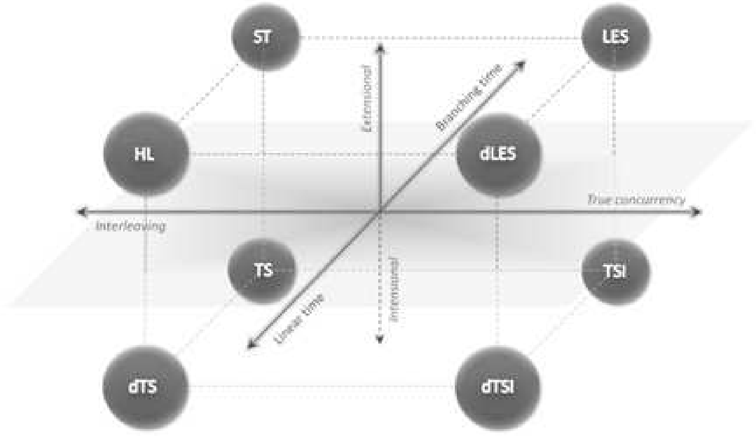


Figure 6. Semantic models for concurrency specification

Executable architecture is a result “product” of the last architecting phase. In the first instance, this term expresses the description of the system’s architecture in a formal notation, the semantic of which is being determined by the logic architecture phase. In the second instance, after using automatic or semi-automatic generation tools, this term may signify a partial implementation of the system – a prototype, which must validate all architecturally significant requirements. A closely related term is the term of *evolutionary prototype*, which is not a work product, but it is stable enough to be considered as a first approximation of a system. According to the article [44], “*producing an evolutionary prototype means that you design, implement, and test a skeleton structure, or architecture, of the system...*”

4 Conclusion

Systems become more concurrent and a new term is needed to define an architecture influenced essentially by concurrency. In this work a new term *concurrency-intensive architecture* is proposed. Also it is being shown that concurrency, as a key factor, can determine a generic architecting framework by providing an architectural prototype. It perfectly fits in the modern architecture-centric development methodologies, such as Rational Unified Process. However concurrency generates difficulties. Still the right methods and tools can decrease the architecting effort. Mature theories and models of concurrency exist; thus the key target of researches is: developing of methods and tools of specification and verification. So an immediate and important objective is to develop and integrate a graphical language of concurrency specification in one of the popular development environments. Language must expressively specify concurrency, thus must use denotational semantic. Such specification language based on events will be presented in the future article. Also, in this article it will be shown how models are verified with operational semantic of CSP# language.

References

[1] S. Herb. *The Free Lunch Is Over. A Fundamental Turn Toward Concurrency in Software.* Dr. Dobb's Journal. [Online] March 2005, Vol. 30, 3. [Cited: April 08 2011] http://www.gotw.ca/publications/concurrency-ddj.htm.

[2] S. Owicki, L. Lamport. *Proving Liveness Properties of Concurrent Programs.* ACM Transactions on Programming Languages and Systems. July 1982, Vol. 4, 3, pp. 455–495.

[3] G. Kiczales, et al. *Aspect-Oriented Programming.* Proceedings of ECOOP'97. s.l.: Springer-Verlag, 1997.

[4] S. Matsuoka, A. Yonezawa. *Analysis of Inheritance Anomaly in Object-Oriented Concurrent Programming Languages.* Research Directions in Concurrent Object-Oriented Programming. s.l.: MIT Press, 1993.

[5] L. Crnogorac, A.S. Rao, K. Ramamohanarao. *Classifying Inheritance Mechanisms in Concurrent Object-Oriented Programming.* Proceedings of ECOOP'98. s.l.: Springer-Verlag, 1998. LNCS 1445.

[6] D.G. Kafura, G. Lavender. *Concurrent Object-Oriented Languages and the Inheritance.* [ed.] T. L. Cassavant. Parallel Computers: Theory and Practice. s.l.: IEEE Press, 1994. pp. 165–198.

