Acta Univ. M. Belii
Math no. 10(2003), pp. 3–8

ASYMPTOTIC EQUIPROBABILITY OF I-PROJECTIONS

M. Grendár, Jr. and M. Grendár

Dedicated to Mar

Abstract. When there is only one I-projection $\hat{\boldsymbol{p}}$ of the probability distribution $\boldsymbol{q}$ on the set Π of types, the conditional probability $\pi(\boldsymbol{\nu} \in B(\hat{\boldsymbol{p}}, \epsilon) | \boldsymbol{\nu} \in \Pi)$ that if $\boldsymbol{q}$ generates a type $\boldsymbol{\nu} \in \Pi$ then the type is close to the I-projection, approaches unity, as size n of type-supporting sequences goes to the infinity. What is the probability $\pi(\boldsymbol{\nu} \in B(\cdot) | \boldsymbol{\nu} \in \Pi)$ when Π admits several I-projections? Asymptotic Equiprobability of I-projections Theorem states that for n growing beyond any limit the conditional probability (measure) becomes split among the I-projections equally.

1. Introduction

In subjects as different as Statistical Physics and Computer Tomography, it is not rare to encounter an applied problem which can be cast into the following generic form:

There is a set Π of empirical probability mass functions (types) which could be obtained from random samples of size n, drawn independently from a supposed probability distribution $\boldsymbol{q}$. The problem (from category of ill-posed inverse problems) lays in making a choice of specific type(s) from the set Π.

A result of the Method of Types, which was developed in the Information Theory (cf. [3]), justifies application of the I-divergence minimization method (or equivalently, maximization of relative entropy method, MaxEnt) for making the choice, when n tends to infinity and Π has certain properties. The result is usually known as Conditioned Weak Law of Large Numbers (CWLLN), or as 'Gibbs' conditioning principle' (in large deviations literature, see [4]). For a convenience it will be recalled here, after a brief survey of necessary terminology. Another probabilistic justification of MaxEnt which is not based on large deviations techniques was proposed at [5].

1.1 Gibbs' conditioning principle, entropy concentration.

Let $\mathcal{X}$ be a discrete finite set with m elements and let $\{X_i, i = 1, 2, \ldots, n\}$ be a sequence of size n of identically and independently drawn random variables taking values in $\mathcal{X}$.

2000 Mathematics Subject Classification. Primary 62B15; Secondary 94A15, 65J20.
Key words and phrases. Conditioned law of large numbers, multiple I-projections, non-convex set.
Received 22. 4. 2003; Accepted 20. 6. 2003

A type $\boldsymbol{\nu} \triangleq [n_1, n_2, \ldots, n_m]/n$ is an empirical probability mass function which can be based on sequence $\{X_i, i = 1, 2, \ldots, n\}$. Thus, n_i denotes number of occurrences of i-th element of $\mathcal{X}$ in the sequence.

Let $\mathcal{P}(\mathcal{X})$ be a set of all probability mass functions (pmf's) on $\mathcal{X}$.

Let the supposed source of the sequences (and hence also of types) be $\boldsymbol{q} \in \mathcal{P}(\mathcal{X})$.

Let $\pi(\boldsymbol{\nu})$ be the probability that $\boldsymbol{q}$ will generate type $\boldsymbol{\nu}$, ie. $\pi(\boldsymbol{\nu}) \triangleq \prod_{i \in \mathcal{X}} q_i^{n\nu_i} \cdot$ $\cdot \Gamma(\boldsymbol{\nu})$, where $\Gamma(\boldsymbol{\nu}) \triangleq \frac{n!}{n_1! n_2! \ldots n_m!}$ is the multiplicity (or Boltzmann's complexion).

Then, $\pi(\boldsymbol{\nu} \in \mathcal{A})$ denotes the probability that $\boldsymbol{q}$ will generate a type $\boldsymbol{\nu}$ which belongs to $\mathcal{A} \subseteq \Pi$, ie. $\pi(\boldsymbol{\nu} \in \mathcal{A}) = \sum_{\boldsymbol{\nu} \in \mathcal{A}} \pi(\boldsymbol{\nu})$. Finally, let $\pi(\boldsymbol{\nu} \in \mathcal{A} | \boldsymbol{\nu} \in \Pi)$ denote the conditional probability that if $\boldsymbol{q}$ generates type $\boldsymbol{\nu} \in \Pi$ then the type belongs to $\mathcal{A}$. It is assumed that the conditional probability exists.

I-projection $\hat{\boldsymbol{p}}$ of $\boldsymbol{q}$ on set $\Pi \subseteq \mathcal{P}(\mathcal{X})$ is such $\hat{\boldsymbol{p}} \in \Pi$ that $I(\hat{\boldsymbol{p}} \| \boldsymbol{q}) = \inf_{\boldsymbol{p} \in \Pi} I(\boldsymbol{p} \| \boldsymbol{q})$, where[1] $I(\boldsymbol{p} \| \boldsymbol{q}) \triangleq \sum_{\mathcal{X}} p_i \log \frac{p_i}{q_i}$ is the I-divergence.

CWLLN. *Let $\hat{\boldsymbol{p}}$ be unique I-projection of $\boldsymbol{q}$ on Π. Let $\boldsymbol{q} \notin \Pi$. Then for any $\epsilon > 0$*

$$\lim_{n \to \infty} \pi \left(|\nu_i - \hat{p}_i| > \epsilon \mid \boldsymbol{\nu} \in \Pi \right) = 0 \qquad i = 1, 2, \ldots, m \tag{1}$$

Well-studied is the case of convex Π, which ensures existence of the unique I-projection (cf. [2], and [7], [8], [9] for further developments).

Without the assumption of uniqueness of the I-projection, a claim known as the Entropy Concentration Theorem (ECT), weaker than (1), can be still made (see [1]):

ECT. *Let $\Pi \subseteq \mathcal{P}(\mathcal{X})$ be nonempty. Let $\hat{I}$ be such that $\hat{I} \leq I(\boldsymbol{\nu} \| \boldsymbol{q})$ for any $\boldsymbol{\nu} \in \Pi$. Then for any $\epsilon > 0$*

$$\lim_{n \to \infty} \pi \left(\left| I(\boldsymbol{\nu} \| \boldsymbol{q}) - \hat{I} \right| < \epsilon \mid \boldsymbol{\nu} \in \Pi \right) = 1 \tag{2}$$

Assumption (of whatever form) which guarantees existence and uniqueness of the I-projection is crucial for coming from statement (2) to the stronger claim (1).

2. Multiple I-projections

Recently, physicists (see for instance [10] and literature cited therein) started to study problems of the generic form mentioned in the Introduction, which led into non-convex Π, with possibly multiple I-projections.

In this note, the following questions are addressed: Do the conditional concentration of types happen on the I-projections? If yes, do types concentrate on each of them? And, if yes, what is the proportion?

In order to investigate these questions, implications of Sanov's Theorem for the case of several I-projections will be studied, at first. Then, a heuristics which suggests answers to the questions will be made. Finally, the answers will be provided by Asymptotic Equiprobability of I-projections Theorem, which will be supported by a proof and illustrative examples.

[1] There, $\log 0 = -\infty$, $\log \frac{b}{0} = +\infty$, $0 \cdot (\pm\infty) = 0$, conventions are assumed. The definition of I-projection was adapted from [2]. Throughout the paper log denotes the natural logarithm.

2.1 Implications of Sanov's Theorem.

Sanov's Theorem (ST) is a fundamental tool for proving both ECT and CWLLN. It reads

$$\lim_{n\to\infty} \frac{\log \pi(\boldsymbol{\nu} \in \Pi)}{n} = -I(\hat{\boldsymbol{p}}||\boldsymbol{q}) \tag{3}$$

provided that Π is closure of its interior, and $\boldsymbol{q} \notin \Pi$. Thus, according to the ST

$$\pi(\boldsymbol{\nu} \in \Pi) \doteq e^{-nI(\hat{\boldsymbol{p}}||\boldsymbol{q})} \tag{4}$$

to first order in the exponent.

An application of the ST to $\pi(\boldsymbol{\nu} \in B|\boldsymbol{\nu} \in \Pi)$, where $B \subseteq \Pi$, gives

$$\lim_{n\to\infty} \frac{\log \pi(\boldsymbol{\nu} \in B|\boldsymbol{\nu} \in \Pi)}{n} = -(I(\hat{\boldsymbol{p}}_B||\boldsymbol{q}) - I(\hat{\boldsymbol{p}}_\Pi||\boldsymbol{q})) \tag{5}$$

where the subscripts B, Π denote the set on which $\boldsymbol{q}$ is I-projected. This can be also stated as

$$\pi(\boldsymbol{\nu} \in B|\boldsymbol{\nu} \in \Pi) \doteq e^{-n(I(\hat{\boldsymbol{p}}_B||\boldsymbol{q}) - I(\hat{\boldsymbol{p}}_\Pi||\boldsymbol{q}))} \tag{6}$$

to first order in the exponent.

Let now Π be such that there are k I-projections $[\hat{\boldsymbol{p}}_1, \hat{\boldsymbol{p}}_2, \ldots, \hat{\boldsymbol{p}}_k]$, of $\boldsymbol{q}$ on Π. What does the ST and (5), (6) imply in this case?

Let $B(\hat{\boldsymbol{p}}_j, \epsilon)$ denote an open Euclidean ball of radius ϵ centered around j-th I-projection, which is assumed to be the only I-projection in there.

Let $C \triangleq \cup_{j=1,2,\ldots,k} B(\hat{\boldsymbol{p}}_j, \epsilon)$. For types outside of C the RHS of (5) is negative and thus, fur such types (6) implies that $\pi(\boldsymbol{\nu} \notin C|\boldsymbol{\nu} \in \Pi)$ goes to zero exponentially fast. In turn, this implies that

$$\lim_{n\to\infty} \pi(\boldsymbol{\nu} \in C|\boldsymbol{\nu} \in \Pi) = 1 \tag{7}$$

Informally, for sufficiently large n it is virtually impossible to find a type which does not belong to union of the I-projection balls. So, now it remains only to find the proportion in which the probability is split among the different I-projections. The next crude heuristics indicates that the split should be equal.

2.2 Heuristics.

According to (6),

$$\pi(\boldsymbol{\nu} \in C|\boldsymbol{\nu} \in \Pi) \doteq e^{-n(I(\hat{\boldsymbol{p}}_C||\boldsymbol{q}) - I(\hat{\boldsymbol{p}}_\Pi||\boldsymbol{q}))} \tag{8}$$

to first order in the exponent. At the same time (6) implies

$$\pi(\boldsymbol{\nu} \in B(\hat{\boldsymbol{p}}_j, \epsilon)|\boldsymbol{\nu} \in \Pi) \doteq e^{-n(I(\hat{\boldsymbol{p}}_C||\boldsymbol{q}) - I(\hat{\boldsymbol{p}}_\Pi||\boldsymbol{q}))} \qquad j = 1, 2, \ldots, k \tag{9}$$

since

$$I(\hat{\boldsymbol{p}}_{B_j}||\boldsymbol{q}) = I(\hat{\boldsymbol{p}}_C||\boldsymbol{q}) \qquad j = 1, 2, \ldots, k \tag{10}$$

However,

$$\pi(\boldsymbol{\nu} \in C|\boldsymbol{\nu} \in \Pi) = \sum_{j=1}^{k} \pi(\boldsymbol{\nu} \in B(\hat{\boldsymbol{p}}_j, \epsilon)|\boldsymbol{\nu} \in \Pi) \tag{11}$$

which together with (8) and (9) implies that for sufficiently large n

$$\pi(\boldsymbol{\nu} \in C|\boldsymbol{\nu} \in \Pi) = k\,\pi(\boldsymbol{\nu} \in B(\hat{\boldsymbol{p}}_j, \epsilon)|\boldsymbol{\nu} \in \Pi) \qquad j = 1, 2, \ldots, k \tag{12}$$

After recalling (7), (12) implies that the conditional measure should be split among the I-projections equally.

2.3 Asymptotic Equiprobability of I-projections.

AEI Theorem. *Let there be k I-projections of $\boldsymbol{q}$ on Π. Let $B(\hat{\boldsymbol{p}}_j, \epsilon)$ be open Euclidean ball of radius ϵ centered at j-th I-projection, which is assumed to be the only I-projection in there. Then*

$$\lim_{n\to\infty} \pi(\boldsymbol{\nu} \in B(\hat{\boldsymbol{p}}_j, \epsilon) | \boldsymbol{\nu} \in \Pi) = \frac{1}{k} \qquad j = 1, 2, \ldots, k \tag{13}$$

Sketch of proof of the Theorem will employ MaxProb justification of MaxEnt (cf. [5]) and the ST.

Proof. Let $\hat{\boldsymbol{\nu}} \triangleq \arg\sup_{\boldsymbol{\nu}^\star \in \Pi} \pi(\boldsymbol{\nu} = \boldsymbol{\nu}^\star | \boldsymbol{\nu} \in \Pi)$. Let there be k such types, $\mathcal{N} \triangleq [\hat{\boldsymbol{\nu}}_1, \hat{\boldsymbol{\nu}}_2, \ldots, \hat{\boldsymbol{\nu}}_k]$. Then, by the Theorem 1 of [5], $\hat{\boldsymbol{\nu}}_j \to \hat{\boldsymbol{p}}_j$, for any $j = 1, 2, \ldots, k$, ie. the most probable type converges to the corresponding I-projection of $\boldsymbol{q}$ on Π. Since there are k types in $\mathcal{N}$ with equal value of the conditional probability π (which is the highest one), and each of the types from $\mathcal{N}$ converges to the corresponding I-projection, the last two facts imply that all the I-projections should be equally probable. Recall (7) to conclude that each of the conditional probabilities π should be equal to $1/k$ for n growing beyond any limit. □

Note that the number of I-projections k can be at most $\sum_{i=1}^{m} \binom{m}{i}$. Since m is assumed finite, k is finite as well.

The Asymptotic Equiprobability of I-projections (AEI) states that the probability that if $\boldsymbol{q}$ generated a type from Π then the type is close to the particular I-projection among k possible I-projections, approaches $\frac{1}{k}$ as n gets large. In other words, the conditional concentration of types happens on each of the I-projections with equal measure.

The AEI will be illustrated by the next two Examples.

2.4 Illustrative examples.

Example 1. Let $\Pi = \{\boldsymbol{p} : \sum_{i=1}^{m} p_i^\alpha - a = 0, \sum_{i=1}^{m} p_i - 1 = 0\}$, where $\alpha, a \in \mathbb{R}$. Note that the first constraint, known as frequency constraint, is non-linear in $\boldsymbol{p}$ and Π is for $|\alpha| > 1$ non-convex.

Let $\alpha = 2$, $m = 3$ and $a = 0.42$ (the value was obtained for $p = [0.5\ 0.4\ 0.1]$). Then there are three I-projections of uniform distribution $\boldsymbol{q} = [1/3\ 1/3\ 1/3]$ on Π: $\hat{\boldsymbol{p}}_1 = [0.5737\ 0.2131\ 0.2131]$, $\hat{\boldsymbol{p}}_2 = [0.2131\ 0.5737\ \ 0.2131]$ and $\hat{\boldsymbol{p}}_3 = [0.2131\ 0.2131\ 0.5737]$ (see [6]). Note that they form a group of permutations. As it will become clear later, it suffices to investigate convergence to say $\hat{\boldsymbol{p}}_1$.

For $n = 30$ there are only two groups of types in Π: G1 comprises [0.5666 0.2666 0.1666] and five other permutations; G2 consists of [0.5 0.4 0.1] and the other five permutations. So, together there are 12 types.

Value of the square of the Euclidean distance δ between $\boldsymbol{\nu}$ and $\hat{\boldsymbol{p}}_1$ attains its minimum $\delta_{G1} = 0.0051$ within G1 group for two types: [0.5666 0.2666 0.1666], [0.5666 0.1666 0.2666]. Within G2 the smallest $\delta_{G2} = 0.0532$ is attained by [0.5 0.4 0.1] and [0.5 0.1 0.4].

