Contents

Methodological papers

ANNALES MATHEMATICAE ET INFORMATICAE 38. (2011)

ANNALES MATHEMATICAE ET INFORMATICAE

TOMUS 38. (2011)

HUNGARIA, EGER

ANNALES MATHEMATICAE ET INFORMATICAE

International journal for mathematics and computer science

Referred by
Zentralblatt für Mathematik
and
Mathematical Reviews

The journal of the Institute of Mathematics and Informatics of Eszterházy Károly College is open for scientific publications in mathematics and computer science, where the field of number theory, group theory, constructive and computer aided geometry as well as theoretical and practical aspects of programming languages receive particular emphasis. Methodological papers are also welcome. Papers submitted to the journal should be written in English. Only new and unpublished material can be accepted.

Authors are kindly asked to write the final form of their manuscript in LaTeX. If you have any problems or questions, please write an e-mail to the managing editor Miklós Hoffmann: hofi@ektf.hu

The volumes are available at http://ami.ektf.hu

ANNALES MATHEMATICAE ET INFORMATICAE

VOLUME 38. (2011)

INSTITUTE OF MATHEMATICS AND INFORMATICS
ESZTERHÁZY KÁROLY COLLEGE
HUNGARY, EGER

HU ISSN 1787-5021 (Print)
HU ISSN 1787-6117 (Online)

A kiadásért felelős az
Eszterházy Károly Főiskola rektora
Megjelent az EKF Líceum Kiadó gondozásában
Kiadóvezető: Kis-Tóth Lajos
Felelős szerkesztő: Zimányi Árpád
Műszaki szerkesztő: Tómács Tibor
Megjelent: 2011. december   Példányszám: 30

Készítette az
Eszterházy Károly Főiskola nyomdája
Felelős vezető: Kérészy László

Annales Mathematicae et Informaticae
38 *(2011) pp. 3–13*
http://ami.ektf.hu

On general strong laws of large numbers for fields of random variables

Cheikhna Hamallah Ndiaye[a], Gane Samb LO[b]

[a]LERSTAD, Université de Saint-Louis
LMA - Laboratoire de Mathématiques Appliquées
Université Cheikh Anta Diop BP 5005 Dakar-Fann Sénégal
e-mail: chndiaye@ufrsat.org

[b]Laboratoire de Statistiques Théoriques et Appliquées (LSTA)
Université Pierre et Marie Curie (UPMC) France
LERSTAD, Université de Saint-Louis
e-mail: ganesamblo@ufrsat.org

Submitted September 6, 2011 Accepted November 25, 2011

Abstract

A general method to prove strong laws of large numbers for random fields is given. It is based on the Hájek-Rényi type method presented in Noszály and Tómács [14] and in Tómács and Líbor [16]. Noszály and Tómács [14] obtained a Hájek-Rényi type maximal inequality for random fields using moments inequalities. Recently, Tómács and Líbor [16] obtained a Hájek-Rényi type maximal inequality for random sequences based on probabilities, but not for random fields. In this paper we present a Hájek-Rényi type maximal inequality for random fields, using probabilities, which is an extension of the main results of Noszály and Tómács [14] by replacing moments by probabilities and a generalization of the main results of Tómács and Líbor [16] for random sequences to random fields. We apply our results to establishing a logarithmically weighted sums without moment assumptions and under general dependence conditions for random fields.

Keywords: Strong laws of large numbers, Maximal inequality, Probability inequalities, Random fields.

MSC: Primary 60F15 60G60. Secondary 62H11 62H35.

1. Introduction and notations

We are concerned in this paper with strong laws of large numbers (SLLN) for random fields. Although this type of problems is entirely settled for sequence of independent real random variables (see for instance [9]) and general strong laws of large numbers for dependent real random variables based on Hájek-Rényi types inequalities. But as for random fields, they are still open. As a reminder, we recall that a family of random elements $(X_n)_{n\in T}$ is said to be a random field if the set is endowed with a partial order $(\leq)$, not necessarily complete. For example, and it is the case in this paper, T may be $\mathbf{N}^d$, where $d > 1$ is an integer and $\mathbf{N}$ is the set of nonnegative integers. For such a real random field $(X_n)_{n\in\mathbf{N}^d}$, we intend to contribute to assessing the more general SLLN's, that is finding general conditions under which there exists a real number μ and a family of normalizing positive numbers $(b_n)_{n\in\mathbf{N}^d}$, named here as a d-sequence, such that, for $S_{(0,\dots,0)} = 0$, and $S_n = \sum_{m\leq n} X_m$ for $n > \mathbf{0}$, one has

$$S_n/b_n \to \mu, \quad \text{a.s.}$$

In the case of random fields, the data may be heavily dependent and then Hájek-Rényi type maximal inequalities are needed to obtain strong laws of large numbers, like in the real case. It seems that providing such inequalities goes back to Móricz [11] and Klesov [8]. Based on such inequalities, many authors established strong laws of large numbers such as Nguyen et al. [13], Tómács [19], Lagodowski [10], Noszály and Tómács [14], Móricz [12], Klesov [8], Fazekas et al. [5], Fazekas [2], [4] and the literature cited herein.

One of the motivations of finding general strong laws of large numbers comes from that the finding, as proved by Cairoli [1], that classical maximal probability inequalities for random sequences are not valid in general for random fields. Besides, nonparametric estimation for random fields or spatial processes was given increasing and simulated attention over the last few years as a consequence of growing demands from applied research areas (see for instance Guyon [6]). This results in the serious motivation to extend the Hájek-Rényi type maximal inequality for probabilities for random sequences, what the cited above authors tackled.

Our objective is to give a nontrivial generalization of some fundamental results of these authors that will lead to positive answers to classical and non solved SLLN's. Before a more precise formulation of our problem, we need a few additional notation.

From now on d is a fixed positive integer. The elements of $\mathbf{N}^d$ will be written in font bold like $\mathbf{n}$ while their coordinate are written in the usual way like $\mathbf{n} = (n_1, \dots, n_d)$. $\mathbf{N}^d$ is endowed with the usual partial ordering, that is $\mathbf{n} = (n_1, \dots, n_d) \leq \mathbf{m} = (m_1, \dots, n_d)$ if and only if or each $1 \leq i \leq d$, one has $n_i \leq m_i$. Further $\mathbf{m} < \mathbf{n}$ means $\mathbf{m} \leq \mathbf{n}$ and $\mathbf{n} \neq \mathbf{m}$. We specially denote $(1, \dots, 1) \equiv \mathbf{1}$ and $(0, \dots, 0) \equiv \mathbf{0}$. All the limits, unless specification, are meant as $\mathbf{n} = (n_1, \dots, n_d) \to \infty$, that is equivalent to say that $n_i \to \infty$ for each $1 \leq i \leq d$. To finish, any family of real numbers $(b_\mathbf{n})_{\mathbf{n}\in A}$ indexed by a subset $\mathbf{N}^d$ is called a

d-sequence. We intensively use product type d-sequences. A d-sequence $(b_{\mathbf{n}})_{\mathbf{n}\in A}$ is of product type if it may be written in the form

$$b_{\mathbf{n}} = \prod_{1\le i\le d} b^{(i)}_{n_i}.$$

This product type d-sequence is unbounded and nondecreasing if and only if each sequence $b^{(i)}_{n_i}$ is unbounded and nondecreasing in n_i. Now with these minimum notation, we are able to state the results of Tómács, Líbor and their co-authors.

On one hand, it is known that the Hájek-Rényi type maximal inequality (see [3]) is an important tool for proving SLLN's for sequences. It is natural that Noszály and Tómács [14] used a generalization of this result for random fields in order to get SLLN's for such objects. They stated

Proposition 1.1. *Let r be a positive real number, $a_{\mathbf{n}}$ be a nonnegative d-sequence. Suppose that $b_{\mathbf{n}}$ is a positive, nondecreasing d-sequence of product type. Then*

$$\mathbb{E}(\max_{\ell\le\mathbf{n}}|S_\ell|^r) \le \sum_{\ell\le\mathbf{n}} a_\ell \quad \forall\mathbf{n}\in\mathbf{N}^d$$

implies

$$\mathbb{E}\left(\max_{\ell\le\mathbf{n}}|S_\ell|^r b_\ell^{-r}\right) \le 4^d \sum_{\ell\le\mathbf{n}} a_\ell b_\ell^{-r} \quad \forall\mathbf{n}\in\mathbf{N}^d.$$

From this, they were led to the following general SLLN for random fields.

Theorem 1.2. *Let $a_{\mathbf{n}}, b_{\mathbf{n}}$ be non-negative d-sequences and let $r>0$. Suppose that $b_{\mathbf{n}}$ is a positive, nondecreasing, unbounded d-sequence of product type. Let us assume that*

$$\sum_{\mathbf{n}} \frac{a_{\mathbf{n}}}{b^r_{\mathbf{n}}} < \infty$$

and

$$\mathbb{E}\left(\max_{\mathbf{m}\le\mathbf{n}} |S_{\mathbf{m}}|^r\right) \le \sum_{\mathbf{m}\le\mathbf{n}} a_{\mathbf{m}} \quad \forall\mathbf{n}\in\mathbf{N}^d.$$

Then

$$\lim_{\mathbf{n}\to\infty} \frac{S_{\mathbf{n}}}{b_{\mathbf{n}}} = 0 \qquad a.s.$$

On an other hand, Tómács and Líbor [16], introduced a Hájek-Rényi inequality for probabilities and, subsequently, strong laws of large numbers for random sequences but not for random fields. They obtained first:

Theorem 1.3. *Let r be a positive real number, a_n be a sequence of nonnegative real numbers. Then the following two statements are equivalent.*
(i) There exists $C>0$ such that for any $n\in\mathbf{N}$ and any $\varepsilon>0$

$$\mathbb{P}(\max_{\ell\le n}|S_\ell| \ge \varepsilon) \le C\varepsilon^{-r}\sum_{\ell\le n} a_\ell.$$

(ii) There exists $C > 0$ such that for any nondecreasing sequence $(b_n)_{n\in \boldsymbol{N}}$ of positive real numbers, for any $n \in \mathbf{N}$ and any $\varepsilon > 0$

$$\mathbb{P}\left(\max_{\ell\leq n}|S_\ell|b_\ell^{-1} \geq \varepsilon\right) \leq C\varepsilon^{-r}\sum_{\ell\leq n} a_\ell b_\ell^{-r}.$$

And next, they derived from it this SLLN.

Theorem 1.4. *Let a_n and b_n are non-negative sequences of real numbers and let $r > 0$. Suppose that b_n is a positive non-decreasing, unbounded sequence of positive real numbers. Let us assume that*

$$\sum_n \frac{a_n}{b_n^r} < \infty$$

and there exists $C > 0$ such that for any $n \in \mathbf{N}$ and any $\varepsilon > 0$

$$\mathbb{P}\left(\max_{m\leq n}|S_m| \geq \varepsilon\right) \leq C\ \varepsilon^{-r}\sum_{m\leq n} a_m.$$

Then

$$\lim_{n\to\infty}\frac{S_n}{b_n} = 0 \quad a.s.$$

As said previously, this paper aims at generalizing the previous results in the following way. First, we give a random fields version for Tómács and Líbor [16] as a first generalization in Proposition 2.1. Next we show that our version of Hájek-Rényi type maximal inequality for probabilities for random fields is a generalization of that of Noszály and Tómács [14] and leads to a more general SLLN.

We apply our method for logarithmically weighted sums without any moment assumption and under general dependence conditions for random fields. This shows that the generalization is not trivial.

The paper is organized as follows. Section 2 is devoted to our main results, a Hájek-Rényi type maximal inequality for probabilities for random fields and automatically a strong law of large numbers are given. Section 3 includes their proofs. Section 4 including applications and illustration of our results, concludes the paper.

2. Results

We first give a Hájek-Rényi type maximal inequality for probabilities for random fields, as an extension of Proposition 1 in Noszály and Tómács [14] and of Theorem 2.1 in Tómács and Líbor [16].

Proposition 2.1. *Let r be a positive real number, $a_{\mathbf{n}}$ be a nonnegative d-sequence. Suppose that $b_{\mathbf{n}}$ is a positive, nondecreasing d-sequence of product type. Then the*

following two statements are equivalent.
(i) There exists $C > 0$ *such that for any* $\mathbf{n} \in \mathbf{N}^d$ *and any* $\varepsilon > 0$

$$\mathbb{P}(\max_{\ell \leq \mathbf{n}} |S_\ell| \geq \varepsilon) \leq C\varepsilon^{-r} \sum_{\ell \leq \mathbf{n}} a_\ell.$$

(ii) There exists $C > 0$ *such for any* $\mathbf{n} \in \mathbf{N}^d$ *and any* $\varepsilon > 0$

$$\mathbb{P}\left(\max_{\ell \leq \mathbf{n}} |S_\ell| b_\ell^{-1} \geq \varepsilon\right) \leq 4^d C\ \varepsilon^{-r} \sum_{\ell \leq \mathbf{n}} a_\ell b_\ell^{-r}.$$

We derive from this proposition a general strong law of large numbers for random fields which includes extensions of Theorem 3 in Noszály and Tómács [14] and of Theorem 2.4 in Tómács and Líbor [16]. But we need this lemma first.

Lemma 2.2 (Lemma 2 in Noszály and Tómács [14])**.** *Let* $a_\mathbf{n}$ *be a nonnegative* d*-sequence and let* $b_\mathbf{n}$ *be a positive, nondecreasing, unbounded* d*-sequence of product type. Suppose that* $\sum_\mathbf{n} \frac{a_\mathbf{n}}{b_\mathbf{n}^r} < \infty$ *with a fixed real* $r > 0$*. Then there exists a positive, nondecreasing, unbounded* d*-sequence* $\beta_\mathbf{n}$ *of product type for which*

$$\lim_{\mathbf{n}} \frac{\beta_\mathbf{n}}{b_\mathbf{n}} = 0 \quad \textit{and} \quad \sum_\mathbf{n} \frac{a_\mathbf{n}}{\beta_\mathbf{n}^r} < \infty.$$

Here is our general strong law of large numbers.

Theorem 2.3. *Let* $a_\mathbf{n}$ *be a non-negative* d*-sequence and let* $r > 0$*. Suppose that* $b_\mathbf{n}$ *is a positive, non-decreasing, unbounded* d*-sequence of product type. If*

$$\sum_\mathbf{n} \frac{a_\mathbf{n}}{b_\mathbf{n}^r} < \infty$$

and there exists $C > 0$ *such that for any* $\mathbf{n} \in \mathbf{N}^d$ *and any* $\varepsilon > 0$

$$\mathbb{P}\left(\max_{\mathbf{m} \leq \mathbf{n}} |S_\mathbf{m}| \geq \varepsilon\right) \leq C\ \varepsilon^{-r} \sum_{\mathbf{m} \leq \mathbf{n}} a_\mathbf{m}$$

then

$$\lim_{\mathbf{n} \to \infty} \frac{S_\mathbf{n}}{b_\mathbf{n}} = 0 \quad \textit{a.s.}$$

3. Proofs of the main results

We will need Lemma 2.2 and these two following lemmas.

Lemma 3.1. *Let $\{Y_{\mathbf{k}}, \mathbf{k} \in \mathbf{N}^d\}$ be a field of random variables defined on a fixed probability space $(\Omega, \mathcal{F}, \mathbb{P})$. Then for all $x \in \mathbf{R}$,*

$$\mathbb{P}\left(\sup_{\mathbf{k}} Y_{\mathbf{k}} > x\right) = \lim_{\mathbf{n}\to\infty} \mathbb{P}\left(\max_{\mathbf{k}\leq\mathbf{n}} Y_{\mathbf{k}} > x\right).$$

Proof. It is easy to see that, for all $x \in \mathbf{R}$.

$$\left(\sup_{\mathbf{k}} Y_{\mathbf{k}} > x\right) = \bigcup_{\mathbf{n}=\mathbf{1}}^{\infty} \left(\max_{\mathbf{k}\leq\mathbf{n}} Y_{\mathbf{k}} > x\right).$$

Hence, by the monotone convergence theorem for probabilities, we get the statement. □

Lemma 3.2. *Let $\{Y_{\mathbf{k}}, \mathbf{k} \in \mathbf{N}^d\}$ be a field of random variables defined on a fixed probability space $(\Omega, \mathcal{F}, \mathbb{P})$ and $\{\varepsilon_{\mathbf{n}}, \mathbf{n} \in \mathbf{N}^d\}$ a nondecreasing field of real numbers. If*

$$\lim_{\mathbf{n}\to\infty} \mathbb{P}\left(\sup_{\mathbf{k}} Y_{\mathbf{k}} > \varepsilon_{\mathbf{n}}\right) = 0,$$

then $\sup_{\mathbf{k}} Y_{\mathbf{k}} < \infty$ *a.s.*

Proof. By using the monotone convergence theorem for probabilities, we have

$$\mathbb{P}\left(\bigcap_{\mathbf{n}=\mathbf{1}}^{\infty} \left(\sup_{\mathbf{k}} Y_{\mathbf{k}} > \varepsilon_{\mathbf{n}}\right)\right) = \lim_{\mathbf{n}\to\infty} \mathbb{P}\left(\sup_{\mathbf{k}} Y_{\mathbf{k}} > \varepsilon_{\mathbf{n}}\right) = 0$$

which is equivalent to $\mathbb{P}\left(\bigcup_{\mathbf{n}=\mathbf{1}}^{\infty} \left(\sup_{\mathbf{k}}\ Y_{\mathbf{k}} \leq \varepsilon_{\mathbf{n}}\right)\right) = 1$. This implies that there exists $\mathbf{n}_\omega \in \mathbf{N}^d$ for almost every $\omega \in \Omega$ such that $\sup_{\mathbf{k}}\ Y_{\mathbf{k}}(\omega) \leq \varepsilon_{\mathbf{n}_\omega} < \infty$. □

We need more notation for the proofs. In $\mathbf{N}^d$ the maximum is defined coordinate-wise (actually we shall use it only for rectangles). If $\mathbf{n} = (n_1, \dots, n_d) \in \mathbf{N}^d$, then $\langle \mathbf{n} \rangle = \prod_{i=1}^d n_i$. A numerical sequence $a_{\mathbf{n}}, \mathbf{n} \in \mathbf{N}^d$ is called d-sequence. If $a_{\mathbf{n}}$ is a d-sequence then its difference sequence, i.e. the d-sequence $b_{\mathbf{n}}$ for which $\sum_{\mathbf{m}\leq\mathbf{n}} b_{\mathbf{m}} = a_{\mathbf{n}}, \mathbf{n} \in \mathbf{N}^d$, will be denoted by $\Delta a_{\mathbf{n}}$ (i.e. $\Delta a_{\mathbf{n}} = b_{\mathbf{n}}$). We shall say that a d-sequence $a_{\mathbf{n}}$ is of product type if $a_{\mathbf{n}} = \prod_{i=1}^d a_{n_i}^{(i)}$, where $a_{n_i}^{(i)}$ $(n_i = 0, 1, 2, \dots)$ is a (single) sequence for each $i = 1, \dots, d$. Our consideration will be confined to normalizing constants of product type: $b_{\mathbf{n}}$ will always denote $b_{\mathbf{n}} = \prod_{i=1}^d b_{n_i}^{(i)}$, where $b_{n_i}^{(i)}$ $(n_i = 0, 1, 2, \dots)$ is a nondecreasing sequence of positive numbers for each $i = 1, \dots, d$. In this case we shall say that $b_{\mathbf{n}}$ is a positive nondecreasing d-sequence of product type. Moreover, if for each $i = 1, \dots, d$ the sequence $b_{n_i}^{(i)}$ is unbounded, then $b_{\mathbf{n}}$ is called positive, nondecreasing, unbounded d-sequence of product type. As usual, $\log^+(x) := \max\{1, \log(x)\}$, $x > 0$ and $|\log \mathbf{n}| := \prod_{m=1}^d \log^+ n_m$.

Proof of Proposition 2.1. It is clear that (ii) implies (i) by taking $b_{m_j} = 1$ for all $\mathbf{m} \in \mathbf{N}^d$ and $1 \leq j \leq d$. Now we turn to (i) $\Longrightarrow$ (ii). We can assume without loss of generality that $b_{0,j} = 1$ for $1 \leq j \leq d$. If not, we would replace $b_{\mathbf{m}}$ by $\prod_{j=1}^{d} b_{m,j}/b_{0,j}$, $\mathbf{m} \in \mathbf{N}^d$ and (ii) would remain true with a new constant equal to $Cb_0^{-r} = C(\prod_{j=1}^{d} b_{\mathbf{0},j})^{-r}$. Now consider a fixed $\mathbf{n} \in \mathbf{N}^d$ and an arbitrary a real number $c > 1$. Remark by the monotonicity of $(b_{\mathbf{m}})$ that $b_{\mathbf{m}_j} \geq 1$ for all $\mathbf{m} \in N^d$ and that the sequence $(c^p)_{p\geq 0}$ forms a partition of $[1, +\infty[$. This implies that for any $\mathbf{m} \in \mathbf{N}^d$, for any $1 \leq j \leq d$, there exists a nonnegative integer i_j such that $c^{i_j} \leq b_{m_j} < c^{i_j+1}$. Thus for $\mathbf{i} = (i_1, \ldots, i_d)$, we have that $\mathbf{m} \in \mathcal{A}_{\mathbf{i}} = \{\mathbf{s} \in \mathbf{N}^d$ and $c^{i_j} \leq b_{\mathbf{s}_j} < c^{i_j+1}, j = 1, \ldots, d\}$. Since this holds for all $\mathbf{m} \in \mathbf{N}^d$, we get

$$\mathbf{N}^d = \bigcup_{i \in \mathbf{N}^d} \mathcal{A}_{\mathbf{i}}.$$

Let us restrict ourselves to $\mathbf{m} \leq \mathbf{n}$, and let us define

$$\mathcal{A}_{\mathbf{i},\mathbf{n}} = \{\mathbf{s} \in \mathbf{N}^d, \mathbf{s} \leq \mathbf{n} \quad \text{and} \quad c^{i_j} \leq b_{\mathbf{s}_j} < c^{i_j+1}, j = 1, \ldots, d\}.$$

Since $c^p \to \infty$ as $p \to \infty$ and for $\mathbf{m} \in \mathcal{A}_{\mathbf{i},\mathbf{n}}$, for $1 \leq j \leq d$, $b_{\mathbf{s}_j} \leq b_{\mathbf{n}_j} \leq \max\{b_{n_k}, 1 \leq k \leq d\} < \infty$, the sets $\mathcal{A}_{\mathbf{i},\mathbf{n}}$ are empty for large values of i. Then put $\mathbf{k}_{\mathbf{n}} = \max\{\mathbf{i} : \mathcal{A}_{\mathbf{i},\mathbf{n}} \neq \emptyset\} < +\infty$ and we have

$$[0, n] = \bigcup_{\mathbf{i} \leq \mathbf{k}_{\mathbf{n}}} \mathcal{A}_{\mathbf{i},\mathbf{n}}.$$

It is also noticeable that if $\mathbf{m} \leq \mathbf{s} \in \mathcal{A}_{\mathbf{i},\mathbf{n}}$, then necessarily $\mathbf{m}$ is in some $\mathcal{A}_{\mathbf{i}',\mathbf{n}}$ with $\mathbf{i}' \leq \mathbf{i}$. As well let $\mathbf{m}_{\mathbf{i},\mathbf{n}} = \max \mathcal{A}_{\mathbf{i},\mathbf{n}} \leq \mathbf{n}$ and define $D_{\mathbf{i},\mathbf{n}} = \sum_{\mathbf{m} \in \mathcal{A}_{\mathbf{i},\mathbf{n}}} a_{\mathbf{m}}$ where, by convention, $D_{\mathbf{i},\mathbf{n}} = 0$ and $\mathbf{m}_{\mathbf{i},\mathbf{n}} = (0, \ldots, 0)$ when $\mathcal{A}_{\mathbf{i},\mathbf{n}} = \emptyset$. From all that, we have

$$\mathbb{P}\left(\max_{\mathbf{m} \leq \mathbf{n}} |S_{\mathbf{m}}| b_{\mathbf{m}}^{-1} \geq \varepsilon\right) \leq \sum_{\mathbf{i} \leq \mathbf{k}_{\mathbf{n}}} \mathbb{P}\left(\max_{\mathbf{m} \in \mathcal{A}_{\mathbf{i},\mathbf{n}}} |S_{\mathbf{m}}| b_{\mathbf{m}}^{-1} \geq \varepsilon\right).$$

Since for $\mathbf{m} \in \mathcal{A}_{\mathbf{i},\mathbf{n}}$, $b_{\mathbf{m}} = \prod_{j=1}^{d} b_{m_j} \geq \prod_{j=1}^{d} c^{i_j}$ and $\mathcal{A}_{\mathbf{i},\mathbf{n}} \subset [0, m_{\mathbf{i},\mathbf{n}}]$, we get

$$\begin{aligned}\mathbb{P}\left(\max_{\mathbf{m} \leq \mathbf{n}} |S_{\mathbf{m}}| b_{\mathbf{m}}^{-1} \geq \varepsilon\right) \leq \sum_{\mathbf{i} \leq \mathbf{k}_{\mathbf{n}}} \mathbb{P}\left(\max_{\mathbf{m} \in \mathcal{A}_{\mathbf{i},\mathbf{n}}} |S_{\mathbf{m}}| b_{\mathbf{m}}^{-1} \geq \varepsilon\right) \leq \\ \sum_{\mathbf{i} \leq \mathbf{k}_{\mathbf{n}}} \mathbb{P}\left(\max_{\mathbf{m} \in \mathcal{A}_{\mathbf{i},\mathbf{n}}} |S_m| \geq \varepsilon \prod_{j=1}^{d} c^{i_j}\right).\end{aligned}$$

Now by applying (i) one arrives at

$$\begin{aligned}\mathbb{P}\left(\max_{\mathbf{m} \leq \mathbf{n}} |S_{\mathbf{m}}| b_{\mathbf{m}}^{-1} \geq \varepsilon\right) \leq & C\varepsilon^{-r} \sum_{\mathbf{i} \leq \mathbf{k}_{\mathbf{n}}} \prod_{j=1}^{d} c^{-ri_j} \sum_{\mathbf{m} \leq \mathbf{m}_{i,n}} a_m \leq \\ & C\varepsilon^{-r} \sum_{\mathbf{i} \leq \mathbf{k}_{\mathbf{n}}} \prod_{j=1}^{d} c^{-ri_j} \sum_{\mathbf{m} \leq \mathbf{i}} D_{\mathbf{m},\mathbf{n}}.\end{aligned}$$

By the remark made above, $\mathbf{m} \leq \mathbf{m}_{i,n} \in \mathcal{A}_{\mathbf{i},\mathbf{n}}$ implies that $\mathbf{m}$ is in some $\mathcal{A}_{\mathbf{s},\mathbf{n}}$ where $\mathbf{s} \leq \mathbf{i}$ and then by the definition of the $D_{\mathbf{i},\mathbf{n}}$ on has $\sum_{\mathbf{m} \leq \mathbf{m}_{i,n}} a_m \leq \sum_{\mathbf{m} \leq \mathbf{i}} D_{\mathbf{m},\mathbf{n}}$ and next

$$\mathbb{P}\left(\max_{\mathbf{m} \leq \mathbf{n}} |S_{\mathbf{m}}| b_{\mathbf{m}}^{-1} \geq \varepsilon\right) \leq C\varepsilon^{-r} \sum_{\mathbf{i} \leq \mathbf{k_n}} \prod_{j=1}^{d} c^{-ri_j} \sum_{\mathbf{m} \leq \mathbf{i}} D_{\mathbf{m}},$$

which becomes by a straightforward manipulations on the ranges of the sums, and where $k_{\mathbf{n}}(j)$ stands for the j-th coordinate of $k_{\mathbf{n}}$,

$$\mathbb{P}\left(\max_{\mathbf{m} \leq \mathbf{n}} |S_{\mathbf{m}}| b_{\mathbf{m}}^{-1} \geq \varepsilon\right) \leq C\varepsilon^{-r} \sum_{m \leq \mathbf{k_n}} D_{\mathbf{m},\mathbf{n}} \sum_{m \leq \mathbf{i} \leq \mathbf{k_n}} \prod_{j=1}^{d} c^{-ri_j} \leq$$

$$C\varepsilon^{-r} \sum_{m \leq \mathbf{k_n}} D_{\mathbf{m},\mathbf{n}} \prod_{j=1}^{d} \sum_{m_j \leq i_j \leq k_{\mathbf{n}}(j)} c^{-ri_j} =$$

$$C\varepsilon^{-r} \sum_{m \leq \mathbf{k_n}} D_{\mathbf{m},\mathbf{n}} \prod_{j=1}^{d} \frac{c^{-rm_j} - c^{-r(k_{\mathbf{n}}(j)+1)}}{1 - c^{-r}} \leq C\varepsilon^{-r} \sum_{m \leq \mathbf{k_n}} D_{\mathbf{m},\mathbf{n}} \prod_{j=1}^{d} \frac{c^{-rm_j}}{1 - c^{-r}},$$

since $c > 1$ and $k_{\mathbf{n}}(j) + 1 > m_j$. Now, at this last but one step, we have

$$\mathbb{P}\left(\max_{\mathbf{m} \leq \mathbf{n}} |S_{\mathbf{m}}| b_{\mathbf{m}}^{-1} \geq \varepsilon\right) \leq C\varepsilon^{-r} \left(\frac{c^r}{1 - c^{-r}}\right)^d \sum_{m \leq \mathbf{k_n}} D_{\mathbf{m},\mathbf{n}} \prod_{j=1}^{d} c^{-r(m_j+1)} \leq$$

$$C\varepsilon^{-r} \left(\frac{c^r}{1 - c^{-r}}\right)^d \sum_{m \leq \mathbf{k_n}} \sum_{\mathbf{s} \in \mathcal{A}_{\mathbf{m},\mathbf{n}}} a_{\mathbf{s}} \prod_{j=1}^{d} c^{-r(m_j+1)}.$$

Finally, taking into account the fact that for $\mathbf{s} \in \mathcal{A}_{\mathbf{m},\mathbf{n}}$, $c^{m_j+1} \geq b_{s_j}$, $1 \leq j \leq d$, that is $\prod_{j=1}^{d} c^{r(m_j+1)} \geq b_{\mathbf{s}}^r$, we arrive at

$$\mathbb{P}\left(\max_{\mathbf{m} \leq \mathbf{n}} |S_{\mathbf{m}}| b_{\mathbf{m}}^{-1} \geq \varepsilon\right) \leq C\varepsilon^{-r} \left(\frac{c^r}{1 - c^{-r}}\right)^d \sum_{m \leq \mathbf{k_n}} \sum_{\mathbf{s} \in \mathcal{A}_{\mathbf{m},\mathbf{n}}} \frac{a_{\mathbf{s}}}{b_{\mathbf{s}}^r} \leq$$

$$C\varepsilon^{-r} \left(\frac{c^r}{1 - c^{-r}}\right)^d \sum_{\mathbf{m} \leq \mathbf{n}} \frac{a_{\mathbf{m}}}{b_{\mathbf{m}}^r}.$$

Since c is arbitrary $c > 1$ and $\min_{c>1} \frac{c^r}{1-c^{-r}} = 4$, we achieve the proof by

$$\mathbb{P}\left(\max_{\mathbf{m} \leq \mathbf{n}} |S_{\mathbf{m}}| b_{\mathbf{m}}^{-1} \geq \varepsilon\right) \leq 4^d \; C \;\; \varepsilon^{-r} \sum_{\mathbf{m} \leq \mathbf{n}} \frac{a_{\mathbf{m}}}{b_{\mathbf{m}}^r}. \qquad \square$$

Proof of Theorem 2.3. Let $\beta_{\mathbf{n}}$ be the d-sequence obtained in the Lemma 2.2. According to Proposition 2.1

$$\mathbb{P}\left(\max_{\ell \leq \mathbf{m}} |S_\ell| \beta_\ell^{-1} \geq \varepsilon_{\mathbf{k}}\right) \leq 4^d C \varepsilon_{\mathbf{k}}^{-r} \sum_{\ell \leq \mathbf{m}} a_\ell \beta_\ell^{-r} \quad \forall \mathbf{m} \leq \mathbf{n}.$$

By this fact we get for any fixed $\mathbf{k} \in \mathbf{N}^d$

$$\mathbb{P}\left(\sup_{\ell \leq \mathbf{m}} |S_\ell|\beta_\ell^{-1} \geq \varepsilon_\mathbf{k}\right) \leq \lim_{\mathbf{m}\to\infty} \mathbb{P}\left(\max_{\ell \leq \mathbf{m}} |S_\ell|\beta_\ell^{-1} \geq \varepsilon_\mathbf{k}\right) \leq 4^d C \varepsilon_\mathbf{k}^{-r} \sum_\mathbf{n} a_\mathbf{n} \beta_\mathbf{n}^{-r},$$

where $\{\varepsilon_\mathbf{k}, \mathbf{k} \in \mathbf{N}^d\}$ a positive, nondecreasing, unbounded field of real numbers. So we have by Lemma 2.2

$$\lim_{\mathbf{k}\to\infty} \mathbb{P}\left(\sup_{\ell} |S_\ell|\beta_\ell^{-1} \geq \varepsilon_\mathbf{k}\right) = 0.$$

Using Lemma 3.1

$$\mathbb{P}\left(\sup_{\ell} |S_\ell|\beta_\ell^{-1} \geq \varepsilon_\mathbf{k} \text{ for all } \mathbf{k} \in \mathbf{N}^d\right) = 0.$$

So we have by Lemma 3.2 $\sup_\ell |S_\ell|\beta_\ell^{-1} < \infty$ a.s. Finally by Lemma 2.2

$$0 \leq \frac{|S_\mathbf{n}|}{b_\mathbf{n}} = \frac{|S_\mathbf{n}|}{\beta_\mathbf{n}} \frac{\beta_\mathbf{n}}{b_\mathbf{n}} \leq \sup_\ell |S_\ell|\beta_\ell^{-1} \frac{\beta_\mathbf{n}}{b_\mathbf{n}} \to 0 \quad \text{a.s.} \qquad \square$$

4. Conclusion

4.1. A first application: Logarithmically weighted sums

The following result is an extension of Theorem 7 in Noszály and Tómács [14] and of Theorem 4.2 in Fazekas et al. [5]. In this Theorem, we do not need any moment assumption in contrary of these above cited theorems.

Theorem 4.1. *Let $\{X_\mathbf{n}, \mathbf{n} \in \mathbf{N}^d\}$ be a field of random variables. Let $r > 1$. We assume there exists $C > 0$ such that for any $\mathbf{m} \in \mathbf{N}^d$ and any $\varepsilon > 0$*

$$\mathbb{P}\left(\max_{\ell \leq \mathbf{m}} \sum_{\mathbf{k} \leq \ell} \frac{X_\mathbf{k}}{\langle \mathbf{k} \rangle} \geq \varepsilon\right) \leq C\varepsilon^{-r} \sum_{\ell \leq \mathbf{m}} \frac{1}{\langle \ell \rangle}.$$

Then

$$\frac{1}{|\log \mathbf{n}|} \sum_{\mathbf{k} \leq \mathbf{n}} \frac{X_\mathbf{k}}{\langle \mathbf{k} \rangle} \to 0 \quad (\mathbf{n} \to \infty) \quad a.s.$$

Proof. Let us apply Theorem 2.3 with $a_\mathbf{n} = \frac{1}{\langle \mathbf{n} \rangle}$ and $b_\mathbf{n} = |\log \mathbf{n}|$. The proof is achieved by remarking that for $r > 1$

$$\sum_\mathbf{n} \frac{a_\mathbf{n}}{b_\mathbf{n}^r} = \sum_\mathbf{n} \frac{1}{|\log \mathbf{n}|^r} \frac{1}{\langle \mathbf{n} \rangle} < \infty. \qquad \square$$

4.2. A second application

By using Markov's Inequality and applying our results (see Theorem 2.3), under the same assumptions in Noszály and Tómács [14], we rediscover their results.

Acknowledgement. The paper was finalized while the second author was visiting MAPMO, University of Orléans, France, in 2011. He expresses her warm thanks to responsibles of MAPMO for kind hospitality. The authors also thank the referee for his valuable comments and suggestions.

References

[1] Cairoli, R. (1970) Une inégalité pour martingales à indices multiples et ses applications. *Seminaire des Probabilités IV, Université de Strasbourg. Lecture Notes in Math.* **124**, 1–27. Springer, Berlin. (MR0270424)

[2] Fazekas, I. (1983) Convergence of vector valued martingales with multidimensional indices. *Publ. Math. Debrecen* **30/1-2**, 157–164. (MR0733082)

[3] Fazekas, I. and Klesov, O. (2000) A general approach to the strong laws of large numbers. *Teor. Veroyatnost i Primenen* **45** (5), 568-583, *Theory of Probab. Appl.* **45** (3), 436–449. (MR1967791)

[4] Fazekas, I. and Klesov, O. (1998) A general approach to the strong laws of large numbers for random fields. Technical Report $n^o 4$ (1998), Kossuth Lajos University, Hungary.

[5] Fazekas, I., Klesov, O. and Noszály, Cs., Tómács, T. (1999) Strong laws of large numbers for sequences and fields. (Proceedings of the third Ukrainian - Scandinavian. *Conference in Probability theory and Mathematical Statistics 8 - 12. June 1999 Kyiv, Ukraine.*Theory Stoch. Process. 5 (1999), no. 3-4, 91–104. (MR2018403)

[6] Guyon, X. (1995) *Random fields on a Network: Modeling, Statistics and Applications.* Springer, New York. (MR1344683)

[7] Hájék, J. and Rényi, A. (1955) Generalization of an inequality of Kolmogorov. *Acta Math. Acad. Sci. Hungar.* **6/3** - 4, 281 - 283. (MR0076207)

[8] Klesov, O. (1980) The Hájek-Rényi inequality for random fields and strong law of large numbers. *Teor. Veroyatnost. i Mat. Statist.* **22** , 58–66 (Russian). (MR0568238).

[9] Loéve, M. (1977) *Probability Theory I.* Springer-Verlag. New-York.

[10] Lagodowski Zbiguiew A. (2009) Strong laws of large numbers for B-valued random fields. *Discrete Dynamics in Nature and Society.* Article ID 485412, 12 pages. Doi:10.1155/2009/485412.

[11] Móricz, F. (1977) Moment inequalities for the maximum of partial sums of random fields. *Acta Sci. Math.* **39**, 353–366. (MR0458535)

[12] Móricz, F. (1983) A general moment inequality for the maximum of the rectangular partial sum of multiple series. *Acta Math. Hung.* **41**, 337–346. (MR0703745)

[13] Nguyen V. Q. and Nguyen V. H. (2010) A Hájék-Rényi-type Maximal Inequality and strong laws of large numbers for multidimensional arrays. *J. Inequal. Appl.*. Art. ID 569759, 14 pp. (MR2765284)

[14] Noszály, Cs. and Tómács, T. (2000) A general approach to strong laws of large numbers for fields of random variables. *Ann. Univ. Sci. Budapest.* **43**, 61–78. (MR1847869)

[15] Peligrad, M. and Gut, A. (1999) Almost-sure results for a class of dependent random variables. *J. of Theoret. Probab.* **12/I**, 87–104. (MR1674972)

[16] Tómács, T. and Líbor, Z. (2006) A Hájék-Rényi type inequality and its applications. *Ann. Math. Inform.* **33**, 141–149. (MR2385473)

[17] Tómács, T. (2007) A general method to obtain the rate of convergence in the strong law of large numbers. *Ann. Math. Inform.* **34**, 97–102. (MR2385429)

[18] Tómács, T. (2008) Convergence rate in the strong law of large numbers for mixingales and superadditive structures. *Ann. Math. Inform.* **35**, 147–154. (MR2475872)

[19] Tómács, T. (2009) An almost sure limit theorem for α-mixing random fields. *Ann. Math. Inform.* **36**, 123–132. (MR2580908)

Annales Mathematicae et Informaticae
38 *(2011) pp. 15–25*
http://ami.ektf.hu

Harmonic sections on the tangent bundle of order two*

Nour El Houda Djaa, Seddik Ouakkas, Djaa Mustapha

Laboratory of Geometry, Analysis, Control and Applications
Department of Mathematics, University of Saida, Saida, Algeria.
e-mail: Lgaca_saida2009@hotmail.com

Submitted July 25, 2010 Accepted May 30, 2011

Abstract

The problem studied in this paper is related to the Harmonicity of sections from a Riemannian manifold (M, g) into its tangent bundle of order two T^2M equipped with the Diagonal metric g^D. First we introduce a connection on $\Gamma(T^2M)$ and we investigate the geometry and the harmonicity of sections as maps from (M, g) to (T^2M, g^D).

Keywords: Horizontal lift, vertical lift, harmonic maps.

MSC: 53A45, 53C20, 58E20

1. Introduction

Consider a smooth map $\phi : (M^m, g) \to (N^n, h)$ between two Riemannian manifolds, then the energy functional is defined by

$$E(\phi) = \frac{1}{2} \int_M |d\phi|^2 dv_g \tag{1.1}$$

(or over any compact subset $K \subset M$).

A map is called harmonic if it is a critical point of the energy functional E (or $E(K)$ for all compact subsets $K \subset M$). For any smooth variation $\{\phi\}_{t\in I}$ of ϕ with $\phi_0 = \phi$ and $V = \left.\frac{d\phi_t}{dt}\right|_{t=0}$, we have

$$\left.\frac{d}{dt} E(\phi_t)\right|_{t=0} = -\int_M h(\tau(\phi), V)\, dv_g, \tag{1.2}$$

*The authors would like to thank the referee for his useful remarks.

where

$$\tau(\phi) = \mathrm{trace}_g \nabla d\phi \tag{1.3}$$

is the tension field of ϕ. Then we have

Theorem 1.1. *A smooth map $\phi : (M^m, g) \to (N^n, h)$ is harmonic if and only if*

$$\tau(\phi) = 0. \tag{1.4}$$

If $(x^i)_{1\leq i\leq m}$ and $(y^\alpha)_{1\leq \alpha\leq n}$ denote local coordinates on M and N respectively, then equation (1.4) takes the form

$$\tau(\phi)^\alpha = \left(\Delta\phi^\alpha + g^{ij\,N}\Gamma^\alpha_{\beta\gamma}\frac{\partial\phi^\beta}{\partial x^i}\frac{\partial\phi^\gamma}{\partial x^j}\right) = 0, \quad 1\leq\alpha\leq n, \tag{1.5}$$

where $\Delta\phi^\alpha = \frac{1}{\sqrt{|g|}}\frac{\partial}{\partial x^i}\left(\sqrt{|g|}g^{ij}\frac{\partial\phi^\alpha}{\partial x^j}\right)$ is the Laplace operator on (M^m, g) and ${}^N\Gamma^\alpha_{\beta\gamma}$ are the Christoffel symbols on N. One can refer to [1, 4, 6, 7, 8, 9] for background on harmonic maps.

2. Some results on horizontal and vertical lifts

Let (M, g) be an n-dimensional Riemannian manifold and (TM, π, M) be its tangent bundle. A local chart

$$(U, x^i)_{i=1\ldots n}$$

on M induces a local chart $(\pi^{-1}(U), x^i, y^j)_{i,j=1,\ldots,n}$ on TM. Denote by Γ^k_{ij} the Christoffel symbols of g and by ∇ the Levi-Civita connection of g.
We have two complementary distributions on TM, the vertical distribution $\mathcal{V}$ and the horizontal distribution $\mathcal{H}$, defined by:

$$\begin{aligned}
\mathcal{V}_{(x,u)} &= \mathrm{Ker}(d\pi_{(x,u)})\\
&= \{a^i\frac{\partial}{\partial y^i}|_{(x,u)}; \quad a^i\in\mathbb{R}\}\\
\mathcal{H}_{(x,u)} &= \{a^i\frac{\partial}{\partial x^i}|_{(x,u)} - a^i u^j\Gamma^k_{ij}\frac{\partial}{\partial y^k}|_{(x,u)}; \quad a^i\in\mathbb{R}\},
\end{aligned}$$

where $(x,u)\in TM$, such that $T_{(x,u)}TM = \mathcal{H}_{(x,u)}\oplus\mathcal{V}_{(x,u)}$.
Let $X = X^i\frac{\partial}{\partial x^i}$ be a local vector field on M. The vertical and the horizontal lifts of X are defined by

$$X^V = X^i\frac{\partial}{\partial y^i} \tag{2.1}$$

$$X^H = X^i\frac{\delta}{\delta x^i} = X^i\{\frac{\partial}{\partial x^i} - y^j\Gamma^k_{ij}\frac{\partial}{\partial y^k}\} \tag{2.2}$$

For consequences, we have $(\frac{\partial}{\partial x^i})^H = \frac{\delta}{\delta x^i}$, $(\frac{\partial}{\partial x^i})^V = \frac{\partial}{\partial y^i}$ and $(\frac{\delta}{\delta x^i}, \frac{\partial}{\partial y^j})_{i,j=1,\ldots,n}$ a local frame on TM.

Remark 2.1.

1. if $w = w^i \frac{\partial}{\partial x^i} + \overline{w}^j \frac{\partial}{\partial y^j} \in T_{(x,u)}TM$, then its horizontal and vertical parts are defined by
$$w^h = w^i \frac{\partial}{\partial x^i} - w^i u^j \Gamma^k_{ij} \frac{\partial}{\partial y^k} \in \mathcal{H}_{(x,u)}$$
$$w^v = \{\overline{w}^k + w^i u^j \Gamma^k_{ij}\} \frac{\partial}{\partial y^k} \in \mathcal{V}_{(x,u)}$$

2. if $u = u^i \frac{\partial}{\partial x^i} \in T_x M$ then its vertical and horizontal lifts are defined by
$$u^V = u^i \frac{\partial}{\partial y^i}$$
$$u^H = u^i \{\frac{\partial}{\partial x^i} - y^j \Gamma^k_{ij} \frac{\partial}{\partial y^k}\}.$$

Proposition 2.2 (see [10]). *Let $F \in \mathfrak{T}^1_p(M)$ be a tensor of type (1,p) (respectively, $G \in \mathfrak{T}^0_p(M)$ a tensor of type (0,p)), then there exist a tensor $\gamma(F) \in \mathfrak{T}^1_{p-1}(TM)$ (respectively, $\gamma(G) \in \mathfrak{T}^0_{p-1}(TM)$), localy defined by*

$$\gamma(F) = F^k_{h_1 h_2 \dots h_p} y^{h_1} \frac{\partial}{\partial y^k} \otimes dx^{h_2} \otimes \cdots \otimes dx^{h_p} \tag{2.3}$$

$$\gamma(G) = G_{h_1 h_2 \dots h_p} y^{h_1} dx^{h_2} \otimes \cdots \otimes dx^{h_p} \tag{2.4}$$

where $F = F^j_{i_1 \dots i_p} \frac{\partial}{\partial x^j} \otimes dx^{i_1} \otimes \cdots \otimes dx^{i_p}$ and $G = G_{i_1 \dots i_p} dx^{i_1} \otimes \cdots \otimes dx^{i_p}$.