[7] G. Milicia, Vl. Sassone. *The inheritance anomaly: ten years after.* Proceedings of Symposium on Applied Computing (SAC). Nicosia, Cyprus : ACM, 2004.

[8] D.M. Suciu. *Techniques for Implementing Concurrency in Object-Oriented Analysis and Design.* Ph. D. thesis. [http://cs.ubbcluj.ro/∼tzutzu/Research/Teza.pdf]. Cluj-Napoca: Babeş-Boleay University, 2001.

[9] L. Bergmans, M. Aksit. *Composing Software from Multiple Concerns: A Model and Composition Anomalies.* Proceedings of ICSE 2000 (2nd) Workshop on Multidimensional Separation of Concerns. 2000.

[10] C. Hewitt. *Actor Model of Computation.* arXiv.org. [Online] Noiembrie 08, 2010. [Cited: Noiembrie 26, 2010.] http://arxiv.org/abs/1008.1459v8. arXiv:1008.1459v8.

[11] B. Goetz. *Concurrency: Past and Present.* InfoQ. [Online] August 26, 2006. [Cited: April 08, 2011.] http://www.infoq.com/presentations/goetz-concurrency-past-present.

[12] G. Bracha, W. Cook. *Mixin-based inheritance.* Proceedings of OOPSLA/ECOOP '90. s.l.: ACM Press, 1990.

[13] V. Pavlov. *Aspect-oriented programming.* IT-Archiv. [Online] June 2003. [Cited: November 21, 2010.] http://www.javable.com/columns/aop/workshop/01/index.pdf.

[14] R. Pawlak, L. Seinturier, Ph. Retaillé. *Foundations of AOP for J2EE development.* [http://books.google.com] s.l.: Apress, 2005. ISBN 1-59059-507-6.

[15] E. Gamma, et al. *Design Patterns: Elements of Reusable Object-Oriented Software*. s.l.: Addison-Wesley, 1994. ISBN 0-201-63361-2.

[16] D. Lea. *Concurrent programming in Java: design principles and patterns*. [http://books.google.com] s.l.: Addison-Wesley, 2000. ISBN 0-201-31009-0.

[17] M. E. Fayad, D. C. Schmidt. *Object-Oriented Application Frameworks*. Communications of the ACM. 1997, Vol. 40, 10.

[18] R.E. Johnson. *Frameworks = (components + patterns)*. Communications of the ACM. 1997, Vol. 40, 10.

[19] D.C. Schmidt, M. E. Fayad. *Building Reusable OO Frameworks for Distributed Software*. Communications of the ACM. 1997, Vol. 40, 10.

[20] J. Kienzle, R. Guerraoui. *AOP: Does it Make Sense? The Case of Concurrency and Failures*. École Polytechnique Fédérale De Lausanne (EPFL). [Online] 2002. [Cited: April 08, 2011.] http://ic2.epfl.ch/publications/documents/IC_TECH_REPORT_200216.pdf.

[21] R. T. Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD dissertation. Irvine: University of California, 2000.

[22] M. S. Corneliussen. *IT Architecturing – Reconceptualizing Current Notions of Architecture in IS Research*. European Conference on Information Systems. Mart 04, 2008.

[23] D. Garlan. *Software Architecture: a Roadmap*. [ed.] A. Finkekstein. The Future of Software Engineering. s.l.: ACM Press, 2000.

[24] M. W. Maier, D. Emery, R. Hilliard. *Software Architecture: Introducing IEEE Standard 1471*. Computer. April 2001, Vol. 34, 4, pp. 107–109.

[25] International Organization for Standardization. *Recommended Practice for Architectural Description of Software-Intensive Systems.* [Online] 2007. [Cited: Decembre 2, 2010.] http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=45991. ISO/IEC 42010:2007.

[26] Ph. Kruchten. *Architectural Blueprints — The "4+1" View Model of Software Architecture.* IEEE Software. 1995. Vol. 12, 6.

[27] W. Hürsch, Cr. Lopes. *Separation of Concerns.* College of Computer Science, Northeastern University. Boston, USA: s.n., 1995. Technical report.