Thus, if $\epsilon = \epsilon_1$ is chosen so that the ball $B(\hat{\boldsymbol{p}}_1, \epsilon_1)$ contains only the two types from G1 (which at the same time guarantees that $\hat{\boldsymbol{p}}_1$ is the only I-projection in the ball), then $\pi(\boldsymbol{\nu} \in B(\hat{\boldsymbol{p}}_1, \epsilon_1) | \boldsymbol{\nu} \in \Pi) = 2 * 0.1152 = 0.2304$. In words: probability that if $\boldsymbol{q}$ generated a type from Π than the type falls into the ball containing only types which are closest to the I-projection is 0.2304. If $\epsilon = \epsilon_2$ is chosen so that also

the two types from G2 are included in the ball and also so that it is the only I-projection in the ball (any $\epsilon_2 \in (\sqrt{0.0532}, \sqrt{0.1253})$ satisfies both the requirements), then $\pi(\boldsymbol{\nu} \in B(\hat{\boldsymbol{p}}_1, \epsilon_2) | \boldsymbol{\nu} \in \Pi) = \frac{1}{3}$.

For $n = 330$ there are four groups of types in Π: G1, G2 and a couple of new one: G3 consists of [0.4727 0.4333 0.0939] and all its permutations; G4 comprises [0.5727 0.2333 0.1939] and its permutations. Hence, the total number of types from Π which are supported by random sequences of size $n = 330$ is 24.

δ_{G3} for the two types from G3 which are closest to $\hat{\boldsymbol{p}}_1$ is 0.0729. The smallest $\delta_{G4} = 0.00077$ is attained by [0.5727 0.2333 0.1939] and by [0.5727 0.1939 0.2333]. Thus, clearly, the two types from G4 have the smallest Euclidean distance to $\hat{\boldsymbol{p}}_1$ among all types from Π which are based on samples of size $n = 330$. Again, setting ϵ such that the ball $B(\hat{\boldsymbol{p}}_1, \epsilon)$ contains only the two types which are closest to $\hat{\boldsymbol{p}}_1$ leads to the 0.261 value of the conditional probability. Note the important fact, that the probability has risen, as compared to the corresponding value 0.2304 for $n = 30$.

Moreover, if ϵ is set such that besides the two types from G4 also the second closest types (i.e. the two types from G1) are included in the ball then the conditional probability is indistinguishable from $\frac{1}{3}$. Hence, there is virtually no conditional chance of observing any of the remaining 4 types. The same holds for the types which concentrate around $\hat{\boldsymbol{p}}_2$ or $\hat{\boldsymbol{p}}_3$. Thus, in total, a half of the 24 types is almost impossible to observe.

So, this Example illustrates, that indeed, as n gets large, $\pi(\boldsymbol{\nu} \notin C | \boldsymbol{\nu} \in \Pi)$ goes to zero, and that the conditional probability of finding a type which is close (in the Euclidean distance) to one of the three I-projections goes to $\frac{1}{3}$.

Example 2. Let $\Pi = \Pi_1 \cup \Pi_2$, where $\Pi_j = \{\boldsymbol{p} : \sum_{i=1}^m p_i x_i = a_j; \sum_{i=1}^m p_i = 1\}, j = 1, 2$. Thus Π is union of two sets, each of whose is given by the classical moment consistency constraint. If $\boldsymbol{q}$ is chosen to be the uniform distribution, then there is no difficulty to find values a_1, a_2 such that there will be two different I-projections of the uniform $\boldsymbol{q}$ on Π with the same value of I-divergence (as well as of the Shannon's entropy). This is indeed true for any $a_1 = \mu + \Delta$, $a_2 = \mu - \Delta$, where $\mu \triangleq EX$ and $\Delta \in (0, (X_{\max} - X_{\min})/2)$, since then $\hat{\boldsymbol{p}}_1$ is just a permutation of $\hat{\boldsymbol{p}}_2$, and as such attains the same value of Shannon's entropy. To see that types which are based on random samples of size n from Π indeed concentrate on the I-projections with equal measure note, that for any n to each type in Π_1 corresponds a unique permutation of the type in Π_2. Thus, types in ϵ-ball with center at $\hat{\boldsymbol{p}}_1$ have the same conditional probabilities π as types in the ϵ-ball centered at $\hat{\boldsymbol{p}}_2$. This, together with convexity of both Π_j, for which the conditional concentration of types on the respective I-projection is well-established, directly implies that

$$\lim_{n\to\infty} \pi(\boldsymbol{\nu} \in B(\hat{\boldsymbol{p}}_j, \epsilon) | \boldsymbol{\nu} \in \Pi) = \frac{1}{2} \qquad j = 1, 2$$

The same reasoning could be made for arbitrary $\boldsymbol{q}$. Convexity of I-divergence guarantees that there exists a pair a_1, a_2 with the desired property. For general $\boldsymbol{q}$ the sought values of a_1, a_2 are not displaced around μ equally.

3. Concluding notes

Probabilistic justification of MaxEnt by CWLLN as well as several axiomatic and non-axiomatic foundations of MaxEnt require assumption of uniqueness of I-projection. As [10] indicates, problems which lead to non-convex Π with possibly

non-unique I-projection appear in Physics. This facts prompted the presented study of convergence of types to multiple I-projections. In particular, we were interested in assessing the proportion by which types conditionally concentrate on each of the I-projections. As the AEI Theorem states, concentration of types happens on each of the I-projections with equal conditional measure. To support the sketched proof of the AEI Theorem, two examples were developed.

Informally, the AEI can be described using the Statistical Physics terminology, by stating that each of equilibrium points (I-projections) is asymptotically conditionally equally possible. Yet another informal formulation: If a random generator (probability distribution $\boldsymbol{q}$) is confined to produce types in Π then, as n gets large, the generator hides itself equally likely behind any of its I-projections on Π.

Asymptotic Equiprobability of I-projections enhances the Conditioned Weak Law of Large Numbers.

Aknowledgement

Hospitality of Banach Centre of the Institute of Mathematics of Polish Academy of Sciences, where a part of this study was performed as a part of the European Community Center of Excellence programme (package 'Information Theory and its Applications to Physics, Finance and Biology') is gratefully acknowledged. The work was also supported by the grant VEGA 1/7295/20 from the Scientific Grant Agency of the Slovak Republic. It is a pleasure to thank Brian R. La Cour for valuable discussions and an anonymous referee for careful reading and constructive criticism of the manuscript.

References

1. Cover, T. and Thomas, J., *Elements of Information Theory*, John Wiley and Sons, NY, 1991.
2. Csiszár. I., *Sanov Property, Generalized I-projection and a Conditional Limit Theorem*, Ann. Probab. **12** (1984), 768–793.
3. ———, *The Method of Types*, IEEE Trans. IT **44** (1998), 2505–2523.
4. Dembo, A and Zeitouni, O., *Large Deviations Techniques and Applications*, 2-nd ed., Springer, Application of Mathematics, vol. 38,, NY, 1998.
5. Grendár, M., Jr. and Grendár, M., *What is the question that MaxEnt answers? A probabilistic interpretation*, Bayesian Inference and Maximum Entropy Methods in Science and Engineering (A. Mohammad-Djafari, ed.), AIP, Melville, NY, 2001, pp. 83–94.
6. ———, *Maximum Entropy method: Shannon-Kullback vs. Rényi-Tsallis*, Advances of Mathematical Physics, NSP, NY, 2003 (to appear).
7. La Cour, B. R. and Schieve, W. C., *Macroscopic determinism in interacting systems using Large Deviations Theory*, Jour. Stat. Phys. **107 3/4** (2002), 729-755.
8. Leonard, Ch. and Najim, J., *An extension of Sanov's Theorem: Application to the Gibbs Conditioning Principle*, Bernoulli **8 (6)** (2002), 721–743.
9. Lewis, J. T., Pfister, C.-E. and Sullivan, W. G., *Entropy, concentration of probability and conditional theorems*, Markov Proc. Rel. Field. **1** (1995), 319–386.
10. Romera, E., Angulo, J. C. and Dehesa, J. S., *Reconstruction of a density from its entropic moments*, Bayesian Inference and Maximum Entropy Methods in Science and Engineering (R. L. Fry, ed.), AIP, Melville, NY, 2002, pp. 449–457.

Institute of Mathematics and Computer Science (of Matej Bel University and of Mathematical Institute of Slovak Academy of Sciences); Severná 5; SK-974 00 Banska Bystrica; Slovak Republic
Institute of Measurement Science of Slovak Academy of Sciences; Dúbravská cesta 9; SK-841 04 Bratislava; Slovak Republic
marian.grendar@savba.sk

Acta Univ. M. Belii
Math no. 10(2003), pp. 9–15

ON INTEGRAL BALANCED ROOTED TREES OF DIAMETER 10

Pavel Híc and Milan Pokorný

Abstract. A graph G is called integral if all the roots of the characteristic polynomial $P(G;x)$ are integers. A tree T is called balanced if the vertices at the same distance from the centre of T have the same degree. In the present paper the infinite class of integral balanced rooted trees of diameter 10, which has not been known so far, is given. The problem of the existence of integral balanced rooted trees of arbitrarily large diameter remains open.

1.Introduction

Let $G = (V, E)$ be a graph. The characteristic polynomial $P(G;x)$ of the graph G is defined to be characteristic polynomial of the adjacency matrix of G. The spectrum of the adjacency matrix is also called the spectrum of G, and is denoted by $Sp(G)$. We assume that the eigenvalues of G are given in non-increasing order

$$\lambda_1(G) \geq \lambda_2(G) \geq \cdots \geq \lambda_n(G).$$

$\lambda_1(G)$ is called the index of G, where $n = |V|$. Given $v \in V(G), G - v$ denotes the subgraph of G obtained by deleting the vertex v. For all other facts on graph spectra (or terminology), see [1] (or [2]).

We say that G has an integral spectrum if all the roots of $P(G;x)$ are integers. A graph G is called integral if it has an integral spectrum. In general, the problem of characterizing integral graphs seems to be difficult. In this paper we restrict our investigations to integral balanced rooted trees, which present one interesting family of graphs. It is known that there are infinitely many integral trees. In 1998 Híc and Nedela (see[4]) published the problem if there are integral balanced trees of arbitrarily large diameter. There exist integral balanced trees with diameter 2, 3, 4, 6, 8. Integral balanced trees with diameter 5, 7, and 9 do not exist, as well as integral balanced trees with diameter $4k + 1$ (k is an arbitrary integer). Hence, the first unsolved case of above problem is diameter 10. The main concern of this paper is to investigate integral balanced rooted trees of diameter 6, 8, and 10. The infinite class of integral balanced rooted trees with diameter 10 is given here (see Proposition 6). In general, the problem remains still open.

2000 Mathematics Subject Classification: 05C50.
Key words and phrases. tree, characteristic polynomial, integral tree.
Received 24. 9. 2002; Accepted 16. 4. 2003

The structure of a balanced tree (without vertices of degree 2) is determined by the parity of its diameter and the sequence $(n_k, n_{k-1}, ..., n_1)$, where k is the radius of T and n_j $(1 \leq j \leq k)$ denotes the number of successors of the vertex at the distance $k-j$ from the centre $Z(T)$. In what follows, n_i $(i = 1, 2, ...)$ always stands for an integer ≥ 2. The balanced trees of diameter $2k$ are encoded by the sequence $(n_k, n_{k-1}, ...n_1)$ and denoted by $T_k = T(n_k, n_{k-1}, ...n_1)$. These trees are called balanced rooted trees. (see [4])

Sequence $(n_k, n_{k-1}, ..., n_1)$ is called integral if the corresponding balanced rooted tree $T(n_k, n_{k-1}, ..., n_1)$ is integral. In 1974 Harary and Schwenk (see [3]) proved that (n_1) is integral if and only if n_1 is a square. Later, Schwenk and Watanabe (see [6]) proved that the sequence (n_2, n_1) is integral if and only if both n_1 and $n_1 + n_2$ are squares.

Híc and Nedela (see [4], [5]) published the following results:

(1) If $(n_k, n_{k-1}, ..., n_1)$ is integral, then $(n_j, n_{j-1}, ..., n_1)$ is integral for $1 \leq j \leq k-1$.

(2) The sequence $(n_k, n_{k-1}, ..., n_1)$ of positive integers is integral if and only if for every $q \in \mathrm{N}$ the sequence $(q^2 n_k, q^2 n_{k-1}, ..., q^2 n_1)$ is integral.

(3) All roots of the characteristic polynomial of the balanced tree with the sequence $(n_k, n_{k-1}, ..., n_1)$ are roots of the following recursively defined polynomial $P_k(x)$:
$P_0(x) = x$
$P_1(x) = x^2 - n_1$
$P_j(x) = x.P_{j-1}(x) - n_j P_{j-2}(x)$ where $j = 2, ..., k$.

The integral sequence $(n_k, n_{k-1}, ..., n_1)$ such that the g.c.d. $(n_k, n_{k-1}, ..., n_1)$ is square-free is called the primitive integral sequence and the corresponding integral tree $T(n_k, n_{k-1}, ..., n_1)$ is called the primitive integral tree.

2.Preliminaries

Here, we give some useful results from spectral graph theory, and then deduce some elementary facts about the spectrum of the integral balanced trees of even diameter.

Theorem 1. (see [1, Theorem 0.6, p. 19]) Let G be a connected graph. Then $\lambda_1(G) > \lambda_1(G - v)$ for any vertex $v \in V(G)$.

Lemma 2. Let $\{P_i(x)\}$ be the sequence of polynomials (3) and $\{T_i\}$ be the sequence of corresponding balanced trees. Then the sequence $\{\lambda_1(T_i)\}$ is increasing.

Proof. Let i be a positive integer, and let v be the central vertex of T_i. Now, since $T_i - v = n_i T_{i-1}$, where $n_i T_{i-1}$ are n_i disjoint copies of T_{i-1}, the proof follows from Theorem 1. □

Now, let μ_i be the smallest positive root of the polynomial $P_i(x)$, $i = 1, 2, ...$ Denote by $\{\mu_k\}$ the sequence of the smallest positive roots corresponding to the sequence $\{P_k(x)\}$. The following theorem is given in [5].

Theorem 3. (see [5, Theorem 1]) For every $i \geq 1$, there exists a positive root of the polynomial $P_i(x)$. Moreover, the following statements hold:

a. $\{\mu_{2k+1}\}$ is decreasing;

b. $\{\mu_{2k}\}$ is decreasing;

c. $\mu_{2k+2} > \mu_{2k+1}$ for $k = 0, 1, ...$

Lemma 4. Let $f_i(x) = \frac{xP_i(x)}{P_{i-1}(x)}$ for $i = 1, 2, ...$, where $P_i(x)$ is the polynomial of (3). Then $f_i(x)$ is increasing and positive for $x \in (\lambda_1(T_i), \infty)$.

Proof. We proceed by mathematical induction.

1. If $i = 1$, then $f_1(x) = \frac{xP_1(x)}{P_0(x)} = x^2 - n_1$. Notice that $\sqrt{n_1} = \lambda_1(T_1)$. Clearly, the function $f_1(x)$ is increasing and positive for any $x \in (\lambda_1(T_1), \infty)$.

2. Now, let $f_{i-1}(x) = \frac{xP_{i-1}(x)}{P_{i-2}(x)}$ be increasing and positive for any $x \in (\lambda_1(T_{i-1}), \infty)$. Using (3), after some routine calculations from the formula

$f_i(x) = \frac{xP_i(x)}{P_{i-1}(x)}$

we get

$f_i(x) = x^2(1 - \frac{n_i}{f_{i-1}(x)})$.

By the induction hypothesis, the function $f_{i-1}(x)$ is increasing and positive for $x \in (\lambda_1(T_{i-1}), \infty)$. Using lemma 2, the function $f_{i-1}(x)$ is increasing and positive also for $x \in (\lambda_1(T_i), \infty)$. Hence, the function $\frac{1}{f_{i-1}(x)}$ is decreasing and positive for $x \in (\lambda_1(T_i), \infty)$. Since n_i is a positive integer, the function $\frac{n_i}{f_{i-1}(x)}$ is decreasing and positive for $x \in (\lambda_1(T_i), \infty)$, too. We will prove that $\frac{n_i}{f_{i-1}(x)} < 1$ for every $x \in (\lambda_1(T_i), \infty)$.