Definition 2.3. The Sasaki metric g^s on the tangent bundle TM of M is given by

1. $g^s(X^H, Y^H) = g(X, Y) \circ \pi$
2. $g^s(X^H, Y^V) = 0$
3. $g^s(X^V, Y^V) = g(X, Y) \circ \pi$

for all vector fields $X, Y \in \Gamma(TM)$.

In the general case, Sasaki metrics is considered in [2, 5, 7, 10].

Proposition 2.4 (see [6]). *A vector fields $X : (M, g) \to (TM, g^s)$ is harmonic iff*

$$\sum_{i=1} X^k_{ii} = 0, \qquad \sum_{i=1} R^k_{ilj} X^j_i = 0.$$

where X^k_i (resp X^k_{ij}) are the components of the first (resp second) covariant differential of the vector field X.

From Proposition 2.4 we deduce

Proposition 2.5. *If $X : (M, g) \to (TM, g^s)$ is a harmonic vector field, then*

$$\mathrm{trace}_g \nabla^2 X = 0, \qquad \mathrm{trace}_g R(X, \nabla_* X)* = 0.$$

Let M be an n-dimensional manifold. The tangent bundle of order 2 is the natural bundle of 2-jets of differentiable curves, defined by:

$$\begin{aligned} T^2M = \{j_0^2\gamma \quad ; \quad \gamma : \mathbb{R}_0 \to M, \text{ is a smooth curve at } 0 \in \mathbb{R}\} \\ \pi_2 : T^2M \to M \\ j_0^2\gamma \mapsto \gamma(0) \end{aligned}$$

A local chart $(U, x^i)_{i=1\ldots n}$ on M induces a local chart $(\pi_2^{-1}(U), x^i, y^i, z^i)_{i=1\ldots n}$ on T^2M by the following formulae

$x^i = \gamma^i(0)$.

$y^i = \frac{d}{dt}\gamma^i(0)$.

$z^i = \frac{d^2}{dt^2}\gamma^i(0)$.

Proposition 2.6. *Let M, be an n-dimensional manifold, then TM is sub-bundle of T^2M, and the map*

$$\begin{aligned} i : TM \to T^2M \\ j_0^1 f = j_0^2 \widetilde{f} \end{aligned} \tag{2.5}$$

is an injective homomorphism of a natural bundles (not of vector bundles), where

$$\widetilde{f^i} = \int_0^t f^i(s)ds - tf^i(0) + f^i(0) \quad i = 1\ldots n.$$

Proof. Locally if (U, x^i) is a chart on M and (U, x^i, y^i) and (U, x^i, y^i, z^i) are the induced chart on TM and T^2M respectivelly, then we have $i : (x^i, y^i) \mapsto (x^i, 0, y^i)$, it follows that i is an injective homomorphism. Remains to show that i is well defined.
Let (U, φ) and (V, ψ) are a charts on M, for any vector $j_0^1 f \in TM$, if we denote

$$\widetilde{f}(t) = \varphi^{-1}(\int_0^t \varphi \circ f(s)ds - t\varphi \circ f(0) + \varphi \circ f(0))$$
$$\widehat{f}(t) = \psi^{-1}(\int_0^t \psi \circ f(s)ds - t\psi \circ f(0) + \psi \circ f(0))$$

then we obtain

$$\varphi \circ \widetilde{f}(0) = \varphi \circ f(0)$$

$$
\begin{aligned}
&= \varphi \circ \widehat{f}(0) \\
\frac{d}{dt}(\varphi \circ \widetilde{f})(0) &= 0 \\
&= \frac{d}{dt}(\varphi \circ \widehat{f})(0) \\
\frac{d^2}{dt^2}(\varphi \circ \widetilde{f})(0) &= \frac{d}{dt}(\varphi \circ f)(0) \\
&= \frac{d^2}{dt^2}(\varphi \circ \widehat{f})(0)
\end{aligned}
$$

which proves that $j_0^2\widetilde{f} = j_0^2\widehat{f}$.

Theorem 2.7. *Let (M, g) be a Riemannian manifold and ∇ be the Levi-Civita connection. If $TM \oplus TM$ denotes the Whitney sum, then*

$$
\begin{aligned}
S : T^2M &\to TM \oplus TM \\
j_0^2\gamma &\mapsto (\dot{\gamma}(0), (\nabla_{\dot{\gamma}(0)}\dot{\gamma})(0))
\end{aligned}
\tag{2.6}
$$

is a diffeomorphism of natural bundles.

In the induced coordinate, we have

$$
S : (x^i, y^i, z^i) \mapsto (x^i, y^i, z^i + y^j y^k \Gamma^i_{jk}) \tag{2.7}
$$

In the more general case, the difeomorphism S is considered in [3].

Remark 2.8. The diffeomorphism S determines a vector bundle structure on T^2M, for which S be an isomorphism of vector bundles, and $i : TM \to T^2M$ is an injective homomorphism of vector bundles.

Definition 2.9. Let (M, g) be a Riemannian manifold and T^2M be its tangent bundle of order 2 endowed with the vectorial structure induced by the diffeomorphism S. For any section $\sigma \in \Gamma(T^2M)$, we define two vector fields on M by:

$$
\begin{aligned}
X_\sigma &= P_1 \circ S \circ \sigma \\
Y_\sigma &= P_2 \circ S \circ \sigma
\end{aligned}
\tag{2.8}
$$

where P_1 and P_2 denotes the first and the second projection from $TM \oplus TM$ on TM.

Remark 2.10. We can easily verify that for all sections $\sigma, \varpi \in \Gamma(T^2M)$ and $\alpha \in \mathbb{R}$, we have

$$
\begin{aligned}
X_{\alpha\sigma+\varpi} &= \alpha X_\sigma + X_\varpi \\
Y_{\alpha\sigma+\varpi} &= \alpha Y_\sigma + Y_\varpi
\end{aligned}
$$

From the Remarks 2.8 and 2.10 we can define a connection on $\Gamma(T^2M)$.

Definition 2.11. Let (M,g) be a Riemannian manifold and T^2M be its tangent bundle of order 2 endowed with the vectorial structure induced by the diffeomorphism S. We define a connection on $\Gamma(T^2M)$ by:

$$\begin{aligned} \widehat{\nabla} : \Gamma(TM) \times \Gamma(T^2M) &\to \Gamma(T^2M) \\ (Z,\sigma) &\mapsto \widehat{\nabla}_Z\sigma = S^{-1}((\nabla_Z X_\sigma, \nabla_Z Y_\sigma)) \end{aligned} \tag{2.9}$$

where ∇ is the Levi-Civita connection on M.

From formula 2.7 and Definition 2.9 , it follows

Proposition 2.12. *If (U,x^i) is a chart on M and $(\sigma^i,\overline{\sigma}^i)$ are the components of section $\sigma \in \Gamma(T^2M)$ then*

$$\begin{aligned} X_\sigma &= \sigma^i \frac{\partial}{\partial x^i} \\ Y_\sigma &= (\overline{\sigma}^k + \sigma^i\sigma^j\Gamma^k_{ij})\frac{\partial}{\partial x^k} \end{aligned}$$

From Theorem 2.7 and Remark 2.10 we have

Proposition 2.13. *Let (M,g) be a Riemannian manifold and T^2M be its tangent bundle of order 2, then*

$$\begin{aligned} J : \Gamma(TM) &\to \Gamma(T^2M) \\ Z &= S^{-1}(Z,0) \end{aligned} \tag{2.10}$$

is an injective homomorphism of vector bundles.

Locally if (U,x^i) is a chart on M and (U,x^i,y^i) and (U,x^i,y^i,z^i) are the induced chart on TM and T^2M respectivelly, then we have

$$J : (x^i,y^i) \mapsto (x^i,y^i,-y^jy^k\Gamma^i_{jk}) \tag{2.11}$$

Definition 2.14. Let (M,g) be a Riemannian manifold and $X \in \Gamma(TM)$ be a vector field on M. For $\lambda = 0,1,2$, the λ-lift of X to T^2M is defined by

$$\begin{aligned} X^0 &= S_*^{-1}(X^H, X^H) \\ X^1 &= S_*^{-1}(X^V, 0) \\ X^2 &= S_*^{-1}(0, X^V) \end{aligned} \tag{2.12}$$

In the more general case, the λ-lift is considered in [3].

Theorem 2.15 (see [3]). *Let (M,g) be a Riemannian manifold and R its tensor curvature, then for all vector fields $X,Y \in \Gamma(TM)$ and $p \in T^2M$ we have*

1. $[X^0,Y^0]_p = [X,Y]^0_p - (R(X,Y)u)^1 - (R(X,Y)w)^2$

2. $[X^0, Y^i] = (\nabla_X Y)^i$

3. $[X^i, Y^j] = 0.$

where $(u, w) = S(p)$ *and* $i, j = 1, 2$.

Definition 2.16. Let (M, g) be a Riemannian manifold. For any section $\sigma \in \Gamma(T^2M)$ we define the vertical lift of σ to T^2M by

$$\sigma^V = S_*^{-1}(X_\sigma^V, Y_\sigma^V) \in \Gamma(T(T^2M)). \tag{2.13}$$

Remark 2.17. From Definition 2.9 and the formulae (2.5), (2.10), (2.12) and (2.13), for all $\sigma \in \Gamma(T^2M)$ and $Z \in \Gamma(TM)$, we obtain

- $\sigma^V = X_\sigma^1 + Y_\sigma^2$
- $(\widehat{\nabla}_Z \sigma)^V = (\nabla_Z X_\sigma)^1 + (\nabla_Z Y_\sigma)^2$
- $Z^1 = J(Z)^V$
- $Z^2 = i(Z)^V$

3. Metric diagonal and harmonicity

Using Definition 2.3 and formula (2.12), we have

Theorem 3.1. *Let* (M, g) *be a Riemannian manifold and* TM *its tangent bundle equipped with the Sasakian metric* g^s*, then*

$$g^D = S_*^{-1}(\widetilde{g} \oplus \widetilde{g})$$

is the only metric that satisfies the following formulae

$$g^D(X^i, Y^j) = \delta_{ij} \cdot g(X, Y) \circ \pi_2 \tag{3.1}$$

for all vector fields $X, Y \in \Gamma(TM)$ *and* $i, j = 0, \ldots, 2$*, where* $\widetilde{g}$ *is the metric defined by*

$$\begin{aligned}
\widetilde{g}(X^H, Y^H) &= \frac{1}{2} g^s(X^H, Y^H) \\
\widetilde{g}(X^H, Y^V) &= g^s(X^H, Y^V) \\
\widetilde{g}(X^V, Y^V) &= g^s(X^V, Y^V),
\end{aligned}$$

g^D is called the diagonal lift of g to T^2M.

Theorem 3.2. *Let* (M, g) *be a Riemannian manifold and* $\widetilde{\nabla}$ *be the Levi-Civita connection of the tangent bundle of order two* T^2M *equipped with the diagonal metric* g^D*. Then*

1. $(\widetilde{\nabla}_{X^0}Y^0)_p = (\nabla_X Y)^0 - \frac{1}{2}(R(X,Y)u)^1 - \frac{1}{2}(R(X,Y)w)^2$,
2. $(\widetilde{\nabla}_{X^0}Y^1)_p = (\nabla_X Y)^1 + \frac{1}{2}(R(u,Y)X)^0$,
3. $(\widetilde{\nabla}_{X^0}Y^2)_p = (\nabla_X Y)^2 + \frac{1}{2}(R(w,Y)X)^0$,
4. $(\widetilde{\nabla}_{X^1}Y^0)_p = \frac{1}{2}(R_x(u,X)Y))^0$,
5. $(\widetilde{\nabla}_{X^2}Y^0)_p = \frac{1}{2}(R_x(w,X)Y))^0$,
6. $(\widetilde{\nabla}_{X^i}Y^j)_p = 0$

for all vector fields $X, Y \in \Gamma(TM)$ *and* $p \in \Gamma(T^2M)$*, where* $i, j = 1, 2$ *and* $(u, w) = S(p)$*.*

The proof of theorem 3.2 follows directly from Theorem 3.1 and the Kozul formula.

Lemma 3.3. *Let* (M, g) *be a Riemannian manifold and* (TM, g^s) *be the tangent bundle equipped with the Sasaki metric. If* $X, Y \in \Gamma(TM)$ *are a vector fields and* $(x, u) \in TM$ *such that* $X_x = u$*, then we have*

$$d_x X(Y_x) = Y^H_{(x,u)} + (\nabla_Y X)^V_{(x,u)}.$$

Proof. Let (U, x^i) be a local chart on M in $x \in M$ and $(\pi^{-1}(U), x^i, y^j)$ be the induced chart on TM, if $X_x = X^i(x)\frac{\partial}{\partial x^i}|_x$ and $Y_x = Y^i(x)\frac{\partial}{\partial x^i}|_x$, then

$$d_x X(Y_x) = Y^i(x)\frac{\partial}{\partial x^i}|_{(x,X_x)} + Y^i(x)\frac{\partial X^k}{\partial x^i}(x)\frac{\partial}{\partial y^k}|_{(x,X_x)},$$

thus the horizontal part is given by

$$\begin{aligned}(d_x X(Y_x))^h &= Y^i(x)\frac{\partial}{\partial x^i}|_{(x,X_x)} - Y^i(x)X^j(x)\Gamma^k_{ij}(x)\frac{\partial}{\partial y^k}|_{(x,X_x)}\\ &= Y^H_{(x,X_x)}\end{aligned}$$

and the vertical part is given by

$$\begin{aligned}(d_x X(Y_x))^v &= \{Y^i(x)\frac{\partial X^k}{\partial x^i}(x) + Y^i(x)X^j(x)\Gamma^k_{ij}(x)\}\frac{\partial}{\partial y^k}|_{(x,X_x)}\\ &= (\nabla_Y X)^V_{(x,X_x)}.\end{aligned}$$

Lemma 3.4. *Let* (M, g) *be a Riemannian manifold and* (T^2M, g^D) *be the tangent bundle equipped with the diagonal metric. If* $Z \in \Gamma(TM)$ *and* $\sigma \in \Gamma(T^2M)$ *, then we have*

$$d_x\sigma(Z_x) = Z^0_p + (\widehat{\nabla}_Z\sigma)^V_p. \tag{3.2}$$

where $p = \sigma(x)$*.*

Proof. Using Lemma 3.3, we obtain

$$\begin{aligned} d_x\sigma(Z) &= dS^{-1}(dX_\sigma(Z), dY_\sigma(Z))_{S(p)} \\ &= dS^{-1}(Z^h, Z^h)_{S(p)} + dS^{-1}((\nabla_Z X_\sigma)^v, (\nabla_Z Y_\sigma)^v)_{S(p)} \\ &= Z_p^0 + (\widehat{\nabla}_Z \sigma)_p^V. \end{aligned}$$

Lemma 3.5. *Let (M, g) be a Riemannian n-dimensional manifold and (T^2M, g^D) be its tangent bundle of order two equipped with the diagonal metric and let $\sigma \in \Gamma(T^2M)$. Then the energy density associated with σ is*

$$e(\sigma) = \frac{n}{2} + \frac{1}{2}\|\widehat{\nabla}\sigma\|^2.$$

where $\|\widehat{\nabla}\sigma\|^2 = \mathrm{trace}_g g(\nabla X_\sigma, \nabla X_\sigma) + \mathrm{trace}_g g(\nabla Y_\sigma, \nabla Y_\sigma)$.

Proof. Let $(e_1, \ldots, e_n)$ be a local orthonormal frame on M, then

$$e(\sigma) = \frac{1}{2}\sum_{i=1}^{n} g^D(d\sigma(e_i), d\sigma(e_i))$$

Using formula 3.2 and Remark 2.17, we obtain

$$\begin{aligned} e(\sigma) &= \frac{1}{2}\sum_{i=1}^{n} g^D(e_i^0, e_i^0) + \frac{1}{2}\sum_{i=1}^{n} g^D((\widehat{\nabla}_{e_i}\sigma)^V, (\widehat{\nabla}_{e_i}\sigma)^V) \\ &= \frac{n}{2} + \frac{1}{2}\|\widehat{\nabla}\sigma\|^2. \end{aligned}$$

Theorem 3.6. *Let (M, g) be a Riemannian manifold and (T^2M, g^D) be its tangent bundle of order two equipped with the diagonal metric. Then the tension field associated with $\sigma \in \Gamma(T^2M)$ is*

$$\tau(\sigma) = (\mathrm{trace}_g \widehat{\nabla}^2\sigma)^V + (\mathrm{trace}_g\{R(X_\sigma, \nabla_* X_\sigma) * + R(Y_\sigma, \nabla_* Y_\sigma) *\})^0. \tag{3.3}$$

Proof. Let $x \in M$ and $\{e_i\}_{i=1}^n$ be a local orthonormal frame on M such that $\nabla_{e_i} e_j = 0$, then

$$\begin{aligned} \tau(\sigma)_x &= \sum_{i=1}^{n} (\nabla_{d\sigma(e_i)} d\sigma(e_i))_{\sigma(x)} \\ &= \sum_{i=1}^{n} \left[\nabla_{e_i^0 + (\nabla_{e_i}\sigma)^V}\left(e_i^0 + (\widehat{\nabla}_{e_i}\sigma)^V\right)\right]_{\sigma(x)} \end{aligned}$$

From Theorem 3.2, we obtain

$$\begin{aligned}\tau(\sigma)_x &= \sum_{i=1}^{n}\Big\{\nabla_{e_i^0}e_i^0 + \nabla_{e_i^0}(\nabla_{e_i}X_\sigma)^1 + \nabla_{e_i^0}(\nabla_{e_i}Y_\sigma)^2 + \nabla_{(\nabla_{e_i}X_\sigma)^1}e_i^0 \\ &\quad + \nabla_{(\nabla_{e_i}Y_\sigma)^2}e_i^0\Big\}_{\sigma(x)} \\ &= \sum_{i=1}^{n}\Big\{(\nabla_{e_i}\nabla_{e_i}X_\sigma)^1_{\sigma(x)} + (\nabla_{e_i}\nabla_{e_i}Y_\sigma)^2_{\sigma(x)} + (R_x(X_\sigma(x),\nabla_{e_i}X_\sigma)e_i)^0 \\ &\quad + (R_x(Y_\sigma(x),\nabla_{e_i}Y_\sigma)e_i)^0\Big\}\end{aligned}$$

Theorem 3.7. *Let (M,g) be a Riemannian manifold and (T^2M,g^D) be its tangent bundle of order two equipped with the diagonal metric. A section $\sigma : M \to T^2M$ is harmonic if and only the following conditions are verified*

$$\begin{aligned}&\mathrm{trace}_g(\nabla^2 X_\sigma) = 0,\\ &\mathrm{trace}_g(\nabla^2 Y_\sigma) = 0,\\ &\mathrm{trace}_g\{R(X_\sigma,\nabla_* X_\sigma)* + R(Y_\sigma,\nabla_* Y_\sigma)*\} = 0.\end{aligned}$$

From Proposition 2.5 and Theorem 3.7 we obtain

Corollary 3.8. *Let (M,g) be a Riemannian manifold and (T^2M,g^D) be its tangent bundle of order two equipped with the diagonal metric. If $\sigma : M \to T^2M$ is a section such that X_σ and Y_σ are harmonic vector fields, then σ is harmonic.*

Corollary 3.9. *Let (M,g) be a Riemannian manifold and (T^2M,g^D) be its tangent bundle of order two equipped with the diagonal metric. If $\sigma : M \to T^2M$ is a section such that X_σ and Y_σ are parallel, then σ is harmonic.*

Theorem 3.10. *Let (M,g) be a Riemannian compact manifold and (T^2M,g^D) be its tangent bundle of order two equipped with the diagonal metric. Then $\sigma : M \to T^2M$ is a harmonic section if and only if σ is parallel (i.e $\widehat{\nabla}\sigma = 0$).*

Proof. If σ is parallel, from Corollary 3.9, we deduce that σ is harmonic. Inversely. Let σ_t be a compactly supported variation of σ defined by $\sigma_t = (1+t)\sigma$. From Lemma 3.5 we have

$$e(\sigma_t) = \frac{n}{2} + \frac{(t+1)^2}{2}\|\widehat{\nabla}\sigma\|^2.$$

If σ is a critical point of the energy functional we have :

$$\begin{aligned}0 &= \frac{d}{dt}E(\sigma_t)_{|t=0},\\ &= \int_M \|\widehat{\nabla}\sigma\|^2 dv_{g^D}\end{aligned}$$

Hence $\widehat{\nabla}\sigma = 0$.

References

[1] Baird, P., Fardoun, A., Ouakkas, S., Conformal and semi-conformal biharmonic maps, *Annals of global analysis and geometry*, 34 (2008),403–414.

[2] Cengiz, N., Salimov, A. A., Diagonal lift in the tensor bundle and its applications, *Appl. Math. Comput*, 142(2003), 309–319 .

[3] Djaa, M., Gancarzewicz, J., The geometry of tangent bundles of order r, *Boletin Academia , Galega de Ciencias ,Espagne*, 4 (1985), 147–165

[4] Eells, J., Sampson, J. H., Harmonic mappings of Riemannian manifolds, *Amer. J. Maths*, 86(1964).

[5] Gudmunsson, S., Kappos, E., On the Geometry of Tangent Bundles, *Expo.Math*, 20 (2002), 1–41.

[6] Ishihara, T., Harmonic sections of tangent bundles, *J. Math. Tokushima Univ*, 13 (1979), 23–27.

[7] Oniciuc, C., Nonlinear connections on tangent bundle and harmonicity, *Ital. J. Pure Appl*, 6 (1999), 109–122 .

[8] Oproiu, V., On Harmonic Maps Between Tangent Bundles, *Rend.Sem.Mat*, 47 (1989), 47–55 .

[9] Ouakkas, S., Biharmonic maps, conformal deformations and the Hopf maps, *Differential Geometry and its Applications*,26 (2008), 495–502.

[10] Yano,K., Ishihara, S.,Tangent and Cotangent Bundles, *Marcel Dekker.INC. New York*, (1973).

Annales Mathematicae et Informaticae
38 *(2011) pp. 27–36*
http://ami.ektf.hu

On $\mathcal{I}_c^{(q)}$-convergence

J. Gogola[a], M. Mačaj[b], T. Visnyai[b]

[a]University of Economics, Bratislava, Slovakia
e-mail: gogola@euba.sk

[b]Faculty of Mathematics, Physics and Informatics
Comenius University, Bratislava, Slovakia
e-mail: visnyai@fmph.uniba.sk, macaj@fmph.uniba.sk

Submitted October 28, 2010 Accepted March 10, 2011

Abstract

In this paper we will study the properties of ideals $\mathcal{I}_c^{(q)}$ related to the notion of $\mathcal{I}$-convergence of sequences of real numbers. We show that $\mathcal{I}_c^{(q)}$ and $\mathcal{I}_c^{(q)*}$-convergence are equivalent. We prove some results about modified Olivier's theorem for these ideals. For bounded sequences we show a connection between $\mathcal{I}_c^{(q)}$-convergence and regular matrix method of summability.

1. Introduction

In papers [9],and [10] the notion of $\mathcal{I}$-convergence of sequences of real numbers is introduced and its basic properties are investigated. The $\mathcal{I}$-convergence generalizes the notion of the statistical convergence (see[5]) and it is based on the ideal $\mathcal{I}$ of subsets of the set $\mathbb{N}$ of positive integers.

Let $\mathcal{I} \subseteq 2^{\mathbb{N}}$. $\mathcal{I}$ is called an admissible ideal of subsets of positive integers, if $\mathcal{I}$ is additive (i.e. $A, B \in \mathcal{I} \Rightarrow A \cup B \in \mathcal{I}$), hereditary (i.e. $A \in \mathcal{I},\ B \subset A \Rightarrow B \in \mathcal{I}$), containing all singletons and it doesn't contain $\mathbb{N}$. Here we present some examples of admissible ideals. More examples can be found in the papers [7, 9, 10, 12].

(a) The class of all finite subsets of $\mathbb{N}$ form an admissible ideal usually denote by $\mathcal{I}_f$.

(b) Let ϱ be a density function on $\mathbb{N}$, then the set $\mathcal{I}_\varrho = \{A \subseteq \mathbb{N} : \varrho(A) = 0\}$ is an admissible ideal. We will use the ideals $\mathcal{I}_d, \mathcal{I}_\delta, \mathcal{I}_u$ related to asymptotic,logarithmic,uniform density,respectively. For those densities for definitions see [9, 10, 12, 13].

(c) For any $q \in (0,1\rangle$ the set $\mathcal{I}_c^{(q)} = \{A \subseteq \mathbb{N} : \sum_{a \in A} a^{-q} < \infty\}$ is an admissible ideal. The ideal $\mathcal{I}_c^{(1)} = \{A \subseteq \mathbb{N} : \sum_{a \in A} a^{-1} < \infty\}$ is usually denoted by $\mathcal{I}_c$. It is easy to see, that for any $q_1 < q_2; q_1, q_2 \in (0,1)$

$$\mathcal{I}_f \subsetneq \mathcal{I}_c^{(q_1)} \subsetneq \mathcal{I}_c^{(q_2)} \subsetneq \mathcal{I}_c \subsetneq \mathcal{I}_d \tag{1.1}$$

In this paper will we study the ideals $\mathcal{I}_c^{(q)}$. In particular the equivalence between $\mathcal{I}_c^{(q)}, \mathcal{I}_c^{(q)*}$, Olivier's like theorems for this ideals and characterization of $\mathcal{I}_c^{(q)}$-convergent sequences by regular matrices.

2. The equivalence between $\mathcal{I}_c^{(q)}$ and $\mathcal{I}_c^{(q)*}$-convergence

Let us recall the notion of $\mathcal{I}$-convergence of sequences of real numbers, (cf.[9, 10]).

Definition 2.1. We say that a sequence $x = (x_n)_{n=1}^{\infty}$ $\mathcal{I}$-converges to a number L and we write $\mathcal{I} - \lim\ x_n = L$, if for each $\varepsilon > 0$ the set $A(\varepsilon) = \{n : |x_n - L| \geq \varepsilon\}$ belongs to the ideal $\mathcal{I}$.

$\mathcal{I}$-convergence satisfies usual axioms of convergence i.e. the uniqueness of limit, arithmetical properties etc. The class of all $\mathcal{I}$-convergent sequences is a linear space. We will also use the following elementary fact.

Lemma 2.2. *Let $\mathcal{I}_1, \mathcal{I}_2$ be admissible ideals such that $\mathcal{I}_1 \subset \mathcal{I}_2$. If $\mathcal{I}_1 - \lim x_n = L$ then $\mathcal{I}_2 - \lim x_n = L$.*

In the papers [9, 10] there was defined yet another type of convergence related to the ideal $\mathcal{I}$.

Definition 2.3. Let $\mathcal{I}$ be an admissible ideal in $\mathbb{N}$. A sequence $x = (x_n)_{n=1}^{\infty}$ of real numbers is said to be $\mathcal{I}^*$-convergent to $L \in \mathbb{R}$ (shortly $\mathcal{I}^* - \lim x_n = L$) if there is a set $H \in \mathcal{I}$, such that for $M = \mathbb{N} \setminus H = \{m_1 < m_2 < \ldots\}$ we have, $\lim\limits_{k \to \infty} x_{m_k} = L$.

It is easy to prove, that for every admissible ideal $\mathcal{I}$ the following relation between $\mathcal{I}$ and $\mathcal{I}^*$-convergence holds:

$$\mathcal{I}^* - \lim x_n = L \Rightarrow \mathcal{I} - \lim x_n = L.$$

Kostyrko, Šalát and Wilczynski in [9] give an algebraic characterization of ideals $\mathcal{I}$, for which the $\mathcal{I}$ and $\mathcal{I}^*$-convergence are equal; it turns out that these ideals are with the property *(AP)*.

Definition 2.4. An admissible ideal $\mathcal{I} \subset 2^{\mathbb{N}}$ is said to satisfy the property *(AP)* if for every countable family of mutually disjoint sets $\{A_1, A_2, \ldots\}$ belonging to $\mathcal{I}$ there exists a countable family of sets $\{B_1, B_2, \ldots\}$ such that $A_j \triangle B_j$ is a finite set for $j \in \mathbb{N}$ and $\bigcup_{j=1}^{\infty} B_j \in \mathcal{I}.(A \triangle B = (A \setminus B) \cup (B \setminus A))$.

For some ideals it is already known whether they have property *(AP)*(see [9, 10, 12, 13]). Now, will show the equivalence between $\mathcal{I}_c^{(q)}$ and $\mathcal{I}_c^{(q)*}$-convergence.

Theorem 2.5. *For any $0 < q \leq 1$ the ideal $\mathcal{I}_c^{(q)}$ has a property (AP).*

Proof. It suffices to prove that any sequences $(x_n)_{n=1}^{\infty}$ of real numbers such that $\mathcal{I}_c^{(q)} - \lim x_n = \xi$ there exist a set $M = \{m_1 < m_2 < \ldots < m_k < \ldots\} \subseteq \mathbb{N}$ such that $\mathbb{N} \setminus M \in \mathcal{I}_c^{(q)}$ and $\lim\limits_{k\to\infty} x_{m_k} = \xi$.

For any positive integer k let $\varepsilon_k = \frac{1}{2^k}$ and $A_k = \{n \in \mathbb{N} : |x_n - \xi| \geq \frac{1}{2^k}\}$. As $\mathcal{I}_c^{(q)} - \lim x_n = \xi$, we have $A_k \in \mathcal{I}_c^{(q)}$, i.e.

$$\sum_{a\in A_k} a^{-q} < \infty.$$

Therefore there exist an infinite sequence $n_1 < n_2 < \ldots < n_k \ldots$ of integers such that for every $k = 1, 2, \ldots$

$$\sum_{\substack{a>n_k \\ a\in A_k}} a^{-q} < \frac{1}{2^k}$$

Let $H = \bigcup_{k=1}^{\infty}[(n_k, n_{k+1}\rangle \cap A_k]$. Then

$$\sum_{a\in H} a^{-q} \leq \sum_{\substack{a>n_1 \\ a\in A_1}} a^{-q} + \sum_{\substack{a>n_2 \\ a\in A_2}} a^{-q} + \ldots + \sum_{\substack{a>n_k \\ a\in A_k}} a^{-q} + \ldots <$$

$$< \frac{1}{2} + \frac{1}{2^2} + \ldots + \frac{1}{2^k} + \ldots < +\infty$$

Thus $H \in \mathcal{I}_c^{(q)}$. Put $M = \mathbb{N} \setminus H = \{m_1 < m_2 < \ldots < m_k < \ldots\}$. Now it suffices to prove that $\lim\limits_{k\to\infty} x_{m_k} = \xi$. Let $\varepsilon > 0$. Choose $k_0 \in \mathbb{N}$ such that $\frac{1}{2^{k_0}} < \varepsilon$. Let $m_k > n_{k_0}$. Then m_k belongs to some interval $(n_j, n_{j+1}\rangle$ where $j \geq k_0$ and doesn't belong to A_j $(j \geq k_0)$. Hence m_k belongs to $\mathbb{N} \setminus A_j$, and then $|x_{m_k} - \xi| < \varepsilon$ for every $m_k > n_{k_0}$, thus $\lim\limits_{k\to\infty} x_{m_k} = \xi$. $\square$

3. Olivier's like theorem for the ideals $\mathcal{I}_c^{(q)}$

In 1827 L. Olivier proved the results about the speed of convergence to zero of the terms of a convergent series with positive and decreasing terms.(cf.[8, 11])

Theorem A. *If $(a_n)_{n=1}^{\infty}$ is a non-increasing sequences and $\sum_{n=1}^{\infty} a_n < +\infty$, then $\lim\limits_{n\to\infty} n \cdot a_n = 0$.*

Simple example $a_n = \frac{1}{n}$ if n is a square i.e. $n = k^2$, $(k = 1, 2, \ldots)$ and $a_n = \frac{1}{2^n}$ otherwise shows that monotonicity condition on the sequence $(a_n)_{n=1}^{\infty}$ can not be in general omitted.

In [14] T.Šalát and V.Toma characterized the class $\mathcal{S}(T)$ of ideals such that

$$\sum_{n=1}^{\infty} a_n < +\infty \Rightarrow \mathcal{I} - \lim_{n\to\infty} n \cdot a_n = 0 \tag{3.1}$$

for any convergent series with positive terms.

Theorem B. *The class $\mathcal{S}(T)$ consists of all admissible ideals $\mathcal{I} \subseteq \mathcal{P}(\mathbb{N})$ such that $\mathcal{I} \supseteq \mathcal{I}_c$.*

From inclusions (1.1) is obvious that ideals $\mathcal{I}_c^{(q)}$ do not belong to the class $\mathcal{S}(T)$.

In what follows we show that it is possible to modify the Olivier's condition $\sum_{n=1}^{\infty} a_n < +\infty$ in such a way that the ideal $\mathcal{I}_c^{(q)}$ will play the role of ideal $\mathcal{I}_c$ in Theorem B.

Lemma 3.1. *Let $0 < q \leq 1$. Then for every sequence $(a_n)_{n=1}^{\infty}$ such that $a_n > 0, n = 1, 2, \dots$ and $\sum_{n=1}^{\infty} {a_n}^q < +\infty$ we have $\mathcal{I}_c^{(q)} - \lim n \cdot a_n = 0$.*

Proof. Let the conclusion of the Lemma 3.1 doesn't hold. Then there exists $\varepsilon_0 > 0$ such that the set $A(\varepsilon_0) = \{n : n \cdot a_n \geq \varepsilon_0\}$ doesn't belong to $\mathcal{I}_c^{(q)}$. Therefore

$$\sum_{k=1}^{\infty} m_k^{-q} = +\infty, \tag{3.2}$$

where $A(\varepsilon_0) = \{m_1 < m_2 < \ldots < m_k < \ldots\}$. By the definition of the set $A(\varepsilon_0)$ we have $m_k \cdot a_{m_k} \geq \varepsilon_0 > 0$, for each $k \in N$. From this $m_k^q \cdot a_{m_k}^q \geq \varepsilon_0^q > 0$ and so for each $k \in N$

$$a_{m_k}^q \geq \varepsilon_0^q \cdot m_k^{-q} \tag{3.3}$$

From (3.2) and (3.3) we get $\sum_{k=1}^{\infty} a_{m_k}^q = +\infty$, and hence $\sum_{n=1}^{\infty} a_n^q = +\infty$. But it contradicts the assumption of the theorem. $\square$

Let's denote by $\mathcal{S}_q(T)$ the class of all admissible ideals $\mathcal{I}$ for which an analog Lemma 3.1 holds. From Lemma 2.2 we have:

Corollary 3.2. *If $\mathcal{I}$ is an admissible ideal such that $\mathcal{I} \supseteq \mathcal{I}_c^{(q)}$ then $\mathcal{I} \in \mathcal{S}_q(T)$.*

Main result of this section is the reverse of Corollary 3.2.

Theorem 3.3. *For any $q \in (0, 1\rangle$ the class $\mathcal{S}_q(T)$ consists of all admissible ideals such that $\mathcal{I} \supseteq \mathcal{I}_c^{(q)}$.*

Proof. It this sufficient to prove that for any infinite set $M = \{m_1 < m_2 < \ldots < m_k < \ldots\} \in \mathcal{I}_c^{(q)}$ we have $M \in \mathcal{I}$, too. Since $M \in \mathcal{I}_c^{(q)}$ we have

$$\sum_{k=1}^{\infty} m_k^{-q} < +\infty.$$

Now we define the sequence $(a_n)_{n=1}^{\infty}$ as follows

$$\begin{aligned} a_{m_k} &= \frac{1}{m_k} \quad (k = 1, 2, \ldots), \\ a_n &= \frac{1}{10^n} \quad \text{for} \quad n \in \mathbb{N} \setminus M. \end{aligned}$$

Obviously $a_n > 0$ and $\sum_{n=1}^{\infty} a_n{}^q < +\infty$ by the definition of numbers a_n. Since $\mathcal{I} \in \mathcal{S}_q(T)$ we have

$$\mathcal{I} - \lim n \cdot a_n = 0.$$

This implies that for each $\varepsilon > 0$ we have

$$A(\varepsilon) = \{n : n \cdot a_n \geq \varepsilon\} \in \mathcal{I},$$

in particular $M = A(1) \in \mathcal{I}$. □

4. $\mathcal{I}_c^{(q)}$-convergence and regular matrix transformations

$\mathcal{I}_c^{(q)}$-convergence is an example of a linear functional defined on a subspace of the space of all bounded sequences of real numbers. Another important family of such functionals are so called matrix summability methods inspired by [1, 6]. We will study connections between $\mathcal{I}_c^{(q)}$-convergence and one class of matrix summability methods. Let us start by introducing a notion of regular matrix transformation (see [4]).

Let $\mathbf{A} = (a_{nk})$ $(n, k = 1, 2, \ldots)$ be an infinite matrix of real numbers. The sequence $(t_n)_{n=1}^{\infty}$ of real numbers is said to be $\mathbf{A}$-limitable to the number s if $\lim\limits_{n\to\infty} s_n = s$, where

$$s_n = \sum_{k=1}^{\infty} a_{nk} t_k \quad (n = 1, 2, \ldots).$$

If $(t_n)_{n=1}^{\infty}$ is $\mathbf{A}$-limitable to the number s, we write $\mathbf{A} - \lim\limits_{n\to\infty} t_n = s$.

We denote by $F(\mathbf{A})$ the set of all $\mathbf{A}$-limitable sequences. The set $F(\mathbf{A})$ is called the convergence field. The method defined by the matrix $\mathbf{A}$ is said *to be regular* provided that $F(\mathbf{A})$ contains all convergent sequences and $\lim\limits_{n\to\infty} t_n = t$ implies $\mathbf{A} - \lim\limits_{n\to\infty} t_n = t$. Then $\mathbf{A}$ is called a *regular matrix.*

It is well-known that the matrix $\mathbf{A}$ is regular if and only if satisfies the following three conditions (see [4]):

(A) $\exists K > 0, \forall n = 1, 2, \ldots \sum_{k=1}^{\infty} |a_{nk}| \leq K$;

(B) $\forall k = 1, 2, \ldots \lim\limits_{n\to\infty} a_{nk} = 0$

(C) $\lim\limits_{n\to\infty} \sum_{k=1}^{\infty} a_{nk} = 1$

Let's ask the question: Is there any connection between $\mathcal{I}$-convergence of sequence of real numbers and **A**-limit of this sequence? It is well know that a sequence $(x_k)_{k=1}^{\infty}$ of real numbers $\mathcal{I}_d$-converges to real number ξ if and only if the sequence is strongly summable to ξ in Caesaro sense. The complete characterization of statistical convergence ($\mathcal{I}_d$-convergence) is described by Fridy-Miller in the paper [6]. They defined a class of lower triangular nonnegative matrices $\mathcal{T}$ with properties:

$$\sum_{k=1}^{n} a_{nk} = 1 \quad \forall n \in \mathbb{N}$$

$$\text{if } \; C \subseteq \mathbb{N} \; \text{ such that } \; d(C) = 0, \quad \text{then } \; \lim_{n\to\infty} \sum_{k\in C} a_{nk} = 0.$$

They proved the following assertion:

Theorem C. *The bounded sequence $x = (x_n)_{n=1}^{\infty}$ is statistically convergent to L if and only if $x = (x_n)_{n=1}^{\infty}$ is $\boldsymbol{A}$-summable to L for every $\boldsymbol{A}$ in $\mathcal{T}$.*

Similar result for $\mathcal{I}_u$-convergence was shown by V. Baláž and T. Šalát in [1]. Here we prove analogous result for $\mathcal{I}_c^{(q)}$-convergence. Following this aim let's define the class $\mathcal{T}_q$ lower triangular nonnegative matrices in this way:

Definition 4.1. Matrix $\mathbf{A} = (a_{nk})$ belongs to the class $\mathcal{T}_q$ if and only if it satisfies the following conditions:

(I) $\lim\limits_{n\to\infty} \sum_{k=1}^{n} a_{nk} = 1$

(q) If $C \subset \mathbb{N}$ and $C \in \mathcal{I}_c^{(q)}$, then $\lim\limits_{n\to\infty} \sum_{k\in C} a_{nk} = 0, \quad 0 < q \leq 1.$

It is easy to see that every matrix of class $\mathcal{T}_q$ is regular. As the following example shows the converse does not hold.

Example 4.2. Let $C = \{1^2, 2^2, 3^2, 4^2, \ldots, n^2, \ldots\}$ and $q = 1$. Obviously $C \in \mathcal{I}_c^{(1)} = \mathcal{I}_c$. Now define the matrix **A** by:

$$a_{11} = 1, a_{1k} = 0, \quad k > 1$$

$$a_{nk} = \frac{1}{2k \cdot \ln n}, \quad k \neq l^2, k \leq n$$

$$a_{nk} = \frac{1}{l \ln n}, \quad k = l^2, k \leq n$$

$$a_{nk} = 0, \quad k > n$$

It is easy to show that **A** is lower triangular nonnegative regular matrix but does not satisfy the condition (q) from Definition 4.1.

$$\sum_{\substack{k<n^2 \\ k\in C}} a_{n^2k} = \frac{1}{\ln n^2}(1 + \frac{1}{2} + \ldots + \frac{1}{n}) \geq \frac{\ln n}{2\ln n} = \frac{1}{2} \nrightarrow 0$$

for $n \to \infty$. Therefore $\mathbf{A} \notin \mathcal{T}_1$.

Lemma 4.3. *If the bounded sequence $x = (x_n)_{n=1}^{\infty}$ is not $\mathcal{I}$-convergent then there exist real numbers $\lambda < \mu$ such that neither the set $\{n \in \mathbb{N} : x_n < \lambda\}$ nor the set $\{n \in \mathbb{N} : x_n > \mu\}$ belongs to ideal $\mathcal{I}$.*

As the proof is the same as the proof on Lemma in [6] we will omit it.

Next theorem shows connection between $\mathcal{I}_c^{(q)}$-convergence of bounded sequence of real numbers and $\mathbf{A}$-summability of this sequence for matrices from the class $\mathcal{T}_q$.

Theorem 4.4. *Let $q \in (0, 1\rangle$. Then the bounded sequence $x = (x_n)_{n=1}^{\infty}$ of real numbers $\mathcal{I}_c^{(q)}$-converges to $L \in \mathbb{R}$ if and only if it is $\boldsymbol{A}$-summable to $L \in \mathbb{R}$ for each matrix $\boldsymbol{A} \in \mathcal{T}_q$.*

Proof. Let $\mathcal{I}_c^{(q)} - \lim x_n = L$ and $\mathbf{A} \in \mathcal{T}_q$. As $\mathbf{A}$ is regular there exists a $K \in \mathbb{R}$ such that $\forall n = 1, 2, \dots \sum_{k=1}^{\infty} |a_{nk}| \le K$.

It is sufficient to show that $\lim\limits_{n\to\infty} b_n = 0$, where $b_n = \sum_{k=1}^{\infty} a_{nk}.(x_k - L)$. For $\varepsilon > 0$ put $B(\varepsilon) = \{k \in \mathbb{N} : |x_k - L| \ge \varepsilon\}$. By the assumption we have $B(\varepsilon) \in \mathcal{I}_c^{(q)}$. By condition (q) from Definition 4.1 we have

$$\lim_{n\to\infty} \sum_{k\in B(\varepsilon)} |a_{nk}| = 0 \tag{4.1}$$

As the sequence $x = (x_n)_{n=1}^{\infty}$ is bounded, there exists $M > 0$ such that

$$\forall k = 1, 2, \dots : |x_k - L| \le M \tag{4.2}$$

Let $\varepsilon > 0$. Then

$$\begin{aligned}
|b_n| &\le \sum_{k\in B(\frac{\varepsilon}{2K})} |a_{nk}||x_k - L| \;+ \sum_{k\notin B(\frac{\varepsilon}{2K})} |a_{nk}||x_k - L| \le \\
&\le M \sum_{k\in B(\frac{\varepsilon}{2K})} |a_{nk}| \;+\; \frac{\varepsilon}{2K} \sum_{k\notin B(\frac{\varepsilon}{2K})} |a_{nk}| \le \\
&\le M \sum_{k\in B(\frac{\varepsilon}{2K})} |a_{nk}| \;+\; \frac{\varepsilon}{2}
\end{aligned} \tag{4.3}$$

By part (q) of Definition 4.1 there exists an integer n_0 such that for all $n > n_0$

$$\sum_{k\in B(\frac{\varepsilon}{2K})} |a_{nk}| < \frac{\varepsilon}{2M}$$

Together by (4.3) we obtain $\lim\limits_{n\to\infty} b_n = 0$.

Conversely, suppose that $\mathcal{I}_c^{(q)} - \lim x_n = L$ doesn't hold. We show that there exists a matrix $\mathbf{A} \in \mathcal{T}_q$ such that $\mathbf{A} - \lim\limits_{n\to\infty} x_n = L$ does not hold, too. If $\mathcal{I}_c^{(q)} -$

$\lim x_n = L^{'} \neq L$ then from the firs part of proof it follows that $\mathbf{A} - \lim\limits_{n\to\infty} x_n = L^{'} \neq L$ for any $\mathbf{A} \in \mathcal{T}_q$. Thus, we may assume that $(x_n)_{n=1}^{\infty}$ is not $\mathcal{I}_c^{(q)}$-convergent, and by the above Lemma 4.3 there exist λ and μ ($\lambda < \mu$), such that neither the set $U = \{k \in \mathbb{N} : x_k < \lambda\}$ nor $V = \{k \in \mathbb{N} : x_k > \mu\}$ belongs to the ideal $\mathcal{I}_c^{(q)}$. It is clear that $U \cap V = \emptyset$. If $U \notin \mathcal{I}_c^{(q)}$ then $\sum_{i\in U} i^{-q} = +\infty$ and if $V \notin \mathcal{I}_c^{(q)}$ then $\sum_{i\in V} i^{-q} = +\infty$. Let $U_n = U \cap \{1, 2, \ldots, n\}$ and $V_n = V \cap \{1, 2, \ldots, n\}$.

Now we define the matrix $\mathbf{A} = (a_{nk})$ by the following way: Let $s_{(1)n} = \sum_{i\in U_n} i^{-q}$ for $n \in U$, $s_{(2)n} = \sum_{i\in V_n} i^{-q}$ for $n \in V$ and $s_{(3)n} = \sum_{i=1}^{n} i^{-q}$ for $n \notin U \cap V$. As $U, V \notin \mathcal{I}_c^{(q)}$ we have $\lim\limits_{n\to\infty} s_{(j)n} = +\infty, j = 1, 2, 3.$

$$a_{nk} = \begin{cases} a_{nk} = \frac{k^{-q}}{s_{(1)n}} & n \in U \text{ and } k \in U_n, \\ a_{nk} = 0 & n \in U \text{ and } k \notin U_n, \\ a_{nk} = \frac{k^{-q}}{s_{(2)n}} & n \in V \text{ and } k \in V_n, \\ a_{nk} = 0 & n \in V \text{ and } k \notin V_n, \\ a_{nk} = \frac{k^{-q}}{s_{(3)n}} & n \notin U \cap V, \\ a_{nk} = 0 & k > n, \end{cases}$$

Let's check that $\mathbf{A} \in \mathcal{T}_q$. Obviously $\mathbf{A}$ is a lower triangular nonnegative matrix. Condition (I) is clear from the definition of matrix $\mathbf{A}$. Condition (q): Let $B \in \mathcal{I}_c^{(q)}$ and $b = \sum_{k\in B} k^{-q} < +\infty$. Then

$$\sum_{k\in B} a_{nk} \leq \frac{1}{s_{(3)n}} \sum_{k\in B\cap\{1,\ldots,n\}} k^{-q}\chi_B(k) \leq \frac{b}{s_{(3)n}} \to 0$$

for $n \to \infty$. Thus $\mathbf{A} \in \mathcal{T}_q$.