[28] C. A. Constantinides, T. Elrad. *On the Requirements for Concurrent Software Architectures to Support Advanced Separation of Concerns.* OOPSLA'2000, Workshop on Advanced Separation of Concerns in Object-Oriented Systems. 2000.

[29] G. Kiczales, et al. *Aspect-Oriented Programming.* Proceedings of ECOOP'97. s.l.: Springer-Verlag, 1997.

[30] D. Ciorba, et al. *Arch-pattern based design and aspect-oriented implementation of Readers-Writers concurrent problem.* The Computer Science Journal of Moldova. 2007, Vol. 15, 3, pp. 338–353.

[31] G. Hohpe. *Integration Patterns Overview. Enterprise intergration pstterns.* [Online] 2010. [Cited: Dec 06, 2010.] http://www.eaipatterns.com/eaipatterns.html.

[32] M. Fowler. *Event Collaboration.* Development of Further Patterns of Enterprise Application Architecture. [Online] Jun 19, 2006. [Cited: Dec 05, 2010.] http://martinfowler.com/eaaDev/EventCollaboration.html.

[33] J. Bacon, T. Harris. *Operating systems: concurrent and distributed software design.* s.l.: Pearson Education, 2003. ISBN 0-321-11789-1.

[34] IBM Rational Software. *Rational Unified Process. Best Practices for Software Development Teams.* developerWorks®:

IBM's resource for developers and IT professionals. [Online] Jul 23, 2005. [Cited: May 11, 2011.] http://www.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251_bestpractices_TP026B.pdf.

[35] H. Yim, et al. *Architecture-Centric Object-Oriented Design Method for Multi-Agent Systems.* Fourth International Conference on Multi-Agent Systems (ICMAS'00). Los Alamitos, CA, USA: IEEE Computer Society, 2000. ISBN: 0-7695-0625-9.

[36] R. Malan, D. Bredemeyer. *The Visual Architecting Process.* [http://www.bredemeyer.com/papers.htm] s.l.: Bredemeyer Consulting, January 2005.

[37] J.M. Broekman, *Structuralism.* Synthese library. Monographs on epistemology, logic, methodology, philosophy of science, sociology of science and of knowledge. s.l.: Springer, 1974. Vol. 67. ISBN 9-027-70478-3.

[38] N. Rozanski, E. Woods. *Software Systems Architecture: Working with Stakeholders Using Viewpoints and Perspectives.* s.l.: Addison-Wesley Professional, 2005. ISBN 0-321-11229-6.

[39] P. Clements. *Documenting software architectures: views and beyond.* s.l.: Addison-Wesley, 2003. ISBN 0-201-70372-6.

[40] P. Avgeriou, U. Zdun. *Architectural Patterns Revisited – A Pattern Language.* In Proceedings of 10th European Conference on Pattern Languages of Programs (EuroPlop 2005). Irsee, Germany: s.n., July 2005. pp. 1–39.

[41] R. Cleaveland, S. A. Smolka. *Strategic Directions in Computing Research Concurrency Working Group Report.* Stony Brook University – Department of Computer Science. [Online] September 30, 1996. [Cited: January 31, 2011.] http://www.cs.sunysb.edu/~sas/sdcr/report/final/final.html.

[42] T. T. Hildebrandt. *Categorical Models for Concurrency: Independence, Fairness and Dataflow.* [http://www.brics.dk/DS/00/1/BRICS-DS-00-1.pdf] Aarhus, Denmark: University of Aarhus, February 2000. BRICS Dissertation Series. ISSN 1396-7002.

[43] Vl. Sassone, M. Nielsen, G. Winskel. *Models for Concurrency: Towards a Classification.* Theoretical Computer Science. 1996. Vol. 170, 1-2, pp. 297–348.

[44] P. Krol. *Transitioning from waterfall to iterative development.* developerWorks®: IBM's resource for developers and IT professionals. [Online] April 16, 2004. [Cited: December 18, 2010.] http://www.ibm.com/developerworks/rational/library/4243.html.