Using substitution $x = \lambda_1(T_i)$ in the equation (3) we have

$P_i(\lambda_1(T_i)) = \lambda_1(T_i).P_{i-1}(\lambda_1(T_i)) - n_iP_{i-2}(\lambda_1(T_i))$. Since $\lambda_1(T_i)$ is the root of $P_i(x)$, we have

$n_i = \frac{\lambda_1(T_i)P_{i-1}(\lambda_1(T_i))}{P_{i-2}(\lambda_1(T_i))} = f_{i-1}(\lambda_1(T_i))$.

Notice that the function $f_{i-1}(x)$ is increasing for $x \in (\lambda_1(T_{i-1}), \infty)$. Using the previous formula, we have $f_{i-1}(x) > n_i$ for $x \in (\lambda_1(T_i), \infty)$. Therefore, $\frac{n_i}{f_{i-1}(x)} < 1$ for $x \in (\lambda_1(T_i), \infty)$. Now, the function $1 - \frac{n_i}{f_{i-1}(x)}$ is increasing and positive for $x \in (\lambda_1(T_i), \infty)$. Since the function x^2 is also increasing and positive for $x \in (\lambda_1(T_i), \infty)$, the function $f_i(x) = x^2(1 - \frac{n_i}{f_{i-1}(x)})$ is increasing and positive for $x \in (\lambda_1(T_i), \infty)$, too. The proof is complete. □

Theorem 5. (see [6, Theorem 1 and Theorem 2])

a) The balanced rooted tree $T(n_1)$ is integral if and only if $n_1 = k^2$ for some $k \in N$.

b) The balanced rooted tree $T(n_2, n_1)$ is integral if and only if $n_1 = k^2$ and $n_2 = n^2 + 2nk$ for some $k, n \in N$.

3. Results

3.1 Integral balanced rooted trees of diameter 6

Theorem 5 enables us to construct all integral balanced rooted trees $T(n^2 + 2nk, k^2)$ of diameter 4 for given $k, n \in N$. It follows from (1) that every integral balanced rooted tree of diameter 6 can be expressed as $T(n_3, n^2 + 2nk, k^2)$ for $k, n \in N$. For the roots of its characteristic polynomial the following equations hold:

(4) $$P_0(x) = x$$

$$P_1(x) = x^2 - k^2$$
$$P_2(x) = x.P_1(x) - n_2P_0(x) = x(x^2 - (k+n)^2)$$
$$P_3(x) = x.P_2(x) - n_3P_1(x)$$

The roots of the first three equations are $0, \pm k, \pm(n+k)$. Now, it is easy to see that every root of equation $x.P_2(x) - n_3P_1(x) = 0$ is also the root of the equation

$$n_3 = \frac{x.P_2(x)}{P_1(x)}. \tag{5}$$

The equation $P_3(x) = 0$ is the polynomial equation of degree 4 with the roots $\pm a, \pm b$ (because the spectrum of every tree is symmetric). Using (4) and (5), we have $0 < a < k, n + k < b$.

The problem of finding all integral balanced rooted trees $T(n_3, n^2 + 2nk, k^2)$ for given $k, n \in N$ is identical to the problem of finding all integers n_3, for which the equation (5) has only integer roots. This problem can be solved by the following algorithm:

readln(k);
readln(n);
for a:=1 to $k-1$ do
if $\frac{a.P_2(a)}{P_1(a)}$ is integer then
begin
$b := n + k + 1$;
while $\frac{b.P_2(b)}{P_1(b)} \leq \frac{a.P_2(a)}{P_1(a)}$ do begin { end condition follows from lemma 4 }
if $\frac{b.P_2(b)}{P_1(b)} = \frac{a.P_2(a)}{P_1(a)}$ then write $(T(\frac{b.P_2(b)}{P_1(b)}, n^2 + 2nk, k^2))$;
$b := b + 1$;
end;
end;

Using the programme based on this algorithm we have found all integral balanced rooted trees $T(n_3, n^2 + 2nk, k^2)$ of diameter 6 for $n = 1...10000$, $k = 1...2000$. The number of them is 270 814, but only 96 720 of them are primitive. That means that we have found 96 720 infinite classes of integral balanced rooted trees of diameter 6. Their list you can get by e-mail from authors.

3.2 Integral balanced rooted trees of diameter 8

To find integral balanced rooted trees of diameter 8, we will use similar method as we did in the section 3.1. It follows from (1) that every integral balanced rooted tree of diameter 8 can be expressed as $T(n_4, n_3, n^2 + 2nk, k^2)$, where $T(n_3, n^2 + 2nk, k^2)$ is the integral balanced rooted tree of diameter 6. The roots of its characteristic polynomial have to satisfy the equations (4) and the equation

$$P_4(x) = x.P_3(x) - n_4P_2(x) = 0.$$

The roots of the equations (4) are $0, \pm a, \pm k, \pm(n+k), \pm b$. Every root of the equation $x.P_3(x) - n_4P_2(x) = 0$ is the root of the equation

$$n_4 = \frac{x.P_3(x)}{P_2(x)} \tag{6}$$

The equation $P_4(x) = 0$ is the polynomial equation of degree 5 with the roots $0, \pm c, \pm d$. The problem of finding all integral balanced rooted trees $T(n_4, n_3, n^2 + 2nk, k^2)$ for given integral balanced rooted tree $T(n_3, n^2 + 2nk, k^2)$ of diameter 6 is identical to the problem of finding all integers n_4, for which the equation (6) has only integer roots. This problem can be solved by the following algorithm (the values of variables k, n and n_3 are taken from the results of the algorithm, which is described in the section 3.1):

readln(k);
readln(n);
readln(n_3);
for c:=$a + 1$ to $n + k - 1$ do
if $\frac{c.P_3(c)}{P_2(c)}$ is integer then
begin
$d := b + 1$;
while $\frac{d.P_3(d)}{P_2(d)} \leq \frac{c.P_3(c)}{P_2(c)}$ do begin { end condition follows from lemma 4 }
if $\frac{d.P_3(d)}{P_2(d)} = \frac{c.P_3(c)}{P_2(c)}$then write $(T(\frac{d.P_3(d)}{P_2(d)}, n_3, n^2 + 2nk, k^2))$;
$d := d + 1$;
end;
end;

Using the programme based on this algorithm we have found all integral balanced rooted trees $T(n_4, n_3, n^2 + 2nk, k^2)$ of diameter 8 for $n = 1...10000$, $k = 1...2000$. The number of them is 31 558, but only 5 784 of them are primitive. That means that we have found 5 784 infinite classes of integral balanced rooted trees of diameter 8. Their list you can get by e-mail from authors.

3.3 Integral balanced rooted trees of diameter 10

Every integral balanced rooted tree of diameter 10 can be expressed as $T(n_5, n_4, n_3, n^2 + 2nk, k^2)$, where $T(n_4, n_3, n^2 + 2nk, k^2)$ is the integral balanced rooted tree of diameter 8. The roots of its characteristic polynomial have to satisfy the equations (4), (6), and the equation $P_5(x) = x.P_4(x) - n_5P_3(x)$. The roots of the equations (4) and (6) are $0, \pm a, \pm c, \pm k, \pm(n + k), \pm b, \pm d$. If x is the root of the equation

$$x.P_4(x) - n_5P_3(x) = 0,$$

then x is also the root of the equation

$$n_5 = \frac{x.P_4(x)}{P_3(x).} \tag{7}$$

The equation $P_5(x) = 0$ is polynomial equation of degree 6 with the roots $\pm e, \pm f, \pm g$. The problem of finding all integral balanced rooted trees $T(n_5, n_4, n_3, n^2 + 2nk, k^2)$ for given integral balanced rooted tree $T(n_4, n_3, n^2 + 2nk, k^2)$ of diameter 8 is identical to the problem of finding all integers n_5, for which the equation (7) has only integer roots. This problem can be solved by the following algorithm (the values of variables k, n, n_3 and n_4 are taken from the results of the algorithm, which is described in the section 3.2):

readln(k);
readln(n);

readln(n_3);
readln(n_4);
for e:=1 to $a-1$ do
if $\frac{e.P_4(e)}{P_3(e)}$ is integer then
for $f := c+1$ to $b-1$ do
if $\frac{f.P_4(f)}{P_3(f)} = \frac{e.P_4(e)}{P_3(e)}$ then
begin
$g := d+1$;
while $\frac{g.P_4(g)}{P_3(g)} \leq \frac{e.P_4(e)}{P_3(e)}$ do begin { end condition follows from lemma 4 }
if $\frac{g.P_4(g)}{P_3(g)} = \frac{e.P_4(e)}{P_3(e)}$ then write $(T(\frac{g.P_4(g)}{P_3(g)}, n_4, n_3, n^2+2nk, k^2))$;
$g := g+1$;
end;
end;

Using the programme based on this algorithm we have found all integral balanced rooted trees $T(n_5, n_4, n_3, n^2+2nk, k^2)$ of diameter 10 for $n = 1...10000, k = 1...2000$. Their list is in the table below. Only the tree in the first column after the heading is primitive, the other trees can be construct from it using (2).

n_5	3006756	12027024	27060804	48108096	75 168 900
n_4	1051960	4207840	9467640	16831360	26 299 000
n_3	751689	3006756	6765201	12027024	18 792 225
n_2	283360	1133440	2550240	4533760	7 084 000
n_1	133956	535824	1205604	2143296	3 348 900
k	366	732	1098	1464	1 830
n	280	560	840	1120	1 400
a	306	612	918	1224	1 530
b	1037	2074	3111	4148	5 185
c	527	1054	1581	2108	2 635
d	1394	2788	4182	5576	6 970
e	289	578	867	1156	1 445
f	918	1836	2754	3672	4 590
g	2074	4148	6222	8296	10 370

Proposition 6.

For any $q \in N$, the tree $T(q^2 3006756, q^2 1051960, q^2 751689, q^2 283360, q^2 133956)$ is integral balanced rooted tree of diameter 10, and its spectrum

$Sp = \{0, \pm 280q, \pm 289q, \pm 306q, \pm 366q, \pm 527q, \pm 918q, \pm 1037q, \pm 1394q, \pm 2074q\}$.

Conclusion

The method which is written in this paper can be used for finding integral balanced rooted trees with arbitrarily large even diameter. The disadvantage of this method is that we have not been able to find any integral balanced rooted tree with diameter larger than 10 yet because we have to work with extremely huge numbers and investigate a huge number of possibilities. This problem can be solved by development of computers only partially.

We have used 10 computers with Pentium II processors and with the operating system Windows 98. The computation has lasted for two months, but only at weekends. The programmes were made using Delphi Pascal.

About the problems connected with finding integral balanced rooted tree of diameter larger than 10 tells the fact that from 20 million integral balanced rooted trees of diameter 4 only 270 814 can be expanded to integral balanced rooted trees of diameter 6, from them only 31 558 can be expanded to integral balanced rooted trees of diameter 8, and from them only 5 can be expanded to integral balanced rooted trees of diameter 10.

We can suppose that if we wanted to find integral balanced rooted tree of diameter 12, we would have to investigate much more integral balanced rooted trees of diameter 4, 6, 8, and 10, and work with much larger numbers than we have worked so far.

Acknowledgement

The authors wish to express their thanks to the referees for their help in improving the quality of this paper.

Research partially supported by VEGA grant No. 1/7152/20 (the first author).

References

[1] Cvetkovič, D.M. - Doob, M. - Sachs, H., *Spectra of graphs - Theory and Application*, Deutscher Verlag der Wissenschaften - Academic Press, Berlin-New York, 1980.

[2] Harary, F., *Graph Theory*, Addison - Wesley Publishing Company, Reading, MA, 1969.

[3] Harary, F. - Schwenk, A. J., *Which graphs have integral spectra? In: Graphs and Combinatorics. Lecture Notes in Math* **406** (1974), Springer-Verlag, Berlin, pp. 45-51.

[4] Híc, P. - Nedela, R., *Balanced integral trees. Math. Slovaca* **48** (1998), No. 5, pp. 429-445.

[5] Híc, P. - Nedela, R., *Note on zeros of the characteristic polynomial of balanced trees, Acta Univ. M. Bellii, Math.* (1996), No. 3, pp. 31-35.

[6] Schwenk, A. J. - Watanabe, M., *Integral starlike trees, J. Austral. Math. Soc.* **28** (1979), Ser. A, pp. 120-128.

Department of mathematics and informatics; Faculty of Education; Trnava University; Priemyselná 4; SK–918 43 Trnava; Slovak Republic
E-mail: phic@truni.sk and mpokorny@truni.sk

Acta Univ. M. Belii
Math no. 10(2003), pp. 17–19

A NEW PROOF FOR CHORDAL GRAPHS

Petr Hliněný

Abstract. Chordal graphs are those without induced cycles longer than three. It is a classical fact that every nonempty chordal graph contains a vertex, the neighbourhood of which induces a clique. This was first proved by Lekkerkerker and Boland in 1962. Few more proofs are known nowadays. We present yet another, very short and elementary proof.

Chordal Graphs

We consider nonempty finite simple graphs. A *clique* is a complete (sub)graph. A *chordal graph* is a graph containing no induced cycle of length greater than three. For a graph G and vertex $v \in V(G)$, we denote by $N_G(v)$ the subgraph induced on the neighbours of v in G (not including v itself – an "open" neighbourhood). A vertex v of G is called *simplicial* if $N_G(v)$ is a clique.

Simplicial vertices in chordal graphs were, perhaps, first considered by Lekkerkerker and Boland in an old paper [2] describing interval graphs. One of their results – a key fact in a characterization of chordal graphs via a simplicial decomposition, reads:

Theorem 1. (Lekkerkerker and Boland, 1962) If G is a chordal graph, then G has a simplicial vertex.

We say that a graph G is *bisimplicial* if G is either a clique, or G has two nonadjacent simplicial vertices. The "folklore" short proof of Theorem 1 establishes the following claim by induction.

Lemma 2. Every chordal graph is bisimplicial.

The inductive step in this proof finds a minimal vertex cut X which separates the remaining vertices in G into sets Y_1, Y_2. It is proved that X induces a clique in G, and then the claim is applied to the smaller graphs $G - Y_1$ and $G - Y_2$.

2000 Mathematics Subject Classification. 05C38.
Key words and phrases. chordal graph, simplicial vertex.
Received 17. 11. 2002; Accepted 16. 4. 2003

New Proof

The above sketched proof of Lemma 2 is short and elegant, but it is not elementary — the proof that a minimal vertex cut X in a chordal graph induces a clique needs the Menger theorem. This had shown to be a major obstacle when we tried to extend the notion of chordality to represented matroids (cf. [1]). That is why we have looked for another, short and elementary proof of Theorem 1.

Our alternative proof of Lemma 2 proceeds in a sequence of three simple (and elementary) claims, which lead to a contradiction showing that no minimal counterexample to Lemma 2 exists.

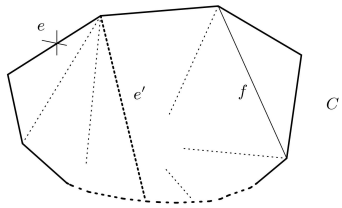

FIG. 1. An illustration to a chordal cycle C

Lemma 3. Let G be a chordal graph. If $C \subseteq G$ is a cycle, and e is an edge of C, then there is an edge $f \in E(G)$ such that f forms a triangle with two edges of $C - \{e\}$. (Fig 1.)