For $n \in U$

$$\sum_{k=1}^{\infty} a_{nk}x_k = \frac{1}{s_{(1)n}} \sum_{k=1}^{n} k^{-q}\chi_U(k)x_k < \frac{\lambda}{s_{(1)n}} \sum_{k=1}^{n} k^{-q}\chi_U(k) = \lambda$$

on other hand for $n \in V$

$$\sum_{k=1}^{\infty} a_{nk}x_k = \frac{1}{s_{(2)n}} \sum_{k=1}^{n} k^{-q}\chi_V(k)x_k > \frac{\mu}{s_{(2)n}} \sum_{k=1}^{n} k^{-q}\chi_V(k) = \mu.$$

Therefore $\mathbf{A} - \lim\limits_{n\to\infty} x_n$ does not exist. □

Corollary 4.5. *If* $0 < q_1 < q_2 \leq 1$, *then* $\mathcal{T}_{q_2} \subsetneq \mathcal{T}_{q_1}$.

Proof. Let $B \in \mathcal{I}_c^{(q_2)} \setminus \mathcal{I}_c^{(q_1)}$ and let $(x_n) = \chi_B(n)$, $n = 1, 2, \ldots$ Clearly $\mathcal{I}_c^{(q_2)} - \lim x_n = 0$ and $\mathcal{I}_c^{(q_1)} - \lim x_n$ does not exist. Let $\mathbf{A}$ be the matrix constructed from the sequence $(x_n)_{n=1}^{\infty}$ as in the proof of Theorem 4.4. In particular $\mathbf{A} \in \mathcal{T}_{q_1}$ and $\mathbf{A} - \lim\limits_{n\to\infty} x_n$ does not exist. Therefore $\mathbf{A} \notin \mathcal{T}_{q_2}$. □

Further we show some type well-known matrix which fulfills condition (I). Let $(p_j)_{j=1}^{\infty}$ be the sequence of positive real numbers. Put $P_n = p_1 + p_2 + \ldots + p_n$.
Now we define matrix $\mathbf{A} = (a_{nk})$ in this way:

$$a_{nk} = \frac{p_k}{P_n} \quad k \leq n$$

$$a_{nk} = 0 \quad k > n.$$

This type of matrix is called Riesz matrix.

Especially we put $p_n = n^\alpha$, where $0 < \alpha < 1$. Then

$$a_{nk} = \frac{k^\alpha}{1^\alpha + 2^\alpha + \ldots + n^\alpha} \quad k \leq n$$

$$a_{nk} = 0 \quad k > n.$$

This special class of matrix we denote by $(\mathbf{R}, n^\alpha)$. It is clear that this matrix fulfills conditions (I) and (q). For this class of matrix is true following implication:

$$\mathcal{I}_c^{(q)} - \lim x_k = L \Rightarrow (\mathbf{R}, n^\alpha) - \lim x_k = L$$

where $(x_k)_{k=1}^{\infty}$ is a bounded sequence, $0 < q \leq 1$, $0 < \alpha < 1$. Converse does not hold. It is sufficient to choose the characteristic function of the set of all primes $\mathbb{P}$. Then $(\mathbf{R}, n^\alpha) - \lim x_k = 0$, but $\mathcal{I}_c^{(q)} - \lim x_k$ does not exist, because $\sum_{n\in\mathbb{P}} n^{-q} = +\infty$, where $\mathbb{P}$ is a se of all primes. Hence the class $(\mathbf{R}, n^\alpha)$ of matrices belongs to $\mathcal{T} \setminus \mathcal{T}_q$.

Problem 4.6. If we take any admissible ideal $\mathcal{I}$ and define the class $\mathcal{T}_{\mathcal{I}}$ of matrices by replacing the condition (I) in Definition 4.1 by condition:if $C \subset \mathbb{N}$ and $C \in \mathcal{I}$, $\mathcal{I}$ admissible ideal on $\mathbb{N}$ then $\lim\limits_{n\to\infty} \sum_{k\in C} |a_{nk}| = 0$ then it is easy to see that the if part of Theorem 4.4 holds for $\mathcal{I}$ too. The question is what about only if part.

References

[1] Baláž, V. - Šalát, T.: Uniform density u and corresponding I_u convergence, *Math. Communications* **11** (2006), 1–7.

[2] Brown,T.C. - Freedman,A.R.: The uniform density of sets of integers and Fermat's Last Theorem, *C.R. Math. Rep. Acad. Sci. Canada* **XII** (1990), 1–6.

[3] Buck,R.C.: The measure theoretic approach to density, *Amer. J. Math.* **68** (1946), 560–580.

[4] Cooke, C.: Infinite matrices and sequences spaces, *Moskva* (1960).

[5] Fast, H.: Sur la convergence statistique, *Colloq. Math.* **2** (1951), 241–244.

[6] Fridy,J.A.-Miller,H.I.: A matrix characterization of statistikal convergence *Analysis* **11** (1986), 59–66.

[7] Gogola, J. - Červeňanský, J.: $\mathcal{I}_c^{(q)}$-convergence of real numbers, *Zborník vedeckých prác MtF STU v Trnave* **18** (2005), 15–18

[8] Knopp, K: Theorie ubd Anwendung der unendlichen Reisen, *Berlin* (1931).

[9] Kostyrko,P. - Šalát, T. - Wilczyński W.: $\mathcal{I}$-convergence, *Real Anal. Exchange* **26** (2000-2001), 669–686.

[10] Kostyrko, P. - Mačaj, M. - Šalát, T. - Sleziak, M.: $\mathcal{I}$-convergence and extremal $\mathcal{I}$-limit poits, *Mathematica Slovaca* **55** (2005), No. 4, 443–464.

[11] Olivier, L: Remarques sur les series infinies et lem convergence, *J. reine angew. Math* **2** (1827), 31–44.

[12] Pašteka, M. - Šalát, T. - Visnyai, T.: Remark on Buck's measure density and generalization of asymptotic density, *Tatra Mt. Math. Publ.* **31** (2005), 87–101.

[13] Sember,J.J. - Freedman,A.R.: On summing sequences of 0's and 1's, *Rocky Mount J. Math* **11** (1981), 419–425.

[14] Šalát, T. - Toma, V.: A classical Olivier's theorem and statistical convergence, *Annales Math. Blaise Pascal* **1** (2001), 10–18.

[15] Šalát, T. - Visnyai,T.: Subadditive measures on $\mathbb{N}$ and the convergence of series with positive terms, *Acta Math.(Nitra)* **6** (2003), 43–52.

Annales Mathematicae et Informaticae
38 *(2011) pp. 37–44*
http://ami.ektf.hu

A convexity test for control point based curves*

Imre Juhász

Department of Descriptive Geometry, University of Miskolc, Egyetemváros Miskolc, Hungary
agtji@uni-miskolc.hu

Submitted November 15, 2011 Accepted December 06, 2011

Abstract

We study the convexity of curves defined by the combination of control points and blending functions, that are globally controlled. We provide a method using which the convexity of the curve can be determined by the location of one of its control points.

Keywords: convex curve, convexity test, singularity, inflection point, self-intersection, cusp

MSC: 65D17, 68U07

1. Introduction

Convexity of curves is an important concept in Computer Aided Geometric Design. We adopt the following definition of convex curves (see e.g. [4]).

Definition 1.1. A curve is convex if it is (a part of) the boundary of a convex plane figure.

One can find other approaches, such as

- A (directed) plane curve is convex if it is on the same side of its (directed) tangents (cf. [3]).

*This research was carried out as a part of the TAMOP-4.2.1.B-10/2/KONV-2010-0001 project with support by the European Union, co-financed by the European Social Fund.

- A curve is convex, if it is intersected by any hyperplane in at most two points or it lies completely in the hyperplane. (c.f. [5], [1]) Therefore, convex curves are plane curves (lie in two-dimensional planes). This is a bit more restrictive than Definition 1.1, since excludes curves that contain straight line segments.

In [3] there is a comprehensive study on the convexity of directed parametric curves. According to that treatment the same plane curve can be convex or concave depending on its direction.

Based on the more traditional Definition 1.1 we study the global convexity of control point based curves

$$\mathbf{g}(u) = \sum_{j=0}^{n} F_j(u)\, \mathbf{d}_j, u \in [a, b], \tag{1.1}$$

where functions $\{F_i(u)\}_{i=0}^{n}$ are assumed to be at least twice continuously differentiable.

Applying the moving control point concept, we propose a method that provides both a visual aid for interactive convex curve design and a simple convexity check algorithm for all curves defined by the combination of control points and blending functions. In comparison with the already published results, the proposed method is rather intuitive and easy to implement and use.

2. Singularity

We briefly summarize those results of [2] that we will utilize in the sequel. We let control point $\mathbf{d}_i$, $i \in \{0, 1, \ldots, n\}$ vary and fix the rest. Separating the fixed and varying parts of (1.1) we obtain

$$\mathbf{g}(u) = F_i(u)\, \mathbf{d}_i + \mathbf{r}_i(u), \ \mathbf{r}_i(u) = \sum_{j=0, j\neq i}^{n} F_j(u)\, \mathbf{d}_j. \tag{2.1}$$

We assume that control point $\mathbf{d}_i$ has influence on the shape of the whole curve, i.e. we suppose that the curve is globally controlled. Consequently, the proposed method is not suitable for spline curves, i.e. for locally controlled curves.

Curve

$$\mathbf{c}_i(u) = -\frac{\dot{\mathbf{r}}_i(u)}{\dot{F}_i(u)} \tag{2.2}$$

is called the ith discriminant of curve (1.1).

Conditions for singularity are as follows.

- The locus of control point $\mathbf{d}_i$ that results a cusp on the curve (1.1) is the ith discriminant (2.2).

- The locus of control point $\mathbf{d}_i$ that results a zero curvature point on the curve (1.1) is the region in the plane that is covered by the tangent lines of the ith discriminant (2.2).

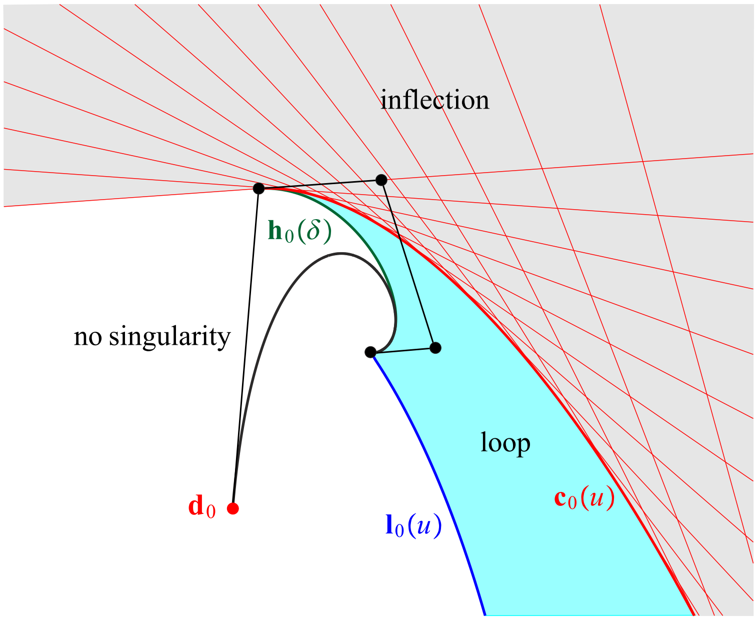


Figure 1: Singularity regions of a quartic Bézier curve with respect to its control point $\mathbf{d}_0$.

- The locus of control point $\mathbf{d}_i$ that results a self-intersection point on the curve (1.1) is the triangular region in the plane bounded by the curves

$$\mathbf{l}_i(u) = -\frac{\mathbf{r}_i(b) - \mathbf{r}_i(u)}{F_i(b) - F_i(u)}, u \in [a,b], \tag{2.3}$$

$$\mathbf{h}_i(\delta) = -\frac{\mathbf{r}_i(a+\delta) - \mathbf{r}_i(a)}{F_i(a+\delta) - F_i(a)}, \delta \in (0, b-a]$$

  and by the ith discriminant (2.2).

These singularities are illustrated in Fig. 1 for a quartic Bézier curve with respect to its control point $\mathbf{d}_0$.

3. Convexity

As is known, if the curve (1.1) shares the variation diminishing property, then any convex control polygon results a convex curve. However, the converse is not true in general.

We assume that curve (1.1) has no inflection point, cusp and self-intersection point, i.e. it is free of singularity. Under these circumstances, closed curves ($\mathbf{g}(a) = \mathbf{g}(b)$) are convex. In case of open curves, however this is not necessarily true, the two possible types of counterexamples can be seen in Fig. 2.

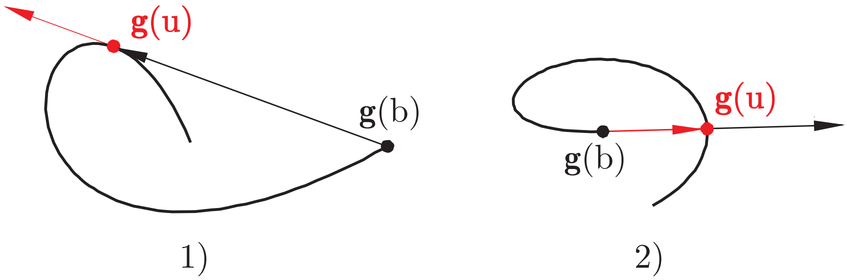


Figure 2: Singularity free nonconvex open curves.

3.1. Case 1

The left hand side figure of Figure 2 shows that case when a tangent line can be drawn from the endpoint $\mathbf{g}(b)$ to the curve, which means that $\exists u \in [a,b)$ for which

$$\dot{\mathbf{g}}(u) \times (\mathbf{g}(u) - \mathbf{g}(b)) = \mathbf{0},$$

i.e., $\exists \lambda \in \mathbb{R}$ such that

$$\lambda \dot{\mathbf{g}}(u) = \mathbf{g}(u) - \mathbf{g}(b).$$

Substituting (2.1) we obtain

$$\mathbf{d}_i = \frac{\mathbf{r}_i(u) - \lambda \dot{\mathbf{r}}_i(u) - \mathbf{r}_i(b)}{\lambda \dot{F}_i(u) + F_i(b) - F_i(u)},$$

which is the parametric form of a straight line (with parameter λ). The $\lambda = 0$ point of this line is

$$\frac{\mathbf{r}_i(u) - \mathbf{r}_i(b)}{F_i(b) - F_i(u)}$$

that is on the curve (2.3), and the $\lambda \to \infty$ point (which is a singularity of the parametrization)

$$-\frac{\dot{\mathbf{r}}_i(u)}{\dot{F}_i(u)}$$

is on the discriminant (2.2). The direction vector of this line is

$$(\mathbf{r}_i(u) - \mathbf{r}_i(b)) \dot{F}_i(u) + \dot{\mathbf{r}}_i(u) (F_i(b) - F_i(u)),$$

therefore this line is the tangent line of curve (2.3) at $u \in [a,b)$.

Thus, the locus of control point $\mathbf{d}_i$ that results case 1 is the plane region covered by the tangent lines of the curve (2.3).

3.2. Case 2

The right hand side figure of Figure 2 illustrates the case when the tangent line at the endpoint $\mathbf{g}(b)$ intersects the curve, which means that $\exists u \in [a, b)$ for which

$$\dot{\mathbf{g}}(b) \times (\mathbf{g}(u) - \mathbf{g}(b)) = \mathbf{0},$$

i.e., $\exists \lambda \in \mathbb{R}$

$$\lambda \dot{\mathbf{g}}(b) = \mathbf{g}(u) - \mathbf{g}(b).$$

After the substitution (2.1) we obtain

$$\mathbf{d}_i = \frac{\mathbf{r}_i(u) - \lambda \dot{\mathbf{r}}_i(b) - \mathbf{r}_i(b)}{\lambda \dot{F}_i(b) + F_i(b) - F_i(u)}$$

which is the parametric form of a line, with parameter λ. The $\lambda = 0$ point of this line is

$$\frac{\mathbf{r}_i(u) - \mathbf{r}_i(b)}{F_i(b) - F_i(u)},$$

which is on the curve (2.3), and the $\lambda \to \infty$ point (which is a singularity of the parametrization)

$$-\frac{\dot{\mathbf{r}}_i(b)}{\dot{F}_i(b)}$$

is on the discriminant (2.2). The direction vector of this line is

$$\mathbf{q}_i(u) = (\mathbf{r}_i(u) - \mathbf{r}_i(b)) \dot{F}_i(b) + \dot{\mathbf{r}}_i(b) (F_i(b) - F_i(u)). \tag{3.1}$$

Therefore, the locus of control point $\mathbf{d}_i$ that results case 2 is the plane region covered by straight lines that are parallel to the direction (3.1) and pass through the points of curve (2.3).

4. Convexity test

It is obvious, that if a curve has a self-intersection point then it is possible to draw a tangent line from its endpoint $\mathbf{g}(b)$ to the curve such that the point of contact differs from the endpoint itself, or the tangent line at the endpoint intersects the curve. Therefore, self-intersecting curves fall into Case 1 or 2 above.

As a consequence of this, the following questions have to be answered in order to determine convexity.

1. Does the curve (1.1) have a cusp or an inflection point? It comprises the following steps:

    - Is the control point $\mathbf{d}_i$ on the discriminant (2.2)?
    - Is it possible to draw a tangent line from control point $\mathbf{d}_i$ to the discriminant (2.2)? If the answer is yes, does the curvature change sign in the neighborhood of the corresponding point on the curve (1.1)?

2. Is it possible to draw a tangent line from control point $\mathbf{d}_i$ to the curve (2.3)? (Case 1)

3. Is there a straight line parallel to the direction (3.1) that passes through the corresponding point of curve (2.3)? (Case 2)

If all answers are negative then the curve is convex, otherwise it is concave. The corresponding equations are

$$(\mathbf{d}_i - \mathbf{c}_i(u)) \times \dot{\mathbf{c}}_i(u) = \mathbf{0},\ u \in (a,b), \tag{4.1}$$

$$(\mathbf{d}_i - \mathbf{l}_i(u)) \times \dot{\mathbf{l}}_i(u) = \mathbf{0},\ u \in (a,b), \tag{4.2}$$

$$(\mathbf{d}_i - \mathbf{l}_i(u)) \times \mathbf{q}_i(u) = \mathbf{0},\ u \in (a,b), \tag{4.3}$$

respectively. Since, the curve is planar, we can assume that it is in the x, y coordinate plane, therefore only the third component of the cross products above may differ from zero. Thus, equations (4.1–4.3) can be reduced to scalar equations, i.e. we have to find zeros of functions of a single variable.

Actually, in cases (4.2) and (4.3) we do not need the zeros themselves, only their existence is of interest. For the determination of their existence it is enough to bracket the zero of the function, i.e. to find two values in the domain the corresponding function values to which have different sign. Bracketing is also used as a pre-processor to zero finding methods like bisection, secant or false position (c.f. [6]).

In case of inflection point, however we have to find the zeros themselves, since equality (4.1) guarantees only a vanishing curvature. A point with zero curvature is an inflection point if the curvature changes its sign in the neighborhood of the point.

This test works also for closed curves, and it is easier to answer question 2 than to check self-intersection by means of the loop region described in Section 2.

5. Implementation

In principle, we can use any control point of the curve for the convexity test but the usage of $\mathbf{d}_0$ seems to be the best choice in several cases, especially curves with endpoint interpolation, such as the Bézier curve and its various extensions and generalizations. (It is explained in more detail in [2].) In case of Bézier curves basis functions are the Bernstein polynomials, i.e.

$$F_j(u) = B_j^n(u) = \binom{n}{j} u^j (1-u)^{n-j}, (j = 0, 1, \ldots, n)$$

with $a = 0, b = 1$. We describe the consideration above for the control point $\mathbf{d}_0$. In this case

$$\mathbf{r}_0(u) = \sum_{j=1}^{n} B_j^n(u)\, \mathbf{d}_j,$$

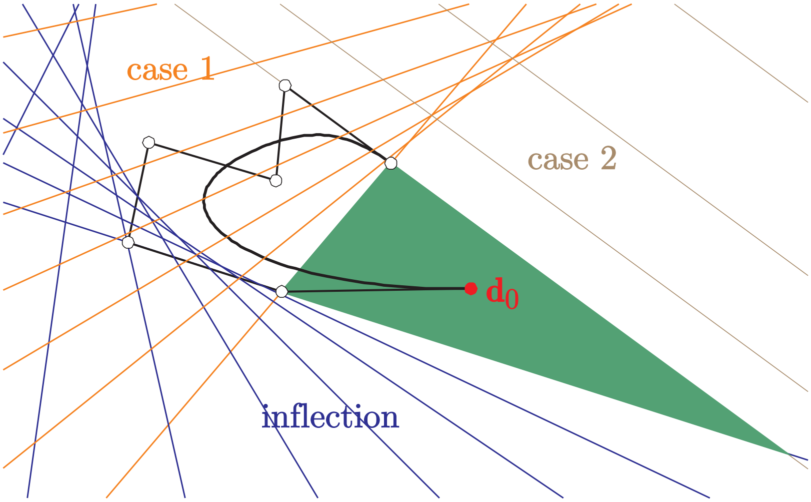


Figure 3: A convex Bézier curve with concave control polygon. While control point $\mathbf{d}_0$ is in the green region the curve remains convex.

curve (2.3) becomes

$$\mathbf{l}_0(u) = -\frac{\mathbf{r}_0(1) - \mathbf{r}_0(u)}{B_0^n(1) - B_0^n(u)} = \frac{\mathbf{d}_n - \mathbf{r}_0(u)}{B_0^n(u)}$$

and direction $\mathbf{q}_0(u)$ is

$$\begin{aligned}\mathbf{q}_0(u) &= (\mathbf{r}_0(u) - \mathbf{r}_0(1))\,\dot{B}_0^n(1) + \dot{\mathbf{r}}_0(1)\,(B_0^n(1) - B_0^n(u)) \\ &= -nB_0^n(u)\,(\mathbf{d}_n - \mathbf{d}_{n-1}),\end{aligned}$$

i.e. $\mathbf{q}_0(u)$ is parallel to the direction $\mathbf{d}_n - \mathbf{d}_{n-1}$ for any permissible value of u. Fig. 3 illustrates the different regions that are used for the convexity test for a Bézier curve of degree 6.

References

[1] Farin, G., *Curves and surfaces for CAGD: A practical guide*, 4th ed., Academic Press, San Diego, 1999.

[2] Juhász, I., On the singularity of a class of parametric curves, *Computer Aided Geometric Design*, 23 (2006) 146–156.

[3] Liu, C., Traas, C. R., On convexity of planar curves and its application in CAGD, *Computer Aided Geometric Design*, 14 (1997) 653–669 .

[4] Pottmann, H., Wallner, J., *Computational line geometry*, Springer-Verlag, Berlin, 2001.

[5] Prautzsch, H., Boehm, W., Paluszny, M., *Bézier and B-spline techniques*, Springer-Verlag, Berlin, 2002.

[6] Ralston, A., Rabinowitz, P., *A first course in numerical analysis*, 2nd ed., McGraw-Hill, New York, 1978.

Annales Mathematicae et Informaticae
38 *(2011) pp. 45–57*
http://ami.ektf.hu

The Roman (k, k)-domatic number of a graph*

A.P. Kazemi[a], S.M. Sheikholeslami[bc], L. Volkmann[d]

[a]Department of Mathematics
University of Mohaghegh Ardabili, Ardabil, Iran
e-mail: adelpkazemi@yahoo.com

[b]Department of Mathematics
Azarbaijan University of Tarbiat Moallem, Tabriz, Islamic Republic of Iran
e-mail: s.m.sheikholeslami@azaruniv.edu

[c]School of Mathematics
Institute for Research in Fundamental Sciences (IPM), Tehran, Islamic Republic of Iran

[d]Lehrstuhl II für Mathematik
RWTH-Aachen University, Aachen, Germany
e-mail: volkm@math2.rwth-aachen.de

Submitted February 21, 2011 Accepted April 19, 2011

Abstract

Let k be a positive integer. A *Roman k-dominating function* on a graph G is a labelling $f : V(G) \longrightarrow \{0, 1, 2\}$ such that every vertex with label 0 has at least k neighbors with label 2. A set $\{f_1, f_2, \ldots, f_d\}$ of distinct Roman k-dominating functions on G with the property that $\sum_{i=1}^{d} f_i(v) \leq 2k$ for each $v \in V(G)$, is called a *Roman (k, k)-dominating family* (of functions) on G. The maximum number of functions in a Roman (k, k)-dominating family on G is the *Roman (k, k)-domatic number* of G, denoted by $d_R^k(G)$. Note that the Roman $(1, 1)$-domatic number $d_R^1(G)$ is the usual Roman domatic number $d_R(G)$. In this paper we initiate the study of the Roman (k, k)-domatic number in graphs and we present sharp bounds for $d_R^k(G)$. In addition, we determine the Roman (k, k)-domatic number of some graphs. Some of our results extend those given by Sheikholeslami and Volkmann in 2010 for the

*This research was in part supported by a grant from IPM (No. 90050043).

Roman domatic number.

Keywords: Roman domination number, Roman domatic number, Roman k-domination number, Roman (k, k)-domatic number.

MSC: 05C69

1. Introduction

In this paper, G is a simple graph with vertex set $V = V(G)$ and edge set $E = E(G)$. The order $|V|$ of G is denoted by $n = n(G)$. For every vertex $v \in V$, the *open neighborhood* $N(v)$ is the set $\{u \in V(G) \mid uv \in E(G)\}$ and the *closed neighborhood* of v is the set $N[v] = N(v) \cup \{v\}$. The *degree* of a vertex $v \in V(G)$ is $\deg_G(v) = \deg(v) = |N(v)|$. The *minimum* and *maximum degree* of a graph G are denoted by $\delta = \delta(G)$ and $\Delta = \Delta(G)$, respectively. The *open neighborhood* of a set $S \subseteq V$ is the set $N(S) = \cup_{v \in S} N(v)$, and the *closed neighborhood* of S is the set $N[S] = N(S) \cup S$. The complement of a graph G is denoted by $\overline{G}$. We write K_n for the *complete graph* of order n and C_n for a *cycle* of length n. Consult [4, 15] for the notation and terminology which are not defined here.

Let k be a positive integer. A subset S of vertices of G is a *k-dominating set* if $|N_G(v) \cap S| \geq k$ for every $v \in V(G) - S$. The *k-domination number* $\gamma_k(G)$ is the minimum cardinality of a k-dominating set of G. A *k-domatic partition* is a partition of V into k-dominating sets, and the *k-domatic number* $d_k(G)$ is the largest number of sets in a k-domatic partition. The k-domatic number was introduced by Zelinka [16]. Further results on the k-domatic number can be found in the paper [5] by Kämmerling and Volkmann. For a good survey on the domatic numbers in graphs we refer the reader to [1]. Recently more domatic parameters are studied (see for instance [10, 11, 12]).

Let $k \geq 1$ be an integer. Following Kämmerling and Volkmann [6], a *Roman k-dominating function* (briefly RkDF) on a graph G is a labelling $f : V(G) \to \{0, 1, 2\}$ such that every vertex with label 0 has at least k neighbors with label 2. The *weight* of a Roman k-dominating function is the value $f(V(G)) = \sum_{v \in V(G)} f(v)$. The minimum weight of a Roman k-dominating function on a graph G is called the *Roman k-domination number*, denoted by $\gamma_{kR}(G)$. Note that the Roman 1-domination number $\gamma_{1R}(G)$ is the usual Roman domination number $\gamma_R(G)$. A *$\gamma_{kR}(G)$-function* is a Roman k-dominating function of G with weight $\gamma_{kR}(G)$. A Roman k-dominating function $f : V \to \{0, 1, 2\}$ can be represented by the ordered partition (V_0, V_1, V_2) (or (V_0^f, V_1^f, V_2^f) to refer to f) of V, where $V_i = \{v \in V \mid f(v) = i\}$. In this representation, its weight is $\omega(f) = |V_1| + 2|V_2|$. Since $V_1^f \cup V_2^f$ is a k-dominating set when f is an RkDF, and since placing weight 2 at the vertices of a k-dominating set yields an RkDF, in [6], it was observed that

$$\gamma_k(G) \leq \gamma_{kR}(G) \leq 2\gamma_k(G). \tag{1.1}$$

A set $\{f_1, f_2, \ldots, f_d\}$ of distinct Roman k-dominating functions on G with the property that $\sum_{i=1}^{d} f_i(v) \leq 2k$ for each $v \in V(G)$ is called a *Roman (k, k)-*

dominating family (of functions) on G. The maximum number of functions in a Roman (k,k)-dominating family (briefly R(k,k)D family) on G is the *Roman* (k,k)-*domatic number* of G, denoted by $d_R^k(G)$. The Roman (k,k)-domatic number is well-defined and

$$d_R^k(G) \geq 1 \tag{1.2}$$

for all graphs G since the set consisting of any RkDF forms an R(k,k)D family on G and if $k \geq 2$, then

$$d_R^k(G) \geq 2 \tag{1.3}$$

since the functions $f_i : V(G) \to \{0,1,2\}$ defined by $f_i(v) = i$ for each $v \in V(G)$ and $i = 1,2$ forms an R(k,k)D family on G of order 2. In the special case when $k = 1$, $d_R^1(G)$ is the Roman domatic number $d_R(G)$ investigated in [8] and has been studied in [9].

The definition of the Roman dominating function was given implicitly by Stewart [14] and ReVelle and Rosing [7]. Cockayne et al. [3] as well as Chambers et al. [2] have given a lot of results on Roman domination.

Our purpose in this paper is to initiate the study of the Roman (k,k)-domatic number in graphs. We first study basic properties and bounds for the Roman (k,k)-domatic number of a graph. In addition, we determine the Roman (k,k)-domatic number of some classes of graphs.

The next known results are useful for our investigations.

Proposition A (Kämmerling, Volkmann [6] 2009)**.** *Let $k \geq 1$ be an integer, and let G be a graph of order n. If $n \leq 2k$, then $\gamma_{kR}(G) = n$. If $n \geq 2k+1$, then $\gamma_{kR}(G) \geq 2k$.*

Proposition B (Kämmerling, Volkmann [6] 2009)**.** *Let G be a graph of order n. Then $\gamma_{kR}(G) < n$ if and only if G contains a bipartite subgraph H with bipartition X, Y such that $|X| > |Y| \geq k$ and $\deg_H(v) \geq k$ for each $v \in X$.*

Proposition C (Kämmerling, Volkmann [6] 2009)**.** *If G is a graph of order n and maximum degree $\Delta \geq k$, then*

$$\gamma_{kR}(G) \geq \left\lceil \frac{2n}{\frac{\Delta}{k}+1} \right\rceil.$$

Proposition D (Sheikholeslami, Volkmann [8] 2010)**.** *If G is a graph, then*

$$d_R(G) = 1$$

if and only if G is empty.

Proposition E (Sheikholeslami, Volkmann [8] 2010)**.** *If G is a graph of order $n \geq 2$, then $d_R(G) = n$ if and only if G is the complete graph on n vertices.*

Proposition F (Sheikholeslami, Volkmann [8] 2010)**.** *Let K_n be the complete graph of order $n \geq 1$. Then $d_R(K_n) = n$.*

Proposition G (Sheikholeslami, Volkmann [13]). *Let $K_{p,q}$ be the complete bipartite graph of order $p+q$ such that $q \geq p \geq 1$. Then $\gamma_{kR}(K_{p,q}) = p+q$ when $p < k$ or $q = p = k$, $\gamma_{kR}(K_{p,q}) = k+p$ when $p+q \geq 2k+1$ and $k \leq p \leq 3k$ and $\gamma_{kR}(K_{p,q}) = 4k$ when $p \geq 3k$.*

We start with the following observations and properties. The first observation is an immediate consequence of (1.3) and Proposition D.

Observation 1.1. *If G is a graph, then $d_R^k(G) = 1$ if and only if $k = 1$ and G is empty.*

Observation 1.2. *If G is a graph and $k \geq 2$ is an integer, then $d_R^k(G) = 2$ if and only if G is trivial.*

Proof. If G is trivial, then obviously $d_R^k(G) = 2$. Now let G be nontrivial and let $v \in V(G)$. Define $f, g, h : V(G) \to \{0,1,2\}$ by

$$f(v) = 1 \text{ and } f(x) = 2 \text{ if } x \in V(G) - \{v\},$$

$$g(v) = 2 \text{ and } g(x) = 1 \text{ if } x \in V(G) - \{v\},$$

and

$$h(x) = 1 \text{ if } x \in V(G).$$

It is clear that $\{f, g, h\}$ is an R(k,k)D family of G and hence $d_R^k(G) \geq 3$. This completes the proof. □

Observation 1.3. *If G is a graph and $k \geq \Delta(G)+1$ is an integer, then $d_R^k(G) \leq 2k-1$.*

Proof. If $d_R^k(G) = 1$, then the statement is trivial. Let $d_R^k(G) \geq 2$. Since $k \geq \Delta(G)+1$, we have $\gamma_{kR}(G) = n$. Let $\{f_1, f_2, \ldots, f_d\}$ be an R(k,k)D family on G such that $d = d_R^k(G)$. Since $f_1, f_2, \ldots, f_d$ are distinct, we may assume $f_i(v) = 2$ for some i and some $v \in V(G)$. It follows from $\sum_{j=1}^{d} f_j(v) \leq 2k$ that $\sum_{j \neq i} f_j(v) \leq 2k-2$. Thus $d-1 \leq 2k-2$ as desired. □

Observation 1.4. *If $k \geq 2$ is an integer, and G is a graph of order $n \geq 2k-2$, then $d_R^k(G) \geq 2k-1$.*

Proof. If $V(G) = \{v_1, v_2, \ldots, v_n\}$, then define $f_j : V(G) \to \{0,1,2\}$ by $f_j(v_j) = 2$ and $f_j(x) = 1$ for $x \in V(G) - \{v_j\}$ and $1 \leq j \leq 2k-2$ and $f_{2k-1} : V(G) \to \{0,1,2\}$ by $f_{2k-1}(x) = 1$ for each $x \in V(G)$. Then $f_1, f_2, \ldots, f_{2k-1}$ are distinct with $\sum_{i=1}^{2k-1} f_i(x) = 2k$ for each $x \in \{v_1, v_2, \ldots, v_{2k-2}\}$ and $\sum_{i=1}^{2k-1} f_i(x) = 2k-1$ otherwise. Therefore $\{f_1, f_2, \ldots, f_{2k-1}\}$ is an R(k,k)D family on G, and thus $d_R^k(G) \geq 2k-1$. □

The last two observations lead to the next result immediately.

Corollary 1.5. *Let $k \geq 2$ be an integer. If G is a graph of order $n \geq 2k-2$ and $k \geq \Delta(G)+1$, then $d_R^k(G) = 2k-1$.*

Observation 1.6. *If $k \geq 3$ is an integer, and G is a graph of order $n \geq 2k-4$, then $d_R^k(G) \geq 2k-2$.*

Proof. If $V(G) = \{v_1, v_2, \ldots, v_n\}$, then define $f_j : V(G) \to \{0,1,2\}$ by $f_j(v_j) = 2$ and $f_j(x) = 1$ for $x \in V(G) - \{v_j\}$ and $1 \leq j \leq 2k-4$, $f_{2k-3} : V(G) \to \{0,1,2\}$ by $f_{2k-3}(x) = 1$ for each $x \in V(G)$ and $f_{2k-2} : V(G) \to \{0,1,2\}$ by $f_{2k-2}(x) = 2$ for each $x \in V(G)$. Then $f_1, f_2, \ldots, f_{2k-2}$ are distinct with $\sum_{i=1}^{2k-2} f_i(x) = 2k$ for each $x \in V(G)$. Therefore $\{f_1, f_2, \ldots, f_{2k-2}\}$ is an R(k,k)D family on G, and thus $d_R^k(G) \geq 2k-2$. □

Observation 1.7. *Let $k \geq 2$ be an integer. If G is a graph of order $n \leq 2k-3$ and $k \geq \Delta(G)+1$, then $d_R^k(G) \leq 2k-2$.*

Proof. If $n = 1$, then $d_R^k(G) = 2 \leq 2k-2$. Assume now that $n \geq 2$. Let $\{f_1, f_2, \ldots, f_d\}$ be an R(k,k)D family on G such that $d = d_R^k(G)$. Since $k \geq \Delta(G)+1$, we observe that $f_i(x) \geq 1$ for each $1 \leq i \leq d$ and each $x \in V(G)$. Suppose to the contrary that $d \geq 2k-1$. Since $f_1, f_2, \ldots, f_d$ are distinct, there exists a vertex $u \in V(G)$ such that $f_s(u) = f_t(u) = 2$ for two indices $s, t \in \{1, 2, \ldots, d\}$ with $s \neq t$. However, this leads to

$$\sum_{i=1}^{d} f_i(u) \geq \sum_{i=1}^{2k-1} f_i(u) \geq 4 + 2k - 3 = 2k+1,$$

a contradiction. Therefore $d_R^k(G) \leq 2k-2$, and the proof is complete. □

Theorem 1.8. *Let $k \geq 1$ be an integer, and let G be a graph of order n. If $k \geq 3 \cdot 2^{n-2}$, then $d_R^k(G) = 2^n$.*

Proof. Let $\{f_1, f_2, \ldots, f_d\}$ be the set of all pairwise distinct functions from $V(G)$ into the set $\{1,2\}$. Then f_i is a Roman k-dominating function on G for $1 \leq i \leq d$, and it is well-known that $d = 2^n$. The hypothesis $k \geq 3 \cdot 2^{n-2}$ leads to

$$\sum_{i=1}^{d} f_i(v) = \sum_{i=1}^{2^n} f_i(v) = 2^{n-1} + 2^n = 3 \cdot 2^{n-1} \leq 2k$$

for each vertex $v \in V(G)$. Therefore $\{f_1, f_2, \ldots, f_d\}$ is an R(k,k)D family on G and thus $d_R^k(G) \geq 2^n$.

Now let $f : V(G) \longrightarrow \{0,1,2\}$ be a Roman k-dominating function on G. Since $k \geq 3 \cdot 2^{n-2} > n > \Delta(G)$, it is impossible that $f(x) = 0$ for any vertex $x \in V(G)$. Hence the number of Roman k-dominating functions on G is at most 2^n and so $d_R^k(G) \leq 2^n$. This yields the desired identity. □

Observation 1.9. *If $k \geq 1$ is an integer, then $\gamma_{kR}(K_n) = \min\{n, 2k\}$.*

Proof. If $n \leq 2k$, then Proposition A implies that $\gamma_{kR}(K_n) = n$.

Assume now that $n \geq 2k+1$. It follows from Proposition A that $\gamma_{kR}(K_n) \geq 2k$. Let $V(K_n) = \{v_1, v_2, \ldots, v_n\}$, and define $f : V(K_n) \to \{0,1,2\}$ by $f(v_1) = f(v_2) = \ldots = f(v_k) = 2$ and $f(v_j) = 0$ for $k+1 \leq j \leq n$. Then f is an RkDF on K_n of weight $2k$ and thus $\gamma_{kR}(K_n) \leq 2k$, and the proof is complete. □

2. Properties of the Roman $(\mathbf{k}, \mathbf{k})$-domatic number

In this section we present basic properties of $d_R^k(G)$ and sharp bounds on the Roman (k,k)-domatic number of a graph.

Theorem 2.1. *Let G be a graph of order n with Roman k-domination number $\gamma_{kR}(G)$ and Roman (k,k)-domatic number $d_R^k(G)$. Then*

$$\gamma_{kR}(G) \cdot d_R^k(G) \leq 2kn.$$

Moreover, if $\gamma_{kR}(G) \cdot d_R^k(G) = 2kn$, then for each $R(k,k)D$ family $\{f_1, f_2, \ldots, f_d\}$ on G with $d = d_R^k(G)$, each function f_i is a $\gamma_{kR}(G)$-function and $\sum_{i=1}^{d} f_i(v) = 2k$ for all $v \in V$.

Proof. Let $\{f_1, f_2, \ldots, f_d\}$ be an R(k,k)D family on G such that $d = d_R^k(G)$ and let $v \in V$. Then

$$\begin{aligned} d \cdot \gamma_{kR}(G) &= \sum_{i=1}^{d} \gamma_{kR}(G) \\ &\leq \sum_{i=1}^{d} \sum_{v \in V} f_i(v) \\ &= \sum_{v \in V} \sum_{i=1}^{d} f_i(v) \\ &\leq \sum_{v \in V} 2k \\ &= 2kn. \end{aligned}$$

If $\gamma_{kR}(G) \cdot d_R^k(G) = 2kn$, then the two inequalities occurring in the proof become equalities. Hence for the R(k,k)D family $\{f_1, f_2, \ldots, f_d\}$ on G and for each i, $\sum_{v \in V} f_i(v) = \gamma_{kR}(G)$, thus each function f_i is a $\gamma_{kR}(G)$-function, and $\sum_{i=1}^{d} f_i(v) = 2k$ for all $v \in V$. □

Theorem 2.2. *Let G be a graph of order $n \geq 2$ and $k \geq 1$ be an integer. Then $\gamma_{kR}(G) = n$ and $d_R^k(G) = 2k$ if and only if G does not contain a bipartite subgraph H with bipartition X, Y such that $|X| > |Y| \geq k$ and $\deg_H(v) \geq k$ for each $v \in X$ and G has $2k$ or $2k-1$ connected bipartite subgraphs $H_i = (X_i, Y_i)$ with $|X_i| = |Y_i|$, $\deg_{H_i}(v) \geq k$ for each $v \in X_i$ and $|\{i \mid u \in Y_i\}| = |\{i \mid u \in X_i\}| = k$ for each $u \in V(G)$.*

Proof. Let $\gamma_{kR}(G)=n$ and $d_R^k(G)=2k$. It follows from Proposition B that G does not contain a bipartite subgraph H with bipartition X,Y such that $|X|>|Y|\geq k$ and $\deg_H(v)\geq k$ for each $v\in X$. Let $\{f_1,\ldots,f_{2k}\}$ be a Roman (k,k)-dominating family on G. By Theorem 2.1, $\gamma_{kR}(G)=\omega(f_i)=n$ for each i. First suppose for each i, there exists a vertex x such that $f_i(x)\neq 1$. Assume that H_i is a subgraph of G with vertex set $V_0^{f_i}\cup V_2^{f_i}$ and edge set $E(V_0^{f_i},V_2^{f_i})$. Since $\omega(f_i)=n$ and f_i is a Roman k-dominating function, $|V_2^{f_i}|=|V_0^{f_i}|$ and $\deg_{H_i}(v)\geq k$ for each $v\in V_0^{f_i}$. By Theorem 2.1, $\sum_{i=1}^{2k} f_i(v)=2k$ for each $v\in V(G)$ which implies that $|\{i\mid v\in V_2^{f_i}\}|=|\{i\mid v\in V_0^{f_i}\}|=k$ for each $v\in V(G)$. Now suppose $f_i(x)=1$ for each $x\in V(G)$ and some i, say $i=2k$. Define the bipartite subgraphs H_i for $1\leq i\leq 2k-1$ as above.

Conversely, assume that G does not contain a bipartite subgraph H with bipartition X,Y such that $|X|>|Y|\geq k$ and $\deg_H(v)\geq k$ for each $v\in X$ and G has $2k$ or $2k-1$ connected bipartite subgraphs $H_i=(X_i,Y_i)$ with $|X_i|=|Y_i|$ and $\deg_{H_i}(v)\geq k$ for each $v\in X_i$. Then by Proposition B, $\gamma_{kR}(G)=n$. If G has $2k$ connected bipartite subgraphs H_i, then the mappings $f_i:V(G)\to\{0,1,2\}$ defined by

$$f_i(u)=2 \text{ if } u\in Y_i,\ f_i(v)=0 \text{ if } v\in X_i, \text{and } f_i(x)=1 \text{ for each } x\in V-(X_i\cup Y_i)$$

are Roman k-dominating functions on G and $\{f_i\mid 1\leq i\leq 2k\}$ is a Roman (k,k)-dominating family on G. If G has $2k-1$ connected bipartite subgraphs H_i, then the mappings $f_i,g:V(G)\to\{0,1,2\}$ defined by $g(x)=1$ for each $x\in V(G)$ and

$$f_i(u)=2 \text{ if } u\in Y_i,\ f_i(v)=0 \text{ if } v\in X_i, \text{and } f_i(x)=1 \text{ for each } x\in V-(X_i\cup Y_i)$$

are Roman k-dominating functions on G and $\{g,f_i\mid 1\leq i\leq 2k-1\}$ is a Roman (k,k)-dominating family on G.

Thus $d_R^k(G)\geq 2k$. It follows from Theorem 2.1 that $d_R^k(G)=2k$, and the proof is complete. □

The next corollary is an immediate consequence of Proposition C, Observation 1.3 and Theorem 2.1.

Corollary 2.3. *For every graph G of order n, $d_R^k(G)\leq\max\{\Delta,k-1\}+k$.*

Let $A_1\cup A_2\cup\ldots\cup A_d$ be a k-domatic partition of $V(G)$ into k-dominating sets such that $d=d_k(G)$. Then the set of functions $\{f_1,f_2,\ldots,f_d\}$ with $f_i(v)=2$ if $v\in A_i$ and $f_i(v)=0$ otherwise for $1\leq i\leq d$ is an R(k,k)D family on G. This shows that $d_k(G)\leq d_R^k(G)$ for every graph G. Since $\gamma_{kR}(G)\geq\min\{n,\gamma_k(G)+k\}$ (cf. [6]), for each graph G of order $n\geq 2$, Theorem 2.1 implies that $d_R^k(G)\leq\frac{2kn}{\min\{n,\gamma_k(G)+k\}}$. Combining these two observations, we obtain the following result.