Dumitru Ciorbă, Victor Beşliu
Received May 28, 2011

Technical University of Moldova
Automation and Information Technology Department
Str. Studenţilor, 7/3, corp 3, 504 Chişinău, MD-2068
Phone: (+373 22) 509908
E–mail: *victor.besliu@ati.utm.md*, *besliu@mail.utm.md*, *vbesliu@yahoo.com*
E–mail: *diciorba@yahoo.com*, *dumitru.ciorba@ati.utm.md*

Preface

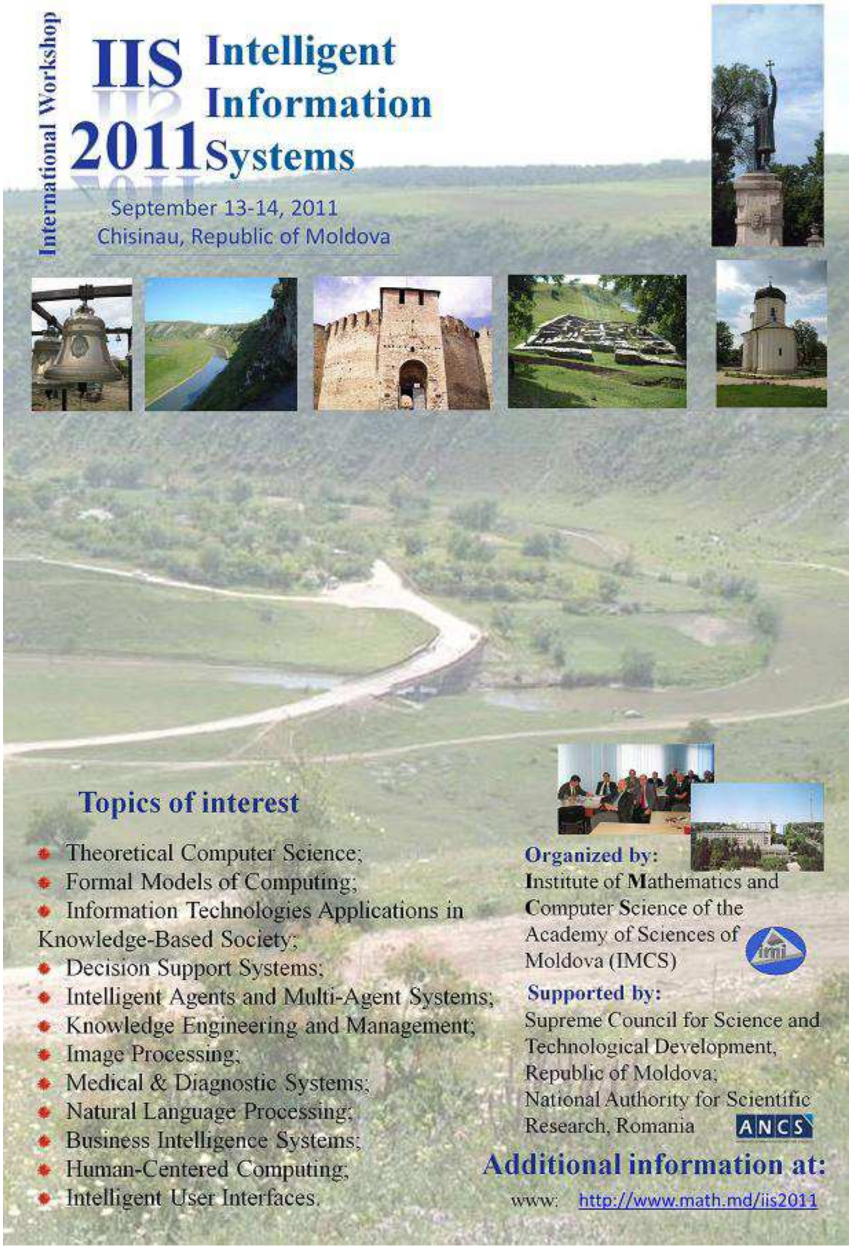
Between September 13 and 14, 2011 at the Institute of Mathematics and Computer Science (IMCS) of Academy of Sciences of Moldova (ASM) the first edition of the International Workshop on Intelligent Information Systems (IIS-2011) took place. It was organized by the IMCS of the ASM with support of the Supreme Council for Science and Technological Development (Republic of Moldova), National Authority for Scientific Research (Romania) and Union of Scientific and Technical Societies of Moldova (Republic of Moldova).