Proof. We proceed by induction on $|C|$. If C is a triangle, then the claim holds for $f = e$. So suppose a cycle $C \subseteq G$ of length greater than three. Since C is not induced in G, there is an edge $e' \in E(G - C)$ having both ends on C (a "chord" of C). The graph $(C - \{e\}) \cup \{e'\}$ contains a unique cycle C' which is shorter than C. We find the edge f inductively for C' and e'. □

Lemma 4. Let G be a graph, and let u, v be adjacent vertices in G such that the neighbourhood subgraph $N_G(v)$ is bisimplicial. If v is simplicial in the neighbourhood subgraph $N_G(u)$, but v is not simplicial in the whole graph G, then there is a vertex w which is adjacent to v but not to u, and w is simplicial in $N_G(v)$.

Proof. By the assumption, there are two nonadjacent simplicial vertices w, w' in the neighbourhood subgraph $N_G(v)$. At least one of them, say w, does not belong to the clique $N_G(u) \cap N_G(v)$. (See also Fig. 2.) □

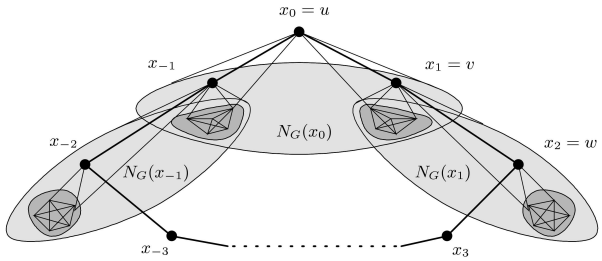

FIG 2. An illustration to the proofs

Lemma 5. Let G be a non-bisimplicial graph such that the neighbourhood subgraph $N_G(v)$ is bisimplicial for each vertex $v \in V(G)$. Then there is a cycle $C \subseteq G$ and an edge e of C, such that no triangle of G has two edges on $C - \{e\}$.

Proof. By the assumptions, G contains a non-simplicial vertex x_0. Let x_1 and x_{-1} be two nonadjacent neighbours of x_0 that are simplicial in $N_G(x_0)$. For $i = 1, 2, 3, \ldots$, we apply Lemma 4 to $u = x_{i-1}$ and $v = x_i$, and we set $x_{i+1} = w$. Notice that x_{i-1}, x_i, x_{i+1} induce a 2-path but *not a triangle*. (See Fig. 2.) The sequence proceeds inductively until x_i is simplicial. In the other direction we analogously get x_{i-1} from x_i, x_{i+1} for $i = -1, -2, -3, \ldots$.

Since G is finite, we eventually find indices $i, j \in \mathbb{Z}$, $i - j > 2$; such that both x_i, x_j are simplicial or that $x_{i+1} = x_j$. Then, unless G is bisimplicial in the first case, the vertices x_i, x_j are adjacent. Denote by C the cycle on $x_j, x_{j+1}, \ldots, x_i$ in G, and by $e = x_j x_i$. We have found the required objects C, e. □

Suppose that G_0 is a counterexample to Lemma 2 on the smallest possible number of vertices. Then G_0 is a non-bisimplicial chordal graph on more than one vertex. Since an induced subgraph of a chordal graph is chordal by definition, the (smaller) neighbourhood subgraphs $N_{G_0}(v)$ are bisimplicial for each $v \in V(G_0)$. We have a contradiction between Lemmas 3 and 5 for $G = G_0$.

REFERENCES

[1] P. Hliněný and G. Whittle, *Tree-Width and Superchordal Matroids*, in preparation.
[2] C.B. Lekkerkerker and J.C. Boland, *Representation of finite graphs by a set of intervals on the real line*, Fund. Math. **51** (1962), 45–64.

INSTITUTE OF MATHEMATICS AND COMP. SCIENCE (IMI SAV & UMB); SEVERNÁ 5; SK–974 00 BANSKÁ BYSTRICA; SLOVAK REPUBLIC
E-mail: hlineny@ member.ams.org

Acta Univ. M. Belii
Math no. 10(2003), pp. 21–44

MINIMAL REPRESENTATIVES OF $\mathcal{G}$-CLASSES OF 3-MANIFOLDS OF GENUS TWO

Ján Karabáš and Roman Nedela

Abstract. One of the central problems for 3-manifolds is the isomorphism problem. Since 70's several methods to attack it were developed. The method introduced in a paper of Ferri and Gagliardi is not easy to use, since no bound for the number of steps in a computer representation is known. Some approximations were introduced in the paper of Grasselli, Mulazzani and Nedela. The present method based on these approximations leads to a simple algorithm finding representatives of a given equivalence classes of 3-manifolds of genus two. We have applied the algorithm to reduce a known list of representatives of 3-manifolds of genus 2 and to derive some new results as well.

Introduction

A n-manifold ($n \geq 1$) is the topological space, in which every point has a neighbourhood $O(x)$ homeomorphic to the n-dimensional Euclidean space. Next, every compact connected n-manifold, $n \leq 3$, can be expressed as a simplicial complex containing a finite set of simplices of dimension n. For instance, a compact connected surface can be triangulated. However, we can form a triangulation of a surface by infinitely many ways. For example, we can choose a point in a triangle of a given triangulation, connect it with the vertices of that triangle and form a new triangulation of the same surface. A general problem is to decide, whether two different simplicial complexes represent the same n-manifold. In what follows we shall only consider compact connected piecewise-linear 3-manifolds. Each such a 3-manifold can be triangulated as was already mentioned.

There is a well defined equivalence relation on the set of n-dimensional complexes representing n-manifolds based on *wave moves* [4]. This equivalence allows us to decide, which simplicial complexes represent the same n-manifold. Unfortunately, the straight use of wave moves to solve the above isomorphism problem seems to be intractable. In fact, if $n > 2$ no limit for the number of steps (moves) needed to decide whether two complexes determine the same compact connected manifold is known.

2000 *Mathematics Subject Classification.* 57M15; Secondary 57M27,05C10.
Key words and phrases. manifold, simplicial complex, graph.
This work was supported by Science and Technology Assistance Agency under the contract No. APVT-51-012502
Received 15. 11. 2002; Accepted 16. 4. 2003

Let $\mathcal{M}$ be an n-manifold. It is well known, that a simplicial complex representing $\mathcal{M}$ can be represented by a graph $\Gamma(\mathcal{M})$, which vertices represents simplices of dimension n of the complex and edges represent a "gluing" of maximal simplices of the complex in subsimplices of dimension $n-1$. Given graph $\Gamma(\mathcal{M})$ can be "drawn" on an orientable surface. The *genus* of $\mathcal{M}$ is the minimal genus of an orientable surface into which $\Gamma(\mathcal{M})$ embeds in a particular way described in the next section.

Following [3] we represent a 3-manifold of genus two as a vector of integers of length six and consider certain equivalence relations defined on 6-tuples and preserving the associated 3-manifold of genus two. There is an equivalence on the set of 6-tuples introduced in [2] called $\mathcal{H}$-equivalence. In [5] other equivalence on the set of 6-tuples is defined. This equivalence is called $\mathcal{G}$-equivalence and it extends $\mathcal{H}$-equivalence. If f and g are $\mathcal{G}$-equivalent 6-tuples then they represent isomorphic 3-manifolds of genus two. Hence the $\mathcal{G}$-equivalence provides an approximation of the "isomorphism problem".

Main aim of this paper is to investigate the $\mathcal{G}$-equivalence in details. As an application a list of representations of "small" 3-manifolds of genus two is produced.

Preliminaries

Each 3-dimensional simplicial complex can be represented by a bipartite 4-edge-coloured graph. Let T be any simplicial triangulation and T' be its first barycentric subdivision. Each vertex $\hat{\omega}$, which is the barycenter of the simplex ω of T is labelled by the dimension of ω. Take the dual graph Γ of T' and if uv is an edge and $\{i,j,k\}$ are the colours of respective triangle in T use the colour complementary to $\{i,j,k\}$ to colour the edge uv. The labelling of vertices of T induces a decomposition of the tetrahedrons of T into two classes, where adjacent tetrahedrons belong to different classes. Thus Γ is bipartite. The dual graph Γ of T', together with the edge-colouring ν, is a 4-coloured graph, representing T.

Definition 1. Let $\Gamma = (V(\Gamma), E(\Gamma)))$ be a bipartite graph and let there exist a mapping $\nu : E(\Gamma) \to \Delta_4 = \{1,2,3,4\}$ such that for all incident edges $f, g \in E(\Gamma)$: $\nu(f) \neq \nu(g)$. This mapping called a *graph colouring* and the graph Γ_{Δ_4} *4-coloured graph*.

It is proved [4] that the above mentioned simplicial complexes can be represented by a 4-coloured bipartite and connected graph Γ_{Δ_4} (next, the graph). The colouring is regular, i.e. two incident edges share distinct colours. Since the colouring is regular a factor induced by two colours is a disjoint union of bicoloured cycles. Let $\mathcal{I}$ denotes the set of 2-cell embeddings of Γ_{Δ_4} into a closed orientable surface such that the local rotation of colours induced by the embedding in "black" vertices is the same, say ρ, while the local rotation of colours in "white" vertices is ρ^{-1}. Note that there are six possibilities for choosing ρ. It follows that faces of such embedding are bounded by bicoloured cycles. Out of these six possibilities for ρ we choose such ρ that the genus of the underlying surface is minimal in $\mathcal{I}$. The integer g is an invariant of a 3-manifold $\mathcal{M}$ represented by Γ_{Δ_4} and it is called the *regular genus* of $\mathcal{M}$ (or shortly the genus of $\mathcal{M}$). It is known that the regular genus of $\mathcal{M}$ is equal to the Heegaard genus of $\mathcal{M}$ [1].

Let Γ_{Δ_4} is a 4-coloured graph and let Θ is subgraph of Γ_{Δ_4} contains of vertices X, Y joined by h edges ($1 \leq h \leq 3$) coloured by colours $c_1, \ldots, c_h$. If X and Y are in two different components of graph $\Gamma_{\Delta_4 - \{c_1,\ldots,c_h\}}$ induced by the set of

complementary colours $\Delta_4 - \{c_1, \ldots, c_h\}$ then the subgraph Θ will be called a *dipole of type* h.

There is a well defined operator over the set of 4-coloured graphs [4] called *wave move*. Note that a wave move can be defined for $(n+1)$-coloured, connected and bipartite graphs (n-manifolds) generally.

Definition 2. If Θ is a dipole of type h in Γ_{Δ_4} coloured by colours $\{c_1, \ldots, c_h\}$ we define a wave move as follows (see Fig. 1):

(a) Cutting of Θ

- remove edges and vertices of Θ
- glue "hanging" edges of graph Γ_{Δ_4} of same colour

(b) Adding of Θ as inverse to cutting

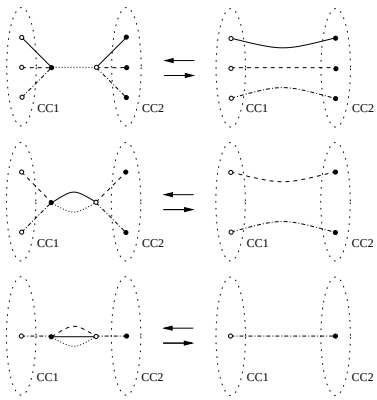

FIG. 1. Wave moves

The main result of [4] states that graphs Γ_{Δ_4} and Γ'_{Δ_4} represent isomorphic 3-manifolds if and only if there is a finite sequence of wave-moves transforming Γ_{Δ_4} to Γ'_{Δ_4}. Hence the "isomorphism problem" reduces to the problem to decide whether two 4-coloured graphs are "wave-move equivalent".

It follows from [3] that each (closed) genus two 3-manifold can be represented by a graph Γ_{Δ_4} which structure can be coded by a 6-tuple of integers satisfying certain conditions. Let $\widetilde{\mathcal{F}}_2$ is set of 6-tuples:

$$f = (h_0, h_1, h_2; q_0, q_1, q_2), h_i, q_i \in \mathbb{N}.$$

The set of 6-tuples representing genus two 3-manifolds satisfy the following axioms:

(i) $\forall i \in \mathbb{Z}_3 : h_i > 0$,
(ii) all h_i has the same parity,
(iii) $\forall i \in \mathbb{Z}_3 : 0 \leq q_i < h_{i-1} + h_i = 2l_i$,
(iv) all q_i has the same parity.

Remark 1. From here all operations with numbers q_i will be considered modulo $2l_i$, and according to (iii), q_i will be always the least non-negative integer of the class.

Now let us define the set $V(f)$ for a 6-tuple $f \in \widetilde{\mathcal{F}}_2$:

$$V(f) = \bigcup_{i \in \mathbb{Z}_3} \{i\} \times \mathbb{Z}_{2l_i}$$

and the following permutations on $V(f)$:

$$\begin{aligned}
\alpha_0(i,j) &= (i, j + (-1)^j), \\
\alpha_1(i,j) &= (i, j - (-1)^j), \\
\alpha_2(i,j) &= \begin{cases} (i+1, 2l_k - j - 1)\,;\, k = i+1\,; & 0 \le j < h_i \\ (i-1, 2l_k - j - 1)\,;\, k = i\,; & h_i \le j < 2l_i \end{cases}, \\
\alpha_3(i,j) &= \rho \circ \alpha_2 \circ \rho^{-1},
\end{aligned}$$

where $\rho : V(f) \to V(f)$ is a bijection defined by rule

$$\rho(i,j) = (i, j + q_i).$$

Now let $f \in \mathcal{F}_2$ satisfy the following conditions:

(v) $\forall i \in \mathbb{Z}_3 : h_i + q_i$ is odd, h_i and q_i have different parity,

(vi) the group $\langle \alpha_2, \alpha_3 \rangle$ has exactly three orbits.

Given 6-tuple f we define the associated graph $\Gamma_{\Delta_4}(f)$ as follows. Let $V = V(f)$ be the set of vertices of $\Gamma_{\Delta_4}(f)$. Then the permutations $\alpha_0, \alpha_1, \alpha_2$ and α_3 define the decomposition of the edge set into four colours, the orbits of α_i form the edges of Γ_{Δ_i} coloured by i, for $i = 0, 1, 2, 3$. Observe that the subgraphs $\Gamma_{\{0,1,2\}}$, $\Gamma_{\{0,1,3\}}$ induced by the respective sets of colours are isomorphic planar graphs.

Vice-versa let Γ_{Δ_4} be a 4-coloured graph with a bicoloured 2-factor containing three circles of even length C_0, C_1, C_2 coloured by colours 0 and 1. Other edges coloured by colours 2 and 3 join vertices of Γ_{Δ_4} such that the induced subgraphs $\Gamma_{\{0,1,2\}}$ and $\Gamma_{\{0,1,3\}}$ are planar and isomorphic. Now, let us code the graph by the 6-tuple $f = (h_0, h_1, h_2; q_0, q_1, q_2)$ [2]. The first three items code the numbers of edges coloured by 2 (3) joining the circles C_{i-1} and C_i, $(i = 0, 1, 2)$ of Γ_{Δ_4}. Clearly, the planar subgraphs $\Gamma_{\{0,1,2\}} \simeq \Gamma_{\{0,1,3\}}$ of Γ_{Δ_4} are uniquely determined by the integers h_0, h_1 and h_2. Then Γ_{Δ_4} arises by gluing $\Gamma_{\{0,1,2\}}$ with $\Gamma_{\{0,1,3\}}$ in the three cycles C_0, C_1 and C_2 coloured by 0 and 1. The integers q_0, q_1 and q_2 determine the rotations of cycles C_0, C_1, C_2 in $\Gamma_{\{0,1,3\}}$ before the gluing is done. In this way we get an embedding of Γ_{Δ_4} into bitorus (see Fig. 2)

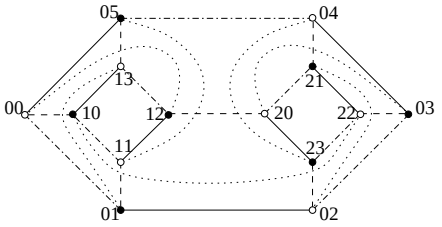

FIG. 2. The graph represented by 6-tuple $(3, 1, 3; 2, 2, 0)$

The conditions (i) – (vi) come in part from the interpretation while in part they are forced by the requirement that Γ represents a compact connected 3-manifold of genus 2.