Corollary 2.4. *For any graph G of order n,*

$$d_k(G)\leq d_R^k(G)\leq\frac{2kn}{\min\{n,\gamma_k(G)+k\}}.$$

Theorem 2.5. *Let K_n be the complete graph of order n and k a positive integer. Then $d_R^k(K_n) = n$ if $n \geq 2k$, $d_R^k(K_n) \leq 2k-1$ if $n \leq 2k-1$ and $d_R^k(K_n) = 2k-1$ if $k \geq 2$ and $2k-2 \leq n \leq 2k-1$.*

Proof. By Proposition F, we may assume that $k \geq 2$. Assume that $V(K_n) = \{x_1, x_2, ..., x_n\}$. First let $n \geq 2k$. Since Observation 1.9 implies that $\gamma_{kR}(K_n) = 2k$, it follows from Theorem 2.1 that $d_R^k(K_n) \leq n$. For $1 \leq i \leq n$, define now $f_i : V(K_n) \to \{0, 1, 2\}$ by

$$f_i(x_i) = f_i(x_{i+1}) = \ldots = f_i(x_{i+k-1}) = 2 \text{ and } f_i(x) = 0 \text{ otherwise},$$

where the indices are taken modulo n. It is easy to see that $\{f_1, f_2, \ldots, f_n\}$ is an R$(k,k)D$ family on G and hence $d_R^k(K_n) \geq n$. Thus $d_R^k(K_n) = n$.

Now let $n \leq 2k-1$. Then Observation 1.9 yields $\gamma_{kR}(K_n) = n$, and it follows from Theorem 2.1 that $d_R^k(K_n) \leq 2k$. Suppose to the contrary that $d_R^k(K_n) = 2k$. Then by Theorem 2.1, each Roman k-dominating function f_i in any R(k,k)D family $\{f_1, f_2, , \ldots, f_{2k}\}$ on G is a $\gamma_{kR}(G)$-function. This implies that $f_i(x) = 1$ for each $x \in V(K_n)$. Hence $f_1 \equiv f_2 \equiv \cdots \equiv f_{2k}$ which is a contradiction. Thus $d_R^k(K_n) \leq 2k-1$.

In the special case $k \geq 2$ and $2k-2 \leq n \leq 2k-1$, Observation 1.4 shows that $d_R^k(K_n) \geq 2k-1$ and so $d_R^k(K_n) = 2k-1$. □

In view of Proposition G and Theorem 2.1 we obtain the next upper bounds for the Roman (k,k)-domatic number of complete bipartite graphs.

Corollary 2.6. *Let $K_{p,q}$ be the complete bipartite graph of order $p+q$ such that $q \geq p \geq 1$, and let k be a positive integer. Then $d_R^k(K_{p,q}) \leq 2k$ if $p < k$ or $q = p = k$, $d_R^k(K_{p,q}) \leq \frac{2k(p+q)}{k+p}$ if $p+q \geq 2k+1$ and $k \leq p \leq 3k$ and $d_R^k(K_{p,q}) \leq \frac{p+q}{2}$ if $p \geq 3k$.*

For some special cases of complete bipartite graphs, we can prove more.

Corollary 2.7. *Let $K_{p,p}$ be the complete bipartite graph of order $2p$, and let k be a positive integer. If $p \geq 3k$, then $d_R^k(K_{p,p}) = p$. If $p < k$, then $d_R^k(K_{p,p}) \leq 2k-1$. In particular, if $p = k-1$, then $d_R^k(K_{p,p}) = 2k-1$, and if $p = k-2$, then $d_R^k(K_{p,p}) = 2k-2$.*

Proof. Assume first that $p \geq 3k$. Let $X = \{u_1, u_2, \ldots, u_p\}$ and $Y = \{v_1, v_2, \ldots, v_p\}$ be the partite sets of the complete bipartite graph $K_{p,p}$. For $1 \leq i \leq p$, define $f_i : V(K_{p,p}) \to \{0, 1, 2\}$ by

$$f_i(u_i) = f_i(u_{i+1}) = \ldots = f_i(u_{i+k-1}) = f_i(v_i) = f_i(v_{i+1}) = \ldots = f_i(v_{i+k-1}) = 2$$

and $f_i(x) = 0$ otherwise, where the indices are taken modulo p. It is a simple matter to verify that $\{f_1, f_2, \ldots, f_p\}$ is an R$(k,k)D$ family on $K_{p,p}$ and hence $d_R^k(K_{p,p}) \geq p$. Using Corollary 2.6 for $p = q \geq 3k$, we obtain $d_R^k(K_{p,p}) = p$.

Assume next that $p < k$. Since $k > p = \Delta(K_{p,p})$, it follows from Observation 1.3 that $d_R^k(K_{p,p}) \leq 2k-1$.

Assume now that $p = k-1$. Then $k \geq 2$ and $n(K_{p,p}) = 2k-2$, and we deduce from Observation 1.4 that $d_R^k(K_{p,p}) \geq 2k-1$ and so $d_R^k(K_{p,p}) = 2k-1$.

Finally, assume that $p = k-2$. Then $k \geq 3$ and $n(K_{p,p}) = 2k-4$. It follows from Observation 1.6 that $d_R^k(K_{p,p}) \geq 2k-2$ and from Observation 1.7 that $d_R^k(K_{p,p}) \leq 2k-2$ and thus $d_R^k(K_{p,p}) = 2k-2$. □

Theorem 2.8. *If G is a graph of order $n \geq 2$, then*

$$\gamma_{kR}(G) + d_R^k(G) \leq n + 2k \tag{2.1}$$

with equality if and only if $\gamma_{kR}(G) = n$ and $d_R^k(G) = 2k$ or $\gamma_{kR}(G) = 2k$ and $d_R^k(G) = n$.

Proof. If $d_R^k(G) \leq 2k-1$, then obviously $\gamma_{kR}(G) + d_R^k(G) \leq n + 2k - 1$. Let now $d_R^k(G) \geq 2k$. If $\gamma_{kR}(G) \geq 2k$, Theorem 2.1 implies that $d_R^k(G) \leq n$. According to Theorem 2.1, we obtain

$$\gamma_{kR}(G) + d_R^k(G) \leq \frac{2kn}{d_R^k(G)} + d_R^k(G). \tag{2.2}$$

Using the fact that the function $g(x) = x + (2kn)/x$ is decreasing for $2k \leq x \leq \sqrt{2kn}$ and increasing for $\sqrt{2kn} \leq x \leq n$, this inequality leads to the desired bound immediately.

Now let $\gamma_{kR}(G) \leq 2k-1$. Since $\min\{n, \gamma_k(G) + k\} \leq \gamma_{kR}(G)$, we deduce that $\gamma_{kR}(G) = n$. According to Theorem 2.1, we obtain $d_R^k(G) \leq 2k$ and hence $d_R^k(G) = 2k$. Thus

$$\gamma_{kR}(G) + d_R^k(G) = n + 2k.$$

If $\gamma_{kR}(G) = n$ and $d_R^k(G) = 2k$ or $\gamma_{kR}(G) = 2k$ and $d_R^k(G) = n$, then obviously $\gamma_{kR}(G) + d_R^k(G) = n + 2k$.

Conversely, let equality hold in (2.1). It follows from (2.2) that

$$n + 2k = \gamma_{kR}(G) + d_R^k(G) \leq \frac{2kn}{d_R^k(G)} + d_R^k(G) \leq n + 2k,$$

which implies that $\gamma_{kR}(G) = \frac{2kn}{d_R^k(G)}$ and $d_R^k(G) = 2k$ or $d_R^k(G) = n$. This completes the proof. □

The special case $k = 1$ of the next result can be found in [8].

Theorem 2.9. *For every graph G and positive integer k,*

$$d_R^k(G) \leq \delta(G) + 2k.$$

Moreover, the upper bound is sharp.

Proof. If $d_R^k(G) \leq 2k$, the result is immediate. Let now $d_R^k(G) \geq 2k+1$ and let $\{f_1, f_2, \ldots, f_d\}$ be an R(k,k)D family on G such that $d = d_R^k(G)$. Assume that v is a vertex of minimum degree $\delta(G)$. Let ℓ be the number of sums $\sum_{u \in N[v]} f_i(u) = 1$ and let m be the number of those sums in which $\sum_{u \in N[v]} f_i(u) = 2$. Obviously, $l + 2m \leq 2k$.

We may assume, without loss of generality, that the equality $\sum_{u \in N[v]} f_i(u) = 1$ holds for $i = 1, \ldots, \ell$, if any, and the equality $\sum_{u \in N[v]} f_i(u) = 2$ holds for $i = \ell+1, \ldots, \ell+m$ when $m \geq 1$. In this case $f_i(v) = 1$ and $f_i(u) = 0$ for each $u \in N(v)$ and $i = 1, \ldots, \ell$ and $f_i(v) = 2$ and $f_i(u) = 0$ for each $u \in N(v)$ and $i = \ell+1, \ldots, \ell+m$. Thus $f_i(v) = 0$ for $\ell+m+1 \leq i \leq d$, and thus $\sum_{u \in N[v]} f_i(u) \geq 2k$ for $\ell+m+1 \leq i \leq d$. Altogether we obtain

$$\begin{aligned}
2k(d-(\ell+m)) + \ell + 2m &\leq \sum_{i=1}^{d} \sum_{u \in N[v]} f_i(u) \\
&= \sum_{u \in N[v]} \sum_{i=1}^{d} f_i(u) \\
&\leq \sum_{u \in N[v]} 2k \\
&= 2k(\delta(G)+1).
\end{aligned}$$

If $m = 0$, then the above inequality chain leads to

$$d \leq \delta(G) + 1 + \ell - \ell/(2k).$$

Since the function $g(x) = x + x/(2k)$ is increasing for $0 \leq x \leq 2k$, we deduce the desired bound as follows

$$d \leq \delta(G) + 1 + \ell - \ell/(2k) \leq \delta(G) + 1 + 2k - (2k)/(2k) = \delta(G) + 2k.$$

Now let $m \geq 1$. Then we obtain

$$d \leq \delta(G) + (\ell+m) + \frac{2k - \ell - 2m}{2k}.$$

Since the last fraction in the sum is a rational number in $[0,1]$ and since $m \geq 1$, we deduce that

$$d \leq \delta(G) + (\ell+m) + \frac{2k-\ell-2m}{2k} \leq \delta(G) + (\ell+m) + 1 \leq \delta(G) + \ell + 2m \leq \delta(G) + 2k$$

as desired.

To prove the sharpness of this inequality, let G_i be a copy of $K_{k^3+(2k+1)k}$ with vertex set $V(G_i) = \{v_1^i, v_2^i, \ldots, v_{k^3+(2k+1)k}^i\}$ for $1 \leq i \leq k$ and let the graph G be obtained from $\cup_{i=1}^k G_i$ by adding a new vertex v and joining v to each $v_1^i, \ldots, v_k^i$.

Define the Roman k-dominating functions f_i^s, h_l for $1 \le i \le k, 0 \le s \le k-1$ and $1 \le l \le 2k$ as follows:

$$f_i^s(v_1^i) = \cdots = f_i^s(v_k^i) = 2,\ f_i^s(v^j_{(i-1)k^2+(s+1)k+1}) = \cdots = f_i^s(v^j_{(i-1)k^2+(s+1)k+k}) = 2$$

$$\text{if } j \in \{1,2,\ldots,k\} - \{i\} \text{ and } f_i^s(x) = 0 \text{ otherwise}$$

and for $1 \le l \le 2k$,

$$h_l(v) = 1, h_l(v^i_{k^3+lk+1}) = \ldots = h_l(v^i_{k^3+lk+k}) = 2\ (1 \le i \le k),$$
$$\text{and } h_l(x) = 0 \text{ otherwise.}$$

It is easy to see that f_i^s and g_l are Roman k-dominating function on G for each $1 \le i \le k, 0 \le s \le k-1, 1 \le l \le 2k$ and $\{f_i^s, g_l \mid 1 \le i \le k, 0 \le s \le k-1 \text{ and } 1 \le l \le 2k\}$ is a Roman (k,k)-dominating family on G. Since $\delta(G) = k^2$, we have $d_R^k(G) = \delta(G) + 2k$. □

For regular graphs the following improvement of Theorem 2.9 is valid.

Theorem 2.10. *Let k be a positive integer. If G is a $\delta(G)$-regular graph, then*

$$d_R^k(G) \le \max\{2k-1, \delta(G)+k\} \le \delta(G) + 2k - 1.$$

Proof. If $k > \Delta(G) = \delta(G)$ then by Observation 1.7, $d_R^k(G) \le 2k-1$ and the desired bound is proved. If $k \le \Delta(G)$, then it follows from Corollary 2.3 that

$$d_R^k(G) \le \delta(G) + k,$$

and the proof is complete. □

As an application of Theorems 2.9 and 2.10, we will prove the following Nordhaus-Gaddum type result.

Theorem 2.11. *Let $k \ge 1$ be an integer. If G is a graph of order n, then*

$$d_R^k(G) + d_R^k(\overline{G}) \le n + 4k - 2, \tag{2.3}$$

with equality only for graphs with $\Delta(G) - \delta(G) = 1$.

Proof. It follows from Theorem 2.9 that

$$d_R^k(G) + d_R^k(\overline{G}) \le (\delta(G)+2k) + (\delta(\overline{G})+2k) = (\delta(G)+2k) + (n - \Delta(G) - 1 + 2k).$$

If G is not regular, then $\Delta(G) - \delta(G) \ge 1$, and hence this inequality implies the desired bound $d_R^k(G) + d_R^k(\overline{G}) \le n + 4k - 2$. If G is $\delta(G)$-regular, then we deduce from Theorem 2.10 that

$$d_R^k(G) + d_R^k(\overline{G}) \le (\delta(G)+2k-1) + (\delta(\overline{G})+2k-1) = n + 4k - 3,$$

and the proof of the Nordhaus-Gaddum bound (2.3) is complete. Furthermore, the proof shows that we have equality in (2.3) only when $\Delta(G) - \delta(G) = 1$. □

Corollary 2.12 ([8]). *For every graph G of order n,*

$$d_R(G) + d_R(\overline{G}) \leq n + 2,$$

with equality only for graphs with $\Delta(G) = \delta(G) + 1$.

For regular graphs we prove the following Nordhaus-Gaddum inequality.

Theorem 2.13. *Let $k \geq 1$ be an integer. If G is a δ-regular graph of order n, then*

$$d_R^k(G) + d_R^k(\overline{G}) \leq \max\{4k-2, n+2k-1, n+3k-2-\delta, 3k+\delta-1\}. \tag{2.4}$$

Proof. Let $\delta(G) = \delta$ and $\delta(\overline{G}) = \overline{\delta}$. We distinguish four cases.

If $k \geq \delta + 1$ and $k \geq \overline{\delta} + 1$, then it follows from Observation 1.7 that

$$d_R^k(G) + d_R^k(\overline{G}) \leq (2k-1) + (2k-1) = 4k - 2.$$

If $k \leq \delta$ and $k \leq \overline{\delta}$, then Corollary 2.3 implies that

$$d_R^k(G) + d_R^k(\overline{G}) \leq (\delta + k) + (\overline{\delta} + k) = \delta + 2k + n - 1 - \delta = n + 2k - 1.$$

If $k \geq \delta + 1$ and $k \leq \overline{\delta}$, then we deduce from Observation 1.7 and Corollary 2.3 that

$$d_R^k(G) + d_R^k(\overline{G}) \leq (2k-1) + (\overline{\delta} + k) = 3k - 1 + n - 1 - \delta = n + 3k - 2 - \delta.$$

If $k \leq \delta$ and $k \geq \overline{\delta} + 1$, then Observation 1.7 and Corollary 2.3 lead to

$$d_R^k(G) + d_R^k(\overline{G}) \leq (\delta + k) + (2k-1) = 3k + \delta - 1.$$

This completes the proof. □

If G is a δ-regular graph of order $n \geq 2$, then Theorem 2.13 leads to the following improvement of Theorem 2.11 for $k \geq 2$.

Corollary 2.14. *Let $k \geq 2$ be an integer. If G is a δ-regular graph of order $n \geq 2$, then*

$$d_R^k(G) + d_R^k(\overline{G}) \leq n + 4k - 4.$$

References

[1] Bouchemakh, I., Ouatiki, S., Survey on the domatic number of a graph, Manuscript.

[2] Chambers, E. W., Kinnersley, B., Prince, N., West, D. B., Extremal problems for Roman domination, *SIAM J. Discrete Math.*, 23 (2009) 1575-1586.

[3] Cockayne, E. J., Dreyer Jr., P. M., Hedetniemi, S. M., Hedetniemi, S. T., On Roman domination in graphs, *Discrete Math.*, 278 (2004) 11-22.

[4] Haynes, T. W., Hedetniemi, S. T., Slater, P. J., Fundamentals of Domination in graphs, *Marcel Dekker, Inc., New york*, 1998.

[5] Kämmerling, K., Volkmann, L., The k-domatic number of a graph, *Czech. Math. J.*, 59 (2009) 539-550.

[6] Kämmerling, K., Volkmann, L., Roman k-domination in graphs, *J. Korean Math. Soc.*, 46 (2009) 1309-1318.

[7] Revelle, C. S., Rosing, K. E., Defendens imperium romanum: a classical problem in military strategy, *Amer. Math. Monthly*, 107 (2000) 585–594.

[8] Sheikholeslami, S. M., Volkmann, L., The Roman domatic number of a graph, *Appl. Math. Lett.*, 23 (2010) 1295–1300.

[9] Sheikholeslami, S. M., Volkmann, L., The Roman k-domatic number of a graph, *Acta Math. Sin. (Engl. Ser.)*, 27 (2011) 1899–1906.

[10] Sheikholeslami, S. M., Volkmann, L., Signed (k, k)-domatic number of a graph, *Ann. Math. Inform.*, 37 (2010) 139–149.

[11] Sheikholeslami, S. M., Volkmann, L., The k-rainbow domatic number of a graph, *Discuss. Math. Graph Theory*, (to appear)

[12] Sheikholeslami, S. M., Volkmann, L., The k-tuple total domatic number of a graph, *Util. Math.*, (to appear)

[13] Sheikholeslami, S. M., Volkmann, L., On the Roman k-bondage number of a graph, AKCE Int. J. Graphs Comb., 8 (2011), (to appear).

[14] Stewart, I., Defend the Roman Empire, *Sci. Amer.*, 281 (1999) 136–139.

[15] West, D. B., Introduction to Graph Theory, *Prentice-Hall, Inc*, 2000.

[16] Zelinka, B., On k-ply domatic numbers of graphs, *Math. Slovaka*, 34 (1984) 313–318.

Annales Mathematicae et Informaticae
38 *(2011) pp. 59–74*
http://ami.ektf.hu

Metrics based optimization of functional source code*

[a]**Roland Király**, [b]**Róbert Kitlei**

[a]Institute of Mathematics and Informatics, Eszterházy Károly College
e-mail: kiraly.roland@inf.elte.hu

[b]Eötvös Loránd University, Faculty of Informatics, Department of Programming Languages and Compilers
e-mail: kitlei@elte.hu

Submitted November 22, 2011 Accepted December 21, 2011

Abstract

In order to control software development, usually a set of criteria is fixed, among other things defining limits for the size of modules and functions, guiding layout principles etc. These criteria are not always observed, especially if the criteria are specified after pieces of the code are already written – that is, handling legacy code.

In this paper, we describe a method how the code base can semi automatically improved to conform more to the development criteria. We define a usage of a query language with which the user can employ our software complexity metrics to identify the out-of-line code parts, and select a transformation strategy that are automatically used by the tool to improve the identified parts.

Keywords: erlang, refactoring, structural complexity metrics, metric, functional language

1. Introduction

Measuring metrics in order to assist software development is not a new idea. In his seminal paper, Thomas J. McCabe [10] reasoned about the importance of source code measurement. He was investigating how programs can be modularised in order to decrease the costs of testing.

*Supported by KMOP-1.1.2-08/1-2008-0002 and Ericsson Hungary

The application of complexity and other kinds of metrics yield interesting results if we use them to measure large projects. Currently, we have measured the development of *RefactorErl* [1, 2] itself, comparing various versions. Since *RefactorErl* underwent a big change about one year ago, gaining a new layer with well-designed interfaces, and some refactorings were greatly simplified with much tighter connections to the interface modules, it was expected that the values of some metrics would change substantially.

Table 1 shows the measurement results before and after the changes in the analyzed code body. The rows labelled *max* show the three largest values.

RefactorErl	Before upgrade	After upgrade
Effective line of code (sum)	14518	32366
Effective line of code (avg)	308.89	425.868
Effective line of code (max)	812/701/745	8041/1022/770
Number of functions (sum)	1329	2648
Number of functions (avg)	17.038	21.7
Number of functions (max)	85/70/49	551/91/64
Max depth of cases (avg)	1.7	1.6
Max depth of cases (max)	4	4
Branches of recursion (sum)	201	750
Branches of recursion (avg)	6.931	18.75
Branches of recursion (max)	20/18/15	211/43/35
Num. of function clauses (sum)	1725	6778
Num. of function clauses (avg)	36.70	89.18
Num. of function clauses (max)	139/133/53	3251/303/165
Number of fun-expression (sum)	185	271
Number of fun-expression (avg)	5.138	4.839
Number of fun-expression (max)	20/18/17	31/26/22
Number of funpath (sum)	3911	10854
Number of funpath (avg)	83.21	142.81
Number of funpath (max)	261/224/196	3752/399/333

Table 1: Values of metrics in two versions of RefactorErl

Refactorings make use of the new interface layer, which abstracts away a lot of code. Since these parts of the code are removed from the code of the refactorings, the modules of the refactorings have become smaller in size and complexity. Conversely, the number of connections between these modules and the query and interface modules have increased.

The complexity of the code has decreased in many parts, increased in other parts, but it is clear from the results that the complexity of the source code has increased. Another factor that indicates an increase in complexity is that loading the tool using takes an order of magnitude longer than before.

On the other hand, if we observe which metrics have increased and which ones have decreased, we can realize a correlation between the change in complexity metrics and the rise of load time, namely the time spent on syntactic and semantic analysis.

Measurements of this type can help solve several optimization problems, which are connected to the duration of loading and that of analysis.

Knowing the relation among modules, the chains of function calls and the depth of call chains help us show which sets of modules can be considered as clusters, or which parts of the code are heavily connected. In earlier work this type of measurements has been used to cluster modules and to carry out changes related to clustering, but it has not been made available yet as a distinct metric.

Regarding measurements we can realize that certain metrics often change together. For example, if the *Number of functions* rises, then the *number of funpath*, *Number of funclauses* and the *Cohesion* of modules and the *Cohesion* between modules follows, and this has an influence on load time, by increasing the depth and complexity of the syntax tree and the amount of the related semantic information.

We can observe that the changes in the source code have brought significant changes in the software metrics at several points. Some of the changes have improved the values of the metrics, and some metrics have changed for the worse. For large code bases, it is very hard to estimate the effect of individual changes in the source code on the overall quality of the code without proper tool support. An appropriate tool, on the other hand, indicates the current complexity of the source code and potential error sources. Such feedback helps the software designer to decide whether the development process is progressing as desired.

For large programs, it is imperative to be able to restructure the program so that it becomes clearer, easier to maintain and to test. In order to achieve these goals, it would be advantageous if the aforementioned tool would also support code restructuring.

In the rest of the paper, can we find answers to the following questions.

1. Is it possible to track the changes in the complexity of the code when transformations are made, and detect any problems that may arise?

2. We are seeking a method to automatically or semi-automatically improve the source code based on the measured complexity.

3. How can we use the measured values of the metrics to enhance the process of software development?

As an answer to these questions, we have implemented a system that can measure the structural complexity of Erlang programs.

The rest of the paper is structured as follows. In Section 2, we discuss how complexity can be measured in functional languages. In Section 3, we describe our representation and how we measure and store the values of the metrics. In Section 4, we give an extension to our previous metrics query language with which it is possible to run automated transformations based on the measured values of the metrics. This section contains the main contribution of the paper. Section 5 discusses related works and Section 6 concludes the paper.

2. Measuring functional languages

Several metrics developed for measuring imperative and object oriented languages can readily be applied to measuring functional languages. This is possible because there are similarities in several constructs when regarded with a degree of abstraction. As an example, a similar aspect of a library, a namespace, a class and a module is that they all can be regarded as collections of functions. If the chosen metric does not take the distinctive properties of these constructs into account (variables, method overrides, dynamic binding, visibility etc.), then it can be applied to these apparently diverse constructs. Some other properties of functional languages which bear such adaptable similarities to features in imperative languages are: nesting levels (blocks, control structures), function relations (call graph, data flow, control flow), inheritance versus cohesion, and simple cardinality metrics (number of arguments).

Functional programming languages contain several constructs and properties which are generally not present in imperative languages, thus require special attention during adaptation:

- list comprehensions,
- expression evaluation laziness, lack of destructive assignment,
- lack of loop construct, which evokes heavy use of either
  - tail recursion, or
  - higher order functions,
- referential transparency of pure functions,
- pattern matching,
- currying.

While these features raise the expressive power of functional languages, most of the existing complexity metrics require some changes before they become applicable to functional languages. So far, we have been successful in converting the metrics that we have encountered.

In addition to adapting existing metrics, we have introduced metrics that are well suited in general and Erlang in particular. We would like to point out the following findings.

- *Branches of recursion* measures the number of different cases where a function calls itself. This metric can be applied to non-functional languages as well, yet we did not see it defined elsewhere.

- Several cardinality measures, such as the number of *fun expressions*, and *message passing constructs.*
- The number of different *return points of a function.*
- We can measure metrics on a single clause of a function.
- We have extended metrics to take *higher order functions* into account, for example, how many times a fun expression is called. Due to the dynamic nature of Erlang, runtime function calls are hard to inspect, and we still have to improve this aspect of this feature.
- We are planning to investigate message passing further, which will enable us to make our metrics more precise.
- We are planning to measure OTP (Open Telecom Platform) [3] behaviours, which will uncover currently hidden function calls.

2.1. Short description of the metrics

Here we present a short overview about our implemented metrics. Hereinafter there is an enumeration of metrics which can be used as property in the extended query language. In the tables 2, 3 and 4 we can find the original name of the metric and its synonyms (one can use either the original name or any of the synonyms), afterwards we can find their short definitions.

Metrics for functions and modules	
Name	*Synonyms*
`line_of_code`	`loc`
`char_of_code`	`coc`
`max_depth_of_calling`	`max_depth_calling, max_depth_of_call, max_depth_call`
`max_depth_of_cases`	`max_depth_cases`
`number_of_funclauses`	`num_of_funclauses, number_of_funclaus, num_of_funclaus`
`branches_of_recursion`	`branches_of_rec, branch_of_recursion, branch_of_rec`
`mcCabe`	`mccabe`
`number_of_messpass`	-
`fun_return_points`	`fun_return_point, function_return_points, function_return_point`

Table 2: List of metrics for modules and functions

Effective Line of code The number of lines in the text of the module's or the function's source code excluding the empty lines.

Characters of the code The number of characters in the text of the module's or the function's source code.

Max depth of calling The length of function call-paths, namely the path with the maximum depth. It gives the depth of non-recursive calls. Recursive calls are covered by *depth_of_recursion/1* function.

Max depth of cases Gives the maximum of case control structures nested in *case* of the concrete function (how deeply are the case control structures nested). In case of a module it measures the same regarding all the functions in the module. Measuring does not break in *case* of *case* expressions, namely when the *case* is not embedded into a *case* structure.

Number of funclauses The number of the given function's function clauses (which have the same name and same arity). In case of module it counts all of the function clauses in the given module.

Branches of recursion Gives the number the given function's branches i.e., how many times a function calls itself, and not the number of clauses it has besides definition.

McCabe McCabe cyclomatic complexity metric. We define the complexity metric in a control flow graph with the number of defined basic edges, namely the number of outputs a function can have disregarding the number of function outputs functions within the function can have. Functions called each count as one possible output.

The sum of the results measured on the given module's functions is the same as the sum measured on the module itself. This metric was developed to measure procedural programs, but it can be used to measure the text of functional programs as well. (in case of functional programs we measure functions).

Number of funexpr The number of function expressions in the given function or module. (It does not count the call of function expressions, only their creation.)

Number of message passings In case of functions it counts the number of code snippets implementing messages from a function, while in case of modules it counts the total number of messages in all of the functions of the given module.

Function return points The number of the given function's possible return points. In case of module it is the sum of its function return points.

Calls for the function The number of calls for the given function. (It is not equivalent with the number of other functions calling the given function, because all of these other functions can refer to the measured one more than once.)

Metrics only for functions	
Name	*Synonyms*
calls_for_function	calls_for_fun, call_for_function, call_for_fun
calls_from_function	calls_from_fun, call_from_function, call_from_fun
function_sum	fun_sum

Table 3: List of metrics only for functions

Calls from the function The number of calls from a certain function, namely how many times a function refers to another one. The result includes recursive calls as well.

Function sum The sum calculated from the function's complexity metrics that characterises the complexity of the function. It is calculated using various metrics together.

Metrics only for modules	
Name	*Synonyms*
number_of_fun	num_of_fun, num_of_functions, number_of_functions
number_of_macros	num_of_macros, num_of_macr
number_of_records	num_of_records, num_of_rec
included_files	inc_files
imported_modules	imp_modules, imported_mod, imp_mod
number_of_funpath	number_of_funpathes, num_of_funpath, num_of_funpathes
function_calls_out	fun_calls_out
cohesion	coh
otp_used	otp
min_depth_of_calling	min_depth_calling, min_depth_of_call, min_depth_call
module_sum	mod_sum

Table 4: List of metrics only for modules

Number of functions The number of functions implemented in the module, excluding the non-defined functions.

Number of macros The number of defined macros in the module.

Number of records The number of defined records in the module.

Number of included files The number of visible header files in the module.

Imported modules The number of imported modules used in the given module. The metric does does not take into account the number of qualified calls (calls that have the following form: *module:function*).

Number of funpath The total number of function paths in the given module. The metric, besides the number of internal function links, also contains the number of external paths, or the number of paths that lead outward from the module.

Function calls into the module The number of function calls into the given module from other modules.

Function calls from the module The number of function calls from the given module towards other modules.

Cohesion of the module The number of call-paths of functions that call each other in the module. By call-path we mean that an *f1* function calls *f2* (e.g. `f1()->f2().`). If *f2* also calls *f1*, then the two calls still count as one callpath.

Max depth of calling The maximum depth of function call chains within the given module. It gives the depth of non-recursive calls.

Module sum The sum of *function_sum* for all functions in the given module.

3. Program graph representation

In [4], we have introduced an extensible architecture in which the definition and acquisition of important attributes of the source code can be conveniently formulated. When the source code is loaded, it is parsed into an abstract syntax tree, which is then turned into a program graph by adding static semantic nodes and edges. These semantic nodes and edges describe the important attributes and connections of the source code: the call graph, the statically analysable properties of dynamic constructs, the data flow necessary to track the spreading of values. The semantic nodes currently comprise all information that is necessary for the calculation of metrics, but the architecture is extensible: new semantic constructs can be added easily.

The program graph, for our purposes, contains syntactic and semantic nodes and edges. The abstract syntax tree built upon the represented source code forms a subgraph of the program graph. In addition to this subgraph, the program graph also contains nodes that describe semantical information, such as the binding structure of variables. The edges of the program graph are directed, labelled, and for each node, the outgoing edges having the same label are ordered.

Low level query language. Information retrieval is supported by a low level query language that makes it easy to traverse graph structures. This low level query language consists of fixed length *path expressions*, which run starting from a single node, can traverse edges in a forward or backward direction, and can filter the resulting nodes in each step based on their contents.

Metrics are calculated by running several queries that collect syntactic and semantic constructs, and then evaluating the information content of the resulting nodes.

Summing it up, the calculation of complexity metrics takes place in three steps:

1. We construct the program graph of the source code. As we measure several metrics on the same program graph, the program graph is already available. During the static analysis of the source code, we construct the Abstract Syntax Tree of the code, then we expand it from all the semantic information gained with all of the static analyses. If we already have the semantic graph at hand, then the process only takes two steps.

2. We execute the path expression that is appropriate for the metric. The result of the path expressions defined on the constructed graph will be a list of nodes. In most cases, the characteristic complexity metric can be calculated from the result.

3. We calculate the value of the metric. For some metrics, this step is simply the calculation of the cardinality of the resulting list (e.g. number of functions), whereas for other metrics, filtering has to be done (e.g. internal cohesion of the module). The result of this expression is a list of all the function nodes, which are available on the defined graph path. The length of the list gives the total number of function paths. The result contains all of the function calls within the module and the for and from calls. If we wanted to measure regarding the internal cohesion of the module, then we would have to filter the result.

Caching calculated values of metrics As metrics have to be recalculated each time the code is changed, it is desirable to make this process as fast as possible. Fortunately, most metrics can be calculated incrementally, if we store the measured values in the associated *module* or *function* semantic node. This way, only the values of those metrics have to be adjusted that are affected by the change in the code.

Number of function clauses

```
show number_of_funclauses for module('exampmod')
```

Figure 1: Query language example

Function calls into the module

```
show function_calls_out for module('exampmod')
```

Figure 2: Query for modules

Example Erlang module

```
-module(exampmod).

abs(X) when X >= 0 -> X;
abs(X)             -> -X.

sign(0)            -> 0;
sign(X) when X > 0 -> 1;
sign(X)            -> -1.

manhattan(Xs, Ys) ->
 Pairs = lists:zip(Xs, Ys),
   List =
     [abs(X-Y) || {X, Y} <- Pairs],
   lists:sum(List).
```

Figure 3: An example module in Erlang

Textual query language Figure 1 and Figure 2 show two metrics queries. The former shows the number of all function clauses in the module; for the module *exampmod*, whose code can be seen in Figure 3, this value is 6, as the function *abs* has two clauses, *sign* has three and *manhattan* has one. The latter shows the number of calls of external functions in the module; for *exampmod*, this value is 2, since calls to *lists:sum* and *lists:zip* (functions of the module *lists*) are included, but the call to *abs*, a local function, is not.

In Section 4.1, we give an extension to this query language that enables the user to write transformations based on the measured metric values. Batches of such queries can be stored as scripts that automatically improve the source code when executed.

4. Metrics driven transformations

Most of the metrics can be associated to a node in the program graph so that the value of the metric can be calculated using only the syntax subtree below the node. We store the current values of metrics in the associated node, which serves two purposes when the code is transformed. Firstly, since most of the metrics are compositional, we can use the stored values as caches, and only recalculate the parts that have changed, thereby making the calculation of the new values faster. Secondly, we can compare the old and the new values of the metrics, and we can make the necessary arrangements if the code is changed in an undesired way:

1. We can leave the transformation of the code to the user. This task is time consuming and error prone, especially if the code base is large, difficult, or unknown to the user. In this case, RefactorErl can help the user by displaying the values of metrics measured on the current code, and warns the user if a value goes beyond a specified limit.

2. The user may use the semi-automatical transformations of RefactorErl to improve the code. With this option, the user regains control of the process of transformation: he chooses what gets transformed and in what way. Using RefactorErl ensures that the code is transformed in all necessary places, and that the resulting code is syntactically valid, and semantically equivalent to the original.

3. As the main contribution of the paper, we introduce a new approach: metrics driven automatic code optimization. We elaborate it in the following section.

4.1. Metrics driven automatic code optimization

We introduce an extension to our query language in which the transformation engine of RefactorErl can be instructed to improve the source code based on the calculated metrics. The grammar of the original query language is shown in Figure 4.

Optimization query language Figure 5 shows the grammar of the optimization extension language. In Figures 4 and 5, `Id`, `Ids`, `ArRel`, `LogCon Var` and `Int` stand for an identifier, a list of identifiers, an arithmetic relation (e.g. $<$), a logical connector (e.g. *and*), a variable and an integer, respectively. The extension language is quite straightforward, describing which modules are to be transformed (*optimize*), which transformations are to be used (*using*), where the transformations are to be attempted (*where*), and at most how many steps are to be attempted (*limit*). In the *where* clause, the identifiers indicate a metric; variables may only be used if the query is part of a script, and the variable is bound to a value of a metric.

$$
\begin{aligned}
\textbf{MetricQuery} &\rightarrow \textbf{Show Loc} \\
\textbf{Show} &\rightarrow \textbf{show}\ Id \\
\textbf{Loc} &\rightarrow \textbf{module}\ Id \mid \textbf{function}\ Id
\end{aligned}
$$

Figure 4: Slightly abridged grammar of the metrics query language

$$
\begin{array}{rcl}
\textbf{Query} & \rightarrow & \textbf{MetricQuery} \mid \textbf{OptQuery} \\
\textbf{OptQuery} & \rightarrow & \textbf{Opti Trs Where Limit} \\
\textbf{Opti} & \rightarrow & \textbf{optimize all} \mid \textbf{optimize } \mathit{Ids} \\
\textbf{Trs} & \rightarrow & \textbf{using } \mathit{Ids} \\
\textbf{Where} & \rightarrow & \textbf{where Cond} \\
\textbf{Cond} & \rightarrow & \textbf{Expr } \mathit{ArRel} \textbf{ Expr} \\
 & \mid & \textbf{Cond } \mathit{LogCon} \textbf{ Cond} \\
\textbf{Expr} & \rightarrow & \mathit{Id} \mid \mathit{Var} \mid \textbf{MetricQuery} \\
\textbf{Limit} & \rightarrow & \textbf{limit } \mathit{Int}
\end{array}
$$

Figure 5: Grammar of the metrics query language with optimization

Metrics driven transformation example The first code snippet in Figure 7 shows a function that contains too deeply nested *case* expressions. Figure 6 shows the script we are going to use to instruct the engine to improve the code.

The script consists of two steps. The first step calculates the maximum level of case nesting in module *not_present* (not appearing in this paper); let this value to be 1. This value is assigned to the variable *P1*. The second step starts the transformation engine, which tries to decrease the number of nodes in module *to_refactor* where the condition holds. Since the *number_of_functions* is only one, the significant part of the condition selects those nodes where *max_depth_of_cases* is larger than one. In the original code, the function *f* contains a case construct of depth 3, which is then refactored using the *introduce function* transformation (*introduce_fun*). The transformation takes the body of the innermost case construct, and extracts it to a new function *f0*.[1]

As we have not reached the step limit, the condition is reevaluated: the *number_of_functions* has grown to 2, and the *max_depth_of_cases* is decreased to 2. Since this value is still over the desired value, a similar transformation step is applied as depicted in Figure 7. This is the last transformation step: we have reached the step limit. Incidentally, we have also eliminated all nodes where the condition of the query would hold.

Since the transformation engine executes the script without external help, it might transform the code in an inferior way to an expert. If the result of the script execution does not turn out to be desirable, the user want have to revert the code to the state before the execution of the script. It is also possible to revert only some steps that the script took.

[1]The name of the function is generated. If the name of the function is not to the liking of the user, it can be changed using another available transformation later.

```
P1 = show
       max_depth_of_cases
     for
       module not_present
optimize
  module to_refactor
where
  max_depth_of_cases > P1
  and
  number_of_functions < 10
using
  introduce_fun
limit 2
```

Figure 6: Metric query language example code

5. Related work

Several IDEs for object oriented languages (e.g. Eclipse [7], NetBeans, IntelliJ Idea) provide both metrics and refactorings, however, we are unaware that the two areas are connected in any of them.

Simon, Steinbrückner and Lewerentz [5] have created a tool that visualizes several metrics based on Java and C++ code, thereby helping the user to make decisions about transforming his code. They show that well chosen metrics can support the decision of the user before he confirms a refactoring.

The goal of the project *Crocodile* [6] is to provide concepts and tools for an effective usage of quantitative product measurement to support and facilitate design and code reviews particularly for object oriented programs and reusable frameworks. While Crocodile is useful as a measurement tool, it also can interactively assist the programmer in executing transformation steps.

Tidier [8, 9] is a software tool that makes a series of fully automated code transformations which improve the performance, quality and/or structure of the code. Tidier uses simple semantics preserving transformations with an emphasis on easy validability. The transformations are universally applicable, and do not rely on metrics for guidance.

6. Conclusion and future work

In this paper, we have presented a way to improve software by applying automated transformations based on complexity metrics. We have defined a query language which makes the transformations accessible to the end user, and we have imple-

The original code

```
f({A, B})->
  case A of
    send -> case B of
      {Pid, Data} ->
         case Pid of
           {pid, P} -> P ! Data;
                  _ -> Pid ! B end;
       _ -> null ! B end};
    _ -> mod:wait() end.
```

↓

Code after the first step

```
f({A, B})->
  case A of
        send -> f0(B);
    _ -> mod:wait()
  end.

f0(B) ->
  case B of
   {Pid, Data} ->
      case Pid of
         {pid, P} -> P ! Data;
                _ -> Pid ! B
      end;
   _ -> null ! B end.
```

↓

The result of the transformation

```
f({A, B})->
  case A of
        send -> f0(B);
           _ -> mod:wait()
  end.

f0(B) ->
    case B of
        {Pid, Data} -> f1(B, Data, Pid);
                  _ -> {null ! B}
    end.

f1(B, Data, Pid) ->
    case Pid of
        {pid, P} -> P ! Data;
               _ -> Pid ! B
    end.
```

Figure 7: Two steps of automatic source code transformation

mented an engine that improves the source code by executing scripts written in the query language.

We have implemented the method described in the paper as part of RefactorErl, an Erlang analyser and transformation tool. The back end of the tool builds the program graph representation discussed in Section 3.

Using the information in the graph, we collect the values of the metrics and we store them in the corresponding nodes of the graph. When the graph is changed, these values are updated incrementally. The values are shown in the user interface, and are also queryable. The user may define a limit for a metric or a combination of metrics; when the code is measured to be outside this limit, the user is alerted. Analysis of how such limits should be defined may constitute a promising new line of research.

In addition to the calculation of the metrics values, we have implemented the metrics driven code optimization discussed in Section 4. We have extended the previous metrics query language as seen in Figure 5 to support query based automatic code optimization, and we have implemented the engine that runs the transformations according to the query. The engine is also capable of running scripts that contain batches of queries. If unsatisfied with the result, the user can fully or partially revoke these transformations.

References

[1] R. Kitlei, L. Lövei, M Tóth, Z. Horváth, T. Kozsik, T. Kozsik, R. Király, I. Bozó, Cs. Hoch, D. Horpácsi. Automated Syntax Manipulation in RefactorErl. *14th International Erlang/OTP User Conference. Stockholm, (2008)*

[2] Horváth, Z., Lövei, L., Kozsik, T., Kitlei, R., Víg, A., Nagy, T., Tóth, M., and Király, R.: Building a refactoring tool for Erlang *In Workshop on Advanced Software Development Tools and Techniques, WASDETT 2008, (2008)*

[3] Erlang - Dynamic Functional Language, `http://www.erlang.org`

[4] Kitlei, R., Lövei, L., Nagy, T., Víg, A., Horváth, Z., and Csörnyei, Z. *Generic syntactic analyser: ParsErl*, In Proceedings of the 13th International Erlang/OTP User Conference, EUC 2007, Stockholm, Sweden, November 2007

[5] Frank Simon, Frank Steinbrückner, Claus Lewerentz *Metrics based refactoring* IEEE Computer Society Press 2001 30–38.

[6] Claus Lewerentz, Frank Simon *A Product Metrics Tool Integrated into a Software Development Environment* Object-Oriented Technology (ECOOP'98 Workshop Reader), LNCS 1543 Springer-Verlag 256–257.

[7] Eclipse Foundation, `http://www.eclipse.org/`

[8] Thanassis Avgerinos, Konstantinos F. Sagonas *Cleaning up Erlang code is a dirty job but somebody's gotta do it.* Erlang Workshop 2009: 1–10.

[9] Konstantinos F. Sagonas, Thanassis Avgerinos *Automatic refactoring of Erlang programs.* PPDP '09 Proceedings of the 11th ACM SIGPLAN conference on Principles and practice of declarative programming 2009: 13–24.

[10] McCabe T. J. A Complexity Measure, *IEE Trans. Software Engineering, SE-2(4), pp.308–320 (1976).*

Annales Mathematicae et Informaticae
38 *(2011) pp. 75–86*
http://ami.ektf.hu

C++ Standard Template Library by infinite iterators*

Tamás Kozsik, Norbert Pataki, Zalán Szűgyi

Department of Programming Languages and Compilers
Eötvös Loránd University, Budapest, Hungary
e-mail: kto@pnyf.inf.elte.hu, patakino@elte.hu, lupin@ludens.elte.hu

Submitted August 31, 2010 Accepted December 8, 2011

Abstract

The C++ Standard Template Library (STL) is an essential part of professional C++ programs. STL is a type-safe template library that is based on the generic programming paradigm and helps to avoid some possible dangerous C++ constructs. With its usage, the efficiency, safety and quality of the code is increased.

However, professional C++ programmers are eager for some missing STL-related opportunities. For instance, infinite ranges are hardly supported by C++ STL. STL does not contain iterators that use a predicate during traversal. STL's design is not good at all from the view of predicates. In this paper we present some extensions of C++ STL that supports effective generic programming. We show scenarios where these extensions can be used pretty gracefully. We present the implementation of our infinite iterators.

Keywords: C++, STL, iterators

MSC: 68N15

1. Introduction

The *C++ Standard Template Library* (STL) was developed by *generic programming* approach [1]. In this way containers are defined as class templates and many algorithms can be implemented as function templates. Furthermore, algorithms are implemented in a container-independent way, so one can use them with different

*The Research is supported by the European Union and co-financed by the European Social Fund (grant agreement no. TÁMOP 4.2.1./B-09/1/KMR-2010-0003).

containers [19]. C++ STL is widely-used inasmuch as it is a very handy, standard C++ library that contains beneficial containers (like list, vector, map, etc.), a large number of algorithms (like sort, find, count, etc.) among other utilities.

The STL was designed to be extensible [2]. We can add new containers that can work together with the existing algorithms. On the other hand, we can extend the set of algorithms with a new one that can work together with the existing containers. The expression problem [23] is solved with this approach.

Iterators bridge the gap between containers and algorithms. They provide a standard interface to the algorithms to access the elements of the containers. Iterators are distinguished based on their capabilities and a hierarchy is formed based on these categories [11], too. The following categories defined in the STL:

- input iterator: the elements are reachable sequentially and they are just readable for the algorithms.
- output iterator: the elements are reachable sequentially and they are just writeable for the algorithms.
- forward iterator: the elements are reachable sequentially and the algorithms can both read and write them.
- bidirectional iterator: the elements are reachable sequentially, but the algorithms can read them forward and backward too, and the elements are both readable and writeable. For example: the container `list` provides this kind of iterator.
- random access iterator: the elements are reachable in any order and the algorithms can read and write the elements. For example: the container `vector` provides iterators with these capabilities.

STL also includes adaptor types which transform standard elements of the library for a different functionality. There are iterator adaptors allowing to read an input stream or write an output stream. These iterator adaptors instead of access an existing element of a container read or write them to a stream. Other iterator adaptor allows to insert a new element into a container instead of access an existing one. These iterator adaptors are mainly input or output iterators [9].

Functor objects make STL more flexible as they enable the execution of user-defined code parts inside the library without significant overhead [10]. Basically, functors are usually simple classes with an `operator()`. Inside of the library, the `operator()`s are called to execute user-defined code snippets. This can called a function via pointer to functions or an actual `operator()` in a class. Functors are widely used in the STL inasmuch as they can be inlined by the compilers and they cause no runtime overhead in contrast to function pointers. Moreover, in case of functors extra parameters can be passed to the code snippets via constructor calls.

Functors can be used in various roles: they can define predicates when searching or counting elements, they can define comparison for sorting elements and properly searching, they can define operations to be executed on elements.