A special session of the workshop was dedicated to celebrating the 70th anniversary of Corresponding member Constantin Gaindric.

Objective of the workshop was meeting of experts in the field and people taking their first steps in research, to make a review of current trends and applications related to the main directions of development of intelligent systems: knowledge processing, natural language processing, decision making, formal models of computation, etc.

Figure 1. Plenary session

Experience exchange took place, new ideas were promoted and various forms of collaboration were established.


International Workshop
IIS 2011
Intelligent Information Systems
September 13-14, 2011
Chisinau, Republic of Moldova
Topics of interest
Theoretical Computer Science;
Formal Models of Computing;
Information Technologies Applications in Knowledge-Based Society;
Decision Support Systems;
Intelligent Agents and Multi-Agent Systems;
Knowledge Engineering and Management;
Image Processing;
Medical & Diagnostic Systems;
Natural Language Processing;
Business Intelligence Systems;
Human-Centered Computing;
Intelligent User Interfaces.
Organized by:
Institute of Mathematics and Computer Science of the Academy of Sciences of Moldova (IMCS)
Supported by:
Supreme Council for Science and Technological Development, Republic of Moldova;
National Authority for Scientific Research, Romania
ANCS
Additional information at:
www: http://www.math.md/iis2011

The following scientists attended plenary session of the workshop and made communications which have aroused interest and provoked interesting debates: Academician Florin Gheorghe Filip (Romanian Academy) with ,,Designing and Building Modern Information Systems; A Series of Decisions to be Made"; Professor Antoni Wiliński (Faculty of Computer Science and Information Technology, West Pomeranian University of Technology of Szczecin, Poland) with ,,Prediction Models of Financial Markets Based on Multiregression Algorithms"; Professor Dan Cristea (University "A.I.Cuza", Iasi, Romania) with ,,Romanian Linguistic Resourses On Very Large Scale"; Corresponding member Horia-Nicolai Teodorescu (Romanian Academy) with ,,Modelling Behavior in Social Networks under Disagreement, in Various Logistics". Also, the value of communication made by Professor Gennaro J. Maffia (Department of Chemical Engineering, Manhattan College, USA) "Analysis and Modeling of Vapor Distillation Using ASPEN, HYSYS Recompressive", was not diminished by thousands of miles away, where he had been, as it was successfully done online.

The workshop offered researchers the opportunity to meet with colleagues to share and discuss news and trends of intelligent systems development, with special emphasis on their applicability.

In the framework of the workshop 6 scientific sessions worked: Information systems and technologies, Theory of computing, Image processing, Image processing in medicine, Language technologies, Students' session. There were published and presented as reports 48 papers of authors from Republic of Moldova, Romania, Ukraine, Russia, Poland, Finland and the USA. More than 98 scientists attended the workshop.

A special session for young researchers was organized. Current and future problems of information society development were discussed, as well as applying new information technologies, natural language processing, decision support systems and medical diagnostic systems.

The participants appreciated the high level of organization of the event and specially mentioned that presented scientific results are valuable, particularly those of young researchers, expressing hope that this forum will have continuity, becoming a traditional meeting.

This volume of Computer Science Journal of Moldova contains the

Figure 2. Invited speakers: H.-N. Teodorescu, A. Wiliński, F.G. Filip, D. Cristea

Figure 3. Session for young researchers

peer-reviewed articles that were presented at International Workshop on Intelligent Information Systems – 2011, published in Proceedings IIS-2011 and selected and revised for publication in this edition of CSJM.

S.Cojocaru, I.Ţiţchiev