Theorem 1 [3]**.** For every compact connected 3-manifold $\mathcal{M}$ of genus ≤ 2 there exists $f \in \mathcal{F}_2$ such that $\Gamma(f)$ represents $\mathcal{M}$.

Definition 3. The elements of the set $\mathcal{F}_2 \subset \widetilde{\mathcal{F}}_2$ satisfying conditions (i) – (vi) will be called *admissible 6-tuples*.

Definition 4. Let $f \in \mathcal{F}_2$. The number $z(f) = h_0+h_1+h_2$ is called the *complexity of the 6-tuple* f.

It is easy to design an algorithm to verify the conditions (i) – (vi) for a given integer vector with six items. The most complicated seems to be to verify the condition (vi), but the complexity of this algorithm is polynomial, depending on complexity of given 6-tuple f. Therefore we can construct the set $\mathcal{F}_2$ up to a fixed complexity in an effective way.

Now we introduce the equivalence relations on $\mathcal{F}_2$ defined in [2] and [5]. If $f = (h_0, h_1, h_2; q_0, q_1, q_2)$ is an admissible 6-tuple define the permutations ψ_1, ψ_2, ψ_3 acting on $\mathcal{F}_2$ as follows [2]:

$$\begin{aligned}
\psi_1(h_0, h_1, h_2; q_0, q_1, q_2) &= (h_1, h_2, h_0; q_1, q_2, q_0)\\
\psi_2(h_0, h_1, h_2; q_0, q_1, q_2) &= (h_2, h_1, h_0; q_0, q_2, q_1)\\
\psi_3(h_0, h_1, h_2; q_0, q_1, q_2) &= (h_0, h_1, h_2; 2l_0 - q_0, 2l_1 - q_1, 2l_2 - q_2)
\end{aligned}$$

The above described permutations represents some recolourings of the graph $\Gamma_{\Delta_4}(f)$.

Definition 5. Let $f, g \in \mathcal{F}_2$. Let us define the relation

$$f \overset{\mathcal{H}}{\approx} g : \exists \eta \in \langle \psi_1, \psi_2, \psi_3 \rangle, \eta(f) = g$$

This relation is an equivalence and we will call it $\mathcal{H}$*-equivalence on* $\mathcal{F}_2$. The equivalence classes will be called $\mathcal{H}$*-orbits*.

Lemma 1. [2, Prop. 16] $\mathcal{H}$-equivalence preserves the admissibility of the 6-tuple.

Lemma 2. The group $\mathcal{H} = \langle \psi_1, \psi_2, \psi_3 \rangle$ is isomorphic to $\mathcal{D}_{12}$, where $\mathcal{D}_{12}$ is the group of symmetries of a regular hexagon. In particular, each $\mathcal{H}$-orbit has at most 12 elements.

Proof. It follows from the definition of ψ_1, ψ_2, ψ_3 that $\psi_1^3 = \psi_2^2 = \psi_3^2 = 1$. The group $\langle \psi_1, \psi_2 \rangle$ is isomorphic to the group $\mathcal{S}_3$ of symmetries of a regular triangle because

$$\psi_2 \psi_1 \psi_2 = \psi_1^{-1}.$$

Also ψ_3 commutes with the members of $\langle \psi_1, \psi_2 \rangle$. Hence the group $\mathcal{H}$ satisfies the relations of dihedral group $\mathcal{D}_{12}$. Thus $\mathcal{H}$ is an epimorphic image of $\mathcal{D}_{12}$. To prove that the epimorphism is an isomorphism it is sufficient to find at least one admissible 6-tuple such that the respective $\mathcal{H}$-orbit has 12 different 6-tuples. The 6-tuple $(1, 3, 5; 2, 2, 2)$ is the such a 6-tuple. □

Following [5], let us define mapping $\sigma : \mathcal{F}_2 \to \mathcal{F}_2$:

$$\sigma(h_0, h_1, h_2; q_0, q_1, q_2) = \begin{cases} (h_0, h_1, h_2; q_0, q_1, q_2); & \text{if } q_0 = 0 \\ (h_0', h_1', h_2'; q_0', q_1', q_2'); & \text{if } q_0 \neq 0 \end{cases}$$

where $f = (h'_0, h'_1, h'_2; q'_0, q'_1, q'_2)$ is a 6-tuple defined by the next rules

$$\left.\begin{array}{ll} h'_0 = h_0 + h_1 - q_0 & q'_0 = h_0 + h_1 + h_2 - 2q_0 \\ h'_1 = q_0 & q'_1 = q_0 + q_1 + h_1 \\ h'_2 = h_2 + h_1 - q_0 & q'_2 = q_0 + q_2 + h_1 \end{array}\right\} \quad \text{iff } 0 < q_0 < h_0, h_2$$

$$\left.\begin{array}{ll} h'_0 = q_0 + h_1 - h_2 & q'_0 = h_1 \\ h'_1 = h_0 + h_2 - q_0 & q'_1 = q_0 + q_1 - h_2 \\ h'_2 = q_0 + h_1 - h_0 & q'_2 = q_0 + q_2 - h_0 \end{array}\right\} \quad \text{iff } q_0 > h_0, h_2$$

$$\left.\begin{array}{ll} h'_0 = h_1 & q'_0 = h_1 + h_2 - q_0 \\ h'_1 = h_0 & q'_1 = q_1 \\ h'_2 = h_1 + h_2 - h_0 & q'_2 = 2q_0 + q_2 + h_1 - h_0 \end{array}\right\} \quad \text{iff } h_0 < q_0 < h_2$$

$$\left.\begin{array}{ll} h'_0 = h_1 + h_0 - h_2 & q'_0 = h_1 + h_0 - q_0 \\ h'_1 = h_2 & q'_1 = 2q_0 + q_1 + h_1 - h_2 \\ h'_2 = h_1 & q'_2 = q_2 \end{array}\right\} \quad \text{iff } h_2 < q_0 < h_0$$

The above described operation represents a sequence of wave moves such that applying it to the graph represented by an admissible 6-tuple we get the new graph, which can be represented by an admissible 6-tuple too.

Definition 6. Let $f, g \in \mathcal{F}_2$. We define a relation:

$$f \overset{\mathcal{G}}{\approx} g : \exists \gamma \in \langle \psi_1, \psi_2, \psi_3, \sigma \rangle, \gamma(f) = g$$

This relation will be called $\mathcal{G}$-*equivalence on* $\mathcal{F}_2$. The equivalence classes will be called $\mathcal{G}$-*orbits* and will be marked as usual $[f]_{\mathcal{G}}$.

Agreement. Denote by $[f]_{\mathcal{H}}$ a $\mathcal{H}$-orbit containing f. Similar, denote by $[f]_{\mathcal{G}}$ a $\mathcal{G}$-orbit containing f.

Lemma 3. [5, Th. 5.1] $\mathcal{G}$-equivalence preserves the admissibility of the given 6-tuple.

Obviously, any $\mathcal{G}$-orbit is a union of some $\mathcal{H}$-orbits.

Definition 7. Let $\mathcal{H}, \mathcal{H}'$ be two different $\mathcal{H}$-orbits. Let $f \in \mathcal{H} \wedge g \in \mathcal{H}' : g = \sigma(f)$. Then we define a *derivation of* f as the difference $\delta(f) = z(g) - z(f)$ [5].

Straightforward from Definitions 6 and 7 we get the following lemma.

Lemma 4 [5]**.** With the above notation

$$\delta(f) = \begin{cases} 0 & \text{iff } q_0 = 0 \quad \text{(a)} \\ h_1 - q_0 & \text{iff } 0 < q_0 < h_0, h_2 \quad \text{(b)} \\ h_1 - h_0 & \text{iff } h_0 < q_0 < h_2 \quad \text{(c)} \\ h_1 - h_2 & \text{iff } h_2 < q_0 < h_0 \quad \text{(d)} \\ q_0 + h_1 - h_0 - h_2 & \text{iff } q_0 > h_0, h_2 \quad \text{(e)} \end{cases}$$

Note. We denote by $f_i (i = 1, 2, \ldots, 6)$ the i-th item of the vector representing an admissible 6-tuple.

Definition 8. Let $f, g \in \mathcal{F}_2$ be two 6-tuples. Let $I = \{1, 2, 3, 4, 5, 6\}$ be the set of indexes of components of these vectors. We define the lexical order $\prec$ as follows:

$$f \prec g \Leftrightarrow \text{for } j = \inf\{i | (i \in I) \wedge f(i) \neq g(i)\}, f(j) < g(j)$$

Definition 9. Using the lexical order we derive an order on $\mathcal{F}_2$ in the following way

$$f \ll g \Leftrightarrow (z(f) < z(g)) \vee ((z(f) = z(g)) \wedge (f \prec g))$$

We call this order the *natural order of* $\mathcal{F}_2$.

Finally, we define representatives of $\mathcal{H}$-orbits.

Definition 10. Let $F \subseteq \mathcal{F}_2$.
The member f of F satisfying

$$f \in F : \neg(\exists g \in F), g \ll f$$

is called a *minimal representative of* F.
The member f of F satisfying

$$f \in F : \forall g \in F, f \ll g$$

is called the *least representative of* F.

Since $\mathcal{F}_2$ with respect to $\ll$ is a well-ordered set, for each F there exists a unique minimal representative which is in the same time the least representative of F.

Agreement. In the notation $[f]_{\mathcal{H}}$ denoting an orbit of $\mathcal{H}$-equivalence we shall always assume that 6-tuple f is minimal unless otherwise follows from the context.

To create a $\mathcal{H}$-orbit from a given f is a trivial problem, which can be represented by simple algorithm following i.e. from [2, Prop. 16]. By Lemma 2 the members of $[f]_{\mathcal{H}}$ are

$$\begin{aligned}
f &= (h_0, h_1, h_2; q_0, q_1, q_2) \\
\psi_1 f &= (h_1, h_2, h_0; q_1, q_2, q_0) \\
\psi_2 f &= (h_2, h_1, h_0; q_0, q_2, q_1) \\
\psi_3 f &= (h_0, h_1, h_2; 2l_0 - q_0, 2l_1 - q_1, 2l_2 - q_2) \\
\psi_1^2 f &= (h_2, h_0, h_1; q_2, q_0, q_1) \\
\psi_2\psi_1 f &= (h_0, h_2, h_1; q_1, q_0, q_2) \\
\psi_3\psi_1 f &= (h_1, h_2, h_0; 2l_1 - q_1, 2l_2 - q_2, 2l_0 - q_0) \\
\psi_2\psi_1^2 f &= (h_1, h_0, h_2; q_2, q_1, q_0) \\
\psi_3\psi_1^2 f &= (h_2, h_0, h_1; 2l_2 - q_2, 2l_0 - q_0, 2l_1 - q_1) \\
\psi_3\psi_2 f &= (h_2, h_1, h_0; 2l_0 - q_0, 2l_2 - q_2, 2l_1 - q_1) \\
\psi_3\psi_2\psi_1 f &= (h_0, h_2, h_1; 2l_1 - q_1, 2l_0 - q_0, 2l_2 - q_2) \\
\psi_3\psi_2\psi_1^2 f &= (h_1, h_0, h_2; 2l_2 - q_2, 2l_1 - q_1, 2l_0 - q_0)
\end{aligned}$$

Similarly, it is not complicated to compute an image $\sigma(f)$ for any $f \in \mathcal{F}_2$. On the other hand, a $\mathcal{G}$-orbit may be infinite. In what follows we give a simple method for deciding whether $g \approx_{\mathcal{G}} h$.

Example. The 6-tuple $(1,3,k;2,2,k-1)$ $k \geq 3$ belongs to an infinite $\mathcal{G}$-orbit. Since $\sigma((1,3,k;2,2,k-1)) = (3,1,k+2;k+1,2,2)$ and this 6-tuple is $\mathcal{H}$-equivalent to the 6-tuple $(1,3,k+2;2,2,k+1)$. Hence $(1,3,k+2;2,2,k+1) \approx_{\mathcal{G}} (1,3,k;2,2,k-1)$, and $z(\sigma^{j+1}(f)) > z(\sigma^j(f))$ for every positive integer j. Thus $[(1,3,k;2,2,k-1)]_{\mathcal{G}}$ is infinite.

Representatives of $\mathcal{G}$-orbits

Next lemma appears in [5, Prop. 6.1] without proof.

Lemma 5. Let $f \in \mathcal{F}_2$. Then ψ_2, ψ_3 and σ satisfy the following relations:

$$\begin{aligned} &\text{a) } \sigma^2 = 1 \\ &\text{b) } \psi_2\sigma = \sigma\psi_2 \\ &\text{c) } \psi_3\sigma = \sigma\psi_3 \end{aligned}$$

Proof. The proof is done by direct computation. We have to deal with four cases related with the action of σ-operator. Recall that all the computations with q_i will be done modulo $h_i + h_{i+1}, i \in \mathbb{Z}_3$.

a) Let $f' = \sigma(f)$ and $f'' = \sigma(f')$. We prove $f = f''$.
The case $q_0 = 0$ implies the identity by definition.

$$\text{(I)} \qquad \begin{aligned} &0 < q_0 < h_0 \Rightarrow h_0 + h_1 - q_0 < h_0 + h_1 + h_2 - 2q_0 \Rightarrow h_0' < q_0' \\ &0 < q_0 < h_2 \Rightarrow h_2 + h_1 - q_0 < h_0 + h_1 + h_2 - 2q_0 \Rightarrow h_2' < q_0' \end{aligned}$$

Hence we have to apply Case II in the definition of σ to compute $\sigma(f'')$

$$\begin{aligned} h_0'' &= (h_0 + h_1 + h_2 - 2q_0) + q_0 - (h_2 + h_1 - q_0) = h_0 \\ h_1'' &= (h_0 + h_1 - q_0) + (h_2 + h_1 - q_0) - (h_0 + h_1 + h_2 - 2q_0) = h_1 \\ h_2'' &= (h_0 + h_1 + h_2 - 2q_0) + q_0 - (h_0 + h_1 - q_0) = h_2 \\ q_0'' &= q_0 \bmod (h_0 + h_2) \\ q_1'' &= (h_0 + h_1 + h_2 - 2q_0) + (q_0 + q_1 + h_1) - (h_2 + h_1 - q_0) \equiv q_1 \bmod (h_0 + h_1) \\ q_2'' &= (h_0 + h_1 + h_2 - 2q_0) + (q_0 + q_2 + h_1) - (h_0 + h_1 - q_0) \equiv q_2 \bmod (h_1 + h_2) \end{aligned}$$

$$\text{(II)} \qquad \begin{aligned} &q_0 > h_0, h_2 \Rightarrow q_0 - h_2 > 0 \Rightarrow h_1 < h_1 + (q_0 - h_2) \Rightarrow 0 < q_0' < h_0' \\ &q_0 > h_0, h_2 \Rightarrow q_0 - h_0 > 0 \Rightarrow h_1 < h_1 + (q_0 - h_0) \Rightarrow 0 < q_0' < h_2' \end{aligned}$$

Hence we have to apply Case I in the definition of σ to compute $\sigma(f'')$

$$\begin{aligned} h_0'' &= (q_0 + h_1 - h_2) + (h_0 + h_2 - q_0) - h_1 = h_0 \\ h_1'' &= h_1 \\ h_2'' &= (q_0 + h_1 - h_0) + (h_0 + h_2 - q_0) - h_1 = h_2 \\ q_0'' &= (q_0 + h_1 - h_2) + (h_0 + h_2 - q_0) + (q_0 + h_1 - h_0) - 2h_1 = q_0 \bmod (h_0 + h_2) \\ q_1'' &= h_1 + (q_0 + q_1 - h_2) + (h_0 + h_2 - q_0) \equiv q_1 \bmod (h_0 + h_1) \\ q_2'' &= h_1 + (q_0 + q_2 - h_0) + (h_0 + h_2 - q_0) \equiv q_2 \bmod (h_1 + h_2) \end{aligned}$$