We can distinguish the functors by their behaviour. The functors with no arguments are called *generators*. The functors which return boolean values are *predicates*. We call those functors *unary or binary functor*, which has exactly one or two arguments.

The algorithms usually work on a range of input sequence. The range is defined by a pair of iterators. The first iterator of the pair referring to the beginning of the range and the second one referring to the end. The range is inclusive on the left and exclusive on the right. All containers has two member functions `begin` and `end`, which return an iterator of the first element of the container and a dummy iterator referring to the element after the last element in a container. As the range is exclusive on the right, the range `begin() ... end()` covers the whole container. Moreover the *end iterator* can be used as extremal value, such as the algorithm `find` returns it, when the searched element is not in the range.

The begin and the end of the range are handled specially for those iterators, which do not belong to a container. For example the `istream_iterator` reads elements from the standard input, and it reaches the end of its range when the next read is failed. (E.g.: it reaches the end of file.) `istream_iterator` created by constructor setting the source stream representing the beginning of the range, and the other created by the default constructor will be the end of the range.

The iterators are essential part of STL as they provide the input to the algorithms. Although all the containers provide different classes of iterators, and variety of iterator adaptors are exist in STL, there are several important functionality still missing. There is no support to filter the elements that the iterator traverses, there is no possibility to iterate over an integer range, the elements cannot be transformed by the iterator and at last but not at least there is not possible to work with infinite ranges.

There are ongoing researches to improve the iterator facility of STL (see section 7), no one of them supports infinite ranges.

In this paper we provide an *infinite iterator* type, which is able to generate an infinite sequence of elements. This feature is mainly supported by functional languages, such as Haskell. Functional languages are able to use infinite ranges and lazy evaluation [6]. C++ programmers are eager for these features, too [5, 7, 8, 13, 15, 16, 17]. While infinite sequence is widely used in functional programming realm, the *infinite iterator*s simplifies the initialization of containers, as well.

Our paper is organized as follows: we introduce *infinite iterators* in section 2, and we detaile our enhancements applied on C++0x in section 3. The implementation details about detecting the arity of functors and the stoppage of generation are discussed in section 4 and 5. In section 6 we discuss which kind of infinite ranges are supported by our library. The related works is detailed in section 7 and we conclude our results in section 8.

2. Infinite iterators

The `infinite_iterator`, like the `istream_iterator` of the STL, does not belong to any container, but generates a sequence of elements. Any increase on `infinite_iterator` generates the next element of the sequence. The generator strategy is provided by the user as a functor object. The `infinite_iterator` is an *input iterator*, thus the elements are only readable.

In the hereinafter example, we create an `infinite_iterator` which generates Fibonacci numbers. Then we write the first 10 Fibonacci number to the standard output.

```
struct Fib
{
    Fib() : a(0), b(1) {}
    int operator()()
    {
        int res = a + b;
        a = b;
        b = res;
        return res;
    }
private:
    int a;
    int b;
};

Fib fib;
infinite_iterator<Fib> ii(fib);

for( int i = 0; i < 10; ++i )
    std::cout << *ii++;
```

The struct `Fib` is a generator functor that generates the elements of the sequence. The `infinite_iterator` has one template argument: the type of the generator functor. The type of the generated element is deduced by the compiler of the signature of the functor's member function `operator()`. Its constructor receives only the functor object.

3. C++0x-based approach

The code in the previous subsection works fine, but the functor has to deal with the whole process of generating elements of the sequence. Besides calculation of the next element, it has to take care to save the previous elements which are playing role in computation of the following one. However, the process of saving previous elements is mostly independent of the generated sequence. The only operative question is the number of the previous elements that are needed to compute the next one.

With the features of the new standard of C++, we can provide a more sophisticated `infinite_iterator` that makes easier of writing functors. Our new `infinite_iterator` is able to save the previous elements in a sequence and feed the functor with them. That way the functor only need to take care of computing the next element of sequence. The number of the saved previous elements depends on the number of the functor's `operator()` arguments. The constructor of `infinite_iterator` needs the same number of initial values of the sequence. On increase of `infinite_iterator`, the stored elements are passed to the functor as arguments.

The oldest will be the first argument, and the previously calculated will be the last one. Then the functor computes the next element and returns it.

See the code snippet below, which simplifies the example in previous subsection.

```
struct Fib
{
    int operator()(int a, int b) const
    {
        return a + b;
    }
};

Fib fib;
infinite_iterator<Fib> ii(fib);

for( int i = 0; i < 10; ++i )
    std::cout << *ii++;
```

Now the struct `Fib` needs to take care of the computation of the next element only. Every other is done by the `infinite_iterator`.

Our solution supports the lambda expressions, which are introduced by the C++0x [4]. Lambda expression is also accepted in place of functors [21]. The code snippet below shows the way to apply lambda expression with `infinite_iterator`.

```
auto fib = [](int a, int b){return a + b;};
infinite_iterator<decltype(fib)> ii(fib);
```

4. Specializing by the arity of functors

In C++0x realm the `infinite_iterator` is able to distinguish between the functors by their arity. The arity of a functor is the number of the arguments of its `operator()`. With nullary functor, the `infinite_iterator` does not save the previous elements. That case all the computation process is done by the functor, like in the example in section 2. With unary, binary, trinary, etc. functors our iterator saves the previous one, two, three, etc. elements, and feeds the functor with these values on computation of the next element, like the example in subsection 3.

The code skeleton below shows the way we specializing `infinite_iterator` by the arity of functor. The template argument E is related to the stoppage of the generation of infinite sequence and it is detailed in 5.

```
template<class T>
struct infinite_iterator_base :
    std::iterator<
        std::input_iterator_tag,
        T>
{
  /* the common functionality
     is implemented here */
};

template<
    class G,
    class E = not_specified,
    class P = decltype(&G::operator())>
struct infinite_iterator
{
};

template<class G, class E, class T>
struct infinite_iterator<
    G,
    E,
    T (G::*)()> :
    infinite_iterator_base<T>
{
  /* specialization for nullary functor */
};

template<class G, class E, class T>
struct infinite_iterator<
    G,
    E,
    T (G::*)(T) const> :
      infinite_iterator_base<T>
{
 /* specialization for unary functor */
};
```

```
template<class G, class E, class T>
struct infinite_iterator<
    G,
    E,
    T (G::*)(T, T) const> :
      infinite_iterator_base<T>
{
 /* specialization for binary functor */
};

/* similarly for n-ary functor */
```

The struct `infinite_iterator_base` implements the common functionality of the different `infinite_iterator` specializations, hence these specializations are inherited from the `infinite_iterator_base`. Different specialization belongs to the functors with different arity from 0 to `MAX_ARITY`. `MAX_ARITY` is a preprocessor macro and it sets the upper limit of the supported functor arity. We generate the different specializations from a template using *Boost.Preprocessor* library [24]. Our solution is similar to the way that the *Boost.MPL* library [25] is implemented. While the different arities of functors require different functionalities, thus the general version of `infinite_iterator` is not used and its body remains blank.

5. Stoppage of generation

The `infinite_iterator`s generate an infinite sequence. However, in real problems a finite subsequence of elements is required. Our solution provides *end iterator* to determine a finite range of an infinite sequence. The *end iterator* can be created in two ways.

- *By a constructor with one integer argument:* The argument specifies the length of the finite subsequence.
- *By a constructor with a predicate as its argument.* With this kind of end iterator, the generation of the elements in an infinite range is stopped, when the predicate returns false for the currently generated element. Using this version of `infinite_iterator`, the type of the predicate must be specified as the second template argument during the instantiation of either *normal* or *end* iterator.

The example above fills two arrays of integers `(t, r)` with the numbers from 1 to 10. For array `t` the first kind of end iterator is used, while for array r, the second one is chosen.

```
struct ints
{
    int operator()(int a) const
```

```
    {
        return a + 1;
    }
};

struct pred
{
    pred(int i) : max(i) {}
    bool operator()(int a) const
    {
        return a < max;
    }
private:
    int max;
}

int t[10];
int r[10];

infinite_iterator<ints> tb(ints(), 0);
infinite_iterator<ints> te(10);

infinite_iterator<ints, pred> br(ints(), 0);
infinite_iterator<ints, pred> re(pred(11));

copy(tb, te, t);
copy(rb, re, r);
```

The `infinite_iterator` created by a default constructor is also an end iterator, however, this one represents a real infinite range, thus the generation of the elements needs to be stopped in other way.

6. Supported infinite ranges

While the programmer can apply any kind of functor object to our library, we support a large variety of infinite ranges. The only restriction is that the generated elements must be copy constructable and assignable. (STL also requires this concept.)

With the help of the predefined functors of STL, the most commonly used infinite ranges can be defined without writing any user defined functors. Infinite iterators utilized by:

- functor `binder2nd<plus<int> >(plus<int>(), 1)` with 0 as initial number generates the natural numbers

- functor `binder2nd<plus<int> >(plus<int>(), 2)` with 0 as initial number generates the positive even numbers
- functor `binder2nd<plus<int> >(multiplies<int>(), 2)` with 1 as initial number generates the powers of 2
- functor `plus<int>()` with 1 and 1 as initial numbers generates the Fibonacci numbers
- etc.

Although the usage of the functors provided by the STL covers the most commonly used ranges, our library provides additional functors allowing the user to define infinite ranges easier. Our functors are:

- `inc<T>` which increases an element by prefix `operator++`.
- `dec<T>` which decreases an element by prefix verb|operator–|.
- `constant<N>` which returns always N, thus it can be used to define infinite range of the same elements.

While it is possible to generate any kind of infinite range which elements are copy constructable and assignable, for effeciency reasons our solution mainly focuses on those ranges, where the next element can be determined by the finite number of previous elements. In the latter case the functor itself has to take care about all the generation process. It is a common design rule in STL that the inefficient methods are not supported, for example, there is no index operator for `list`, or `push_front` member function is missing in `vector`.

7. Related work

One known extension of the STL is the View Template Library, which provides a layer over the C++ Standard Template Library [14]. It consists of *views* that are container adaptors, and *smart iterators* that are a kind of iterators provided a different view onto the data that are pointed to by the iterator. Views wrap the container to be filtered and transformed the elements on which the view operates. These transformations and filterings are done during the execution, without taking effect the stored data in the container. The interface provided by the views is a container interface.

Although View Template Library provides *views* that filters the elements on a range its usage is limited to the containers. We cannot filter ranges defined by those iterators which are not belongs to the container. Thus a simple problem: to copy the odd numbers from the standard input to the standard output is not soluble.

The Boost Iterator Library [26] is an extension to the STL with a variety of useful iterator types. It contains two parts. The first one is a system of concepts which extend the C++ standard iterator requirements. The second one is a framework of components for building iterators based on these extended concepts and includes several useful iterator adaptors, such as `filter_iterator`, which traverses only those elements, which satisfy a given requirement; `counting_iterator`, which generates a sequence of an elements; `function_input_iterator`, which invokes a nullary function on dereference operation, etc. The extended iterator concepts have been carefully designed so that old-style iterators can fit in the new concepts and so that new-style iterators will be compatible with old-style algorithms, although algorithms may need to be updated if they want to take full advantage of the new-style iterator capabilities. Several components of this library have been accepted into the C++ standard technical report.

Our solution unifies and extends the functionality of `counting_iterator` and `function_input_iterator` as it is able to accept an arbitray arity of functors. Besides the `infinite_iterator` is able to cooperate the other iterator adaptors of the Boost Library.

Our `infinite_iterators` can be adapted by `filter_iterator` of Boost Library. It is useful, when only elements with a specific property are needed from an infinite sequence. Separating the condition of the specific property and the general generation method may highly simplify to define special infinite sequences. Let us suppose someone needs the infinite sequence of odd Fibonacci numbers. As the addition of last two odd Fibonacci numbers is not the next odd Fibonacci number, the generator functor has to deal with the even Fibonacci numbers as well. However, as the process of checking the number is odd is moved to `filter_iterator`, the generator functor can be a simple Fibonacci sequence generator as in 3.

The example below prints the first ten odd Fibonacci number to the standard output.

```
typedef infinite_iterator<Fib> inf_fib;
inf_fib ib(Fib(), 0, 1);

IsOdd pred;
boost::filter_itertator<IsOdd, inf_fib> fb(pred, ib);

for( int i = 0; i < 10; ++i )
{
    std::cout << *fb++;
}
```

8. Conclusion

C++ Standard Template Library is the most widely-used library based on the generic programming paradigm. It consists of handy containers and general, reusable algorithms. Iterators bridge the gap between containers and algorithms, so algorithms do not depend on the used container. Adaptors are also an important part of the STL, which can change the behaviour of the STL components for special situations.

However, there are functionalities that are missing from the library. Although iterators play a main role in the library, the several features that are make the programmer work easier and fail-safe are missing or limitedly supported.

It this paper we have prompted that the infinite ranges have only a very limited support either in the STL itself or in the other third party libraries, too. We presented a comfortable extension for the STL which supports the usage of infinite ranges in a general way. With the support of the incoming new standard of C++ our library become an highly customizable, easy to use, library which is able to cooperate either the STL or the iterator extensions of the other third party libraries, too.

References

[1] Alexandrescu A., Modern C++ Design, *Addison-Wesley* (2001).

[2] Austern, M. H., Generic Programming and the STL: Using and Extending the C++ Standard Template Library, *Addison-Wesley* (1998).

[3] Buss, A., Fidel, Harshvardhan, Smith, T., Tanase, G., Thomas, N., Xu X., Bianco, M., Amato, N. M., Rauchwerger, L. The STAPL pView, *Proc. of Languages and Compilers for Parallel Computing (LCPL 2010), Lecture Notes in Comput. Sci.* **6548** (2011), 261–275.

[4] Järvi, J., Freeman, J., C++ Lambda Expressions and Closures, *Sci. Comput. Programming* **75(9)** (2010), 762–772.

[5] Király, R., Kitlei, R., Application of Complexity Metrics in Functional Languages, *Proc. of the 8th Joint Conference on Mathematics and Computer Science, Selected Papers*, 267–282.

[6] Kiselyov, O., Functional Style in C++: Closures, Late Binding, and Lambda Abstractions, *Proc. of the third ACM SIGPLAN international conference on Functional programming (ICFP '98)* (1998), p. 337.

[7] McNamara, B., Smaragdakis, Y., Functional Programming with the FC++ Library, *Journal of Functional Programming* **14(4)** (2004), 429–472.

[8] McNamara, B., Smaragdakis, Y., Functional Programming in C++, *Proc. of the fifth ACM SIGPLAN international conference on Functional programming (ICFP '00)*, 118–129.

[9] Meyers, S., Effective STL - 50 Specific Specific Ways to Improve Your Use of the Standard Template Library, *Addison-Wesley* (2001).

[10] Pataki, N., Advanced Functor Framework for C++ Standard Template Library, *Stud. Univ. Babeş-Bolyai, Inform.* **LVI(1)** (2011), 99–113.

[11] Pataki, N., Porkoláb, Z., Istenes, Z., Towards Soundness Examination of the C++ Standard Template Library, *Proc. of Electronic Computers and Informatics* (2006), 186–191.

[12] Pataki, N., Szűgyi, Z., Dévai, G., Measuring the Overhead of C++ Standard Template Library Safe Variants, *Electronic Notes in Theoret. Comput. Sci.* **264(5)** (2011), 71–83.

[13] Porkoláb, Z.: Functional Programming with C++ Template Metaprograms, *Proc. of Central European Functional Programming School (CEFP 2009), Lectures Notes in Comput. Sci.* **6299** (2010), 306–353.

[14] Powell, G., Weiser, M., A New Form of Container Adaptors, *C/C++ Users Journal* **18(4)**, (April 2000) 40–51.

[15] Sipos, Á., Pataki, N., Porkoláb, Z., Lazy Data Types in C++ Template Metaprograms, *Proc. of 6th International Workshop on Multiparadigm Programming with Object-Oriented Language 2007 (MPOOL' 07)* (2007).

[16] Sipos, Á., Porkoláb, Z., Pataki, N., Zsók V., Meta<Fun> – Towards a Functional-Style Interface for C++ Template Metaprograms, *Proc. of 19th International Symposium of Implementation and Application of Functional Languages (IFL 2007)* (2007), 489–502.

[17] Slodičák, V., Szabó Cs., Recursive Coalgebras in Mathematical Theory of Programming, *Proc. of the 8th Joint Conference on Mathematics and Computer Science, Selected Papers*, 385–394.

[18] Smetsers, S., van Weelden, A., Plasmeijer, R., Efficient and Type-Safe Generic Data Storage, *Electronic Notes in Theoret. Comput. Sci.*, **238(2)** (2009), 59–70.

[19] Stroustrup, B., The C++ Programming Language (Special Edition), *Addison-Wesley* (2000).

[20] Szűgyi, Z., Török, M., Pataki, N., Towards a Multicore C++ Standard Template Library, *Proc. of Workshop on Generative Technologies (WGT 2011)* (2011), 38–48.

[21] Szűgyi, Z., Török, M., Pataki, N., Kozsik, T., Multicore C++ Standard Template Library with C++0x, *Proc. of NUMERICAL ANALYSIS AND APPLIED MATHEMATICS ICNAAM 2011: International Conference on Numerical Analysis and Applied Mathematics*, 857–860.

[22] Tanase, G., Buss, A., Fidel, A., Harshvardhan, Papadopoulos, I., Pearce, O., Smith, T., Thomas, N., Xu, X., Mourad, N., Vu, J., Bianco, M., Amato, N. M., The STAPL Parallel Container Framework, *Proc. of ACM SIGPLAN Symp. Prin. Prac. Par. Prog (PPoPP 2011)* (2011), 235–246.

[23] Torgersen, M., The Expression Problem Revisited – Four New Solutions Using Generics, *Proceedings of European Conference on Object-Oriented Programming (ECOOP) 2004, Lecture Notes in Comput. Sci.* **3086** (2004), 123–143.

[24] `http://www.boost.org/doc/libs/1_46_1/libs/preprocessor/doc/`

[25] `http://www.boost.org/doc/libs/1_47_0/libs/mpl/doc/index.html`

[26] `http://www.boost.org/doc/libs/1_47_0/libs/iterator/doc/index.html`

Annales Mathematicae et Informaticae
38 *(2011) pp. 87–93*
http://ami.ektf.hu

Biarc analysis for skinning of circles

Roland Kunkli

Faculty of Informatics, University of Debrecen, Hungary
e-mail: kunkli.roland@inf.unideb.hu

Submitted October 14, 2011 Accepted December 19, 2011

Abstract

By circle skinning we have a discrete set of circles and we would like to find two curves, which touch each of them and satisfy some conditions. There exist methods to give a solution for this problem, but none of them use biarcs for the construction. Meek and Walton published a very deep analysis of biarcs in [1], and they divided them into several families.

Of course one of the basic problems is to find the mentioned curves for *two* circles. In this paper several necessary conditions are given to avoid intersections in this basic case between the skinning curve and the circles using a concrete family of biarcs from [1]. A method is publicated in [3] with which we can find the touching points for the skinning.

Keywords: skinning, biarcs, interpolation, circles

1. Introduction

We have several well-known methods to interpolate a set of points, it is very important in Computer Aided Geometric Design to solve this problem. Skinning is a special case of interpolations, where we have circles instead of points and we would like to find a pair of curves, which touch each of the circles and have certain preferences. There exist actual researches to get a solution for skinning problem, often in higher dimensions with spheres [7, 8, 3, 2]. The results can be very useful by covering problems, geometric design, designing tubular structures or molecular modeling. The mentioned methods use C^1 or G^1 continuous curves by the interpolation. The main idea of this paper is to use biarcs for the construction, and find the skinning curves made by joining biarcs.

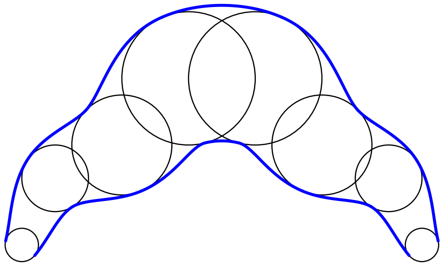


Figure 1: A skinning example

2. Localization of the touching points

Some researchers use an iterational method to find the touching points on the circles [7], others define them exactly before the curve interpolation [3]. In this paper the method is considered from [3]. The authors use Apollonius-circles for the construction, they find touching points on each circles with their neighbours. This method guarantees to find touching points which don't fit on other disks. This is a very powerful attribution, so we can use this method for our construction too.

3. Families of biarcs

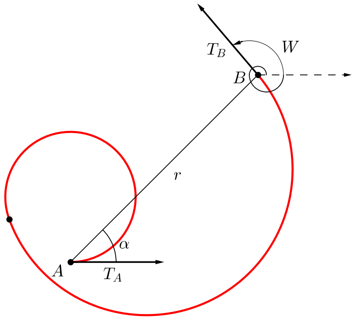
If we join two circular arcs in G^1 continuity, we get a biarc. Biarcs have a long history, there exists a paper from 1937 [4], where biarcs was mentioned first time. We can create so-called *biarc curves* by joining biarcs [5], furthermore there are several methods to approximate a fixed curve with biarc curves [6]. This type of curves are very useful by design, because CNC machines can only cut lines and arcs. Meek and Walton published a very deep analysis of biarcs in [1], and they divided them into several families.

The authors construct a biarc from enhanced G^1 Hermite datas. This means we have two points, two unit tangent vectors and a total rotation of these vectors W, which can be greater, than 2π.

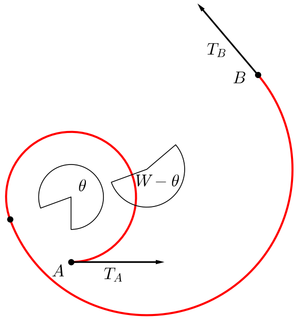
Of course one of the basic problems is to find a biarc for *two* circles. On the following picture (Figure 2) we can see the defining datas with some additional nominations supplemented with two circles. How can we determine θ to avoid intersections between arcs and circles?

The future goal is to analyse all of the families and determine necessary conditions to avoid intersections in each cases. In this paper we consider Case 1.3 (a) from [1]. In this case $r_A, r_B, \theta, W - \theta > 0$ and

$$
\begin{gathered}
2\pi < W < 4\pi, \\
2\pi + 2\alpha < W < 3\pi + \alpha, \\
-2\pi + W < \theta < 4\pi - W + 2\alpha.
\end{gathered}
$$



(a) Enhanced G^1 Hermite data with a biarc



(b) The sum of the two rotation angles must be W.

Figure 2

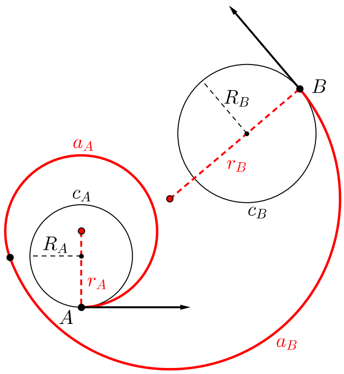


Figure 3

In [1] the authors mention positive and negative radii by the arcs, so we have to follow this convention by the circles. We can suppose that both radius of the circles are positiv, because we can determine the directions of the tangent vectors free by skinning problems. So the circles always can be placed at the "left side" of the tangent vectors. It is easy to see that we can avoid intersection between circle c_A and arc a_A with condition $r_A > R_A$. By similar arguments $r_B > R_B$ helps us to avoid intersection between a_B and c_B.

4. The calculation

4.1. $r_A > R_A$

We know that

$$r_A = \frac{r \cdot \sin\left(\frac{W+\theta}{2} - \alpha\right)}{2\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{W}{2}\right)} = r \cdot \frac{\sin\left(\frac{W}{2} - \alpha + \frac{\theta}{2}\right)}{2\sin\left(\frac{\theta}{2}\right)\sin\left(\frac{W}{2}\right)},$$

and

$$0 < \theta < 2\pi \Rightarrow \sin\frac{\theta}{2} > 0, \quad 2\pi < W < 4\pi \Rightarrow \sin\frac{W}{2} < 0, \quad 0 < \frac{\theta}{2} < \pi,$$

$$2\pi + 2\alpha < W < 3\pi + \alpha \quad \Rightarrow \quad \pi < \frac{W}{2} - \alpha < 2\pi \quad \Rightarrow \quad \sin\left(\frac{W}{2} - \alpha\right) < 0.$$

Now we should analyse the denominator to express θ from the following inequality:

$$\begin{aligned}
r_A &> R_A \\
\frac{r \cdot \sin\left(\frac{W+\theta}{2} - \alpha\right)}{2\sin\frac{\theta}{2}\sin\frac{W}{2}} &> R_A \\
\frac{\sin\left(\left(\frac{W}{2} - \alpha\right) + \frac{\theta}{2}\right)}{\sin\frac{\theta}{2}} &< \frac{2R_A\sin\frac{W}{2}}{r} \\
\frac{\sin\left(\frac{W}{2} - \alpha\right)\cdot\cos\frac{\theta}{2} + \cos\left(\frac{W}{2} - \alpha\right)\sin\frac{\theta}{2}}{\sin\frac{\theta}{2}} &< \frac{2R_A\sin\frac{W}{2}}{r} \\
\sin\left(\frac{W}{2} - \alpha\right)\cot\frac{\theta}{2} &< \frac{2R_A\sin\frac{W}{2}}{r} - \cos\left(\frac{W}{2} - \alpha\right) \\
\cot\frac{\theta}{2} &> \frac{\frac{2R_A\sin\left(\frac{W}{2}\right)}{r} - \cos\left(\frac{W}{2} - \alpha\right)}{\sin\left(\frac{W}{2} - \alpha\right)}
\end{aligned}$$

For brevity let us set

$$K \doteq \frac{\frac{2R_A\sin\left(\frac{W}{2}\right)}{r} - \cos\left(\frac{W}{2} - \alpha\right)}{\sin\left(\frac{W}{2} - \alpha\right)}.$$

We distinguish two cases according as $K > 0$ or $K < 0$.

1. If $K > 0$

$$\cot\frac{\theta}{2} = K \quad \Rightarrow \quad \tan\frac{\theta}{2} = \frac{1}{K} \quad \Rightarrow \quad \theta = 2\arctan\frac{1}{K}$$

$$\boxed{0 < \theta < 2\arctan\frac{1}{K}}$$

2. If $K < 0$

$$\frac{\theta}{2} = \arctan\frac{1}{K} + \pi \quad \Rightarrow \quad \theta = 2\arctan\frac{1}{K} + 2\pi$$

$$\boxed{0 < \theta < 2\arctan\frac{1}{K} + 2\pi}$$

4.2. $r_B > R_B$

We know from [1] that

$$r_B = r \cdot \frac{\sin\left(\alpha - \frac{\theta}{2}\right)}{2\sin\left(\frac{W-\theta}{2}\right)\sin\frac{W}{2}},$$

and

$$W - 2\pi < \theta < W \quad \Rightarrow \quad \frac{W}{2} - \pi < \frac{\theta}{2} < \frac{W}{2} \quad \Rightarrow \quad 0 < \frac{W}{2} - \frac{\theta}{2} < \pi \quad \Rightarrow$$
$$\Rightarrow \quad \sin\left(\frac{W}{2} - \frac{\theta}{2}\right) > 0.$$

Now our inequation is the following:

$$r \cdot \frac{\sin\left(\alpha - \frac{\theta}{2}\right)}{2\sin\left(\frac{W-\theta}{2}\right)\sin\frac{W}{2}} > R_B$$
$$\frac{\sin\left(\alpha - \frac{\theta}{2}\right)}{\sin\left(\frac{W}{2} - \frac{\theta}{2}\right)} < \frac{2R_B\sin\frac{W}{2}}{r}$$
$$\frac{\sin\alpha\cos\frac{\theta}{2} - \cos\alpha\sin\frac{\theta}{2}}{\sin\frac{W}{2}\cos\frac{\theta}{2} - \cos\frac{W}{2}\sin\frac{\theta}{2}} < \frac{2R_B\sin\frac{W}{2}}{r}$$

If we introduce notation $X_B \doteq \frac{2R_B\sin\frac{W}{2}}{r}$, the inequation is

$$\sin\alpha\cos\frac{\theta}{2} - \cos\alpha\sin\frac{\theta}{2} < X_B\sin\frac{W}{2}\cos\frac{\theta}{2} - X_B\cos\frac{W}{2}\sin\frac{\theta}{2}.$$

1. If $\cos\frac{\theta}{2} > 0$ ($\Leftrightarrow 0 < \theta < \pi$)

$$\sin\alpha - \cos\alpha\tan\frac{\theta}{2} < X_B\sin\frac{W}{2} - X_B\cos\frac{W}{2}\tan\frac{\theta}{2}$$
$$\tan\frac{\theta}{2}\left(X_B\cos\frac{W}{2} - \cos\alpha\right) < X_B\sin\frac{W}{2} - \sin\alpha$$

(i) If $X_B \cos\frac{W}{2} - \cos\alpha > 0$

$$\tan\frac{\theta}{2} < \frac{X_B \sin\frac{W}{2} - \sin\alpha}{X_B \cos\frac{W}{2} - \cos\alpha}$$

(a) If $X_B \sin\frac{W}{2} - \sin\alpha > 0$, since $0 < \frac{\theta}{2} < \frac{\pi}{2}$

$$\boxed{0 < \theta < 2\arctan\frac{X_B \sin\frac{W}{2} - \sin\alpha}{X_B \cos\frac{W}{2} - \cos\alpha}}$$

(b) If $X_B \sin\frac{W}{2} - \sin\alpha < 0$, we have a *contradiction* since $\tan\frac{\theta}{2} \not< 0$, if $0 < \frac{\theta}{2} < \frac{\pi}{2}$

(ii) If $X_B \cos\frac{W}{2} - \cos\alpha < 0$

$$\tan\frac{\theta}{2} > \frac{X_B \sin\frac{W}{2} - \sin\alpha}{X_B \cos\frac{W}{2} - \cos\alpha}$$

(a) If $X_B \sin\frac{W}{2} - \sin\alpha < 0$

$$\boxed{2\arctan\frac{X_B \sin\frac{W}{2} - \sin\alpha}{X_B \cos\frac{W}{2} - \cos\alpha} < \theta < \pi}$$

(b) If $X_B \sin\frac{W}{2} - \sin\alpha > 0$

$$\boxed{0 < \theta < \pi}$$

2. If $\cos\frac{\theta}{2} < 0$ (this means that $\pi < \theta < 2\pi$)

$$\tan\frac{\theta}{2}\left(X_B \cos\frac{W}{2} - \cos\alpha\right) > X_B \sin\frac{W}{2} - \sin\alpha$$

(i) If $X_B \cos\frac{W}{2} - \cos\alpha > 0$

$$\tan\frac{\theta}{2} > \frac{X_B \sin\frac{W}{2} - \sin\alpha}{X_B \cos\frac{W}{2} - \cos\alpha}$$

(a) If $X_B \sin\frac{W}{2} - \sin\alpha > 0$, we have a *contradiction.*

(b) If $X_B \sin\frac{W}{2} - \sin\alpha < 0$

$$\boxed{2\arctan\frac{X_B \sin\frac{W}{2} - \sin\alpha}{X_B \cos\frac{W}{2} - \cos\alpha} + 2\pi < \theta < 2\pi}$$

(ii) If $X_B \cos\alpha - \cos\alpha < 0$

$$\tan\frac{\theta}{2} < \frac{X_B \sin\frac{W}{2} - \sin\alpha}{X_B \cos\frac{W}{2} - \cos\alpha}$$

(a) If $X_B \sin \frac{W}{2} - \sin\alpha < 0$

$$\boxed{\pi < \theta < 2\pi}$$

(b) If $X_B \sin \frac{W}{2} - \sin\alpha > 0$

$$\boxed{\pi < \theta < 2\arctan \frac{X_B \sin \frac{W}{2} - \sin\alpha}{X_B \cos \frac{W}{2} - \cos\alpha} + 2\pi}$$

If we consider these conditions together, we get usable interval(s) to create appropriate biarc for two circles.

5. Conclusions

Now we have several conditions to avoid intersections by constructing a biarc from a special family for two circles. Of course we did not eliminate the cases, where we have intersections between a_A and c_B or a_B and c_A. The future goal is to extend the calculation for analysing these cases too, and the other cases from [1]. With these results together we can get a powerful basic to create skinning curves using biarcs.

References

[1] Meek, D.B., Walton, D.J., The family of biarcs that matches planar, two-point G^1 Hermite data, *Journal of Computational and Applied Mathematics*, 212 (2008) 31–45.

[2] Kunkli, R. Localization of touching points for interpolation of discrete circles, *Annales Mathematicae et Informaticae*, 36 (2009) 103–110.

[3] Kunkli, R., Hoffmann, M., Skinning of circles and spheres, *Computer Aided Geometric Design*, 27 (2010) 611–621.

[4] Sandel, G., Zur geometrie der Korbbögen, *Z. Ang. Math. Mech*, 15 (1937) 301–302.

[5] Bolton, K.M. Biarc curves, *Computer Aided Design*, 7 (1975) 89–92.

[6] Piegl, L., Tiller, W. Biarc approximation of NURBS curves, *Computer Aided Design*, 34 (2001) 807–814.

[7] Slabaugh, G., Unal, G., Fang, T., Rossignac, J., Whited, B. Variational Skinning of an Ordered Set of Discrete 2D Balls, *Geometric Modeling and Processing*, (2008) 450–461.

[8] Slabaugh, G., Whited, B., Rossignac, J., Fang, T., Unal, G. 3D Ball Skinning using PDEs for Generation of Smooth Tubular Surfaces, *Computer-Aided Design*, (2010) 18–26.

Annales Mathematicae et Informaticae
38 *(2011) pp. 95–98*
http://ami.ektf.hu

A note on Tribonacci-coefficient polynomials

Ferenc Mátyás[a], **László Szalay**[b]

[a]Institute of Mathematics and Informatics, Eszterházy Károly College
fmatyas@ektf.hu

[b]Institute of Mathematics, University of West Hungary
laszalay@emk.nyme.hu

Submitted May 18, 2011 Accepted November 21, 2011

Abstract

This paper shows, that the Tribonacci-coefficient polynomial $P_n(x) = T_2x^n + T_3x^{n-1} + \cdots + T_{n+1}x + T_{n+2}$ has exactly one real zero if n is odd, and $P_n(x)$ does not vanish otherwise. This improves the result in [1], which provides the upper bound 3 or 2 on the number of zeros of $P_n(x)$, respectively.

Keywords: linear recurrences, zeros of the polynomials with special coefficients

MSC: 11C08, 11B39

1. Introduction

The Fibonacci-coefficient polynomials $\mathcal{F}_n(x) = F_1x^n + F_2x^{n-1} + \cdots + F_nx + F_{n+1}$, $n \in \mathbb{N}^+$ were defined in [2]. The authors determined the number of real zeros of $\mathcal{F}_n(x)$. Generally, but with specific initial values, for binary recurrences and for linear recursive sequences of order $k \geq 2$ the question of the number of real zeros was investigated in [3] and [1], respectively.

As usual, the Tribonacci sequence is defined by the initial values $T_0 = 0$, $T_1 = 0$ and $T_2 = 1$, and by the recurrence relation $T_n = T_{n-1} + T_{n-2} + T_{n-3}$ $(n \geq 3)$. The Corollary 2 of Theorem 1 in [1] states that the possible number of negative zeros of the polynomial

$$P_n(x) = T_2x^n + T_3x^n + \cdots + T_{n+1}x + T_{n+2}$$

does not exceed three. More precisely, $P_n(x)$ possesses 0 or 2 negative zeros if n is even, and 1 or 3 negative zeros when n is odd. Obviously, there is no positive zero of $P_n(x)$, since all coefficients are positive.

The following theorem gives that the number of negative zeros is 0 or 1 depending on the parity of n.

Theorem 1.1. *The polynomial $P_n(x)$ has no real zero if n is even, while $P_n(x)$ possesses exactly one real zero, which is negative, if n is odd.*

In the proof, at the beginning we partially follow the approach of [1].

2. Proof of Theorem 1.1

Proof. Let $f(x) = x^3 - x^2 - x - 1$ denote the characteristic polynomial of the Tribonacci sequence. It is known, that $f(x)$ has one positive real zeros and a pair of complex conjugate zeros. Put

$$Q_n(x) = f(x)P_n(x) = x^{n+3} - T_{n+3}x^2 - (T_{n+2} + T_{n+1})x - T_{n+2}$$

(see Lemma 1 in [1]). Applying the Descartes' rule of signs, $Q_n(x)$ has one positive real zero, which obviously belongs to $f(x)$. (It hangs together with $P_n(x)$ possesses no positive real roots.)

To examine the negative roots, put $q_n(x) = Q_n(-x)$. In order to use Descartes' result again, we must distinguish two cases based on the parity of n.

First suppose that n is even. Now

$$q_n(x) = -x^{n+3} - T_{n+3}x^2 + (T_{n+2} + T_{n+1})x - T_{n+2},$$

and the number of changes of coefficients' signs predicts 2 or 0 positive zeros of $q_n(x)$. We are going to exclude the case of 2 zeros.

Clearly, $q_n(0) = -T_{n+2} < 0$, $q_n(1) = -T_{n+3} + T_{n+1} - 1 < 0$. Further, we have

$$q_n'(x) = -(n+3)x^{n+2} - 2T_{n+3}x + (T_{n+2} + T_{n+1}).$$

The values $q_n'(0) = T_{n+2} + T_{n+1} > 0$, $q_n'(1) = -(n+3) - 2T_{n+3} + T_{n+2} + T_{n+1} < 0$ show that the function $q_n(x)$ strictly monotone increasing locally in 0, while strictly monotone decreasing in 1. Since $q_n''(x) = -T_2(n+3)(n+2)x^{n+1} - 2T_{n+3}$ is negative for all non-negative $x \in \mathbb{R}$, then $q_n(x)$ is concave on $\mathbb{R}^+$. Consequently, if exist, the positive zeros of the polynomial $q_n(x)$ are in the interval $(0;1)$.

Therefore, to show that $q_n(x)$ does not cross the x-axes it is sufficient to prove that intersection point of the tangent lines $e\ :\ y = (T_{n+2} + T_{n+1})x - T_{n+2}$ and $f\ :\ y = (-(n+3) - 2T_{n+3} + T_{n+2} + T_{n+1})(x-1) - T_{n+3} + T_{n+1} - 1$ is under the x-axes. To reduce the calculations we simply justify that $x_0 > x_1$, where x_0 is defined by $e \cap x$-axes and x_1 is given by $f \cap x$-axes (see Figure 1).

First, $(T_{n+2} + T_{n+1})x - T_{n+2} = 0$ implies

$$x_0 = \frac{T_{n+2}}{T_{n+2} + T_{n+1}} > \frac{T_{n+2}}{T_{n+2} + T_{n+2}} = \frac{1}{2}.$$

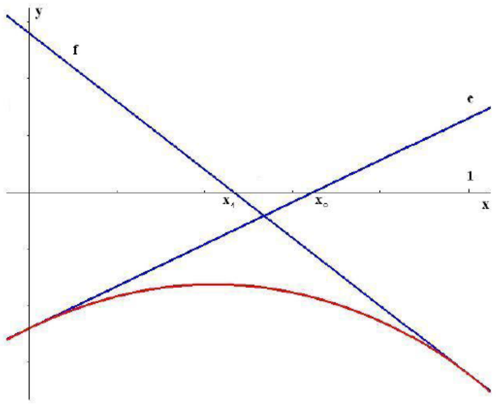


Figure 1

On the other hand,

$$x_1 = \frac{T_{n+3} - T_{n+1} + 1}{-(n+3) - 2T_{n+3} + T_{n+2} + T_{n+1}} + 1 \leq \frac{1}{2} \tag{2.1}$$

holds if $n \geq 5$. Indeed, (2.1) is equivalent to

$$\frac{1}{2} \leq \frac{T_{n+3} - T_{n+1} + 1}{(n+3) + 2T_{n+3} - T_{n+2} - T_{n+1}},$$

where both the numerator and the denominator are positive. Hence $n+1 \leq T_{n+2} - T_{n+1}$ remains to show, and it can be easily deduced, for example, by induction if $n \geq 5$.

The case $n = 4$ can be separately investigated. Now $T_5 = 4$, $T_6 = 7$, and $11x - 7 = 0$ provides $x_0 = 7/11$. Moreover, $T_7 = 13$ and $-22(x-1) - 10 = 0$ gives $x_1 = 6/11$. Thus $x_1 < x_0$.

Assume now, that n is odd. We partially repeat the procedure of the previous case.

The polynomial

$$q_n(x) = x^{n+3} - T_{n+3}x^2 + (T_{n+2} + T_{n+1})x - T_{n+2}$$

may have 3 or 1 positive zeros (by Descartes' rule of signs again).

Obviously, $q_n(0) = -T_{n+2} < 0$ and $q_n(1) = -T_{n+3} + T_{n+1} + 1 < 0$. Now

$$q_n'(x) = (n+3)x^{n+2} - 2T_{n+3}x + (T_{n+2} + T_{n+1}),$$

which together with $q_n'(0) = T_{n+2} + T_{n+1} > 0$, $q_n'(1) = (n+3) - 2T_{n+3} + T_{n+2} + T_{n+1} < 0$ implies the same monotonity behaviour in $(0;1)$ as before.

Since the equation $q_n''(x) = (n+3)(n+2)x^{n+1} - 2T_{n+3} = 0$ holds if and only if

$$x_{inf} = \sqrt[n+1]{\frac{T_{n+3}}{\binom{n+3}{2}}},$$

then $q_n(x)$ is concave on the interval $(0; x_{inf})$, and convex for $x > x_{inf}$. However, $x_{inf} > 1$ if $n \geq 9$, and in this case we can show that $q_n(x)$ does not intersect the x-axes in the interval $(0; 1)$ but there is exactly one zero if $x > 1$. The second part is an immediate consequence of the existence of unique positive inflection point $x_{inf} > 1$. Concentrating on the interval $(0; 1)$, similarly to the previous part $e : y = (T_{n+2} + T_{n+1})x - T_{n+2}$ and $f : y = ((n+3) - 2T_{n+3} + T_{n+2} + T_{n+1})(x - 1) - T_{n+3} + T_{n+1} + 1$ intersect each other under the x-axes, because of $x_0 > \frac{1}{2}$ holds again, and

$$x_1 = \frac{T_{n+3} - T_{n+1} - 1}{(n+3) - 2T_{n+3} + T_{n+2} + T_{n+1}} + 1 \leq \frac{1}{2}$$

follows, since $-(n+1) \leq T_{n+2} - T_{n+1}$.

For $n = 3$ or 5 or 7 we can easily check the required property. Thus the proof is complete. □

References

[1] Filep, F., Liptai, K., Mátyás, F., Tóth, J.T., Polynomials with special coefficients, *Ann. Math. Inf.*, 37 (2010), 101–106.

[2] Garth, D., Mills, D., Mitchell, P., Polynomials generated by the Fibonacci sequence, *J. Integer Sequences*, Vol. 10 (2007), Article 07.6.8.

[3] Mátyás,F., Further generalization of the Fibonacci-coefficient polynomials, *Ann. Math. Inf.*, 35 (2008), 123–128.

Annales Mathematicae et Informaticae
38 *(2011) pp. 99–110*
http://ami.ektf.hu

A clustering algorithm for multiprocessor environments using dynamic priority of modules

Pramod Kumar Mishra[a], **Kamal Sheel Mishra**[b]
Abhishek Mishra[c]

[a]Department of Computer Science & DST Centre for Interdisciplinary Mathematical Sciences, Banaras Hindu University, Varanasi-221005, India
e-mail: mishra@bhu.ac.in

[b]Department of Computer Science, School of Management Sciences, Varanasi-221011, India
e-mail: ksmishra@smsvaranasi.com

[c]Department of Computer Engineering, Institute of Technology, Banaras Hindu University, Varanasi-221005, India
e-mail: abhishek.rs.cse@itbhu.ac.in

Submitted March 3, 2011 Accepted October 25, 2011

Abstract

In this paper, we propose a task allocation algorithm on a fully connected homogeneous multiprocessor environment using dynamic priority of modules. This is a generalization of our earlier work in which we used static priority of modules. Priority of modules is dependent on the computation and the communication times associated with the module as well as the current allocation. Initially the modules are allocated in a single cluster. We take out the modules in decreasing order of priority and recalculate their priorities. In this way we propose a clustering algorithm of complexity $O(|V|^2(|V|+|E|)log(|V|+|E|))$, and compare it with Sarkar's algorithm.

Keywords: Clustering; Distributed Computing; Homogeneous Systems; Task Allocation.

MSC: 68W40, 68Q25.

1. Introduction

A homogeneous computing environment (HoCE) consists of a number of machines generally fully connected through a communication backbone. They consist of identical machines that are connected through identical communication links. In contrast, a heterogeneous computing environment (HeCE) consists of different types of machines as well as possibly different types of communication links (e.g., [8], [4], [13]). In the remainder of this paper our discussion is based on a HoCE that is fully connected and having an unlimited supply of machines.

A task to be executed on a HoCE may consist of a set of software modules having interdependencies between them. The interdependencies between software modules in a task can be represented as a task graph that is a weighted directed acyclic graph (DAG). The vertices in this DAG represent software modules and have a weight associated with them that represents the time of execution for the software module. The directed edges represent data dependencies between software modules. For example, if there is a directed edge of weight w_{ij} from module M_i to the module M_j, then this means that M_j can start its execution only when M_i has finished its execution and the data has arrived from M_i to M_j. The time taken for communication is 0 if M_i and M_j are allocated on the same machine, while it is w_{ij} in the case when M_i and M_j are allocated to different machines. We are using the same computational model that was used in the CASC algorithm by Kadamuddi and Tsai [7], in which a software module immediately starts sending its data simultaneously along its outgoing edges.

When all the software modules of a task are allocated to the same machine, then the time taken for completing the task is called sequential execution time. When the modules are distributed among more than one machine, then the time taken for completing the task is called parallel execution time. We use parallelism so that the parallel execution time can be less than the sequential execution time of a task.

The parallel execution time of a task may depend on the way in which the software modules of the task are allocated to the machines. The task allocation problem is to find an allocation so that the parallel execution time can be minimized. When the HoCE is fully connected, and having an unlimited supply of processors (as is in our case), the task allocation problem is also called the clustering problem in which we make clusters of modules and allocate them to different machines. The task allocation problem on a HeCE may consist of two steps. In the first step, a clustering of modules are found aiming at minimizing the parallel execution time of the task on a HoCE that is fully connected, and having an unlimited supply of machines. In the second step, the clusters are allocated to different machines so that the parallel execution time of the task on the given HeCE can be minimized.

The problem of finding a clustering of software modules of a task that takes minimum time is an NP-Complete problem (Sarkar [12], Papadimitriou [11]). So, for solving clustering problems in time that is polynomial in size of task graph, we need to develop some heuristic. The solution provided by using a polynomial time algorithm is generally suboptimal.