(III)
$$q_0 < h_2 \Rightarrow h_1 < (h_2 - q_0) + h_1 \Rightarrow h'_0 < q'_0$$
$$q_0 > h_0 \Rightarrow (h_1 + h_2) - q_0 < (h_1 + h_2) - h_0 \Rightarrow q'_0 < h'_2$$

Hence we have to apply Case III in the definition of σ to compute $\sigma(f'')$

$$\begin{aligned}
h''_0 &= h_0 \\
h''_1 &= h_1 \\
h''_2 &= h_0 + (h_1 + h_2 - h_0) - h_1 = h_2 \\
q''_0 &= h_0 + (h_1 + h_2 - h_0) - (h_1 + h_2 - q_0) = q_0 \bmod (h_0 + h_2) \\
q''_1 &= q_1 \bmod (h_0 + h_1) \\
q''_2 &= 2(h_1 + h_2 - q_0) + (2q_0 + q_2 + h_1 - h_0) + h_0 - h_1 \equiv q_2 \bmod (h_1 + h_2)
\end{aligned}$$

(IV)
$$q_0 > h_2 \Rightarrow (h_1 + h_0) - h_2 > (h_1 + h_0) - q_0 \Rightarrow h'_0 > q'_0$$
$$h_0 > q_0 \Rightarrow (h_0 - q_0) + h_1 > h_1 \Rightarrow q'_0 > h'_2$$

Hence we have to apply Case IV in the definition of σ to compute $\sigma(f'')$

$$\begin{aligned}
h''_0 &= h_2 + (h_1 + h_0 - h_2) - h_1 = h_0 \\
h''_1 &= h_1 \\
h''_2 &= h_2 \\
q''_0 &= h_2 + (h_1 + h_0 - h_2) - (h_1 + h_0 - q_0) = q_0 \bmod (h_0 + h_2) \\
q''_1 &= 2(h_1 + h_0 - q_0) + (2q_0 + q_1 + h_1 - h_2) + h_2 - h_1 \equiv q_1 \bmod (h_0 + h_1) \\
q''_2 &= q_2 \bmod (h_1 + h_2)
\end{aligned}$$

b) In the following calculations the usage of the respective Case in computation of images under σ is signed as follows $\ldots \stackrel{I.}{=} \ldots, \ldots \stackrel{IV.}{=} \ldots$. Let $f' = \sigma(f)$ and $f'' = \psi_2(f)$

(I) $0 < q_0 < h_0, h_2 \Rightarrow \psi_2(f') \stackrel{I.}{=}$

$\stackrel{I.}{=} \psi_2(h_0 + h_1 - q_0, q_0, h_2 + h_1 - q_0; h_0 + h_1 + h_2 - 2q_0, q_0 + q_1 + h_1, q_0 + q_1 + h_1) =$

$= (h_2 + h_1 - q_0, q_0, h_0 + h_1 - q_0; h_2 + h_1 + h_0 - 2q_0, q_0 + q_2 + h_1, q_0 + q_1 + h_1) \stackrel{I.}{=}$

$\stackrel{I.}{=} \sigma(h_2, h_1, h_0; q_0, q_2, q_1) = \sigma(f'')$

(II) $q_0 > h_0, h_2 \Rightarrow \psi_2(f') \stackrel{II.}{=}$

$\stackrel{II.}{=} \psi_2(q_0 + h_1 - h_2, h_0 + h_2 - q_0, q_0 + h_1 - h_0; h_1, q_0 + q_1 - h_2, q_0 + q_2 - h_0) =$

$= (q_0 + h_1 - h_0, h_2 + h_0 - q_0, q_0 + h_1 - h_2; h_1, q_0 + q_2 - h_0, q_0 + q_1 - h_2) \stackrel{II.}{=}$

$\stackrel{II.}{=} \sigma(h_2, h_1, h_0; q_0, q_2, q_1) = \sigma(f'')$

Note, that using ψ_2 in Cases (III) and (IV) of σ swaps the input conditions. However, we need to prove the following equalities:

Using Case (IV) in the definition of σ we get:

(III) $h_0 < q_0 < h_2 \Rightarrow \psi_2(f') \overset{III.}{=}$

$\overset{III.}{=} \psi_2(h_1, h_0, h_1 + h_2 - h_0; h_1 + h_2 - q_0, q_1, 2q_0 + q_2 + q_2 + h_1 - h_0) =$

$= (h_1 + h_2 - h_0, h_0, h_1; h_1 + h_0 - q_0, 2q_0 + q_2 + h_1 - h_0, q_1) \overset{IV.}{=}$

$\overset{IV.}{=} \sigma(h_2, h_1, h_0; q_0, q_2, q_1) = \sigma(f'')$

Using Case (III) in the definition of σ we get:

(IV) $h_2 < q_0 < h_0 \Rightarrow \psi_2(f') \overset{IV.}{=}$

$\overset{IV.}{=} \psi_2(h_1 + h_0 - h_2, h_2, h_1; h_1 + h_0 - q_0, q_2, 2q_0 + q_1 + h_1 - h_2) =$

$= (h_1, h_2, h_1 + h_0 - h_2; h_1 + h_0 - q_0, q_2, 2q_0 + q_1 + h_1 - h_2) \overset{III.}{=}$

$\overset{III.}{=} \sigma(h_2, h_1, h_0; q_0, q_2, q_1) = \sigma(f'')$

c) To prove commutativity of ψ_3 and σ note that for the minimum non-negative representatives q_0, q_1, q_2 of the repective residual classes the following relations hold (see Remark 1):

$$-q_i = (h_i + h_{i-1}) - q_i \bmod (h_i + h_{i-1}); i \in \mathbb{Z}_3$$

$q_i < h_{i-1} \Rightarrow (h_i + h_{i-1}) - q_i > (h_i + h_{i-1}) - h_{i-1} \Rightarrow$

$\Rightarrow -q_i > h_i \bmod (h_i + h_{i-1})$

$q_i < h_i \Rightarrow (h_i + h_{i-1}) - q_i > (h_i + h_{i-1}) - h_i \Rightarrow$

$\Rightarrow -q_i > h_{i-1} \bmod (h_i + h_{i-1})$

Let $f' = \psi_3 \sigma f$ and $f'' = \sigma \psi_3 f$. We have to prove $f' = f''$.

I) $0 < q_0 < h_0, h_2$

$h'_0 = h_0 + h_1 - q_0$

$h'_1 = q_0$

$h'_2 = h_2 + h_1 - q_0$

$q'_0 = (h_0 + h_1 - q_0) + (h_2 + h_1 - q_0) - (h_0 + h_1 + h_2 - 2q_0) \equiv h_1 \bmod (2h_1 + h_0 + h_2 - 2q_0)$

$q'_1 = (h_0 + h_1 - q_0) + q_0 - (q_0 + q_1 + h_1) = h_0 - q_0 - q_1 \bmod (h_0 + h_1)$

$q'_2 = q_0 + (h_2 + h_1 - q_0) - (q_0 + q_2 + h_1) \equiv h_2 - q_0 - q_2 \bmod (h_1 + h_2)$

Since $0 < q_0 < h_0, h_2 \Rightarrow -q_0 > h_0, h_2 \bmod (h_0 + h_2)$ Case II in calculation of f'' applies.

$h''_0 = (h_0 + h_2) - q_0 + h_1 - h_2 = h_0 + h_1 - q_0$

$h''_1 = h_0 + h_2 - ((h_0 + h_2) - q_0) = q_0$

$h''_2 = (h_0 + h_2) - q_0 + h_1 - h_0 = h_2 + h_1 - q_0$

$q''_0 = h_1 \bmod (2h_1 + h_0 + h_2 - 2q_0)$

$q''_1 = (h_0 + h_2 - q_0) + (h_0 + h_1 - q_1) - h_2 \equiv h_0 - q_0 - q_1 \bmod (h_0 + h_1)$

$q''_2 = (h_0 + h_2 - q_0) + (h_1 + h_2 - q_2) \equiv h_2 - q_0 - q_2 \bmod (h_1 + h_2)$

II) $h_0, h_2 < q_0$

$h'_0 = h_1 - h_2 + q_0$

$h'_1 = h_0 + h_2 - q_0$

$h'_2 = h_1 - h_0 + q_0$

$q'_0 = (h_1 - h_2 + q_0) + (h_1 - h_0 + q_0) - h_1 =$

$= h_1 - h_0 - h_2 + 2q_0 \equiv -h_1 \bmod (2h_1 - h_0 - h_2 + 2q_0)$

$q'_1 = (h_1 + h_2 + q_0) + (h_1 - h_0 + q_0) - (q_0 + q_1 - h_2) =$

$= h_0 + h_1 + h_2 - q_0 - q_1 \equiv h_2 - q_0 - q_1 \bmod (h_1 + h_0)$

$q'_2 = (h_0 + h_2 - q_0) + (h_1 - h_0 + q_0) - q_0 + q_2 - h_0 =$

$= h_0 + h_1 + h_2 - q_0 - q_2 \equiv h_0 - q_0 - q_0 \bmod (h_1 + h_2)$

Since $h_0, h_2 < q_0 \Rightarrow 0 < -q_0 < h_0, h_2 \bmod (h_0 + h_2)$ Case I in calculation of f'' applies.

$h''_0 = h_0 + h_1 - (h_0 + h_2 - q_0) = h_1 - h_2 + q_0$

$h''_1 = h_0 + h_2 - q_0$

$h''_2 = h_2 + h_1 - (h_0 + h_2 - q_0) = h_1 - h_0 + q_0$

$q''_0 = h_0 + h_1 + h_2 - 2(h_0 + h_2 - q_0) =$

$= h_1 - h_0 - h_2 + 2q_0 \equiv -h_1 \bmod (2h_1 - h_0 - h_2 + 2q_0)$

$q''_1 = (h_0 + h_2 - q_0) + (h_0 + h_1 - q_1) + h_1 =$

$= h_0 + h_1 + h_2 - q_0 - q_1 \equiv h_2 - q_0 - q_1 \bmod (h_0 + h_1)$

$q''_2 = (h_0 + h_2 - q_0) + (h_1 + h_2 - q_2) + h_1 =$

$= h_0 + h_1 + h_2 - q_0 - q_2 \equiv h_0 - q_0 - q_2 \bmod (h_1 + h_2)$

III) $h_0 < q_0 < h_2$

$h'_0 = h_1$

$h'_1 = h_0$

$h'_2 = h_1 + h_2 - h_0$

$q'_0 = [h_1 + (h_1 + h_2 - h_0)] - (h_1 + h_2 - q_0) = h_1 - h_0 + q_0 \bmod (2h_1 + h_2 - h_0)$

$q'_1 = [h_1 + h_0 - q_1] \equiv -q_1 \bmod (h_1 + h_0)$

$q'_2 = [h_0 + (h_1 + h_2 - h_0)] - (2q_0 + q_2 + h_1 - h_0) = h_2 + h_0 - 2q_0 - q_2 \bmod (h_1 + h_2)$

Since $h_0 < q_0 < h_2 \Rightarrow h_0 < -q_0 < h_2 \bmod (h_0 + h_2)$ Case III in calculation of f'' applies.

$h''_0 = h_1$

$h''_1 = h_0$

$h''_2 = h_1 + h_2 - h_0$

$q''_0 = h_1 + h_2 - (h_0 + h_2 - q_0) = h_1 - h_0 + q_0 \bmod (2h_1 + h_2 - h_0)$

$q''_1 = h_0 + h_1 - q_1 = h_0 + h_1 - q_1 \equiv -q_1 \bmod (h_0 + h_1)$

$q''_2 = 2(h_0 + h_2 - q_0) + (h_1 + h_2 - q_2) + h_1 - h_0 = h_2 + h_0 - 2q_0 - q_2 \bmod (h_1 + h_2)$

IV) $h_2 < q_0 < h_0 \Rightarrow h_2 < -q_0 < h_0 \bmod (h_0 + h_2)$

$h'_0 = h_1 + h_0 - h_2$

$h'_1 = h_2$

$h'_2 = h_1$

$q'_0 = [(h_1 + h_0 - h_2) + h_1] - (h_1 + h_0 - q_0) = h_1 - h_2 + q_0 \bmod (2h_1 + h_0 - h_2)$

$q'_1 = [(h_1 + h_0 - h_2) + h_2] - (2q_0 + q_1 + h_1 - h_2) = h_0 + h_2 - 2q_0 - q_1 \bmod (h_1 + h_0)$

$q'_2 = [h_1 + h_2] - q_2 \equiv -q_2 \bmod (h_1 + h_2)$

Since $h_2 < q_0 < h_0 \Rightarrow h_2 < -q_0 < h_0 \bmod (h_0 + h_2)$ Case IV in calculation of f'' applies.

$h''_0 = h_1 + h_0 - h_2$

$h''_1 = h_2$

$h''_2 = h_1$

$q''_0 = h_1 + h_0 - (h_0 + h_2 - q_0) = h_1 - h_2 + q_0 \bmod (2h_1 + h_0 - h_2)$

$q''_1 = 2(h_0 + h_2 - q_0) + (h_0 + h_1 - q_1) + h_1 - h_2 = h_0 + h_2 - 2q_0 - q_1 \bmod (h_0 + h_1)$

$q''_2 = h_1 + h_2 - q_2 \equiv -q_2 \bmod (h_1 + h_2)$

□

The application of σ is now easier. It follows that to calculate the action of σ it is sufficient to consider the images of the three members σf, $\sigma\psi_1 f$ and $\sigma\psi_1^2 f$ of an $\mathcal{H}$-orbit.

Definition 11. Let $\mathcal{S} = \{V, E\}$ be a graph which vertices are $\mathcal{H}$-orbits and the adjacency relation is given by:

$$[f]_{\mathcal{H}} \sim [g]_{\mathcal{H}} \Leftrightarrow \exists g' \in [g]_{\mathcal{H}} \wedge \exists f' \in [f]_{\mathcal{H}} : g' = \sigma f'.$$

Since $\sigma^2 = 1$, the graph $\mathcal{S}$ is undirected. Note that $\mathcal{S}$ contains loops.

The connectivity components of $\mathcal{S}$ are in a correspondence with the $\mathcal{G}$-orbits. Therefore we call the connectivity components of $\mathcal{S}$, $\mathcal{G}$-orbits too. By the definition, a $\mathcal{G}$-orbit is a class of equivalence. We can describe its minimal representatives.

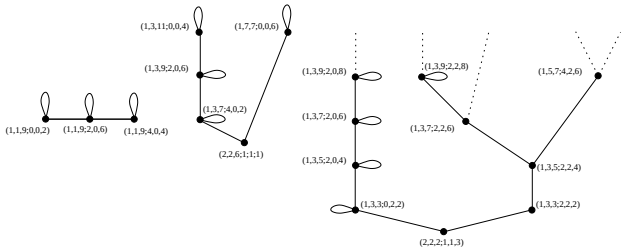

FIG. 3. Some components of connectivity of $\mathcal{S}$.

Agreement. Since the members of each $\mathcal{H}$-orbit have the same complexity, we define the complexity of a $\mathcal{H}$-orbit to be the complexity of its members. Since each $\mathcal{H}$-orbit corresponds to a vertex in $\mathcal{S}$, we can speak about complexity of a vertex. Moreover, we say that $\mathbf{u} \ll \mathbf{v}$ for $\mathbf{u} = [f]_{\mathcal{H}}$ and $\mathbf{v} = [g]_{\mathcal{H}}$, if $f \ll g$.

Lemma 6. The set of neighbours of a vertex $\mathbf{u} = [f]_{\mathcal{H}}$ in the graph $\mathcal{S}$ is $N = \{[\sigma f]_{\mathcal{H}}, [\sigma\psi_1 f]_{\mathcal{H}}, [\sigma\psi_1^2 f]_{\mathcal{H}}\}$. In particular, a vertex in $\mathcal{S}$ has at most 3 neighbours.