In our algorithm, the dynamic priority associated with a module is called the *DCCLoad* (Dynamic-Computation-Communication-Load). *DCCLoad* is approximately a measure of average difference between the module's computation and communication requirements according to the current allocation. Since the modules keep changing the clusters in our algorithm, we need to recalculate their priorities after each allocation. Using the concept of *DCCLoad*, we have developed a clustering algorithm of complexity $O(|V|^2(|V|+|E|)log(|V|+|E|))$.

The remainder of this paper is organized in the following manner. Section 2 discusses some heuristics for solving the clustering problem. Section 3 explains the concept of *DCCLoad*. Section 4 presents the DynamicCCLoad algorithm. Section 5 explains this algorithm with the help of a simple example. In section 6, some experimental results are presented. And finally in section 7, we conclude our work.

2. Current approaches

When the two modules that are connected through a large weight edge, are allocated to different machines, then this will make a large communication delay. To avoid large communication delays, we generally put such modules together on the same machine, thus avoiding the communication delay between them. This concept is called edge zeroing.

Two modules M_i and M_j are called independent if there cannot be a directed path from M_i to M_j as well as from M_j to M_i. A clustering where independent modules are clustered together is called nonlinear clustering. A linear clustering is the clustering in which independent modules are kept on separate clusters.

Sarkar's algorithm [12] uses the concept of edge zeroing for clustering the modules. Edges are sorted in decreasing order of edge weights. Initially each module is in a separate cluster. Edges are examined one-by-one in decreasing order of edge weight. The two clusters connected by the edge are merged together if on doing so, the parallel execution time does not increase. Sarkar's algorithm uses the level information to determine parallel execution time and the levels are computed for each step. This process is repeated till all the edges are examined. The complexity of Sarkar's algorithm is $O(|E|(|V|+|E|))$.

The dominant sequence clustering (DSC) algorithm by Yang and Gerasoulis [14], [15] is based on finding the critical path of the task graph. The critical path is called the dominant sequence (DS). An edge from the DS is used to merge its adjacent nodes, if the parallel execution time is reduced. After merging, a new DS is computed and the process is repeated again. DSC algorithm has a complexity of $O((|V|+|E|)log(|V|))$.

The clustering algorithm for synchronous communication (CASC) by Kadamuddi and Tsai [7], is an algorithm of complexity $O(|V|(|E|^2+log(|V|)))$. It has four stages of Initialize, Forward-Merge, Backward-Merge, and Early-Receive. In addition to achieving the traditional clustering objectives (reduction in parallel execution time, communication cost, etc.), the CASC algorithm reduces the performance degradation caused by synchronizations, and avoids deadlocks during clustering.

Mishra and Tripathi [10] consider the Sarkar's Edge Zeroing heuristic (Sarkar [12]) for scheduling precedence constrained task graphs on parallel systems as a priority based algorithm in which the priority is assigned to edges. In this case, the priority can be taken as the edge weight. They view this as a task dependent priority function that is defined for pairs of tasks. They have extended this idea in which the priority is a cluster dependent function of pairs of clusters (of tasks). Using this idea they propose an algorithm of complexity $O(|V||E|(|V|+|E|))$ and demonstrate its superiority over some well known algorithms.

3. Dynamic Computation-Communication Load of a module

3.1. Notation

We are using the notation of Mishra et al. [9] in which there are n modules $M_i(1 \leq i \leq n)$ where the module M_i is in the cluster $C_i(1 \leq i \leq n)$. The set of modules are given by:

$$M = \{M_i \mid 1 \leq i \leq n\} \tag{3.1}$$

The clusters $C_i \subset M(1 \leq i \leq n)$ are such that for $i \neq j(1 \leq i \leq n, 1 \leq j \leq n)$

$$C_i \bigcap C_j = \emptyset \tag{3.2}$$

and

$$\bigcup_{i=1}^{n} C_i = M \tag{3.3}$$

The label of the cluster C_i is denoted as an integer $cluster[i]$ $(1 \leq i \leq n, 1 \leq cluster[i] \leq n)$. The set of vertices of the task graph are denoted as:

$$V = \{i \mid 1 \leq i \leq n\} \tag{3.4}$$

The set of edges of the task graph are denoted as:

$$E = \{(i,j) \mid i \in V, j \in V, \exists \text{ an edge from } M_i \text{ to } M_j\} \tag{3.5}$$

m_i is the execution time of module M_i. If $(i,j) \in E$, then w_{ij} is the weight of the directed edge from M_i to M_j. If $(i,j) \notin E$, or if $i = j$, then w_{ij} is 0. T is the adjacency list representation of the task graph.

3.2. DCCLoad of a module

In our earlier work (Mishra et al. [9]), we used a static priority of modules that we called *Computation-Communication-Load* (*CCLoad*) of a module. *CCLoad* of a module was defined as follows:

$$CCLoad_i = m_i - max\_in_i - max\_out_i, \tag{3.6}$$

where

$$max\_in_i = MAX(\{w_{ji} \mid 1 \leq j \leq n\}) \tag{3.7}$$

and

$$max\_out_i = MAX(\{w_{ik} \mid 1 \leq k \leq n\}) \tag{3.8}$$

Now we are generalizing this concept so that we can also include the current allocation into the priority of modules. Since the allocation keeps changing in our algorithm, the priority will be dynamic. We will call it the *Dynamic-Computation-Communication-Load* (*DCCLoad*) of a module.

DCCLoad of a module is defined as follows:

$$DCCLoad_i = (c\_in_i + c\_out_i)m_i - sum\_in_i - sum\_out_i, \tag{3.9}$$

where

$$c\_in_i = \sum_{cluster[j] \neq cluster[i], 1 \leq j \leq n} 1 \tag{3.10}$$

$$c\_out_i = \sum_{cluster[i] \neq cluster[k], 1 \leq k \leq n} 1 \tag{3.11}$$

$$sum\_in_i = \sum_{cluster[j] \neq cluster[i], 1 \leq j \leq n} w_{ji} \tag{3.12}$$

and

$$sum\_out_i = \sum_{cluster[i] \neq cluster[k], 1 \leq k \leq n} w_{ik} \tag{3.13}$$

For calculating $DCCLoad_i$ of a module M_i, we first multiply its execution time (m_i) with the number of those incoming edges from, and outgoing edges to, ($c\_in_i + c\_out_i$) that are allocated on different clusters from M_i. Then we subtract the result by the sum of weight of incoming edges that are allocated on different clusters (sum_in_i) subtracted by the sum of weight of outgoing edges that are allocated on different clusters (sum_out_i).

3.3. An example of DCCLoad

In Figure 1 (taken from Mishra et al. [9]), *DCCLoad* of modules are calculated. As an example, for module M_2, we have:

$$m_2 = 4 \tag{3.14}$$

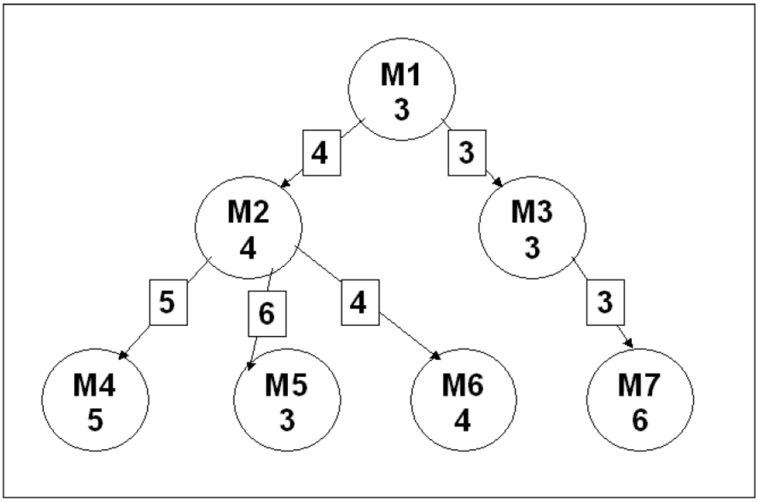


Figure 1: An example task graph for showing the calculation of *DCCLoad* for the allocation $\{M_1, M_3, M_7\}\{M_2, M_6\}\{M_4, M_5\}$. $(DCCLoad_i)_{1\leq i\leq 7}$ = (-1, -3, 0, 0, -3, 0, 0)

The number of incoming edges that are from different clusters are:

$$c\_in_2 = 1 \tag{3.15}$$

The number of outgoing edges that are to different clusters are:

$$c\_out_2 = 2 \tag{3.16}$$

The sum of weight of incoming edges that are from different clusters are:

$$sum\_in_2 = w_{13} = 4 \tag{3.17}$$

The sum of weight of outgoing edges that are to different clusters are:

$$sum\_out_2 = w_{24} + w_{25} = 11 \tag{3.18}$$

Therefore $DCCLoad_2$ is given by:

$$DCCLoad_2 = (c\_in_2 + c\_out_2)m_2 - sum\_in_2 - sum\_out_2 = 12 - 4 - 11 = -3 \tag{3.19}$$

4. The DynamicCCLoad algorithm

4.1. Evaluate-DCCLoad

Evaluate-DCCLoad$(T, cluster)$
01 **for** $i \leftarrow 1$ to $|V|$

```
02     do c_in[i] ← 0
03     c_out[i] ← 0
04     sum_in[i] ← 0
05     sum_out[i] ← 0
06 for i ← 1 to |V|
07     do load[i].index ← i
08     for each (i, j) ∈ E
09        do if cluster[i] ≠ cluster[j]
10           then f ← 1
11           else f ← 0
12        c_in[j] ← c_in[j] + f
13        c_out[i] ← c_out[i] + f
14        sum_in[j] ← sum_in[j] + f w_ij
15        sum_out[i] ← sum_out[i] + f w_ij
16 for i ← 1 to |V|
17     do load[i].value ← (c_in[i] + c_out[i]) m_i − sum_in[i] − sum_out[i]
18 return load
```

Given a task graph T, the algorithm EVALUATE-DCCLOAD calculates the *DCC Load* for each module in the array *load*. Using the notation of Mishra et al. [9], for $(1 \leq j \leq |V|)$, if the *DCCLoad* of module M_j is l_j, and if it is stored in *load*$[i]$, then we have:

$$load[i].value = l_j \tag{4.1}$$

and

$$load[i].index = j \tag{4.2}$$

In lines 01 to 05, the count ($c\_in[i]$) and the sum of weights of incoming edges from different clusters ($sum\_in[i]$), and the count ($c\_out[i]$) and the sum of weight of outgoing edges to different clusters ($sum\_out[i]$) are initialized to 0. In lines 06 to 15, we consider each edge $(i, j) \in E$, and update the values of $c\_out[i]$, $c\_in[j]$, $sum\_out[i]$ and $sum\_in[j]$ accordingly. Finally, in lines 16 to 17, we store the *DCCLoad* of module M_i in *load*$[i]$ for $(1 \leq i \leq |V|)$. Line 18 returns the *load* array.

Lines 01 to 05, and lines 16 to 17 each have complexity $O(|V|)$. Lines 06 to 15 have complexity $O(|E|)$. Line 18 has complexity $O(1)$. Therefore, the algorithm EVALUATE-DCCLOAD has complexity $O(|V| + |E|)$.

4.2. EVALUATE-TIME

Given a task graph T, and a clustering *cluster*, the algorithm EVALUATE-TIME taken from Mishra et al. [9] calculates the parallel execution time of the clustering. It is basically based on the event queue model. There are two type of events: computation completion event, and communication completion event. Events are denoted as 3-tuples (i, j, t). As an example, a computation completion event of module M_i, that completes its computation at time t_i will be denoted as (i, i, t_i), and a communication completion event of a communication from M_i to M_j, that is finished at time t_{ij} will be denoted as (i, j, t_{ij}).

There are a total of $(|V| + |E|)$ events out of which $|V|$ events are computation completion events corresponding to each module, and $|E|$ events are communication completion events corresponding to each edge. Mishra et al. [9] has shown the complexity of the Evaluate-Time algorithm as $O((|V| + |E|)log(|V| + |E|))$.

4.3. DynamicCCLoad Algorithm

```
DynamicCCLoad(T)
01 for j ← 1 to |V|
02     do cluster[j] ← 1
03 load ← Evaluate-DCCLoad(T, cluster)
04 Sort-Load(load)
05 c_max ← 2
06 for j ← 1 to |V|
07     do i ← 1
08     t_min ← Evaluate-Time(T, cluster)
09     for k ← 2 to c_max
10         do cluster[load[j].index] ← k
11         time ← Evaluate-Time(T, cluster)
12         if time < t_min
13             then t_min ← time
14             i ← k
15     cluster[load[j].index] ← i
16     load ← Evaluate-DCCLoad(T, cluster)
17     load[j].value ← −∞
18     Sort-Load(load)
19     if i = c_max
20         then c_max ← c_max + 1
21 return (t_min, cluster)
```

We are using the heuristic of Mishra et al. [9]:

(1) We can keep the computational intensive tasks on separate clusters because they mainly involve computation. Such tasks will heavily load the cluster. If we keep these tasks separated, we can evenly balance the computational load.

(2) We can keep the communication intensive tasks on same cluster because they mainly involve communication. If we keep these tasks on the same cluster, we may reduce the communication delays through edge-zeroing.

The DCCLoad-Clustering algorithm implements the above heuristic using the concept of *DCCLoad*. Initially all modules are kept in the same cluster (cluster 1, also called the *initial cluster*, lines 01 to 02). Given a task graph T, and an initial allocation of modules *cluster*, line 03 evaluates the *DCCLoad* of modules. Line 04 sorts the *load* array in decreasing order. c_{max} (line 05) will be the number of possible clusters that can result, if one module is removed from the *initial cluster*, and put on a different cluster (including the *initial cluster*).

We take the modules out from the *initial cluster* one-by-one (line 06) in decreasing order of $CCLoad$ (line 10). At the same time we also calculate the parallel execution time, when it is put on all possible different clusters (lines 09 to 11). t_{min} is used to record the minimum parallel execution time, and i is used to record the corresponding cluster (lines 12 to 14). Finally we put the module on the cluster that gives the minimum parallel execution time (line 15). In line 16 we also set its $DCCLoad$ value to $-\infty$ to make it invalid so that in future we can not use it. Line 17 re-evaluates the $DCCLoad$ of modules after the change in allocation and line 18 again sorts them in decreasing order.

It may also happen that the parallel execution time was minimum when the module was put alone on a new cluster. In this case we will have to increment c_{max} by 1 (lines 19 to 20). Line 21 finally returns the parallel execution time, and the corresponding clustering.

Lines 01 to 02 have complexity $O(|V|)$. Line 03 has complexity $O(|V| + |E|)$. Line 04 has complexity $O(|V|^2)$ if bubble sort is used [6]. Lines 05 and 21 each have complexity $O(1)$. Lines 08 and 11 have complexity $O((|V| + |E|)log(|V| + |E|))$. For each iteration of the **for** loop in line 06, Evaluate-Time (lines 08 and 11) is called a maximum of $|V|$ times (c_{max} can have a maximum value of $|V|$, when all modules are on separate clusters). The complexity of the **for** loop of lines 06 to 20 is dominated by Evaluate-Time that is called a maximum of $|V|^2$ times. Therefore, the **for** loop has complexity $O(|V|^2(|V|+|E|)log(|V|+|E|))$ that is also the complexity of DynamicCCLoad algorithm.

5. A simple example

Consider the task graph in Figure 2 (taken from Mishra et al. [9]). Initially all modules will be clustered in the *initial cluster* as $(cluster[i])_{1\leq i\leq 4} = (1, 1, 1, 1)$. Parallel execution time will be 8. For the initial allocation we have $(DCCLoad_i)_{1\leq i\leq 4} = (0, 0, 0, 0)$. Then the modules are sorted according to $DCCLoad$ in decreasing order as (M_1, M_2, M_3, M_4).

The first module to be taken out is M_1 which forms the clustering (2, 1, 1, 1). Parallel execution time for this clustering is 9. This is not less than 8. Therefore, module M_1 is kept back in the *initial cluster* as (1, 1, 1, 1). For this allocation we re-calculate $DCCLoad$. After setting the value of $DCCLoad_1$ to $-\infty$ so that it can not be used in future, we get $(DCCLoad_i)_{1\leq i\leq 4} = (-\infty, 0, 0, 0)$. The modules sorted in decreasing order are: (M_2, M_3, M_4, M_1).

We next take out the module M_2 to form the clustering (1, 2, 1, 1). Parallel execution time for this clustering is 8. This is also not less than 8. Therefore, module M_2 is kept back in the *initial cluster* as (1, 1, 1, 1). For this allocation we re-calculate $DCCLoad$. After setting the value of $DCCLoad_2$ to $-\infty$ so that it can not be used in future, we get $(DCCLoad_i)_{1\leq i\leq 4} = (-\infty, -\infty, 0, 0)$. Modules sorted in decreasing order are: (M_3, M_4, M_2, M_1).

We next take out the module M_3 to form the clustering (1, 1, 2, 1). Parallel execution time for this clustering is 7. This is less than 8. Therefore, module M_3

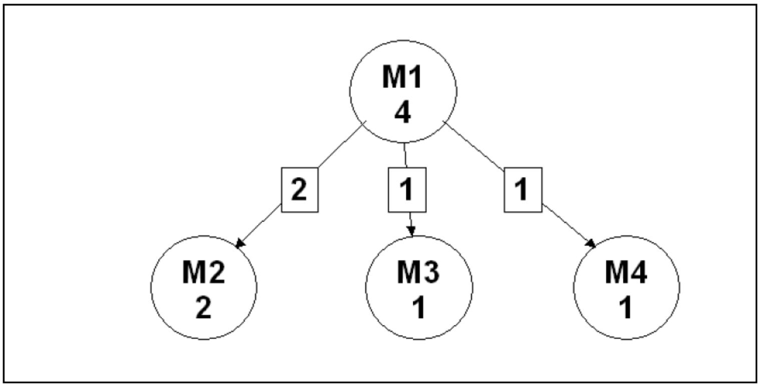


Figure 2: An example task graph for explaining the DYNAMICCCLOAD algorithm. For the initial allocation we have $(DCCLoad_i)_{1\leq i\leq 4} = (0, 0, 0, 0)$. The DYNAMICCCLOAD algorithm clusters the modules as $(M_1, M_2)(M_3)(M_4)$, giving a parallel execution time of 6.

is kept in a *separate cluster* as (1, 1, 2, 1). For this allocation we re-calculate *DCCLoad*. After setting the value of $DCCLoad_3$ to $-\infty$ so that it can not be used in future, we get $(DCCLoad_i)_{1\leq i\leq 4} = (-\infty, -\infty, -\infty, 0)$. Modules sorted in decreasing order are: (M_4, M_3, M_2, M_1).

The last module to be taken out is M_4. Now there are two possible clustering: (1, 1, 2, 2) and (1, 1, 2, 3). Parallel execution time for the clustering (1, 1, 2, 2) is 7. Parallel execution time for the clustering (1, 1, 2, 3) is 6. The minimum parallel execution time comes out to be 6 for the clustering (1, 1, 2, 3) that is also less than 7. Therefore, module M_4 is also kept in a *separate cluster* as (1, 1, 2, 3). For this allocation we re-calculate *DCCLoad*. After setting the value of $DCCLoad_4$ to $-\infty$ so that it can not be used in future, we get $(DCCLoad_i)_{1\leq i\leq 4} = (-\infty, -\infty, -\infty, -\infty)$. Modules sorted in decreasing order are: (M_4, M_3, M_2, M_1). At this point the DYNAMICCCLOAD algorithm stops.

The final clustering of modules is $(M_1, M_2)(M_3)(M_4)$ in which the modules M_1 and M_2 are clustered together, while the modules M_3 and M_4 are kept on separate clusters. This clustering gives a parallel execution time of 6.

6. Experimental results

The DYNAMICCCLOAD algorithm is compared with the Sarkar's edge zeroing algorithm [12]. This algorithm has a complexity of $O(|E|(|V| + |E|))$.

Algorithms are tested on benchmark task graphs of Tatjana and Gabriel [3], [2]. We have tested for 120 task graphs having number of nodes: 50, 100, 200, and 300 respectively. Each task graph has a label as *tn_i_j.td*. Here n is the number

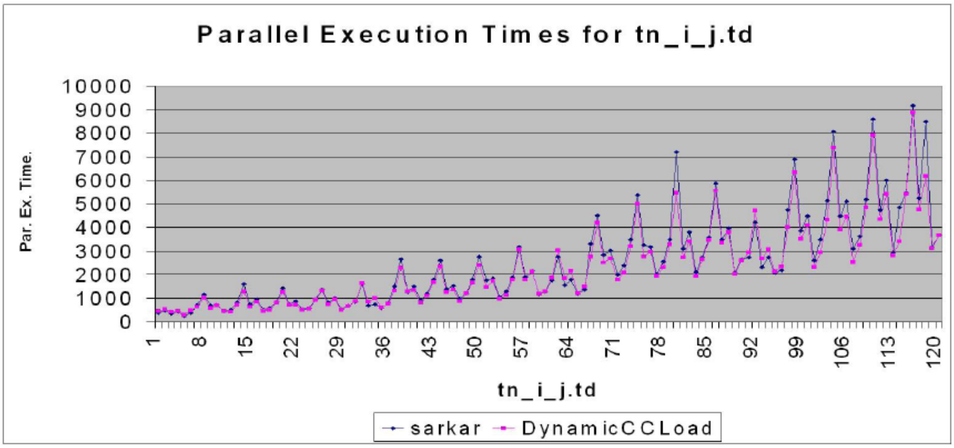


Figure 3: Parallel execution times for tn_i_j.td.

of nodes. i is a parameter depending on the edge density. Its possible values are: 20, 40, 50, 60, and 80. For each combination of n and i, there are 6 task graphs that are indexed by j. j ranges from 1 to 6. Therefore, for each n, there are 30 task graphs.

For the values of n having 50, 100, 200, and 300, Figure 3 shows the comparison between the Sarkar's edge zeroing algorithm and the DynamicCCLoad algorithm for the parallel execution time. It is evident from the figures that the average improvement of DynamicCCLoad algorithm over Sarkar's edge zeroing algorithm ranges from 5.81% for 100-node task graphs to 8.30% for 300-node task graphs.

7. Conclusion

We developed the idea of *DCCLoad* of a module by including the current allocation of modules. This resulted in a dynamically changing priority of modules. We used a heuristic based on it to develop the DynamicCCLoad algorithm of complexity $O(|V|^2(|V| + |E|)log(|V| + |E|))$. We also demonstrated its superiority over the Sarkar's edge zeroing algorithm in terms of parallel execution time. For the future work there are two possibilities: experiment with different dynamic priorities, and experiment with different ways in which we can take the modules out from the initial cluster.

Acknowledgements. The authors are thankful to the anonymous referees for valuable comments and suggestions in revising the manuscript to the present form. Corresponding author greatly acknowledge the financial assistance received under sponsored research project from the CSIR, New Delhi.

References

[1] Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C., *Introduction to Algorithms*, The MIT Press, 2'nd Edition, (2001).

[2] Davidovic, T., Benchmark task graphs available online at: `http://www.mi.sanu.ac.rs/~tanjad/sched_results.htm`, (2006).

[3] Davidovic, T., Crainic, T. G., Benchmark-problem instances for static scheduling of task graphs with communication delays on homogeneous multiprocessor systems, *Computers and Operations Research*, 33(8), (2006) 2155–2177.

[4] Freund, R. F., Siegel, H. J., Heterogeneous processing, *IEEE Computer*, 26(6), (1993) 13–17.

[5] Horowitz, E., Sahni, S., Rajasekaran, S., *Fundamentals of Computer Algorithms*, W. H. Freeman, (1998).

[6] Langsam, Y., Augenstein, M. J., Tenenbaum, A. M., *Data Structures Using C and C++*, Prentice Hall, 2'nd edition, (1996).

[7] Kadamuddi, D., Tsai, J. J. P., Clustering algorithm for parallelizing software systems in multiprocessors environment, *IEEE Transations on Software Engineering*, 26(4), (2000) 340–361.

[8] Maheswaran, M., Braun, T. D., Siegel, H. J., Heterogeneous distributed computing, *J.G. Webster (Ed.), Encyclopedia of Electrical and Electronics Engineering*, 8, (1999) 679–690.

[9] Mishra, P. K., Mishra, K. S., Mishra, A., A clustering heuristic for multiprocessor environments using computation and communication loads of modules, *International Journal of Computer Science & Information Technology*, 2(5), (2010) 170–182.

[10] Mishra, A., Tripathi, A. K., An extension of edge zeroing heuristic for scheduling precedence constrained task graphs on parallel systems using cluster dependent priority scheme, *Journal of Information and Computing Science*, 6(2), (2011) 83–96. *An extended abstract of this paper appears in the Proceedings of IEEE International Conference on Computer and Communication Technology (ICCCT'10), (2010) 647-651.*

[11] Papadimitriou, C. H., Yannakakis, M., Towards an architecture-independent analysis of parallel algorithms, *SIAM Journal on Computing*, 19(2), (1990) 322–328.

[12] Sarkar, V., *Partitioning and Scheduling Parallel Programs for Multiprocessors*, Research Monographs in Parallel and Distributed Computing, MIT Press, (1989).

[13] Siegel, H. J., Dietz, H. G., Antonio, J. K., Software support for heterogeneous computing, *A. B. Tucker Jr. (Ed.), The Computer Science and Engineering Handbook, CRC Press, Boca Raton, FL*, (1997) 1886–1909.

[14] Yang, T., Gerasoulis, A., A fast static scheduling algorithm for DAGs on an unbounded number of processors, *In Proceedings of the 1991 ACM/IEEE Conference on Supercomputing (ICS'91)*, (1991) 633–642.

[15] Yang, T., Gerasoulis, A., PYRROS: Static task scheduling and code generation for message passing multiprocessors, *In Proceedings of the 6'th International Conference on Supercomputing (ICS'92)*, (1992) 428–437.

Annales Mathematicae et Informaticae
***38** (2011) pp. 111–116*
http://ami.ektf.hu

Arithmetic progressions on Huff curves

Dustin Moody

National Institute of Standards and Technology (NIST), Gaithersburg
e-mail: dmoody25@gmail.com

Submitted January 10, 2011 Accepted March 19, 2011

Abstract

We look at arithmetic progressions on elliptic curves known as Huff curves. By an arithmetic progression on an elliptic curve, we mean that either the x or y-coordinates of a sequence of rational points on the curve form an arithmetic progression. Previous work has found arithmetic progressions on Weierstrass curves, quartic curves, Edwards curves, and genus 2 curves. We find an infinite number of Huff curves with an arithmetic progression of length 9.

Keywords: Diophantine equations, arithmetic progressions, elliptic curves

MSC: 11G05, 11B25

1. Introduction

Recently, several researchers have looked at arithmetic progressions on elliptic curves. Bremner [3], Campbell [4], Garcia-Selfa and Tornero [8] used elliptic curves given by a Weierstrass equation, while Campbell [4], MacLeod [12], and Ulas [15] have looked at quartic models. Moody [13] has studied the problem on Edwards curves. Alvarado [1] and Ulas [16] have extended similar results to genus 2 hyperelliptic curves. The historical motivation for this problem is discussed in [8].

Besides Weierstrass equations, quartic curves, and Edwards curves [6], there are other models for elliptic curves. These include Jacobi intersections [5], Hessian curves [10], and Huff curves [9], for example. Originally introduced in 1948, Huff curves have recently been shown to have applications in cryptography [11], [7]. An elliptic curve in Huff's model can be written as

$$H_{a,b} : x(ay^2 - 1) = y(bx^2 - 1).$$

In this work, we look at *arithmetic progressions on Huff curves.* By this we mean a sequence of rational points $(x_1, y_1), \ldots, (x_n, y_n)$ on $H_{a,b}$ with the x_i forming an

Source	Model	Longest progression	Longest progression for infinite family
[3],[4]	Weierstrass curves	8	8
This work	Huff curves	9	9
[13]	Edwards curves	9	9
[2],[12],[15]	quartic curves	14	12
[1],[16]	genus 2 quintics	12	12
[16]	genus 2 sextics	18	16

Table 1: Longest arithmetic progressions on curves

arithmetic progression. The main result of this paper is to show several infinite families of Huff curves with arithmetic progressions of length 9. In comparison, Table 1 gives the length of the longest arithmetic progression for the previously mentioned models. Note in general the length increases as we have more variables in the defining curve equation we can specify.

2. Arithmetic progressions

Huff curves are elliptic curves that can be written as $x(ay^2-1)=y(bx^2-1)$, when $ab(a-b)\neq 0$. Clearly we have symmetry in x and y if we switch a and b, so we only look for arithmetic progressions on the x-coordinates. Note trivially that the point $(0,0)$ is always on the curve. Notice also that an arithmetic progression of x-coordinates of the form $\{-kd,-(k-1)d,\dots,-d,0,d,2d,\dots,(k-1)d,kd\}$ can always be rescaled so that $d=1$. This is seen as follows. If the point (jd,y) is on the curve $H_{a,b}$, then the point $(j,y/d)$ is on the curve H_{ad^2,bd^2}. As a consequence, we will focus on finding Huff curves which have x-coordinates in the set $\{\pm1,\pm2,\pm3,\pm4\}$.

We will repeatedly need the following calculation. If we require a rational point (x,y) on $H_{a,b}$ with $x=n$, then we must have that $any^2-(bn^2-1)y-n=0$. In order for $y\in\mathbb{Q}$, the discriminant $(bn^2-1)^2+4an^2$ must be a rational square. Applying this to $x=1$, we need $(b-1)^2+4a=j^2$ for some rational j. The same equation is true for $x=-1$. Similarly, if we require rational points with x-coordinate ±2 and ±3, then we must have $(4b-1)^2+16a=k^2$, and $(9b-1)^2+36a=l^2$ for some rational k and l. Solving for a in our first equation, we have

$$a=\frac{1}{4}\left(j^2-(b-1)^2\right). \tag{2.1}$$

Eliminating a from the other two equations, we are left with the system

$$12b^2+4j^2-k^2=3, \tag{2.2}$$

$$72b^2+9j^2-l^2=8. \tag{2.3}$$

We now parameterize the solutions in terms of b and a parameter m. Some easy algebra verifies that $j = 3b^2 - 1$ and $k = 6b^2 - 1$ is a solution to (2.2). Let $j = 3b^2 - 1 + t$ and $k = 6b^2 - 1 + mt$. Substituting these values into (2.2) yields

$$t\left((m^2-4)t + 12mb^2 - 24b^2 - 2m + 8\right) = 0.$$

Solving for t, we see $t = -2\frac{(6b^2-1)m-4(3b^2-1)}{m^2-4}$, and thus

$$j = \frac{(3b^2-1)m^2 - 2(6b^2-1)m + 4(3b^2-1)}{m^2-4}, \tag{2.4}$$

$$k = \frac{-(6b^2-1)m^2 + 8(3b^2-1)m - 4(6b^2-1)}{m^2-4}.$$

We substitute this expression for j into (2.3) and seek a rational solution for l. Some more algebra shows that this is equivalent to

$$81(m-2)^4b^4 + 18(m-2)^2(m^2+22m+4)b^2 + m^4 - 36m^3 + 172m^2 - 144m + 16 \tag{2.5}$$

being a rational square. Considering this as a polynomial in b, we first check to see what values of m will lead to the constant term being square. The equation $E : v^2 = m^4 - 36m^3 + 172m^2 - 144m + 16$ clearly has the rational point $(0, 4)$, and so determines an elliptic curve. Using SAGE [14], the curve E is found to have rank 0, and torsion points $(0, \pm 4), (1, \pm 3), (2, \pm 12), (4, \pm 12)$, and $(-2, \pm 36)$. We exclude $m = \pm 2$, as this leads to division by 0 in the expressions for j and k. When $m = 1$ or $m = 4$, then (2.5) is not the square of a polynomial in b. When $m = 0$, then (2.5) is $16(9b^2 + 1)^2$.

So letting $m = 0$, we have $j = -(3b^2 - 1)$, and $a = \frac{1}{4}b(3b-2)(3b-1)(b+1)$ by (2.1). With this expression for a, then the curve $H_{a,b}$ has an arithmetic progression of length 7, namely $x = -3, -2, -1, 0, 1, 2, 3$. In order for $x = \pm 4$ to be a rational point, we are led to the discriminant $144b^4 + 144b^2 + 1$ needing to be a square. As the curve

$$E_1 : v^2 = 144b^4 + 144b^2 + 1$$

clearly has rational point $(0, 1)$, then E_1 is an elliptic curve. By SAGE, this curve has rank 2 with generators $(\frac{1}{12}, \frac{17}{12})$, and $(\frac{1}{8}, \frac{29}{16})$. Each rational point on E_1 leads to a value for b so that the Huff curve $H_{a,b}$ has an arithmetic progression of length 9. We thus have our first infinite family of Huff curves with a progression of length 9.

3. More families

Returning to (2.5),we consider it as a polynomial in m,

$$\begin{aligned}(9b^2+1)^2m^4 - 36(18b^4 - 9b^2 + 1)m^3 + 4(486b^4 - 360b^2 + 43)m^2 \\ - 144(18b^4 - 9b^2 + 1)m + 16(9b^2+1)^2.\end{aligned} \tag{3.1}$$

If we compare this to

$$\Big((9b^2+1)m^2 - \frac{18(18b^4-9b^2+1)}{9b^2+1}m + 4(9b^2+1)\Big)^2,$$

the difference is

$$\frac{160m^2(324b^4-45b^2+1)}{(9b^2+1)^2}.$$

If the difference is equal to 0, then (3.1) is a square. The case $m=0$ was already examined. The other zeroes are when $b=\pm\frac{1}{3},\pm\frac{1}{6}$. Letting $b=-\frac{1}{3}$, then

$$a = -\frac{(3m-4)(m-3)(m+1)(m+4)}{9(m^2-4)^2}.$$

The condition that $x=\pm 4$ is the coordinate of a rational point is equivalent to the corresponding discriminant being a rational square; i.e. we seek a rational point on the curve

$$E_2 : v^2 = 169m^4 - 128m^3 - 264m^2 - 512m + 2704.$$

The choice of $b=\frac{1}{3}$ leads to the same curve. Similarly, when $b=\pm\frac{1}{6}$, we are led to the curve

$$E_3 : v^2 = 46m^4 - 440m^3 + 1968m^2 - 1760m + 736.$$

Both E_2 and E_3 are elliptic curves with rank 2 and 1 respectively. These ranks were computed by SAGE. Each rational point on one of the curves leads to a Huff curve with a rational point having x-coordinate ± 4, and thus a progression of length 9.

By experimentation, we found a few other infinite families. Using the same parameterization as above, let $b=\pm\frac{1}{4}$ or $\pm\frac{1}{8}$. Then it can be checked that $x=\pm 4$ is the x-coordinate of a rational point on the Huff curve $H_{a,b}$ with a determined by (2.1) and (2.4). However, we are no longer guaranteed that $x=\pm 3$ is on the Huff curve. Requiring $x=\pm 3$, we arrive at the following curves

$$E_4 : v^2 = 625m^4 - 4680m^3 + 22936m^2 - 18720m + 10000, \qquad (b=\pm 1/4)$$

$$E_5 : v^2 = 5329m^4 - 127368m^3 + 614296m^2 - 509472m + 85624. \qquad (b=\pm 1/8)$$

These elliptic curves have ranks 1 and 2, leading to two more infinite families of Huff curves with progressions of length 9.

Finally, letting $b=\pm\frac{1}{2}$ the parameterized Huff curve is $H_{a,\pm 1/2}$, with

$$a = -\frac{(3m-2)(m-6)}{64(m-2)^2}. \tag{3.2}$$

The condition that there is a rational point with $x=\pm 3$ leads to a quadratic, instead of a quartic as in previous cases:

$$v^2 = 169m^2 - 604m + 676. \tag{3.3}$$

A parametric solution to (3.3) is given by

$$m = -\frac{4(13s + 151)}{s^2 - 169},$$

$$v = -\frac{2(13s^2 + 302s + 2197)}{s^2 - 169}.$$

Substituting this expression for m into (3.2), and requiring $x = \pm 4$ we have the curve

$$E_6 : r^2 = 46s^4 + 2288s^3 + 42124s^2 + 335712s + 1017846,$$

which has rank 1. Each rational point of E_6 gives a rational s, which in turn determines a rational m and a. The curve $H_{a,\pm 1/2}$ will have rational points with x-coordinates ± 3 and ± 4.

4. Conclusion

In the previous section, we produced six infinite families of Huff curves having the property that each has rational points with x-coordinate $x = -4, -3, -2, -1, 0, 1, 2, 3, 4$. This produces an arithmetic progression of length 9. We have performed computer searches to see if we can find any rational points on these curves leading to $x = \pm 5$ being the x-coordinate of a rational point on $H_{a,b}$. So far these searches have failed to turn up such a point. It is therefore an open problem to find a Huff curve with an arithmetic progression of length 10 (or longer). It would also be interesting to investigate arithmetic progressions on the remaining models of elliptic curves.

Acknowledgments. We would like to thank the anonymous referee for noticing a few minor mistakes in our formulas.

References

[1] Alvarado, A., An arithmetic progression on quintic curves, *J. Integer Seq.*, Paper 09.7.3 (2009).

[2] Alvarado, A., Arithmetic progressions on quartic elliptic curves, *Ann. Math. Inform.*, 37 (2010) 3–6.

[3] Bremner, A., On arithmetic progressions on elliptic curves, *Experiment. Math.*, 8 (1999), 409–413.

[4] Campbell, G., A note on arithmetic progressions on elliptic curves, *J. Integer Seq.*, Paper 03.1.3, (2003).

[5] Chudnovsky, D. and Chudnovsky, G., Sequences of numbers generated by addition in formal groups and new primality and factorization tests, *Adv. App.Math.*, 7 (1986), 385–434.

[6] Edwards, H., A normal form for elliptic curves, *Bull. Amer. Math. Soc.*, 44 (2007), 393–422.

[7] Feng, R. and Wu, H., Elliptic curves in Huff's model, available at `http://eprint.iacr.org/2010/390.pdf`, (2010).

[8] García-Selfa, I. and Tornero, J., Searching for simultaneous arithmetic progressions on elliptic curves, *Bull. Austral. Math. Soc.*, 71 (2005), 417–424.

[9] Huff, G., Diophantine problems in geometry and elliptic ternary forms, *Duke Math. J.*, 15 (1948), 443–453.

[10] Joye, M. and Quisquater, J., Hessian elliptic curves and side-channel attacks, in Ç.K. Koç, D. Naccache, and C. Paar, eds., *Proceedings of Cryptographic Hardware and Embedded Systems - CHES 2001*, Springer-Verlag, (2001), 402–410.

[11] Joye, M., Tibouchi, M., and Vergnaurd, D., Huff's model for elliptic curves, in *Algorithmic Number Theory Symposium (ANTS-IX) proceedings*, LNCS 6197, Springer, (2010), 234–250.

[12] MacLeod, A., 14-term arithmetic progressions on quartic elliptic curves, *J. Integer Seq.*, Paper 06.1.2, (2006).

[13] Moody, D., Arithmetic progressions on Edwards curves, *J. Integer Seq.*, Paper 11.1.7, (2011).

[14] Stein, W. et al., Sage Mathematics Software, The Sage Development Team, (2010), `http://www.sagemath.org`.

[15] Ulas, M., A note on arithmetic progressions on quartic elliptic curves, *J. Integer Seq.*, Paper 05.3.1, (2005).

[16] Ulas, M., On arithmetic progressions on genus two curves, *Rocky Mountain J. Math.*, 39 (2009), 971–980.

Annales Mathematicae et Informaticae
***38** (2011) pp. 117–122*
http://ami.ektf.hu

Upper bounds on van der Waerden type numbers for some second order linear recurrence sequences

Gábor Nyul*, Bettina Rauf

Institute of Mathematics
University of Debrecen
H–4010 Debrecen P.O.Box 12, Hungary
e-mail: gnyul@math.unideb.hu, raufbetti@gmail.com

Submitted July 23, 2011 Accepted October 25, 2011

Abstract

For suitable integers α, γ and $f : [3, +\infty[\cap \mathbb{Z} \to [0, +\infty[\cap \mathbb{Z}$, denote by $w(\mathcal{R}_{\alpha,\gamma,f}, k, r)$ the least positive integer such that for any r-colouring of $[1, w(\mathcal{R}_{\alpha,\gamma,f}, k, r)] \cap \mathbb{Z}$, there exists a monochromatic finite sequence $(x_1, \dots, x_k)$ satisfying $x_i = (\alpha a_i + 2)x_{i-1} + (\gamma a_i - 1)x_{i-2}$ with some integers $a_i = 0$ or $a_i \geq f(i)$ $(i = 3, \dots, k)$. In the present paper we describe the possible values of α and γ. We also derive an upper bound on $w(\mathcal{R}_{\alpha,\gamma,f}, k, 2)$ in these cases. This gives a generalization of a result of B. M. Landman [3].

Keywords: van der Waerden type numbers, linear recurrence sequences

MSC: 05D10, 11B37

1. Introduction

Most results of Ramsey theory in the area of number theory deal with monochromatic sequences or monochromatic solutions of diophantine equations, systems of diophantine equations (for an extensive survey see [4]). In this paper we study the monochromatic properties of some second order linear recurrence sequences.

Let $\mathcal{S}$ be a non-empty set of sequences of positive integers. On a finite sequence of $\mathcal{S}$ of length k we mean the first k elements of a sequence from $\mathcal{S}$. For integers $k \geq 3$ and $r \geq 2$, let $w(\mathcal{S}, k, r)$ be the least positive integer if it exists, such that

*Research was supported in part by Grant 75566 from the Hungarian Scientific Research Fund.

for any r-colouring of $[1, w(\mathcal{S}, k, r)] \cap \mathbb{Z}$, there is a monochromatic finite sequence of $\mathcal{S}$ of length k. We call $w(\mathcal{S}, k, r)$ a van der Waerden type number.

Throughout this paper by arithmetic progression we mean a strictly increasing arithmetic progression of positive integers and denote their set by $\mathcal{A}$. By the classical theorem of B. L. van der Waerden [6], $w(\mathcal{A}, k, r)$ exists for arbitrary k, r. We will use the standard notation $w(k, r)$ for $w(\mathcal{A}, k, r)$.

Obviously, if $\mathcal{S}_1$ and $\mathcal{S}_2$ are non-empty sets of sequences of positive integers such that $\mathcal{S}_1 \subseteq \mathcal{S}_2$ and $w(\mathcal{S}_1, k, r)$ exists, then $w(\mathcal{S}_2, k, r)$ also exists and $w(\mathcal{S}_2, k, r) \leq w(\mathcal{S}_1, k, r)$. In particular, if $\mathcal{S}$ is a non-empty set of sequences of positive integers with $\mathcal{A} \subseteq \mathcal{S}$, then $w(\mathcal{S}, k, r)$ exists and $w(\mathcal{S}, k, r) \leq w(k, r)$.

In our paper we consider the case of linear recurrence sequences. Remark that we can describe $\mathcal{A}$ by a linear recurrence, namely $\mathcal{A}$ is the set of sequences $(x_i)_{i=1}^{\infty}$ satisfying $x_i = 2x_{i-1} - x_{i-2}$ $(i = 3, 4, \ldots)$ with some positive integers $x_1 < x_2$.

Denote by $\mathcal{F}$ the set of strictly increasing sequences of positive integers satisfying the Fibonacci recurrence, that is

$$\mathcal{F} = \{(x_i)_{i=1}^{\infty} \mid x_1 < x_2 \text{ positive integers}, x_i = x_{i-1} + x_{i-2} \ (i = 3, 4, \ldots)\}.$$

H. Harborth, S. Maasberg [2] and H. Ardal, D. S. Gunderson, V. Jungić, B. M. Landman, K. Williamson [1] proved that $w(\mathcal{F}, k, r)$ exists if and only if $k = 3$. The previous authors also examined other binary recurrences. A forthcoming paper of G. Nyul and B. Rauf [5] studies the existence of van der Waerden type numbers for higher order linear recurrence sequences.

B. M. Landman [3] (see also [4], Section 3.6) considered van der Waerden type numbers for three families of some second order linear recurrence sequences, containing $\mathcal{A}$ as a subset. He gave an upper bound for them when $r = 2$. In [4] at the end of Section 3.6, the authors suggest to investigate some similar families of sequences.

The purpose of our paper is to study this question, but not only for some new separate families. We describe all possible families of sequences and give an upper bound for the corresponding van der Waerden type numbers. As we shall see, the three families and the results of B. M. Landman [3] are special cases of our general ones.

2. Description of our families of sequences

Let $\alpha, \gamma \in \mathbb{Z}$, not both zero, and let $f : [3, +\infty[\cap \mathbb{Z} \to [0, +\infty[\cap \mathbb{Z}$. Denote by $\mathcal{R}_{\alpha,\gamma,f}$ the family of sequences $(x_i)_{i=1}^{\infty}$ with positive integers $x_1 < x_2$, satisfying $x_i = (\alpha a_i + 2)x_{i-1} + (\gamma a_i - 1)x_{i-2}$ for some integers a_i where $a_i = 0$ or $a_i \geq f(i)$ $(i = 3, 4, \ldots)$.

Later on we will also consider the special case when f is identically 0. For this we introduce the notation $\mathcal{R}_{\alpha,\gamma} = \mathcal{R}_{\alpha,\gamma,f}$.

According to the slightly different parametrization given by B. M. Landman [3] for families $\mathcal{R}_{0,1,f}$, $\mathcal{R}_{1,0,f}$, $\mathcal{R}_{1,-1,f}$, more generally we could set $\alpha, \beta, \gamma, \delta, A \in \mathbb{Z}$, α, γ

not both zero, such that $\alpha A+\beta = 2$, $\gamma A+\delta = -1$ and $g : [3, +\infty[\cap\mathbb{Z} \to [A, +\infty[\cap\mathbb{Z}$ and consider the collection of sequences $(x_i)_{i=1}^{\infty}$ with positive integers $x_1 < x_2$, satisfying the recurrence $x_i = (\alpha b_i + \beta)x_{i-1} + (\gamma b_i + \delta)x_{i-2}$ where $b_i = A$ or $b_i \geq g(i)$ is an integer ($i = 3, 4, \ldots$). Note that in fact this is not a more general family of sequences, because it can be reparametrized to $\mathcal{R}_{\alpha,\gamma,f}$ with $g(i) = f(i)+A$ and $b_i = a_i + A$.

The van der Waerden type number $w(\mathcal{R}_{\alpha,\gamma,f}, k, r)$ is meaningful only if each element of $\mathcal{R}_{\alpha,\gamma,f}$ consists of positive integers. But in this case $w(\mathcal{R}_{\alpha,\gamma,f}, k, r)$ always exists, since $\mathcal{A} \subseteq \mathcal{R}_{\alpha,\gamma,f}$ (with the choice $a_i = 0$), moreover $w(\mathcal{R}_{\alpha,\gamma,f}, k, r) \leq w(k, r)$. Thus it is natural to prove the following statement.

Proposition 2.1. *Each element of $\mathcal{R}_{\alpha,\gamma,f}$ contains only positive integers if and only if $\alpha \geq 0$, $\gamma > 0$ or $\alpha > 0$, $\gamma \leq 0$, $\alpha \geq |\gamma|$.*

Proof.

I. First let $\alpha \geq 0$ and $\gamma > 0$. In this case we prove by induction that each element $(x_i)_{i=1}^{\infty}$ of $\mathcal{R}_{\alpha,\gamma,f}$ is strictly increasing. It follows from the assumption that $x_1 < x_2$. If we suppose $x_{i-1} < x_i$, then $x_{i+1}-x_i = (\alpha a_{i+1}+1)x_i+(\gamma a_{i+1}-1)x_{i-1} \geq x_i - x_{i-1} > 0$ since $\alpha a_{i+1} + 1 \geq 1$ and $\gamma a_{i+1} - 1 \geq -1$.

In the case $\alpha > 0$, $\gamma \leq 0$, $\alpha \geq |\gamma|$ we can prove it similarly by induction and using $x_{i+1} - x_i = (\alpha a_{i+1} + 1)x_i + (\gamma a_{i+1} - 1)x_{i-1} \geq (|\gamma| a_{i+1} + 1)$ $(x_i - x_{i-1}) > 0$.

II. In the remaining cases we can find a sequence from $\mathcal{R}_{\alpha,\gamma,f}$ which contains a negative number.

In the case $\alpha < 0$, let $x_1 = 1$. Then we have $x_3 = (\alpha x_2 + \gamma)a_3 + 2x_2 - 1$. If x_2 is sufficiently large, then $\alpha x_2 + \gamma < 0$, hence by choosing a sufficiently large a_3, x_3 is negative.

If $\alpha = 0$ and $\gamma < 0$, we get similarly with the choice $x_1 = 1$ that $x_3 = \gamma a_3 + 2x_2 - 1$, which is negative for sufficiently large a_3.

Finally consider $\alpha > 0$, $\gamma < 0$, $\alpha < |\gamma|$, and let $x_2 = x_1 + 1$. Now $x_3 = ((\alpha+\gamma)x_1+\alpha)a_3+x_1+2$ holds. If x_1 is sufficiently large, then $(\alpha+\gamma)x_1+\alpha < 0$, which gives $x_3 < 0$ with a sufficiently large a_3. □

3. Upper bounds on van der Waerden type numbers

Now we prove our main result, an upper bound on van der Waerden type numbers for $\mathcal{R}_{\alpha,\gamma,f}$ when the number of colours is 2.

Theorem 3.1.
***Case 1**: If $\alpha \geq 0$ and $\gamma > 0$, then*

$$w(\mathcal{R}_{\alpha,\gamma,f}, k, 2) \leq w(\mathcal{R}_{\alpha,\gamma,f}, 3, 2) \prod_{j=4}^{k} [(\alpha + \gamma)f(j) + (\alpha + \gamma)j - \alpha - \gamma + 1].$$

Case 2: *If $\alpha > 0$, $\gamma \leq 0$ and $\alpha \geq |\gamma|$, then*

$$w(\mathcal{R}_{\alpha,\gamma,f}, k, 2) \leq w(\mathcal{R}_{\alpha,\gamma,f}, 3, 2) \prod_{j=4}^{k} (\alpha f(j) + \alpha j - \alpha + 2).$$

Proof. For brevity let us use the notation $C_{\alpha,\gamma,f}(k)$ for the right-hand sides of the inequalities. We prove the theorem by induction on k. It is obvious for $k = 3$. Suppose that it is true for $k - 1$ ($k \geq 4$) and prove it for k.

Let χ be an arbitrary 2-colouring of $[1, C_{\alpha,\gamma,f}(k)] \cap \mathbb{Z}$ with colours red and blue. By the induction hypothesis there exists a $(k-1)$-term monochromatic finite sequence $(x_1, \ldots, x_{k-1})$ of $\mathcal{R}_{\alpha,\gamma,f}$ under the colouring χ with elements $x_1, \ldots, x_{k-1} \leq C_{\alpha,\gamma,f}(k-1)$, say it is red.

Let $y_i = [\alpha(f(k) + i - 1) + 2]x_{k-1} + [\gamma(f(k) + i - 1) - 1]x_{k-2}$ $(i = 1, \ldots, k)$. In both cases $y_1 < \ldots < y_k$, $y_i > x_{k-1}$ and $y_i \leq [\alpha(f(k) + k - 1) + 2]x_{k-1} + [\gamma(f(k) + k - 1) - 1]x_{k-2}$ using the assumptions on α and γ. In Case 1 the numbers in brackets are positive and $x_{k-2}, x_{k-1} \leq C_{\alpha,\gamma,f}(k-1)$, hence $y_i \leq [(\alpha + \gamma)f(k) + (\alpha + \gamma)k - \alpha - \gamma + 1]C_{\alpha,\gamma,f}(k-1) = C_{\alpha,\gamma,f}(k)$. In Case 2 the first number in brackets is positive and the other is negative, which gives similarly $y_i \leq [\alpha(f(k) + k - 1) + 2]x_{k-1} \leq [\alpha(f(k) + k - 1) + 2]C_{\alpha,\gamma,f}(k-1) = C_{\alpha,\gamma,f}(k)$. This means $y_i \in [1, C_{\alpha,\gamma,f}(k)] \cap \mathbb{Z}$.

Now we have two possibilities: If some y_i $(i = 1, \ldots, k)$ is red, then $(x_1, \ldots, x_{k-1}, y_i)$ is a red finite sequence from $\mathcal{R}_{\alpha,\gamma,f}$ of length k having elements in the desired interval. On the other hand, if each y_i $(i = 1, \ldots, k)$ is blue, then $(y_1, \ldots, y_k)$ is a k-term monochromatic finite arithmetic progression, hence a finite sequence of $\mathcal{R}_{\alpha,\gamma,f}$ with elements in $[1, C_{\alpha,\gamma,f}(k)] \cap \mathbb{Z}$. □

If f is identically 0, we have the following immediate consequence:

Corollary 3.2.
Case 1: *If $\alpha \geq 0$ and $\gamma > 0$, then*

$$w(\mathcal{R}_{\alpha,\gamma}, k, 2) \leq \frac{w(\mathcal{R}_{\alpha,\gamma}, 3, 2)}{(\alpha + \gamma + 1)(2\alpha + 2\gamma + 1)} \prod_{j=1}^{k} [(\alpha + \gamma)j - \alpha - \gamma + 1].$$

Case 2: *If $\alpha > 0$, $\gamma \leq 0$ and $\alpha \geq |\gamma|$, then*

$$w(\mathcal{R}_{\alpha,\gamma}, k, 2) \leq \frac{w(\mathcal{R}_{\alpha,\gamma}, 3, 2)}{2(\alpha + 2)(2\alpha + 2)} \prod_{j=1}^{k} (\alpha j - \alpha + 2).$$

4. Examples

Finally we show some examples with the most interesting possible values of α and γ. Examples 1 and 2 belong to Case 1, while Examples 3, 4 and 5 belong to Case 2.

We notice that Examples 1, 3 and 4 were the original families treated by B. M. Landman [3].

In each example we describe the recurrence, but omit the conditions of $f : [3, +\infty[\cap \mathbb{Z} \to [0, +\infty[\cap \mathbb{Z}$, and $a_i = 0$ or $a_i \geq f(i)$, since they are common in all cases. Additionally we give a possible reparametrization of the recurrence, together with the corresponding value of A with our earlier notation. (In Examples 2 and 5, $n!!$ denotes the semifactorial of a natural number n.)

Example 1: $\alpha = 0$, $\gamma = 1$.
Recurrence: $x_i = 2x_{i-1} + (a_i - 1)x_{i-2}$
Reparametrization: $x_i = 2x_{i-1} + b_i x_{i-2}$ $(A = -1)$
Upper bounds:

$$w(\mathcal{R}_{0,1,f}, k, 2) \leq w(\mathcal{R}_{0,1,f}, 3, 2) \prod_{j=4}^{k} (f(j) + j)$$

$$w(\mathcal{R}_{0,1}, k, 2) \leq \frac{7}{6} k!, \text{ since } w(\mathcal{R}_{0,1}, 3, 2) = 7.$$

Example 2: $\alpha = 1$, $\gamma = 1$.
Recurrence: $x_i = (a_i + 2)x_{i-1} + (a_i - 1)x_{i-2}$
Reparametrization: $x_i = (b_i + 3)x_{i-1} + b_i x_{i-2}$ $(A = -1)$
Upper bounds:

$$w(\mathcal{R}_{1,1,f}, k, 2) \leq w(\mathcal{R}_{1,1,f}, 3, 2) \prod_{j=4}^{k} (2f(j) + 2j - 1)$$

$$w(\mathcal{R}_{1,1}, k, 2) \leq \frac{3}{5} (2k-1)!!, \text{ since } w(\mathcal{R}_{1,1}, 3, 2) = 9.$$

Example 3: $\alpha = 1$, $\gamma = 0$.
Recurrence: $x_i = (a_i + 2)x_{i-1} - x_{i-2}$
Reparametrization: $x_i = b_i x_{i-1} - x_{i-2}$ $(A = 2)$
Upper bounds:

$$w(\mathcal{R}_{1,0,f}, k, 2) \leq w(\mathcal{R}_{1,0,f}, 3, 2) \prod_{j=4}^{k} (f(j) + j + 1)$$

$$w(\mathcal{R}_{1,0}, k, 2) \leq \frac{1}{3} (k+1)!, \text{ since } w(\mathcal{R}_{1,0}, 3, 2) = 8.$$

Example 4: $\alpha = 1$, $\gamma = -1$.
Recurrence: $x_i = (a_i + 2)x_{i-1} + (-a_i - 1)x_{i-2}$
Reparametrization: $x_i = b_i x_{i-1} + (-b_i + 1)x_{i-2}$ $(A = 2)$
Upper bounds:

$$w(\mathcal{R}_{1,-1,f}, k, 2) \leq w(\mathcal{R}_{1,-1,f}, 3, 2) \prod_{j=4}^{k} (f(j) + j + 1)$$

$$w(\mathcal{R}_{1,-1},k,2)\le\frac{7}{24}(k+1)!,\ \text{since}\ w(\mathcal{R}_{1,-1},3,2)=7.$$

Example 5: $\alpha=2$, $\gamma=-1$.
Recurrence: $x_i=(2a_i+2)x_{i-1}+(-a_i-1)x_{i-2}$
Reparametrization: $x_i=2b_ix_{i-1}-b_ix_{i-2}$ $(A=1)$
Upper bounds:

$$w(\mathcal{R}_{2,-1,f},k,2)\le w(\mathcal{R}_{2,-1,f},3,2)\prod_{j=4}^{k}(2f(j)+2j)$$

$$w(\mathcal{R}_{2,-1},k,2)\le\frac{3}{16}(2k)!!,\ \text{since}\ w(\mathcal{R}_{2,-1},3,2)=9.$$

References

[1] H. Ardal, D. S. Gunderson, V. Jungić, B. M. Landman and K. Williamson, Ramsey results involving the Fibonacci numbers, *Fibonacci Quarterly* **46/47** (2008/2009), 10–17.

[2] H. Harborth and S. Maasberg, Rado numbers for Fibonacci sequences and a problem of S. Rabinowitz, in: Applications of Fibonacci Numbers (eds. G. E. Bergum, A. N. Philippou and A. F. Horadam), Volume 6, Kluwer Academic Publishers, 1996, 143–153.

[3] B. M. Landman, Ramsey functions associated with second order recurrences, *Journal of Combinatorial Mathematics and Combinatorial Computing* **15** (1994), 119–127.

[4] B. M. Landman and A. Robertson, Ramsey Theory on the Integers, American Mathematical Society, 2004.

[5] G. Nyul and B. Rauf, On the existence of van der Waerden type numbers for linear recurrence sequences with constant coefficients, manuscript.

[6] B. L. van der Waerden, Beweis einer Baudetschen Vermutung, *Nieuw Archief voor Wiskunde* **15** (1927), 212–216.

Annales Mathematicae et Informaticae
38 *(2011) pp. 123–126*
http://ami.ektf.hu

A new recursion relationship for Bernoulli Numbers

Abdelmoumène Zékiri, Farid Bencherif

Faculty of Mathematics, USTHB, Algiers, Algeria
e-mail: azekiri@usthb.dz;czekiri@gmail.com,
fbencherif@usthb.dz;fbencherif@yahoo.fr

Submitted March 01, 2011 Accepted September 14, 2011

Abstract

We give an elementary proof of a generalization of the Seidel-Kaneko and Chen-Sun formula involving the Bernoulli numbers.

Keywords: Bernoulli numbers, Bernoulli polynomials, Integer sequences.

MSC: 11B68, 11B83

1. Introduction

The Bernoulli Numbers B_n, $n = 0, 1, 2, \ldots$ are defined by the exponential generating function:

$$B(z) = \frac{z}{e^z - 1} = \sum_{n=0}^{\infty} B_n \frac{z^n}{n!}. \tag{1.1}$$

As (1.1) implies that $B(-z) = z + B(z)$, we have:

$$(-1)^n B_n = B_n + \delta_1^n, \text{ for } n \geq 0. \tag{1.2}$$

where the notation δ_i^n is the classical Kronecker symbol which equals 1 if $n = i$ and 0 otherwise. Consequently, we have $B_1 = -\dfrac{1}{2}$, and $B_n = 0$, when n is odd and $n \geq 3$. Let us define $\epsilon_n := \dfrac{1 + (-1)^n}{2}$, thus:

$$\epsilon_n B_n = B_n + \frac{1}{2}\delta_1^n, \text{ for } n \geq 0. \tag{1.3}$$

Note that the Bernoulli polynomials can be defined by the following function:

$$B(x,z) := \frac{ze^{xz}}{e^z-1} = \sum_{n=0}^{\infty} B_n(x)\frac{z^n}{n!}.$$

Thus, we have:

$$\sum_{n=0}^{\infty} B_n(x)\frac{z^n}{n!} = \left(\sum_{n=0}^{\infty} B_n\frac{z^n}{n!}\right)\left(\sum_{n=0}^{\infty} x^n\frac{z^n}{n!}\right).$$

Therefore the polynomial $B_n(x)$ satisfies the following equality:

$$B_n(x) = \sum_{k=0}^{n} \binom{n}{k} x^{n-k} B_k. \tag{1.4}$$

We note also that:

$$B(x+1,z) - B(x,z) = \sum_{n=0}^{\infty} \left(B_n(x+1) - B_n(x)\right)\frac{z^n}{n!} = ze^{xz}.$$

Consequently, we deduce the following property of $B_n(x)$:

$$B_n(x+1) - B_n(x) = nx^{n-1}, \text{ for } n \geq 1. \tag{1.5}$$

In this paper, we are extending the well-known following formulae involving Bernoulli Numbers. First, the Seidel formula (1877) [4], re-discovered later by Kaneko [3] (1995):

$$\sum_{k=0}^{n} \binom{n+1}{k} (n+k+1)\, B_{n+k} = 0, \text{ for } n \geq 1.$$

And secondly, the Chen-Sun formula [1] (2009):

$$\sum_{k=0}^{n+3} \binom{n+3}{k} (n+k+3)\,(n+k+2)(n+k+1)B_{n+k} = 0. \tag{1.6}$$

Our main result consists on the following:

Theorem 1.1. *For given odd natural q and for natural number $n \geq 0$, we have the equality:*

$$\sum_{k=0}^{n+q} \binom{n+q}{k} (n+k+q)\,(n+k+q-1)\cdots(n+k+1)\, B_{n+k} = 0. \tag{1.7}$$

Obviously, this result gives the Seidel-Kaneko formula when $q = 1$, and the Chen-Sun formula when $q = 3$.

2. Proof of the main result

For a given odd number q and for an integer number $n \geq 0$, we consider the polynomials:

$$H(x) = \frac{1}{2}x^{n+q}(x-1)^{n+q},$$

and

$$K(x) = \sum_{k=0}^{n+q} \frac{\epsilon_{n+k}}{(n+q+k+1)} \binom{n+q}{k} \left(B_{n+q+k+1}(x) - B_{n+q+k+1}\right). \tag{2.1}$$

By the binomial theorem, we deduce:

$$H(x) = \frac{1}{2}\sum_{k=0}^{n+q}(-1)^{n+k+1}\binom{n+q}{k}x^{n+q+k}, \tag{2.2}$$

and

$$H(x+1) = \frac{1}{2}\sum_{k=0}^{n+q}\binom{n+q}{k}x^{n+q+k}. \tag{2.3}$$

Thus, by using the equality property (1.5), we verify that:

$$K(x+1) - K(x) = H(x+1) - H(x) = \sum_{k=0}^{n+q}\epsilon_{n+k}\binom{n+q}{k}x^{n+q+k}. \tag{2.4}$$

Moreover

$$K(0) = H(0) = 0. \tag{2.5}$$

Then, (2.2), (2.3), (2.4) and (2.5) imply:

$$K(x) = H(x).$$

If $[x^n]P(x)$ denotes the coefficient of x^n in the polynomial $P(x)$, we can write:

$$[x^{q+1}]K(x) = [x^{q+1}]H(x). \tag{2.6}$$

So, from (1.4)

$$[x^{q+1}]K(x) = \sum_{k=0}^{n} \frac{\epsilon_{n+k}B_{n+k}}{(n+q+k+1)}\binom{n+q}{k}\binom{n+q+k+1}{q+1}, \tag{2.7}$$

and from (2.2), we have:

$$[x^{q+1}]H(x) = \frac{1}{2}\binom{n+q}{1-n}. \tag{2.8}$$

From (1.3), we know that:

$$\epsilon_{n+k}B_{n+k} = B_{n+k} + \frac{1}{2}\delta_{1-n}^{k}. \tag{2.9}$$

Since

$$\begin{aligned}\sum_{k=0}^{n+q}\frac{\delta_{1-n}^{k}}{2(n+q+k+1)}\binom{n+q}{k}\binom{n+q+k+1}{q+1} &= \frac{1}{2(q+1)}\binom{n+q}{1-n}\binom{q+1}{q}\\ &= \frac{1}{2}\binom{n+q}{1-n}. \end{aligned}\tag{2.10}$$

We deduce, from (2.7), (2.9) and (2.10) that:

$$[x^{q+1}]K(x) = \sum_{k=0}^{n+q}\frac{B_{n+k}}{(n+q+k+1)}\binom{n+q}{k}\binom{n+q+k+1}{q+1} + \frac{1}{2}\binom{n+q}{1-n}. \tag{2.11}$$

It follows from (2.6), (2.8) and (2.11) that:

$$\sum_{k=0}^{n+q}\frac{1}{(n+q+k+1)}\binom{n+q}{k}\binom{n+q+k+1}{q+1}B_{n+k} = 0, \tag{2.12}$$

and by multiplying by $(q+1)!$, we obtain, finally, the aimed result which is:

$$\sum_{k=0}^{n+q}\binom{n+q}{k}(n+k+q)(n+k+q-1)\dots(n+k+1)B_{n+k} = 0.$$

This ends our proof.

References

[1] Chen, W.Y.C., Sun, L.H., Extended Zeilberger's Algorithm for Identities on Bernoulli and Euler Polynomials, *J. Number Theory,* 129 No. 9 (2009) 2111-2132

[2] Cigler, J., q-Fibonacci polynomials and q-Genocchi numbers, arXiv:0908.1219v4 [math.CO].

[3] Kaneko, M., A recurrence formula for the Bernoulli numbers, *Proc. Japan Acad. Ser. A Math. Sci.*, 71 No.8 (1995) 192-193.

[4] Seidel, L., Über eine einfache Entstehungsweise der Bernoullischen Zahlen und einiger verwandten Reihen, *Sitzungsber. Münch. Akad. Math. Phys.* Classe (1877) 157-187.

[5] Wu, K.-J., Sun, Z.-W., Pan, H., Some identities for Bernoulli and Euler polynomials, *Fibonacci Quat.*, 42 (2004) 295-299.

Methodological papers

Annales Mathematicae et Informaticae
38 *(2011) pp. 129–136*
http://ami.ektf.hu

GeoGebra in fifth grade elementary mathematics at rural schools

László Budai

University of Debrecen
e-mail: urlbudai0912@gmail.com

Submitted April 19, 2011 Accepted December 21, 2011

Abstract

In the fifth grade math of elementary school demonstration, representation, and experimentation are of central importance. This can be achieved by dynamic softwares such as GeoGebra. The pivotal point, however, the timetable and the time to learn the management of the interface, navigation tools. It takes a couple of lessons, but in any case, the future prospects for this loss of time. It is also important to keep in mind that, in the case of topics that should be included in the GeoGebra, and on a what kind of manner. Here are the results I will describe in the light of the past 3 years.

Keywords: GeoGebra, ICT in mathematics

1. Introduction

According to George Pólya [1], the mathematics teacher must be a good trader: it is necessary to sale the goods to the customer at all costs, means the mathematics to the students. This can be possible by means of motivation. The motivation is one of the basic principles of didactic methods.

The mathematics teaching sustained intellectual work, which is essential to ensure the proper motivation. It is possible to select tasks that carry the possibility of interest and awareness. It is not only necessary to develop the motivation, but to strengthen it with a planned, determinate development. The inclusion of dynamic software in the learning-teaching process, with the assistance of interactive table, where possible, apparently can create a great motivational base in those schools, where it is difficult to motivate students.

There are several dynamic softwares, but in our experience, the following points have to be taken into account at he choice by the educational institution: if the program is available free of charge, i.e. to be accessible to all, Hungarian-language, cross-platform, easy-to-use, detailed help and available sample programs. GeoGebra [2] is a good and popular choice from this viewpoint.

The Hungarian-language Wiki GeoGebra [5] can provide worksheets, and the palette is very broad: those who want to deal with this issue, can found various samples and lots of information.

According to international research [3, 4] GeoGebra (more precisely dynamic geometry softwares) are the only tool, by which a higher (over 10%) improvement can be achieved in geometry teaching. Similar results of the relevance of GeoGebra have been found in Luxembourg [6, 7], Germany [8] and France [9] in a relatively longer term (3–5 year) within the framework of research projects, specifically at primary school age group.

The aim of this paper is to show the results of a longitudinal research in a rural school, where the effectiveness of GeoGebra-based teaching is proved with the help of a control group with classical teaching methods, even in the case of highly disadvantaged students.

2. The research environment

The school of our experimental research is a rural primary school in a small village, 80% of the students are from minorities, and 68% of the students are highly disadvantaged. The computer is already essentially a kind of motivation for these students. The tests are controlled design, the lineage system were performed: the class 5/a has been involved in the GeoGebra teaching-learning process, while the class 5/b (control group) has been performed by classical tools. At the beginning of the school years on the basis of input from the measurements in parallel classes students' performance were nearly the same in all cases. The following table shows the results:

	year 2008/2009	year 2009/2010	year 2010/2011
5/a	2,61	2,77	2,57
5/b	2,57	2,63	2,72

Table 1: Input measurements in math before application of GeoGebra

In the following section some thematic fields of mathematics will be discussed were GeoGebra has been proved to be extremely useful.

3. Topics and results

The following subjects have been successfully improved by GeoGebra:

1. introduction to geometry
2. geometric drawings
3. geometric transformations
4. Completion of four basic operations by integers or fractions.

3.1. Introduction to geometry

After the introduction of basic geometric concepts and grouping (shapes, flat, space, etc.), the teaching-learning process has been performed in the computer room in class 5/a. At this point 2-3 hours should have been devoted to the learning of the basics of GeoGebra software (the basic structure, interface options, points, lines, polygons, add sections, add zoom options, etc.). The first relevant GeoGebra program was as follows: a straight line through two points to add, and then locate the "end" of the line. GeoGebra worksheet is rather large, so the kids will eventually get bored of searching the straight "ends" of the line. In our experience this short practice is a very successful tool to introduce the concept of the infinity.

In the 5. grade textbook the concept of congruence is appeared in relation to the same shape and size of shapes. Many of the problems in the case of my students arise as to determine if the given shapes are congruent or not. In the GeoGebra it is possible to create an interactive, dynamic worksheet, which helps the children in the recognition process: the combination of congruent shapes (Figure 1).

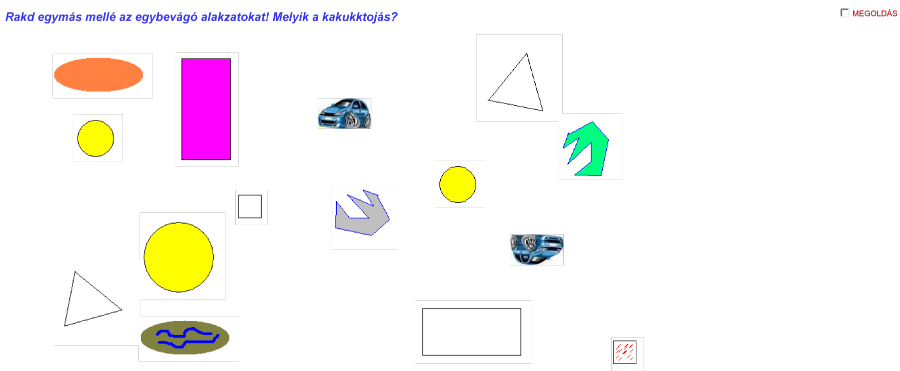


Figure 1: Determine the congruent shapes

Here the children turn on the automatic solution-control (which is a check box, and are good solutions for linking), can filter out the "disturbing" factors: the color of the shape, pattern, location, and to concentrate on the essentials, in all cases (shape and size, which is a mathematical point of view, it is important to us). The introduction of the congruence used to have playful worksheets for the students: find the difference. The two images appear to be congruent but a number of differences are to search in the images. They are very engaging, and the ability to acquire the use of GeoGebra, and these problems arise a doubt, the feeling of need to prove something (in this case, the solution can be achieved by activating a check box).

Is also worth to mention a game called geometric line-game. In addition, the development of reflection operations is a good tool at the beginning to learn the selection of the appropriate toolbars in GeoGebra (c.f. Fig 2.).

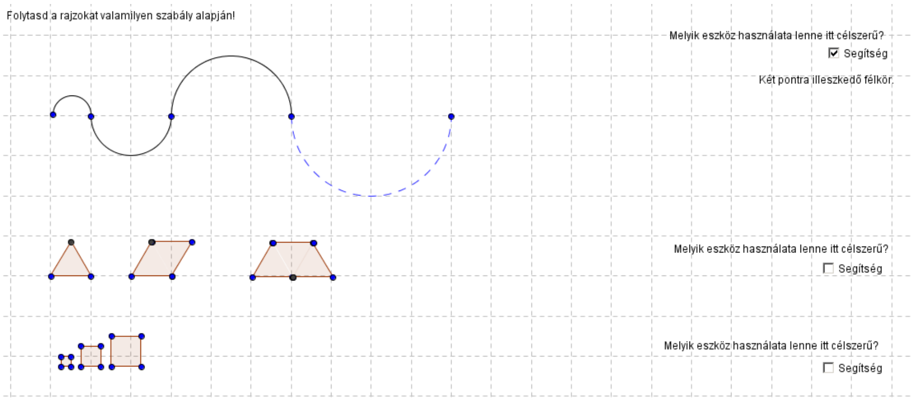


Figure 2: Geometric line-game

After the final test, which has been performed in the traditional way, the GeoGebra group has been provided a result of approximately 9% better than the control group compared to the average, which is certainly significant.

	year 2008/2009	year 2009/2010	year 2010/2011
5/a (GeoGebra group)	3,11	2,92	3,12
5/b (control group)	2,86	2,71	2,92

Table 2: Introduction to the geometry: results

3.2. Geometric constructions

As we have mentioned in the introduction we should not merely rely on the GeoGebra. For children of this age manual activity, motion culture, the development of the aesthetic work are also of great importance to pursue acquisition of competence. Thus, in any case, we start the topic of geometric constructions in the traditional way (ruler and compass). What can one do to provide something extra at this topic by GeoGebra? Once the children confidently carried out various constructions, they can sit to the front of GeoGebra and experience the benefits of the dynamic software compared to paper drawing:

1. faster triangle inequality test

2. faster, more accurate construction

3. resolving a particular construction task, thanks to the dynamics a number of similar construction task can be resolved quickly by changing the positions or values of the data

4. the discussion of the solutions is easier (various surveys, what types of solutions are expected).

The latter happens to be the pivotal point: If they have a single solution in paper, they normally think the problem is solved. An important consideration is to find all of the solutions.

The related lesson types were as follows:

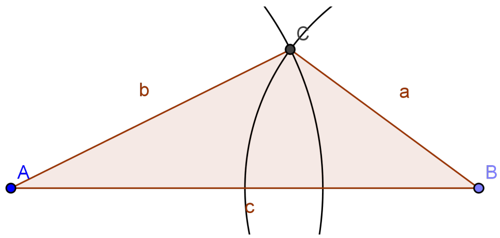


Figure 3: Constructing three sides of a triangle (generally solved)

- Construct a triangle with sides $a = 2\,\mathrm{cm}$, $b = 4\,\mathrm{cm}$, $c = 5\,\mathrm{cm}$. Generalize this construction for arbitrary lengths (Figure 3)!

- Construct a rectangle, if $a = 2\,\mathrm{cm}$, $b = 5\,\mathrm{cm}$. Generalize this construction for arbitrary lengths (Figure 4)!

Since two different device types (ruler – compass and GeoGebra) have been used by the children in the GeoGebra group (5/a), steps like capture data, the systematic, precise, thoughtful work, or the elementary steps of constructions have been clarified in a more confident way (e.g did not work with arcs, but with circles). Finally they have learned how to control their work. Results of the control tests of constructions are shown in the following Table, providing better contribution by the GeoGebra group by an average of 6%:

	year 2008/2009	year 2009/2010	year 2010/2011
5/a (GeoGebra group)	3,62	3,43	3,55
5/b (control group)	3,42	3,27	3,35

Table 3: Geometric constructions: results

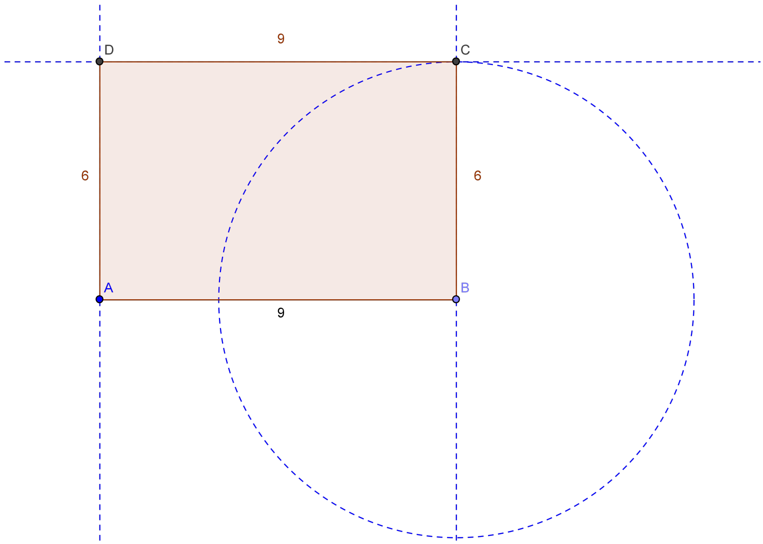


Figure 4: Constructing the rectangle

3.3. Geometric transformations

At this stage students have already been quite confident in use of GeoGebra in class 5/a. In the class 5 we are reflecting objects through a line. This could be introduced in several ways: symmetric images from nature, the use of various interesting games and so on. GeoGebra can be next to go on this way: It is possible to insert pictures to your worksheet. Now the children themselves are able to discover the properties of reflection through the axis (Figure 5).

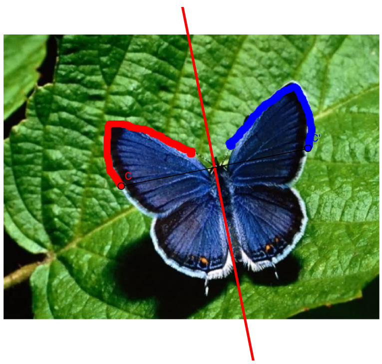


Figure 5: The study of reflecting through axis in a picture

The following task is a stand-alone work: they have to search the internet for a picture with symmetrical shape, find the position of the axis of symmetry and confirm the possibility of turning on the trail on GeoGebra. This type of task highly developed the ability of determining whether there is an axis of symmetry in a picture or shape. In the case of such problems, the results have improved significantly by the application of GeoGebra. For example, 60% of the group was able to reflect a concave heptagon axially in a paper drawing, such that the axis actually intersected the shape, which has been a remarkable performance. In this case the difference between the two groups was so obvious, that the GeoGebra group received more challenging problems in the test. Even this way their results were better, which can be seen in the following table:

	year 2008/2009	year 2009/2010	year 2010/2011
5/a (GeoGebra group) (with harder problems)	3,18	3,07	3,13
5/b (control group)	3,11	2,98	3,04

Table 4: Geometric constructions: results

3.4. Basic operations with integers and fractions

There are several models, which teachers can use in the context of the topic mentioned above: colored-rod stocks, discs, thermometer-model, debt-asset model and so on. Now one can try a new tool, the dynamic software (see Fig. 6).

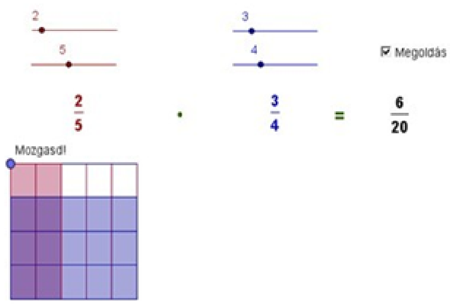


Figure 6: Representation of multiplication

Over the past 3 years our experience gained has shown that the results of the GeoGebra group, have been shifted towards the positive direction, although the extent of the movement was not as large as in the case of the geometry. In our opinion GeoGebra should still be included among the standard equipments in the field as well.

All the GeoGebra worksheets and problems can be found in the Hungarian Wiki GeoGebra page [5], and one can find additional additional worksheets which can be used with similar results.

References

[1] Pólya, Gy., A gondolkodás iskolája, Akkord Kiadó, 2000.

[2] http://www.geogebra.org/cms/ (2011-04-09)

[3] Bakar, K. A., Fauzi, A. and Tarmizi, A., Exploring the effectiveness of using GeoGebra and e-transformation in teaching and learning Mathematics, *Proc. of Intl. Conf. of Advanced Educational Technologies EDUTE 02*, pp. 19–23, 2002.

[4] Lecĺere, P., Raymond, C., Use of the GeoGebra software at upper secondary school, *FICTUP Public case report, INPL, France*, 2010.

[5] http://www.geogebra.org/en/wiki/index.php/Hungarian (2011-04-09)

[6] Kreis, Y., Dording, C., Keller, U., GeoGebraPrim - GeoGebra in der Grundschule, *Beiträge zum Mathematikunterricht* 6 (2010), 1–4.

[7] Kreis, Y., Dording, C., Keller, U., Porro, V., Raynald J., Dynamic mathematics and computer-assisted testing: GeoGebra inside TAO,*International Journal on Mathematics Education*, 43 (2011), 1–10.

[8] Hohenwarter, M., GeoGebra - didaktische Materialien und Anwendungen für den Mathematikunterricht, PhD dissertation Universität Salzburg, 2006.

[9] Richard, P., Fortuny, J., Hohenwarter, M. and Gagnon, M., GeogebraTUTOR : une nouvelle approche pour la recherche sur l'apprentissage compétentiel et instrumenté de la géométrie à l' école secondaire, *Proceedings of the World Conference on E-Learning in Corporate, Government, Healthcare and Higher Education of the Association for the Advancement of Computing in Education. Québec, Canada*, pp. 428–435, 2007.

Annales Mathematicae et Informaticae
38 *(2011) pp. 137–145*
http://ami.ektf.hu

"How to solve it?" – The tsumego session*

Attila Egri-Nagy[ab]

[a]Department of Computing Science, Eszterházy Károly College, Hungary
e-mail: attila@egri-nagy.hu

[b]School of Computer Science, University of Hertfordshire, United Kingdom

Submitted March 21, 2011 Accepted September 14, 2011

Abstract

The so-called 'Pólya's method' is now the canonical way of teaching mathematical problem solving. We would like to show that the method is not restricted to Math classes. Here we apply the method to solving tsumego problems that are isolated, small scale tactical problems in the ancient board game of Go. This new and unusual topic enables the students to get a wider view of the strategies of problem solving and the cognitive and psychological processes involved can also be easily demonstrated.

Keywords: problem solving, game of Go (Wei-Chi, Baduk), Pólya's method

MSC: 97A20,00A08,97D50

1. Introduction

Go is an ancient two player Asian board game with very simple rules (see Appendix A). Despite the simplicity of its description, the game is indeed very complex and requires deep strategical and tactical knowledge. In fact, Go is the last stronghold of natural intelligence, the last board game for which artificial intelligence up to now has failed to produce computer programs that can beat professional players. Go seems to require problem solving techniques that go beyond the brute force search algorithms and learning the game is rumored to be equivalent to take an advanced mathematical course. For younger people playing Go can improve thinking skills and rather surprisingly it can ease their social interactions[8] as well. Similar to chess problems there are Go problems called *tsumego*s (Japanese term adapted to English, see Appendix B). These can be introduced without explaining the

*This project was supported by the Abacus Talent Workshop (NTP-OKA-XII-005).

full complexity of the game, so problem solving can be studied in a very focused setting, unlike mathematical problems that sometimes require some background knowledge.

Pólya's method described in his seminal book titled *How To Solve It?* [6] is now the standard way of teaching mathematical problem solving. The method distinguishes four principles or rather four consecutive stages of problem solving.

Understanding the problem Restating the problem in easier terms with more explanation, drawing diagrams, formulating questions, etc.

Devising a plan Assembling a list of possible steps leading to a solution, guessing and checking, considering special cases, eliminating possibilities, etc.

Carrying out the plan Executing the steps - patience and care is needed.

Looking back, evaluation The real in gain in the learning process comes from reflecting on what has been done and how.

These steps are general enough to apply them in a context different from mathematical problem solving. Here we describe a 90 minutes long session where students solve Go problems using Pólya's method, demonstrating how each step of the method applies to tsumego solving. This description is detailed therefore by using this description, similar sessions can be carried out in different environments. It is important to note that deep knowledge of the game is not required for the instructor.

2. The Tsumego Session

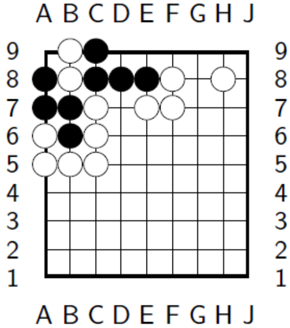
At the Eszterházy Károly College, as part of a one semester programme for fostering talented students from secondary schools we had afternoon sessions on different topics in Mathematics and Computer Science. The pupils were from different schools chosen by teachers from their schools, 12 pupils in total. This particular session on problem solving consisted of two parts (each of them 90 minutes long). The first part contained classical mathematical problems with explicit reflection on the heuristics. Due to the length of the session, the afternoon was very demanding for the students. Therefore it was very important for the second part to be more entertaining, even slightly unusual, thus we chose the game of Go.

It is necessary that real Go boards and Go stones are used during the session. Proper Go equipment has distinct aesthetics: simple regularity of the grid contrasted by the organic shape of woodgrains, the balance of interwoven black and white shapes. Invariably people start fiddling with Go stones when those are within reach of their hands, even when they are not in the situation of playing a game. Therefore the tactile experience of placing a stone on the board is very much part of the game. It is motivating and it gives a natural pace for working on the problems (as opposed to quickly clicking through all the empty intersections by a mouse while staring at a computer screen).

2.1. Understanding the Problem

Without further ado the students are presented with the following tsumego problem. (To save space in later diagrams we omit the coordinates.)

Problem 2.1. Black moves and lives in the corner while white is trying to kill the black groups.



Clearly, this immediate presentation of the problem will have a mild shock on the students as they are most probably used to long introductions before the first exercise. Obviously, this works better if the students have no prior knowledge of the game, or they just played a few games some time before, but they are not regular players. If Go players are present they should be asked not to spoil the effect by telling the solution quickly.

Using the confusion of the students the instructor can point to the first stage of problem solving: understanding the problem. In the ideal situation they have no prior knowledge of the game, so they have to face a situation in which understanding of the problem is completely missing. This never happens in mathematical problem solving, since by the time they first hear about Pólya's method, they already have solved many problems so their background knowledge is indeed quite deep.

Trying not cause any frustration by overexploiting the initial confusion the instructor claims that understanding the problem is just a matter of a few minutes long explanation. Unlike chess, where each piece has its own style of moving, go stones are all the same and once placed they do not move. Fortunately, for life and death problems, only a few concepts needs to be introduced. A *group* is a set of connected stones (along the lines, but not diagonally). A *liberty* of a group is an empty neighbouring intersection. A group is dead if there are no liberties left, so the number of liberties measures how far is the group from being captured (see details in the appendices). For unconditional life a group needs to have at least two liberties, two empty intersections that are not connected along the lines, i.e. they are separated by the group itself (see Fig. 1 and the Appendices).

The goal is now clear: to make moves in a way that the black group eventually survives by building a living shape or captures some white stones.

This introduction of the basic concepts of the game is a good opportunity for introducing the game in a wider philosophical[2] and historical context[3, 4]. In fact,

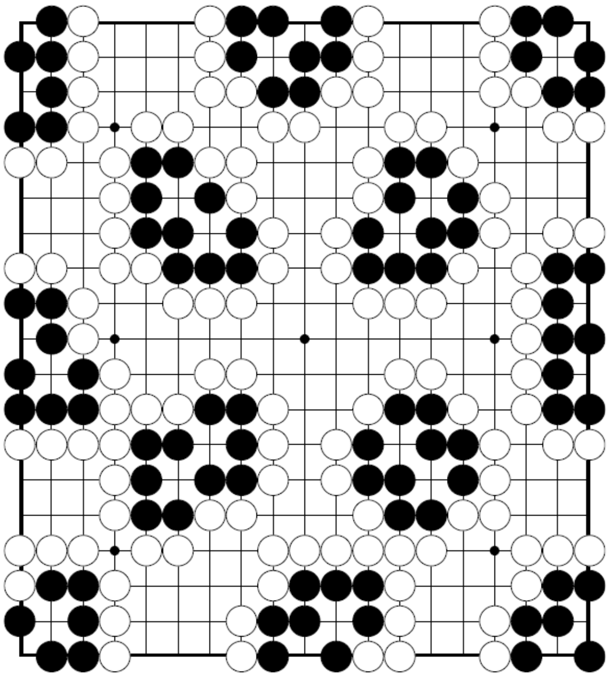


Figure 1: Minimal living shapes. Each black group has only two remaining liberties, but these liberties are well separated, therefore these groups cannot be captured, they are alive unconditionally. For the very first time the concept of unconditional life may not be fully comprehended by the students. This is not a problem, but it is still useful to show this collection as some students may recognize one of these shapes later on the board. [7]

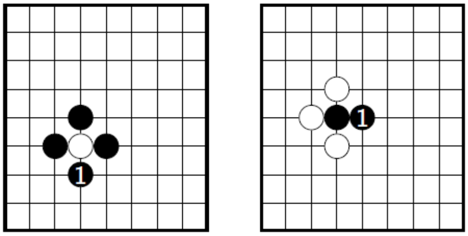


Figure 2: Problems for the concept of capturing and escaping by increasing the number of liberties. The key first step is indicated in both problems. For both problems the solution consists of only one move.

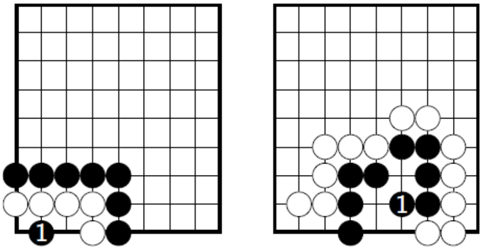


Figure 3: Problems for the idea of having two eyes. In the left problem the white group is not able to form two eyes (comb shape) after Black 1. In the problem on the right Black occupies the only point that can separate two empty points within the tentatively surrounded area.

we do nothing special here, only reversing the usual order of first the introduction and giving background information, then proceeding to exercises. Turning the order around is done for giving more motivation for the pupils and for illustrating more vividly the first step of Pólya's method.

2.2. Devising a plan

By now it is clear for the students that the plan will consist of a sequence of alternating moves. But there are many possible choices and a complete beginner may not have a sense of direction to follow in solving the problem. Another advice from the problem solving method is that one should look for similar but simpler problems. For this purpose the students are given five simpler problems.

The first two problems in this set are just checking the understanding of the basic concepts: capturing and escaping by increasing the number of liberties of a group (Fig. 2). Interestingly, students found these problems too easy and difficult to believe that the answer is just placing a stone. Therefore it is important to

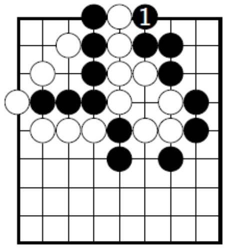


Figure 4: A capturing race. The solution of the problem requires to put white into atari at the right place. Trying to capture the other white group will end up loosing the inner black group.

reiterate that now we look at simplest possible problems.

The next two are about the idea of unconditional life by making two eyes (Fig. 3), referring back to the collection of unconditionally alive shapes (Fig. 1).

The last one in the set is a capturing race (Fig. 4) where for solving the problem one has to count the liberties of each group involved. In all problems black is to make a move. This is just a convenient simplification.

2.3. Carrying out the plan

The best setup is when students work in pairs on one board, one of them taking black, the other one white. After an unsuccessful attempt they may swap sides. If they cannot come up with the correct solution the instructor can take black (or white) and play it out with the students.

During the session approximately one third of the student came up with the correct solution without any further instruction. Others needed feedback on evaluating actual positions, whether the goal is reached or not.

At this point it is good to show the tree structure encoding the variants of a tsumego problem as an illustration. The nodes of the tree are positions, the connecting edges are moves. The variations can be studied after trying to solve the problem on an excellent tsumego site [5]. This also enables a quick explanation of the basic idea of the classical artificial intelligence algorithms: searching the game tree [1]. Humans do exactly the same type step-by-step calculation in an unfamiliar situation just as the participating students did during the session.

2.4. Reviewing the Solution

It is a good attitude in Go if someone is looking for a better move even if a good move has already been found. After successfully solving the tsumego it is important to evaluate the solution. Is there a better variant? Did we overlook something? Maybe white can intervene at a certain move? In case the solution is solid, still there is room for improvement. One can consider whether another shape would be better for further development.

It is also important to mention that the applied search method is not the highest form of problem solving. There is empirical evidence that Go masters come up with the solutions without any discursive thinking, their eyes fixate on the vital point under 300 milliseconds [9]. Some sort high-level pattern matching is done by the master players. For beginners the eye movement traces the steps of the search method. By solving tsumegos the brain develops this ability to recognize patterns on a subconscious level. This level of problem solving is in contrast with the discursive search method mentioned before. Instead of thinking about the problem and creating a plan, we can simply "see" the solution. Clearly the level of intuitive knowledge, the immediate certainty can be reached only by frequent practice of the step-by-step problem solving. Most players agree that this is the best way to get stronger in Go, solving many tsumegos frequently.