Proof. Let $A = \langle \psi_2, \psi_3 \rangle$. Each $\mathcal{H}$-orbit decomposes into the orbits induced by the action of A. Since σ commutes with the elements of A (see Lemma 5), it follows that for $g = \phi f, \phi \in A$ we have $\sigma g = \sigma\phi f = \phi\sigma f$, hence $[\sigma g]_{\mathcal{H}} = [\sigma f]_{\mathcal{H}}$. Hence, the set of neighbours of vertex $\mathbf{u}$ is N. $\square$

Theorem 2. Let $\mathbf{v}, \mathbf{u}, \mathbf{w}$ be three pairwise distinct vertices in $\mathcal{S}$. Let $\mathbf{u}$ and $\mathbf{w}$ be neighbours of $\mathbf{v}$. Then

$$z(\mathbf{u}) < z(\mathbf{v}) \Rightarrow z(\mathbf{w}) > z(\mathbf{v}), \tag{1}$$

$$z(\mathbf{u}) = z(\mathbf{v}) \Rightarrow z(\mathbf{w}) \geq z(\mathbf{v}). \tag{2}$$

Proof. Let us analyse the derivation of complexity $\delta(f)$ for a vertex $\mathbf{v}, f \in [f]_{\mathcal{H}} = \mathbf{v}$. Recall that $f = (h_0, h_1, h_2; q_0; q_1, q_2)$ is the minimal representative of $[f]_{\mathcal{H}}$. It follows that $h_0 \leq h_1 \leq h_2$. By Lemma 6 $\mathbf{u}, \mathbf{w} \in \{[\sigma f]_{\mathcal{H}}, [\sigma\psi_1 f]_{\mathcal{H}}, [\sigma\psi_1^2 f]_{\mathcal{H}}\}$. Hence we need to analyse the three derivations: $\delta(f)$, $\delta(\psi_1 f)$ and $\delta(\psi_1^2 f)$. In the following discussion we refer to Lemma 4.

I. For $\delta(f)$ we get:

(a) $q_0 = 0 \Rightarrow \delta(f) = 0$,
(b) $0 < q_0 < h_0, h_2 \Rightarrow \delta(f) > 0$, therefore $h_1 - q_0 \geq h_0 - q_0 > 0$,
(c) $h_0 < q_0 < h_2$, we consider subcases:
$h_0 < q_0 < h_2 \Rightarrow \delta(f) > 0$
or
$h_0 = h_1 < q_0 < h_2 \Rightarrow \delta(f) = 0$,
(d) $h_2 < h_0$ is in a contradiction with the minimality of f,
(e) $q_0 > h_0, h_2 \Rightarrow \delta(f) > 0$, therefore $q_0 + h_1 - h_0 - h_2 > h_1 - h_0 \geq 0$.

II. For $\delta(\psi_1 f)$ we get:

(a) $q_1 = 0 \Rightarrow \delta(\psi_1 f) = 0$,
(b) $0 < q_1 < h_1, h_0 \Rightarrow \delta(\psi_1 f) > 0$, therefore $h_2 - q_1 \geq h_1 - q_1 > 0$,
(c) $h_1 < h_0$ is in a contradiction with the minimality of f,
(d) $h_0 < q_1 < h_1 \Rightarrow \delta(\psi_1 f) > 0$,
(e) $q_1 > h_1, h_0 \Rightarrow \delta(\psi_1 f) > 0$, therefore $q_1 + h_2 - h_1 - h_0 > h_2 - h_0 \geq 0$.

III. For $\delta(\psi_1^2 f)$ we get:

(a) $q_2 = 0 \Rightarrow \delta(\psi_1^2 f) = 0$,
(b) $0 < q_2 < h_1, h_2$ we must consider following cases:
$q_2 < h_0 \leq h_1, h_2 \Rightarrow \delta(\psi_1^2 f) > 0$
or
$h_0 < q_2 < h_1 \Rightarrow \delta(\psi_1^2 f) < 0$,

(*c*) $h_2 < h_1$, is in a contradiction with the minimality of f,

(*d*) $h_1 < q_2 < h_2$ we consider subcases:

$h_0 < h_1 < q_2 < h_2 \Rightarrow \delta(\psi_1^2 f) < 0$

or

$h_0 = h_1 < q_2 < h_2 \Rightarrow \delta(\psi_1^2 f) = 0,$

(*e*) $q_2 > h_1, h_2$ we consider subcases:

$h_0 \leq h_1 \leq h_2 < q_2 < h_1 + h_2 - h_0 \Rightarrow \delta(\psi_1^2 f) < 0$

or

$h_0 \leq h_1 \leq h_2 < q_2 = h_1 + h_2 - h_0 \Rightarrow \delta(\psi_1^2 f) = 0$

or

$h_1, h_2, h_1 + h_2 - h_0 \leq q_2 \Rightarrow \delta(\psi_1^2 f) > 0.$

The previous discussion describes the derivation of each neighbour of the vertex $\mathbf{v}$. The subcases are pairwise eliminative and they cover all the possibilities. Since only one from the three possible neighbours of a given vertex $\mathbf{x} = [f]_{\mathcal{H}}$ can have a smaller complexity as $\mathbf{x}$, it follows that two edges incident to vertices $\mathbf{u}$, $\mathbf{w}$ with given complexity never enter the vertex $\mathbf{v}$ with higher complexity. This neighbour is $\mathbf{y} = [\sigma\psi_1^2 f]_{\mathcal{H}}$ and $z(\mathbf{y}) < z(\mathbf{x})$ in some subcases of Case **III**. The complexity of a neighbour of $\mathbf{v}$ can be smaller only in Case **III**.

Now assume $z(\mathbf{u}) < z(\mathbf{v})$. We have already observed $z(\mathbf{w}) \geq z(\mathbf{v})$. Assume $z(\mathbf{w}) = z(\mathbf{v})$. Analysing Cases **I**, **II** and **III** we see that $\mathbf{u}$ satisfies one of the conditions **III-b**, **III-d**, **III-e**. Moreover, $\mathbf{w}$ satisfies the condition **I-c**. Combining **I-c** with **III-b**, or **III-d**, or **III-e** we derive the following contradictions:

$$h_0 = h_1 < q_0 < h_2 \wedge h_0 = q_2 < h_1, h_2 \Rightarrow h_1 < q_0 \leq q_2 < h_1$$
$$h_0 = h_1 < q_0 < h_2 \wedge h_0 \leq h_1 \leq h_2 < q_2 = h_1 + h_2 - h_0 \Rightarrow h_2 < q_2 < h_2$$
$$h_0 = h_1 < q_0 < h_2 \wedge h_0 < h_1 < q_2 < h_2.$$

Hence $z(\mathbf{w}) > z(\mathbf{v})$ and we are done. □

Definition 12. A vertex $\mathbf{v}$ is in an *horisontal branch* $\mathcal{B} = \mathcal{B}(\mathbf{v})$ of a $\mathcal{G}$-orbit if the following holds:

$$\mathbf{v} \in \mathcal{B}(\mathbf{v}) \Leftrightarrow \forall \mathbf{u} \in N(\mathbf{v}) : z(\mathbf{u}) \geq z(\mathbf{v})$$

Lemma 7. In every $\mathcal{G}$-orbit there is precisely one horisontal branch $\mathcal{B}$ and $\mathcal{B}$ contains the minimum element $\mathbf{m}$ of the $\mathcal{G}$-orbit with respect to the order $\ll$. The complexity of all elements of $\mathcal{B}$ is equal to $z(\mathbf{m})$.

Proof. By the definition and by Theorem 2 a horisontal branch $\mathcal{B}$ consists of the 6-tuples with a fixed complexity. A minimal representative $\mathbf{m}$ of a $\mathcal{G}$-orbit containing $\mathcal{B}$ belongs to $\mathcal{B}$ as well. Moreover, Theorem 2 implies that the complexity of the 6-tuples in $\mathcal{B}(\mathbf{m})$ is equal to $z(\mathbf{m})$. □

Notice that a horisontal branch may contain only one vertex of $\mathcal{S}$.

Theorem 3. There exists a polynomial-time algorithm to decide whether two 6-tuples in $\mathcal{F}_2$ are $\mathcal{G}$-equivalent.

Proof. Let f and g be 6-tuples in $\mathcal{F}_2$ such that $z(f) \geq z(g)$. Using Theorem 2 we find $f_1 \in N(f)$ and $g_1 \in N(g)$ so that $z(f_1) \leq z(f)$ and $z(g_1) \leq z(g)$. Note that if the complexity of f_1 (g_1) is less than $z(f)$ ($z(g)$), f_1 (g_1) is uniquely determined. By proceeding at most $z(f) = n$ iterations we reach the horisontal branches of the respective $\mathcal{G}$-orbits containing f and g. If $z(f_n) \neq z(g_n)$, the 6-tuples are not $\mathcal{G}$-equivalent. The complexity of this procedure is $O(n)$. If $z(f_n) = z(g_n)$ the algorithm continues. We choose the minimal representatives of horisontal branches $\mathcal{B}_1(f_n)$ and $\mathcal{B}_2(g_n)$ containing f_n and g_n. If the minimal representatives are equal then $f \approx_{\mathcal{G}} g$. The complexity of this part of algorithm can be rougly estimated by $O(z_n(f)^3)$. By Lemma 7 the 6-tuples f and g are not $\mathcal{G}$-equivalent in the other case. □

List of $\mathcal{G}$-minimal representatives of 6-tuples

By using [2], [5] and previous results we have generated two catalogues of minimal representatives of $\mathcal{G}$-classes of 3-manifolds of genus two.

The first one is a reduction of the catalogue introduced in [2]. We have applied the $\mathcal{G}$-equivalence on it. The new version includes only minimal representatives of $\mathcal{G}$-orbits. It is created by a simple algorithm which computes $\sigma(f)$, $\sigma(\psi_1 f)$ and $\sigma(\psi_1^2 f)$ for every 6-tuple. Only 6-tuples satisfying $f \leq \sigma(f) \wedge f \leq \sigma(\psi_1 f) \wedge f \leq \sigma(\psi_1^2 f)$ (see Theorem 2) are listed in this catalogue. The new catalogue contains 309 of 6-tuples with complexity $z \leq 21$ instead of 695 6-tuples of the original.

The second catalogue is formed by computing of all admissible 6-tuples with complexity $z \leq 21$. After creating, a 6-tuple is processed by a similar way as described above and the minimal representatives of horisontal branches were listed. Since we do not use any further criteria to reduce it, this catalogue is more rich as the first one. It contains 433 of minimal 6-tuples. We have excluded the representatives of *traps* defined by a condition introduced in [5].

The catalogue up to complexity $z = 50$ was created and reduced in thirty minutes. Using the same program minimal prepresentatives of $\mathcal{G}$-orbits up to complexity $z = 100$ and higher can be generated in a real time. The program can be parallelised.

References

[1] Bandieri, P., Casali, M. R. and Gagliardi, C., *Representing Manifolds by Crystallisation Theory: Foundations, Improvements and Related Results*, Atti Sem. Mat. Fis. Univ. Modena **IL** (2001), 283–337.

[2] Casali, M. R., *A Catalogue of the Genus Two 3-Manifolds*, Atti Sem. Mat. Fis. Univ. Modena **XXXVII** (1989), 207–236.

[3] Casali, M. R. and Graselli, L., *2-Symmetric crystallisations and 2-fold branched coverings of* S^3, Discrete Mathematics **87** (1991), 9–22.

[4] Ferri, M. and Gagliardi, C., *Crystallisation moves*, Pacific Journal of Mathematics **100** (1982), no. 1, 85–103.

[5] Grasselli, L., Mulazzani, M. and Nedela, R., *2-Symmetric Transformations For Manifolds of Genus Two*, Journal of Combinatorial Theory **79** (2000), 105–130.

[6] Karabáš, J., *Minimálni reprezentanti $\mathcal{G}$-tried 3-variet rodu 2*, Univerzita Mateja Bela, Banská Bystrica, 2000 (Slovak); Diploma thesis.

Institute of Mathematics and Computer Science, Mathematical Institute of Slovak Acad. of Sciences, Severná 5, SK-97401 Banská Bystrica
E-mail: karabas@savbb.sk, nedela@savbb.sk