Similar is true for mathematical problem solving. By working on many different problems one develops the expertise or rather the intuition to see parts of the solution even in more complex problems. This is a well known phenomenon for working mathematicians. When working on a problem the solution comes suddenly, and not when someone tries hard, but when turns away from the problem. This is the next level beyond problem solving as an exercise towards creative research.

Clearly, the success of the session depends on whether the students were capable of solving the tsumegos or not, but since there are really elementary problems, this can be guarateed. The instructor should emphasize that a lot has been learnt and with the acquired knowledge the students can start to play the game themselves. They should be provided with further technical information (good starting point is [7]). At the same time the students should be warned that this is just the beginning and becoming a good Go player or a good at Math is not a quick process.

3. Conclusion

We described the application of Pólya's method to a different domain of problem solving in a form of a special session for selected students. This enables students to get a new perspective on the steps of problem solving (see comparison on Fig. 5). Solving Go problems is a great opportunity to talk about the psychology of problem solving, to introduce algorithmic concepts of artificial intelligence and the inner workings of our pattern matching minds. This fresh view of Pólya's method helps students to apply the steps more efficiently in mathematical context as well. We recommend the tsumego session as a complement to traditional problem solving classes.

	Mathematical Problem Solving	**Studying Tsumegos**
Previous knowledge	Requires extensive background knowledge and experience in Mathematics.	Nothing needed. Only a few simple concepts are to be explained.
Benefit	Good preparation for tests and exams.	Fun, gives new perspective on problem solving, but no immediate payoff.
Reflection	Doing Mathematics is very complex activity, an interplay of numerous cognitive structures.	Due to its simplicity it is easy to point out the cognitive processes involved.

Figure 5: Comparing a mathematical and a tsumego solving session.

A. The Rules of the Game of Go

Go is played by two persons (Black and White) on a board with a 19×19 grid. The game starts with an empty board. A move is placing a stone on an empty intersection point. Black makes the first, then moves alternate. The goal is to surround more territory. Friendly stones on neighbouring points (connected by gridlines but not diagonally) form *groups*. By counting a group's empty neighbouring points we get the *liberty* of the group. If this number becomes zero, i.e. all neighbouring points are occupied by enemy stones, then the group is captured or dead and it is taken off the board. If the liberty count is exactly 1, then we say that the group is in *atari*. It is not compulsory to make a move but suicide moves and those that restore a previous board position are forbidden. The game ends when both player pass. Then the surrounded territory is counted (the number of prisoners subtracted). The winner is the player with more territory.

B. Tsumego

Tsumegos are local battles where in a few moves one side suffers decisive loss or gains overwhelming advantage. The most common type of these all or nothing situations are *life and death* problems in which the goal is to save or kill a group, i.e. increasing liberties/forming two eye groups or filling up liberties of enemy groups. Solving tsumegos is basically finding a few key moves. Go players aim to solve tsumegos within seconds.

Acknowledgments. I am grateful to Ilona Téglási for paving the way to the tsumego session by delivering an excellent problem solving class in the first part, and to Krisztina Barczi for her valuable comments.

References

[1] E.R. Berlekamp, J.H. Conway, and R.K. Guy. *Winning ways for your mathematical plays Volume 1.* A.K. Peters, 2001.

[2] William Cobb. *Reflections on the Game of Go: The Empty Board 1994-2004.* Slate and Shell, 2005.

[3] Michael H. Koulen. *Go. Die Mitte des Himmels - Geschichte, Philosophie, Spielregeln, Meisterpartien.* Hebsacker Verlag, 2006.

[4] Shirakawa Masayoshi. *A Journey in Search of the Origins of Go.* Yutopian Enterprises, 2005.

[5] Adam Miller. Community site for solving go problems. `http://goproblems.comgoproblems.com`, 2011.

[6] George Pólya. *How To Solve It.* Princeton University Press, 1945.

[7] Comprehensive wiki page on the game of go. `http://senseis.xmp.net/senseis.xmp.net`, 2000-2011.

[8] Yasutoshi Yasuda. *Go as Communication: The Educational and Therapeutic Value of Go.* Slate and Shell, 2002.

[9] Atsushi Yoshikawa and Yasuki Saito. Perception in tsumego under 4 seconds time pressure. In *Proceedings of the eighteenth annual conference of the Cognitive Science Society*, page 868, 1996.

Annales Mathematicae et Informaticae
38 *(2011) pp. 147–156*
http://ami.ektf.hu

Teaching of monitoring software

Krisztina Erdélyi

John von Neumann Faculty of Informatics, Óbuda University
e-mail: erdelyi.krisztina@nik.uni-obuda.hu

Submitted September 28, 2010 Accepted May 16, 2011

Abstract

Teaching related to the measurement of IT services, tools and controlling infrastructure has become a common activity in universities and colleges that offer education in IT. To solve these types of IT tasks, software applications are needed that are able to monitor systems, networks or other software. Presented in this paper is an overview of the educational status of teaching monitoring software to students, and also the need to teach monitoring software combined with an expected set of general characteristics. Finally, this paper highlights the advantages of a virtual environment for teaching various 'monitoring software'applications to students.

Keywords: IT Service Management, Monitoring software, Virtual environment

MSC: 97P30

1. Introduction

The definition of 'monitor' in the Oxford Dictionary [1] is, "make continuous observation of, or record or test the operation of". Many activities can be monitored: the state of a system, whether a process has finished, or whether a state or an event is occurred.

According to Kees Jan Koster [2] "there are three basic categories of monitoring; technical monitoring, functional monitoring and business process monitoring". Business process monitoring answers the questions of whether the business is performing well. The IT systems supporting that business are only a part of the solution to that question. The aim of functional monitoring is to assess the performance and availability of a use-case or a set of use-cases related to a system.

Functional monitoring is usually performed by employing software agents to execute scripted operations on a system. Technical monitoring concerns itself with the health of individual pieces of equipment or software, and most monitoring tools are designed to perform these functions.

IT related monitoring can be separated into 'technical' and 'functional'. Every multifunctional operating system contains tools for observing hardware and software continuously and for logging the operation of the system. For example, a CPU's loading and list of current processes can be queried and log files can contain the important events of a system. When computers form a network, operators observe the performance of computers remotely and they monitor the network tools and traffic. Besides general IT, overseeing special areas such as databases, transactional processes are also features of monitoring software. The Monitortools.com website [3] presenting monitoring software gives the following categories:

- PC Monitoring,
- Application Monitoring,
- Performance Monitoring,
- Cloud Monitoring,
- Protocol Analyzing and Packet Capturing,
- Database Monitoring,
- Security Monitoring,
- Service Level Monitoring,
- Environmental Monitoring,
- SNMP Monitoring,
- Event Log Monitoring,
- VoIP Monitoring,
- Network and System Monitoring,
- Web Monitoring, Network
- Traffic Monitoring.

Besides monitoring technical tools and their operations, enterprises often monitor the activities of employees (the portion of effective working hours, which website access and so on). These applications can also be used for parental control at home.

Business process supervision can also be achieved by analyzing data retrieved from technical and functional monitoring and is helped by information technology through analyzing software.

2. Monitoring Software

Technical monitoring software can be categorized many ways. The categorization is made according to the subject requirements, the number of monitored devices, cost, and technology.

User activity, the general operations of computing, network, system, special software or hardware is also related to the subject of monitoring. A differentiation can be made whether Windows, Linux-like systems or both are monitored and the platform where the monitoring software operates is also important. Only the most serious monitoring systems are able to get information from other operating systems (for example from Advanced Interactive eXecutive - IBM AIX). Many vendors make monitoring software with basic functionality for Small and Medium sized Enterprises (SME). These ones generally are only able to monitor limited devices.

There are two different technologies to implement monitoring tools: agent-based and agent-less. It is an older but most robust solution if an agent is deployed in each monitored device. This case the agent collects information and sends them to the server on the grounds of commands from the server. There are different agents for different tasks, hence, "the agent-based approaches can gather more management data and more depth of information because of the agent instrumentation that is sitting on the system" [12]. Agent-less technology appeared because of difficulties with the maintenance of agents, "As the monitoring solution is updated, the agents will need to be updated from time to time. ...If you have a large number of systems, some of them might not be available when it's time to upgrade and then they're running outdated versions." [13]. Agent-less technology is however very rare. Rather, the deployment and maintenance of agents run automatically after network discovery or an in-built agent is used (for example in the Windows systems Windows Management Instrumentation - WMI).

The background knowledge which is necessary to manage monitoring software is also important when choosing the most suitable software. The in-built monitoring tool in Windows systems is the WMI. The Simple Network Management Protocol (SNMP) is used for managing networks or Linux-like systems. One group of software enables setting systems by use of protocols and standards, whilst the other group of software offers previously defined settings for the operators. The second group hides the protocols and standards commonly used.

The background knowledge which is necessary to manage monitoring software is also important when choosing the most suitable software. The in-built monitoring tool in Windows systems is the WMI. The Simple Network Management Protocol (SNMP) is used for managing networks or Linux-like systems. One group of software enables setting systems by use of protocols and standards, whilst the other group of software offers previously defined settings for the operators. The second group hides the protocols and standards commonly used.

Table 1 summarizes the features of technical monitoring software.

Aspects	Features
Subject of monitoring	user activity, general operation of computers, network, database, ...
Number of monitored devices	for home usage, for small and medium businesses, for enterprises
Cost of software	freeware, full trial version, proprietary
Used technology	agent-based, agent-less, hidden agent
Background knowledge	protocols and standards, software management

Table 1: Features of technical monitoring software

3. Educational Importance of Monitoring

The necessity of monitoring computers, networks and other IT tools is an historical issue. The activity of monitoring is comprised in all important methodologies. Hence, it is essential that newly graduated employee in IT service field knows the theoretical and practical base of monitoring.

Teaching the theory and practice of monitoring can come under the subject at technical and economic IT department of universities. (See chapter 5 for details.)

With enriching monitoring knowledge of students on one part their problem solving skills will be improved and on the other part they will get practical skills that can be capitalised at work.

3.1. Improving Problem Solving Skill

Information Technology Infrastructure Library (ITIL) is one of the most applied methodologies of IT service management. ITIL defines the conceptual problem as follows: "A cause of one or more incidents. The cause is not usually known at the time a problem record is created." [5] 'Problem' is distinctly different form 'incident', which are: "An unplanned interruption to an IT service or reduction in the quality of an IT service" [5]. Hence, if something does not operate sufficiently in a system, then it is an 'incident' and its root cause is a 'problem'. Beyond overcoming the incident the task is to discover and solve the problem.

A monitoring system is a great help for discovering and solving a problem.

The Sense-Isolate-Diagnose-Repair way of solution can be applied with software applications [14]. In this way after sensing the incident the next activities determine the place of the problem and discover its root. In this mode failure is then overcome. Monitoring software applications are able to generate events in case of malfunction (sense), and they provide facilities for deep analysis (isolate-diagnose).

ITIL, developed by Charles Kepner and Benjamin Tregoe [5], recommend the theoretical methods to solve a problem. It is.

> Kepner and Tregoe state that Problem Analysis should be a systematic process of problem solving and should maximise the advantage of knowledge and experience. They distinguish the following five phases for Problem Analysis:
>
> - Defining the problem;
> - Describing the problem with regard to identity, location, time and size;
> - Establish possible causes;
> - Testing the most probable cause;
> - Verifying the true cause
>
> [5]

To summarize, theoretical and practical teaching of monitoring develops the problem solving skills of students; it shows a way how to solve problems (Kepner-Tregoe method) that students practise when using software applications (sense-isolation-diagnosis-repair).

3.2. Acquiring knowledge using in everyday professional life

Employees in both IT and economic fields can benefit from monitoring knowledge. All multinational corporations use monitoring software in their everyday business and increasingly more SMEs. Configuring software and continuously changing its settings are both tasks attributable to IT experts. Reports can be made from the state of IT services and from business achievement through measuring services. These are the task of IT experts and business managers respectively.

Many enterprises suffer from the lack of common language between IT and business experts. IT experts do not understand the processes allied business, and business managers do not necessarily understand IT. Teaching monitoring knowledge to students with economic interest mitigates the lack of a common language. Students will know the possibilities of IT and they will have realistic expectations in field of monitoring.

4. The Place of Monitoring in Curricula

In Hungary teaching business processes and any monitoring supported by IT are believed to be more of a specialized topic than general education. Hungarian types of schools and levels of teaching related to technical and functional monitoring are outlined next:

- In elementary and secondary schools students meet task managers and log files of operating systems in a rudimentary approach. There is no further reference to managing systems or networks.
- The exam of European Computer Driving Licence (ECDL) does not require monitoring-like knowledge or competence [9].
- The lowest level where these skills are expected is the advanced level technical training course Maintaining Computer Systems [10]. In the Network Management part of 'Software Deploying on Computer Systems' deals with Basics of Simple Network Management Protocol (SNMP), Configuration Management, Remote Software Deployment, Monitoring, Help Desk, and Remote Console [11].

In core studies of IT colleges and universities the topic of monitoring appears chiefly in teaching operating systems. Retrieving information about operating systems is taught here (for example WMI). There are also possibilities to go deeper into the topic of monitoring and can cover network, system or database monitoring. Also at this level of teaching, standards and protocols of monitoring are reviewed.

To summarize, teaching the theory and practice of monitoring is the task of higher education. It is useful for students with technical and economic IT interests (See on chapter 3), consequently in-depth analysis can only be a specialization.

5. Developing Educational Environment

First step for developing an educational environment is to determine the taught software (see Table 1). Multinational computer technology corporations give a free run of their software for educational purpose to universities, therefore the cost of software is not a significant factor. The advantage of freeware is that students can install and try it at home on their own computing devices. The capacity of the laboratories in a university context is limited so it is reasonable to use software related to SMEs. If students are made familiar with more types of software with different technology platforms and user interfaces would improve teaching and learning.

The first experience about monitoring software that students should be exposed relates to more general purpose tools because main types of monitoring can be presented. There are special monitoring tools, such as applications for monitoring response time of transactions or monitoring user activities. Presenting one special monitoring tool can be the subject of a separate course in which technical issues can be examined more deeply. Alternatively, students can elaborate special software applications independently. At the Óbuda University students deliver a presentation about freely chosen monitoring software as a part of an assignment in accord with their interests; they most often choose network or user activity monitoring software.

The theory part of education about monitoring consists of the basics of ITIL and ITIL's section of monitoring. It also reviews problem solving generally related to IT.

5.1. Virtual Environment

For effective education students ought to experience how monitoring software works, and manage at least one server and one client machine. This condition is achieved by using monitoring software in a virtual environment.

Using virtual technology is also used in the teaching of operating systems in schools. Students are given administrative privileges in virtual machines so that they could not 'destroy' the host machine. For example,

> "Virtual machines provide a secure environment within which students may install, configure, and experiment with operating system, network, and database software." [15]

The effectiveness of such an approach was proved many times,

> "Our experiences in deploying this approach to teach more than nine hundred students have demonstrated the effectiveness of learning about real production operating system kernel development using virtual platforms." [16]

Virtual machines are also used for isolating different pieces of software in teaching laboratories. Software taught in different classes may interfere with others, but deploying software on different virtual machines resolves this challenge. Such a solution is also often useful for teachers as each one can develop their own laboratory environment on a separate virtual machine. Operators deploy only host machines and place virtual machines onto hosts.

Currently hardware is sufficiently powerful to run more VM on one host machine. Typically a server and a client and possibly an additional server (for example a database) or a client with different platform, constitute a virtual network. At the Óbuda University, a virtual network mostly consists of a server and two client machines (Figure 1). The disadvantage of a system made up of VMs relates to efficiency, as a real machine when it accesses the hardware is far superior. However, due to the hardware requirements of client machines, the one server two client "configuration" that executes on one host machine overcomes the issues discussed thus far.

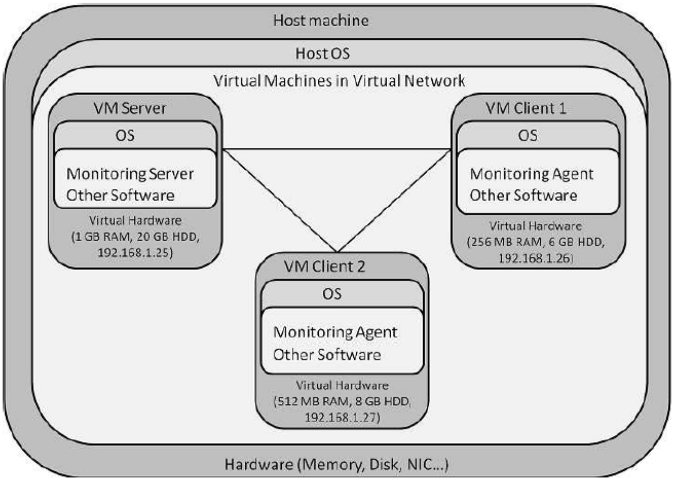


Figure 1: Typical virtual network for teaching monitoring

6. Experience

Within the faculty of Information Technology at the Óbuda University students have many possibilities to familiarize themselves with monitoring software. Many courses contain one or more classes linked to monitoring:

Students can attend the optional course 'Introduction to IT Service Management' whose main topic is monitoring. Additionally, IT Service Management is elaborated as a group of classes for specialisation. ITIL is taught as a theory course with practice of the most important topics included within monitoring and a separate course to represent managing composite applications. Four tutorial classes encompass composite application response time tracking. The topics of classes are the following:

- On the monitoring practice related to the ITIL lectures, students are made familiar with the operator tasks of a general purpose monitoring tools. This provides a sound basis for the monitoring of composite applications.
- In the framework of IT Service Management specialization deeper monitoring practices are taught in a course named as 'Managing Composite Applications'. A transaction monitoring application is used. The subject of monitoring is special in this case but the aim of monitoring and methodologies are identical to general purpose monitoring. Hence students are made familiar with monitoring technologies and a special area of monitoring
- In the course 'Introduction to IT Service Management' three tutorials address monitoring. After an introductory theory section students learn the version

for SMEs as one of their most significant monitoring software applications. In the first unit students use functionalities for operators, managing alerts and understanding different views. In the second unit they come to know the tasks of administrators, creating views and setting alerts. Finally, in the third unit students study the similar functionalities of freeware monitoring software.

Summary: Industrial technologies and best practises have emphasized the importance of monitoring for a long period of time. There are many types of software with monitoring features on the market to satisfy different needs.

In elementary and secondary education monitoring should not be included in the curriculum because of its speciality. But monitoring technologies are sufficiently widespread in industrial use, so students with technical and economic IT interests can exploit their knowledge about monitoring when they seek employment. It is practical to choose different types of software for teaching and to develop networks with virtual machines.

Acknowledgements. It is a pleasure to thank László Zsakó my consultant for significant help he has given to me, and John von Neumann Faculty of Informatics in Óbuda University, my workplace, for ongoing support.

References

[1] Cowie, A P (editor), Oxford Advanced Learner's Dictionary, *Oxford University Press*, 1989 p. 801.

[2] Types of Monitoring, `http://www.kjkoster.org/Blog/Types_of_Monitoring.html`, *last visit 17. 06. 2010*

[3] Network Monitor Software and Windows Development Tools, `http://monitoringtools.com`, *last visit 10. 06. 2010*

[4] IT Service Management Forum, An Introduction Overview of ITIL v3, *ISBN 0-9551245-8-1*

[5] Office of Government Commerce, Service Operation Book (ITIL), *ISBN-13: 978-0113310463*

[6] COBIT, `http://en.wikipedia.org/wiki/COBIT`, *last visit 09. 06. 2010*

[7] IT Governance Blog, `http://www.itgovernanceblog.com/`, *last visit 09. 06. 2010*

[8] Aksoy, Nejat, CobiT Fundamentals, *SF ISACA Fall Conference*, (2005)

[9] ECDL Magyarország, Európai Számítógép-használói Jogosítvány, `http://www.ecdl.hu/index.php?cim=mod2.`, *last visit 09. 06. 2010*

[10] Az Országos Képzési Jegyzék (OKJ), `http://www.szakkepesites.hu/`, *last visit 04. 06. 2010*

[11] PentaSchool Oktatási Központ, `http://www.pentaschool.hu/allami/szgrk.php?gclid=CKWDhdrokqICFQceZwoda3Hjdw/`, *last visit 14. 06. 2010*

[12] Network World. Agent or agentless monitoring? It's your choice, http://www.networkworld.com/newsletters/nsm/2005/0606nsm2.html, *last visit 14. 06. 2010*

[13] Tembria White Paper. Agents vs. Agentless Monitoring, http://www.tembria.com/products/servermonitor/agentless-monitoring.pdf, *last visit 14. 06. 2010*

[14] IBM Redbook, ITCAM for Response Time 6.2 Implementation and Administration Workshop *S150-2724-00*

[15] William I. Bullers, Jr., Burd, Stephen and Seazzu, Alessandro F., Virtual Machines - An Idea Whose Time Has Returned: Application to Network, Security, and Database Courses, *SIGCSE'06*, March 1-5, 2006, Houston, Texas, USA pp. 102–106.

[16] Nieh, Jason and Vaill, Chris, Experiences Teaching Operating Systems Using Virtual Platforms and Linux, *36th SIGCSE Technical Symposium*, pp. 100–104.

Annales Mathematicae et Informaticae
38 *(2011) pp. 157–167*
http://ami.ektf.hu

Technical mathematics in the University of Debrecen

Rita Nagy-Kondor

Faculty of Engineering, University of Debrecen
rita@eng.unideb.hu

Submitted September 13, 2011 Accepted October 25, 2011

Abstract

The presented course book has been written for the lectures and seminars of the subject Mathematics I, which is in the syllabus of the Faculty of Engineering, University of Debrecen. The book has an unusual approach to the curriculum in mathematics. Although the topics follow the usual thematic of the subject, the aim is not the teaching of mathematical concepts and tools, but the demonstration of their application in different engineering and economic fields, where the students will meet them. In this paper, we report on our experiences of this teaching method and on the application of the book.

Keywords: Engineering application, mathematical tools

1. Introduction

In the University of Debrecen Faculty of Engineering – similarly to many higher education institutes – the difficulties of the mass education come forward. Beside the increase of the number of the students, the number of the full-time teachers decreased. For measuring the high school mathematics knowledge of the students we make the first year students to write a test, according to that we can say that in the previous knowledge of our students there are big differences. One part of them cannot meet the earlier created requirement system, cannot bring in his lag.

So many traditional mathematics course book was made in the last years in the Faculty of Engineering, University of Debrecen, which contains the basic mathematics substance. By the experience of the education the authors made decision: some important applications of the mechanics, physics, and economics need to be built into the substance of lectures and practices of mathematics.

The authors have given an answer to the problem, and created a course book – entitled "Mathematical tools in engineering applications" [6] – in which the exercises are related to real technical problems. This way the students can realize that learning mathematics is useful, because they can see the extended application of mathematics in several engineering fields. Our approach emphasises why it is so important to learn mathematical methods and concepts, and where and how they can be applied.

This article reports on this teaching method and on the application of the book. Some problems of teaching mathematics are presented in Section 2. The first part of the course book has already been published; the content of it is reviewed in Section 3. The new method has already been applied in two semesters [5]; the experiences are presented in Section 4.

2. Motivation

International surveys show that the average Hungarian student loses 90% of his mathematical knowledge in the first three years after graduating from secondary school [11].

In the last few years a lot of publications reported on the rapid decreasing of the mathematical knowledge – and in general the educational level – of students in higher education [4] [7] [12]. Our experience is the same. It is also alarming that the skill to apply mathematics and the fundamental mathematical knowledge of graduated engineers show a decline. Sutherland and Pozzi [8] reporting: "There is unprecedented concern amongst mathematicians, scientists and engineers in higher education about the mathematical preparedness of new undergraduates". The situation has not changed since. They identified two main reasons:

- broadening of college and university entrance requirements to enable students to enter through vocational or other non traditional routes,
- the curriculum changes in students' pre-university education.

Teachers dealing with Mathematics have probably had the experience that most students find this subject hard. If they meet a new type of exercise that is a little bit different from the ones they practised you can easily see that their knowledge is superficial. But at the applied level of abstraction less and less student can receive the necessary knowledge. Mathematics works with abstract concepts. This fact makes the learning of it hard and time consuming. But the knowledge and understanding of concepts is vital for students to be able to build up their knowledge. By now the situation has changed, and it has become clear that the traditional teaching methods have to be reformed, because they are insufficient for handling the situation.

European technical universities tried out some potential solutions [1] [2] [3]:

- reducing mathematics syllabus: this made the drastic decrease of the high outstanding students' number.

- developing additional units: the students' loability is low so this gave few results.

- establishing mathematics support centres: but it turned that students with incomplete knowledge would need regular and extensive knowledge replacement.

Each of these has its own disadvantages.

Thus we found it vital to rethink the teaching of mathematics at our Faculty. [9] [10] We have approached the problem from two points of view.

First, we found it essential to summarise and repeat that part of secondary school mathematics and physics which is necessary as basic knowledge at our Faculty of Engineering. Besides, we intended to improve the logical way of thinking and problem solving ability of students. As a result, we introduced the compulsory subject "Basics of Natural Sciences". On the other hand, the teaching of Mathematics I. is based on our new course book, the title of which is "Mathematical Tools in Engineering Applications". That book follows a new approach and uses problems typically occurring in the fields of engineering and economics. However, it is important to note that in spite of the theoretical chapters that were written to clarify the basic concepts and problems of the engineering or economic field it should still be considered a course book of Mathematics, as the main purpose of the book is to help students understand this subject. This can only be realized by simplifying the engineering and economic problems to an appropriate level, so these kinds of problems will not make it more difficult for the students to receive the necessary mathematical knowledge. We hope, that with the didactically well-thought-out usage of this way the level of acquirement of the Mathematics increases. From the teacher's point of view the usage of the system increases the time period that is needed to get prepared for the classes.

3. Mathematical tools in engineering applications

In the course book "Mathematical Tools in Engineering Applications" the topics follow the usual thematic of the subject, the aim is not the teaching of mathematical concepts and tools, but the demonstration of their application in different engineering and economic fields, where the students will meet them.

Our main goals are to help the students to meet the requirements of the curriculum and have a more thorough understanding of Mathematics. We also wish to meet the demands of mass education. We would also like to help our students prepare for MSc level education, as according to our survey, 31% of our students wish to continue their studies after graduation.

The 8 chapters of the course book can be divided into groups of exercises. Each group of exercises starts with a theoretical summary, which provides a brief, but concise and professionally adequate description of the given engineering field. This is followed by a sample exercise with its solution and a series of similar exercises.

Main topics: Plane geometry, Space geometry, Vector algebra, Plane coordinate geometry, Complex numbers, Matrices. Linear functions and transformations, Systems of linear equations. The properties and graph of basic functions.

Figure 1 shows the different subchapters (which refer to the applications) within the main topics. (Engineering Mechanics I = Statics I; Engineering Mechanics II = Statics II; Engineering Mechanics III = Kinematics and Kinetics)

Figure 1

In the following we present two exercises as examples from three different chapters. The titles of these chapters are: Plane geometry (Exercise group: "The geometric relations of plane structures") and Vector algebra (Exercise group: "Forces and their resultant. Equilibrium of a particle").

Example 3.1. The figure below shows a crank-mechanism. Calculate distance. (Figure 2)

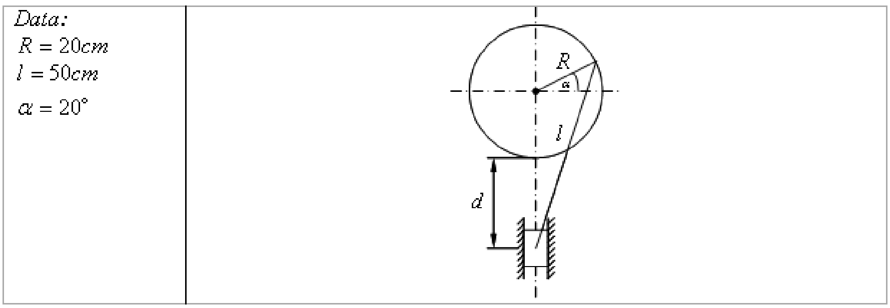


Figure 2

Example 3.2. Three forces act on a screw-head as it is shown in figure. The magnitudes of the forces and their angles relative to the x-axis are given. (Figure 3)

- Calculate the coordinates of the resultant of the three forces.
- Calculate the magnitude of the resultant and its angle relative to the x-axis.
- Construct the resultant of the three forces.

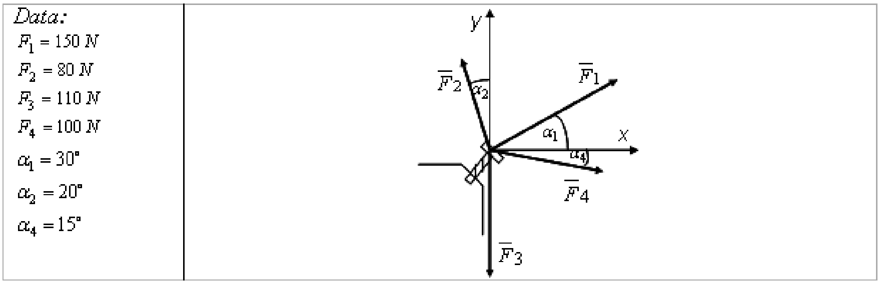


Figure 3

In the book, the same terms and terminology are used, as in the subjects, where the mathematical tools are applied. It is especially important for the students to be able to easily realize, that they meet with the same tools in mathematics as in the other subjects.

4. Results

Mathematics I in the new approach was attended by 105 students, 58 mechanical engineering, 28 building engineering, 16 architect and 3 engineering manager students, in the first semester of 2009/10. At the end of the semester we asked 91 students to take part in our opinion poll about the course book and our new way of teaching Mathematics I. The survey showed that 5,5% of our students consider Mathematics I. to be the most difficult among all the subjects of the first semester, and 50,5% of them see it as one of the three most difficult ones. Our course book "Mathematical Tools in Engineering Applications" was regarded as "easily understandable" by 59,3% of them, and 96,7% of them find it useful in understanding mathematics and also in their further studies (Figure 4).

84,6% of the students declared that the engineering problems helped him in the understanding of mathematics (Figure 5). In the opinion of 27,5% of them accomplishing Mathematics I is more difficult using engineering problems (Figure 6) and 37,4% of the questioned ones said that solving engineering problems is more difficult than mathematical ones (Figure 7). Only 37% of the students said that his secondary school knowledge is a good basis for the understanding of mathematics

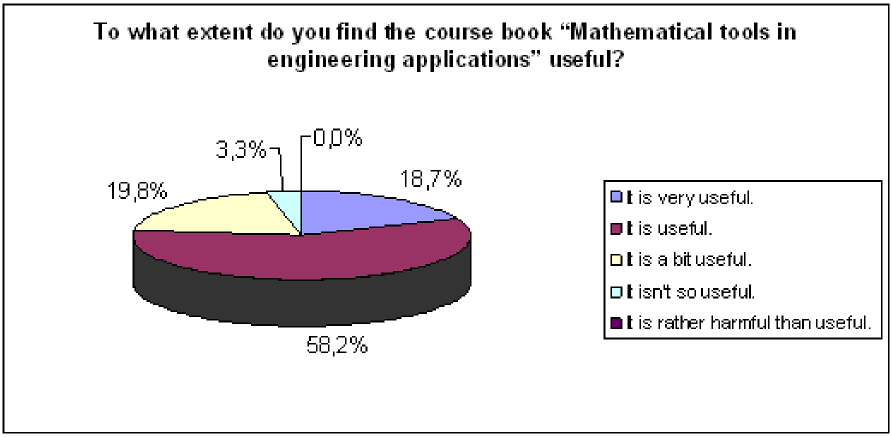


Figure 4

in the first semester. In the opinion Linear functions and transformations, Complex numbers, Vector algebra were the three most difficult topics; and Plane geometry, Matrices and Space geometry were the three easiest topics.

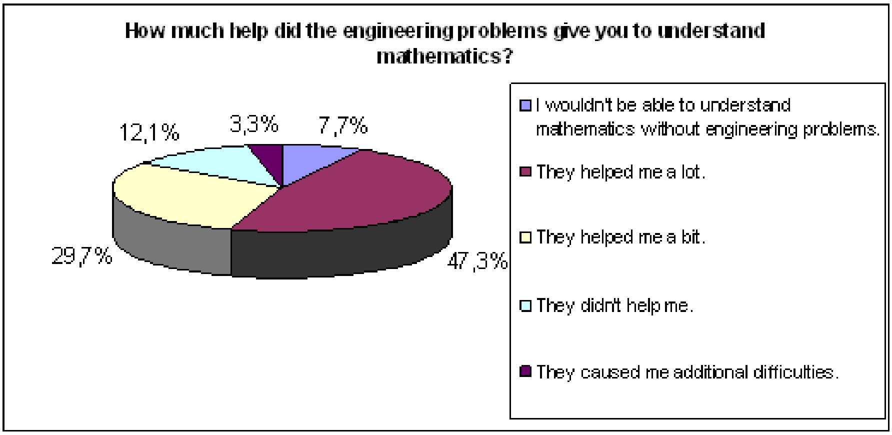


Figure 5

Mechanical Engineering students took the course in Engineering Physics in the first semester of 2009/10, parallel with Mathematics I. From the total 288 Mechanical Engineering students 47 chose to learn mathematics in the new approach. In the midterm writing tests of Engineering Physics we asked exercises from the following topics: free and constrained motion of a particle, electrostatics and DC currents, heat transport (conduction, convection, radiation). The average achievement of the total 288 students in Engineering Physics was 37,7%. The average achievement of

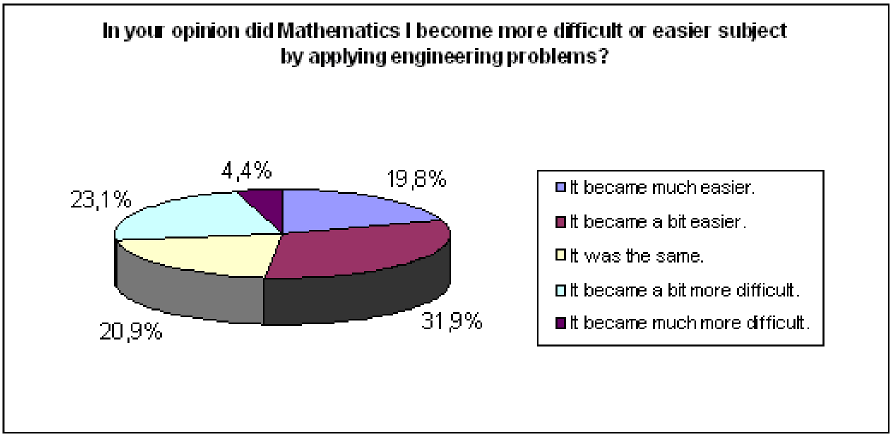


Figure 6

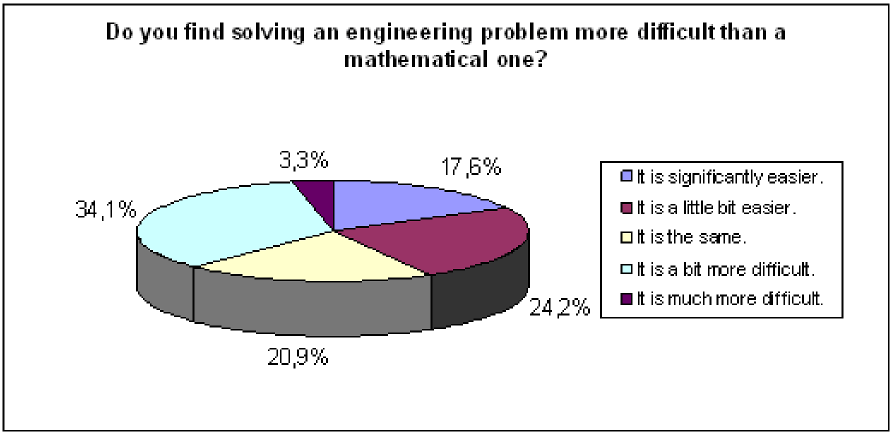


Figure 7

the 47 among them, who attended Mathematics I in the new approach, was better, 42%. The students of Mathematics I in new approach were 1% better in the tasks free and constrained motion of a particle, 6% better in the tasks electrostatics, 4% better in the tasks DC currents, and 7% better in the tasks heat transport. Both groups managed to solve DC currents tasks the best of all. Figure 9 shows the result of tests.

Building Engineering students took the course in Engineering Physics in the second semester. From the total 122 students, 19 attended Mathematics I in the new approach. In the midterm writing tests of Engineering Physics we asked exercises from the following topics: free and constrained motion of a particle, ideal

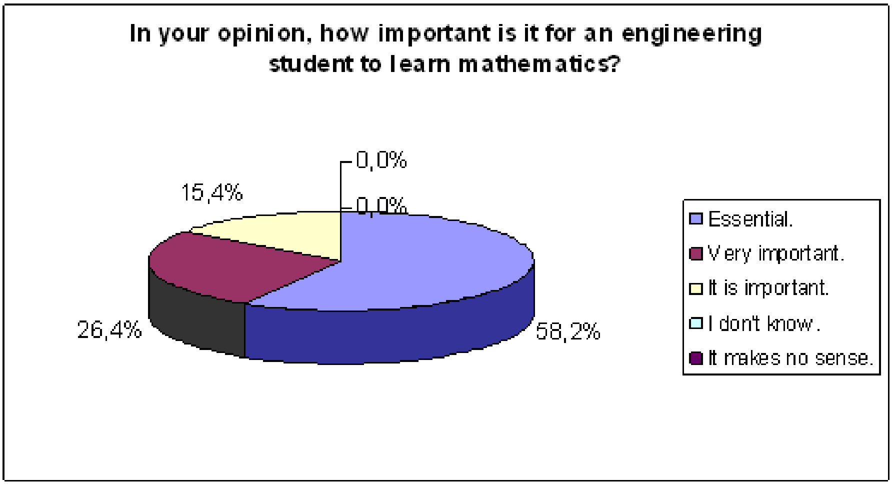


Figure 8

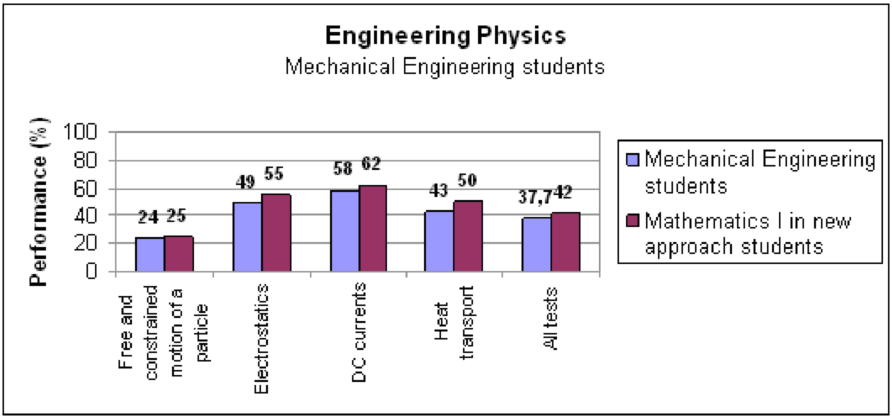


Figure 9

gases and gas mixtures, the processes of ideal gases, heat transport (conduction, convection, radiation). The average achievement of the total 122 students was 54%. The average achievement of those 19 students who attended Mathematics I in the new approach was slightly better, 57%. The students of Mathematics I in new approach were 1% better in the tasks free and constrained motion of a particle, 7% better in the tasks ideal gases, 6% better in the tasks heat transport. The low rate of fulfilling the first tasks show, that the problem solving ability of Mechanical Engineering students and Building Engineering students is poor in the field of free and constrained motion of a particle. We found that there is enough time devoted for teaching of heat transport in Mathematics I in the new approach. The students of Mathematics I scored their worst in the task related to processes of ideal gases. We can admit that there is little time devoted for teaching of processes of ideal gases in Mathematics I. Figure 10 shows the result of tests.

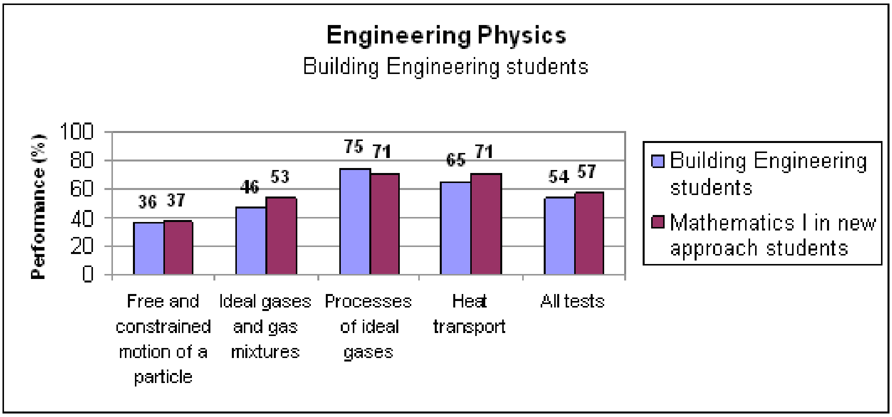


Figure 10

Mechanical Engineering students could take the course in Engineering Mechanics II (Statics II) in the second semester of 2009/10, provided that they had accomplished Engineering Mechanics I before. In the midterm writing test of Engineering Mechanics II we asked exercises from the following topics: state of stress, state of strain, general Hooke's law, Betti-theorem, Castigliano-theorem. From the total 89 students 24 had attended Mathematics I in the new approach. The average achievement of the 89 students was 32,6%. The average achievement of those 24 among them who attended Mathematics I in the new approach was significantly better, 37,2%.

Mechanical Engineering students could take the course in Engineering Mechanics III (Kinematics and Kinetics) in the first semester of 2010/11, provided that they had accomplished Engineering Mechanics I and II before. In the midterm writing tests of Engineering Mechanics III we asked exercises from the following topics: free and constrained motion of a particle, free and constrained motion of a rigid disk. From the total 44 students 10 had attended Mathematics I in the new approach. The average achievement of the 44 students was 34,7%. The average achievement of those 10 among them who attended Mathematics I in the new approach was significantly better, 44,5%. The students of Mathematics I in new approach were 10,5% better in the tasks free and constrained motion of a particle, and were 9,2% better in the tasks free and constrained motion of a rigid disk. In this exercise was the biggest difference between the two groups. We can admit that there is enough time devoted for teaching of free and constrained motion in Mathematics I in the new approach. Figure 11 shows the result of tests.

So we can say that we can reach quality improving with using Mathematics

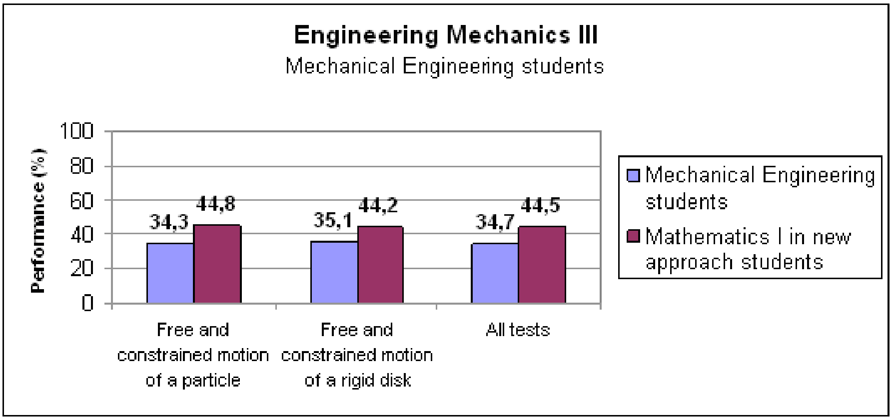


Figure 11

I in the new approach. Organizing the education in this way takes much more time of the teacher, the effective usage of engineering problems requires continuous developing work, but the results of the tests show that the invested work returns. We can talk about mathematical knowledge only in case of those students who can use the definitions and titles in practice as well.

5. Summary

The presented course book has been written for the lectures and seminars of the subject Mathematics I, which is in the syllabus of the Faculty of Engineering, University of Debrecen. The book has an unusual approach to the curriculum in mathematics. This course book underline that why it so important to learn mathematical methods and concepts and where can you use these. The main motive of the authors for writing the course book "Mathematical Tools in Engineering Applications" and for introducing a new kind of teaching method that uses real engineering problems was to make the teaching of mathematics more effective.

We built several important applications from the syllabi of Engineering Mechanics, Physics and Economics into the lectures and seminars of Mathematics. We hope that using this new educational method and the new course book the relationship between Mathematics and the different special engineering subjects becomes more and more clear for our students. In the future, we plan to further develop and revise our new study material on the basis of continual feedback.

On the basis of our results we can conclude that the teaching of mathematics becomes more effective applying engineering problems beside mathematical ones. The achievement and motivation level of students increase this way, and the results of them will be better also in the other engineering subjects that require mathematical knowledge.

If a student studies the book again and again, the connection between mathematics and the other subjects will be clearer for him or her.

References

[1] Körtesi, P., ed. Proceedings of the 10th Sefi-MWG European Seminar on Mathematics in Engineering Education, *Computer-Aided Design*, June 14-16, Miskolc (2002) 139.

[2] Körtesi, P., A matematika oktatása mérnökhallgatóknak, *Doktori (PhD) értekezés, Debreceni Egyetem*, (2005)

[3] Mathematics Working Group of SEFI, *http://sefi.htw-aalen.de*

[4] Nagy-Kondor, R., Special characteristics of engineer studentsÂ´ knowledge of functions, *International Journal for Mathematics Teaching and Learning*, 10 (2005) 1–9.

[5] Nagy-Kondor, R., Sziki, G. Á., Mathematical tools in engineering applications, *33rd International Congress of Teachers of Mathematics, Physics and IT, Conference Volume*, (2009) 50–54.

[6] Nagyné Kondor, R., Sziki, G. Á., Matematikai eszközök mérnöki alkalmazásokban I., *DE MK*,(2009)

[7] Sashalminé K., É., Egy felmérés tanulságai, *Acta Acad. Paed. Agriensis, Sectio Mathematicae*, 26 (1999) 121–126.

[8] Sutherland, R., Pozzi, S., The changing Mathematical Background of Undergraduate Engineers, *The Engineering Council, London* (1995)

[9] Sziki, G. Á., Hüse, E., Statikai problémákból felépülő adatbázis fejlesztése és alkalmazása a műszaki informatika oktatásában, *Proceedings of the 15th Building Services, Mechanical and Building Industry Days International Conference, Debrecen*, (2009) 384–389.

[10] Sziki, G. Á., Számítógépes program elektromobilok menetdinamikai jellemzőinek számításához, *16th "Building Services, Mechanical and Building Industry days" International Conference, 14-15 October 2010, Debrecen, Hungary*, (2010)

[11] Vancsó, Ö., Élő matematika, *Új Pedagógiai Szemle*, 10 (2002) 1–7.

[12] Vízvári, B., Új didaktikai és erkölcsi dilemmák a matematika egyetemi oktatásában, *A Matematika Tanítása*, 5 (1998) 11–14.