Appendix A: Reduced catalogue introuced in [2]

```
+++ z =  7 +++

( 1, 3, 3; 2, 2, 0)

+++ z =  9 +++

( 1, 3, 5; 2, 2, 0)
( 3, 3, 3; 2, 2, 2)

+++ z = 10 +++

+++ z = 11 +++

( 1, 3, 7; 2, 2, 0)
( 1, 5, 5; 2, 2, 0)
( 1, 5, 5; 2, 4, 0)
( 3, 3, 5; 0, 2, 4)
( 3, 3, 5; 2, 2, 4)

+++ z = 12 +++

( 4, 4, 4; 1, 1, 1)
( 4, 4, 4; 1, 1, 5)
( 4, 4, 4; 3, 3, 3)

+++ z = 13 +++

( 1, 3, 9; 2, 2, 0)
( 1, 3, 9; 4, 2, 0)
( 1, 5, 7; 2, 2, 0)
( 1, 5, 7; 2, 4, 0)
( 3, 3, 7; 2, 2, 2)
( 3, 3, 7; 2, 2, 6)
( 3, 5, 5; 2, 4, 0)
( 3, 5, 5; 4, 4, 2)

+++ z = 14 +++

( 4, 4, 6; 1, 1, 1)
( 4, 4, 6; 1, 1, 3)
( 4, 4, 6; 1, 1, 5)
( 4, 4, 6; 1, 1, 7)
( 4, 4, 6; 1, 5, 1)
( 4, 4, 6; 1, 5, 5)
( 4, 4, 6; 3, 1, 5)
( 4, 4, 6; 3, 3, 5)

+++ z = 15 +++

( 1, 3,11; 2, 2, 0)
( 1, 5, 9; 2, 2, 0)
( 1, 5, 9; 2, 4, 0)
( 1, 5, 9; 4, 2, 0)
( 1, 5, 9; 4, 4, 0)
( 1, 7, 7; 2, 2, 0)
( 1, 7, 7; 2, 6, 0)
( 3, 3, 9; 0, 2, 4)
( 3, 3, 9; 0, 2, 8)
( 3, 3, 9; 2, 0, 2)
( 3, 3, 9; 2, 0, 6)
( 3, 3, 9; 2, 2, 8)
( 3, 5, 7; 2, 4, 0)
( 3, 5, 7; 2, 4, 2)
( 3, 5, 7; 4, 4, 0)
( 3, 5, 7; 4, 4,10)
( 5, 5, 5; 0, 4, 4)
( 5, 5, 5; 2, 2, 2)
( 5, 5, 5; 4, 4, 4)

+++ z = 16 +++

( 4, 4, 8; 1, 1, 1)
( 4, 4, 8; 1, 1, 9)
( 4, 4, 8; 1, 5, 1)
( 4, 4, 8; 3, 1, 7)
( 4, 4, 8; 3, 3, 7)
( 4, 4, 8; 3, 7, 3)
( 4, 6, 6; 1, 1, 1)
( 4, 6, 6; 1, 1, 9)
( 4, 6, 6; 1, 7, 1)
( 4, 6, 6; 3, 5,11)
( 4, 6, 6; 5, 5, 3)

+++ z = 17 +++

( 1, 3,13; 2, 2, 0)
( 1, 3,13; 4, 2, 0)
( 1, 3,13; 6, 2, 0)
( 1, 5,11; 2, 2, 0)
( 1, 5,11; 2, 4, 0)
( 1, 7, 9; 2, 2, 0)
( 1, 7, 9; 2, 6, 0)
( 1, 7, 9; 4, 2, 0)
( 1, 7, 9; 4, 6, 0)
( 3, 3,11; 2, 2, 2)
( 3, 3,11; 2, 2, 4)
( 3, 3,11; 2, 2,10)
( 3, 5, 9; 2, 0, 2)
( 3, 5, 9; 2, 4, 0)
( 3, 5, 9; 4, 2, 0)
( 3, 5, 9; 4, 4,12)
( 3, 5, 9; 4, 6, 0)
( 3, 7, 7; 2, 2, 2)
( 3, 7, 7; 2, 6, 2)
( 3, 7, 7; 4, 4,12)
( 5, 5, 7; 0, 2, 6)
( 5, 5, 7; 0, 4, 6)
```

```
( 5, 5, 7; 0, 4,10)
( 5, 5, 7; 2, 0, 4)
( 5, 5, 7; 2, 0, 8)
( 5, 5, 7; 2, 4, 2)
( 5, 5, 7; 2, 4, 6)
( 5, 5, 7; 2, 6, 2)
( 5, 5, 7; 2, 6, 6)
( 5, 5, 7; 4, 0, 6)
( 5, 5, 7; 4, 2, 4)
( 5, 5, 7; 4, 4, 4)
( 5, 5, 7; 4, 4, 6)

+++ z = 18 +++

( 2, 8, 8; 3, 3, 1)
( 2, 8, 8; 3, 5, 1)
( 2, 8, 8; 5, 5, 1)
( 4, 4,10; 1, 1, 1)
( 4, 4,10; 1, 1, 3)
( 4, 4,10; 1, 1, 5)
( 4, 4,10; 1, 1, 7)
( 4, 4,10; 1, 1, 9)
( 4, 4,10; 1, 1,11)
( 4, 4,10; 1, 5, 1)
( 4, 4,10; 1, 5, 7)
( 4, 4,10; 1, 5, 9)
( 4, 4,10; 3, 1, 9)
( 4, 4,10; 3, 3, 3)
( 4, 4,10; 3, 3, 9)
( 4, 6, 8; 1, 1, 1)
( 4, 6, 8; 1, 1, 3)
( 4, 6, 8; 1, 1,11)
( 4, 6, 8; 1, 5, 3)
( 4, 6, 8; 1, 7, 1)
( 4, 6, 8; 3, 5,13)
( 4, 6, 8; 3, 9, 3)
( 4, 6, 8; 3, 9,13)
( 4, 6, 8; 5, 1, 3)
( 4, 6, 8; 5, 5, 3)
( 4, 6, 8; 5, 5,11)
( 6, 6, 6; 1, 1, 1)
( 6, 6, 6; 1, 1, 9)
( 6, 6, 6; 1, 3, 3)
( 6, 6, 6; 1, 3, 7)
( 6, 6, 6; 1, 5, 9)
( 6, 6, 6; 1, 7, 7)
( 6, 6, 6; 3, 3, 5)
( 6, 6, 6; 3, 5, 5)
( 6, 6, 6; 5, 5, 5)

+++ z = 19 +++

( 1, 3,15; 2, 2, 0)
( 1, 3,15; 6, 2, 0)
( 1, 5,13; 2, 2, 0)
( 1, 5,13; 2, 4, 0)
( 1, 5,13; 4, 2, 0)
( 1, 5,13; 4, 4, 0)
( 1, 5,13; 6, 2, 0)
( 1, 5,13; 6, 4, 0)
( 1, 7,11; 2, 2, 0)
( 1, 7,11; 2, 6, 0)
( 1, 9, 9; 2, 2, 0)
( 1, 9, 9; 2, 4, 0)
( 1, 9, 9; 2, 6, 0)
( 1, 9, 9; 2, 8, 0)
( 1, 9, 9; 4, 4, 0)
( 1, 9, 9; 4, 6, 0)
( 3, 3,13; 0, 2, 4)
( 3, 3,13; 0, 2,12)
( 3, 3,13; 2, 0, 2)
( 3, 3,13; 2, 0,10)
( 3, 3,13; 2, 2, 8)
( 3, 3,13; 2, 2,12)
( 3, 5,11; 2, 4, 0)
( 3, 5,11; 2, 4, 2)
( 3, 5,11; 4, 4, 0)
( 3, 5,11; 4, 4, 2)
( 3, 5,11; 4, 4,14)
( 3, 5,11; 6, 4, 0)
( 3, 7, 9; 2, 0, 2)
( 3, 7, 9; 2, 4, 2)
( 3, 7, 9; 4, 2, 0)
( 3, 7, 9; 4, 4, 0)
( 3, 7, 9; 4, 4,14)
( 3, 7, 9; 4, 6, 0)
( 3, 7, 9; 4, 8, 0)
( 3, 7, 9; 4, 8,14)
( 5, 5, 9; 0, 4, 4)
( 5, 5, 9; 0, 4, 6)
( 5, 5, 9; 0, 4,12)
( 5, 5, 9; 2, 0, 2)
( 5, 5, 9; 2, 0,10)
( 5, 5, 9; 2, 2, 2)
( 5, 5, 9; 4, 0, 4)
( 5, 5, 9; 4, 0, 8)
( 5, 5, 9; 4, 4, 8)
( 5, 5, 9; 4, 8, 4)
( 5, 7, 7; 0, 4,12)
( 5, 7, 7; 2, 2, 4)
( 5, 7, 7; 2, 6, 0)
( 5, 7, 7; 2, 8, 2)
( 5, 7, 7; 4, 4, 2)
( 5, 7, 7; 4, 6, 4)
( 5, 7, 7; 4, 6,12)
( 5, 7, 7; 6, 6, 4)

+++ z = 20 +++

( 4, 4,12; 1, 1, 1)
( 4, 4,12; 1, 1, 5)
( 4, 4,12; 1, 1, 9)
( 4, 4,12; 1, 1,13)
( 4, 4,12; 1, 5, 1)
( 4, 4,12; 1, 5, 5)
( 4, 4,12; 3, 1,11)
( 4, 4,12; 3, 3,11)
```

```
( 4, 6,10; 1, 1, 1)                          ( 3, 9, 9; 4, 4,16)
( 4, 6,10; 1, 1,13)    +++ z = 21 +++        ( 5, 5,11; 0, 4,10)
( 4, 6,10; 1, 7, 1)                          ( 5, 5,11; 0, 4,14)
( 4, 6,10; 3, 5, 3)    ( 1, 3,17; 2, 2, 0)   ( 5, 5,11; 2, 0, 8)
( 4, 6,10; 3, 5,15)    ( 1, 3,17; 4, 2, 0)   ( 5, 5,11; 2, 0,12)
( 4, 6,10; 3, 9,15)    ( 1, 3,17; 8, 2, 0)   ( 5, 5,11; 2, 2, 8)
( 4, 6,10; 5, 1, 1)    ( 1, 5,15; 2, 2, 0)   ( 5, 5,11; 2, 4, 2)
( 4, 6,10; 5, 3,15)    ( 1, 5,15; 2, 4, 0)   ( 5, 5,11; 2, 4,10)
( 4, 6,10; 5, 5, 1)    ( 1, 5,15; 6, 2, 0)   ( 5, 5,11; 2, 6, 2)
( 4, 6,10; 5, 5,13)    ( 1, 5,15; 6, 4, 0)   ( 5, 5,11; 2, 6,10)
( 4, 6,10; 5, 9, 3)    ( 1, 7,13; 2, 2, 0)   ( 5, 5,11; 4, 0,10)
( 4, 6,10; 7, 1, 1)    ( 1, 7,13; 2, 6, 0)   ( 5, 5,11; 4, 2, 4)
( 4, 6,10; 7, 3,15)    ( 1, 7,13; 4, 2, 0)   ( 5, 5,11; 4, 2, 8)
( 4, 8, 8; 1, 1, 1)    ( 1, 7,13; 4, 6, 0)   ( 5, 5,11; 4, 4, 8)
( 4, 8, 8; 1, 1,13)    ( 1, 7,13; 6, 2, 0)   ( 5, 5,11; 4, 4,10)
( 4, 8, 8; 1, 7, 3)    ( 1, 7,13; 6, 6, 0)   ( 5, 5,11; 4, 8, 4)
( 4, 8, 8; 1, 9, 1)    ( 1, 9,11; 2, 2, 0)   ( 5, 7, 9; 0, 2, 4)
( 4, 8, 8; 1, 9,13)    ( 1, 9,11; 2, 4, 0)   ( 5, 7, 9; 0, 2,12)
( 4, 8, 8; 3, 3, 1)    ( 1, 9,11; 2, 6, 0)   ( 5, 7, 9; 0, 4,14)
( 4, 8, 8; 5, 5,13)    ( 1, 9,11; 2, 8, 0)   ( 5, 7, 9; 0, 6, 4)
( 4, 8, 8; 5, 7, 3)    ( 3, 3,15; 2, 2, 2)   ( 5, 7, 9; 2, 0, 2)
( 6, 6, 8; 1, 1, 1)    ( 3, 3,15; 2, 2, 6)   ( 5, 7, 9; 2, 2, 4)
( 6, 6, 8; 1, 1, 3)    ( 3, 3,15; 2, 2,14)   ( 5, 7, 9; 2, 4, 4)
( 6, 6, 8; 1, 1, 5)    ( 3, 5,13; 2, 0, 2)   ( 5, 7, 9; 2, 6, 0)
( 6, 6, 8; 1, 1, 7)    ( 3, 5,13; 2, 4, 0)   ( 5, 7, 9; 4, 0, 2)
( 6, 6, 8; 1, 1, 9)    ( 3, 5,13; 4, 2, 0)   ( 5, 7, 9; 4, 0,14)
( 6, 6, 8; 1, 1,11)    ( 3, 5,13; 4, 4,16)   ( 5, 7, 9; 4, 2, 2)
( 6, 6, 8; 1, 5, 5)    ( 3, 5,13; 4, 6, 0)   ( 5, 7, 9; 4, 6, 0)
( 6, 6, 8; 1, 5,11)    ( 3, 5,13; 6, 4, 0)   ( 5, 7, 9; 4, 6, 2)
( 6, 6, 8; 1, 9, 1)    ( 3, 5,13; 8, 4, 2)   ( 5, 7, 9; 4, 6,14)
( 6, 6, 8; 1, 9, 7)    ( 3, 7,11; 2, 2, 2)   ( 5, 7, 9; 4,10, 4)
( 6, 6, 8; 3, 1, 9)    ( 3, 7,11; 2, 6, 2)   ( 5, 7, 9; 4,10,14)
( 6, 6, 8; 3, 7, 7)    ( 3, 7,11; 4, 2, 2)   ( 5, 7, 9; 6, 4, 4)
( 6, 6, 8; 3,11, 3)    ( 3, 7,11; 4, 4,16)   ( 5, 7, 9; 6, 6, 0)
( 6, 6, 8; 3,11, 9)    ( 3, 7,11; 4, 6, 2)   ( 5, 7, 9; 6, 6,12)
( 6, 6, 8; 5, 1, 7)    ( 3, 7,11; 4, 8,16)   ( 5, 7, 9; 6, 6,14)
( 6, 6, 8; 5, 3, 7)    ( 3, 9, 9; 0, 2, 2)   ( 5, 7, 9; 6, 8, 2)
( 6, 6, 8; 5, 5, 7)    ( 3, 9, 9; 2, 4, 0)   ( 5, 7, 9; 6, 8,12)
( 6, 6, 8; 5,11, 7)    ( 3, 9, 9; 2, 8, 0)   ( 5, 7, 9; 6,10,14)
```

```
( 7, 7, 7; 2, 2, 2)    ( 7, 7, 7; 2, 6,10)    ( 7, 7, 7; 4, 4, 8)
( 7, 7, 7; 2, 2, 6)    ( 7, 7, 7; 2, 8, 8)    ( 7, 7, 7; 6, 6, 6)
( 7, 7, 7; 2, 2,10)    ( 7, 7, 7; 4, 4, 4)
```

Appendix B: Our version of catalogue

```
+++ z =  7 +++         ( 3, 3, 7; 2, 2, 6)    ( 3, 3, 9; 2, 2, 8)
                       ( 3, 5, 5; 2, 4, 0)    ( 3, 5, 7; 2, 4, 0)
( 1, 3, 3; 2, 2, 0)    ( 3, 5, 5; 4, 4, 2)    ( 3, 5, 7; 2, 4, 2)
                                              ( 3, 5, 7; 4, 4, 0)
+++ z =  9 +++         +++ z = 14 +++         ( 3, 5, 7; 4, 4,10)
                                              ( 5, 5, 5; 0, 4, 4)
( 1, 1, 7; 2, 0, 2)    ( 2, 2,10; 3, 1, 3)    ( 5, 5, 5; 2, 2, 2)
( 1, 3, 5; 2, 2, 0)    ( 2, 6, 6; 3, 3, 1)    ( 5, 5, 5; 2, 2, 6)
( 3, 3, 3; 2, 2, 2)    ( 2, 6, 6; 3, 5, 1)    ( 5, 5, 5; 4, 4, 4)
                       ( 4, 4, 6; 1, 1, 1)
+++ z = 11 +++         ( 4, 4, 6; 1, 1, 3)    +++ z = 16 +++
                       ( 4, 4, 6; 1, 1, 5)
( 1, 3, 7; 2, 2, 0)    ( 4, 4, 6; 1, 1, 7)    ( 2, 4,10; 3, 3, 1)
( 1, 5, 5; 2, 2, 0)    ( 4, 4, 6; 1, 5, 1)    ( 2, 4,10; 5, 3,13)
( 1, 5, 5; 2, 4, 0)    ( 4, 4, 6; 1, 5, 5)    ( 2, 6, 8; 3, 3, 1)
( 3, 3, 5; 0, 2, 4)    ( 4, 4, 6; 3, 1, 5)    ( 2, 6, 8; 3, 5,13)
( 3, 3, 5; 2, 2, 4)    ( 4, 4, 6; 3, 3, 3)    ( 2, 6, 8; 5, 3, 1)
                       ( 4, 4, 6; 3, 3, 5)    ( 4, 4, 8; 1, 1, 1)
+++ z = 12 +++                                ( 4, 4, 8; 1, 1, 9)
                       +++ z = 15 +++         ( 4, 4, 8; 1, 3, 5)
( 2, 4, 6; 3, 3, 1)                           ( 4, 4, 8; 1, 5, 1)
( 4, 4, 4; 1, 1, 1)    ( 1, 3,11; 2, 2, 0)    ( 4, 4, 8; 3, 1, 7)
( 4, 4, 4; 1, 1, 5)    ( 1, 5, 9; 2, 2, 0)    ( 4, 4, 8; 3, 3, 7)
( 4, 4, 4; 3, 3, 3)    ( 1, 5, 9; 2, 4, 0)    ( 4, 4, 8; 3, 7, 3)
                       ( 1, 5, 9; 4, 2, 0)    ( 4, 6, 6; 1, 1, 1)
+++ z = 13 +++         ( 1, 5, 9; 4, 4, 0)    ( 4, 6, 6; 1, 1, 9)
                       ( 1, 7, 7; 2, 2, 0)    ( 4, 6, 6; 1, 7, 1)
( 1, 1,11; 2, 0, 2)    ( 1, 7, 7; 2, 6, 0)    ( 4, 6, 6; 3, 5, 3)
( 1, 3, 9; 2, 2, 0)    ( 3, 3, 9; 0, 2, 4)    ( 4, 6, 6; 3, 5,11)
( 1, 3, 9; 4, 2, 0)    ( 3, 3, 9; 0, 2, 8)    ( 4, 6, 6; 5, 5, 3)
( 1, 5, 7; 2, 2, 0)    ( 3, 3, 9; 2, 0, 2)
( 1, 5, 7; 2, 4, 0)    ( 3, 3, 9; 2, 0, 4)    +++ z = 17 +++
( 3, 3, 7; 2, 2, 2)    ( 3, 3, 9; 2, 0, 6)
```

```
( 1, 1,15; 2, 0, 2)    ( 5, 5, 7; 4, 0, 6)    ( 4, 6, 8; 5, 5, 3)
( 1, 3,13; 2, 2, 0)    ( 5, 5, 7; 4, 2, 4)    ( 4, 6, 8; 5, 5,11)
( 1, 3,13; 4, 2, 0)    ( 5, 5, 7; 4, 4, 4)    ( 4, 6, 8; 5, 7, 1)
( 1, 3,13; 6, 2, 0)    ( 5, 5, 7; 4, 4, 6)    ( 4, 6, 8; 5, 7,11)
( 1, 5,11; 2, 2, 0)                           ( 4, 6, 8; 5, 7,13)
( 1, 5,11; 2, 4, 0)    +++ z = 18 +++         ( 6, 6, 6; 1, 1, 1)
( 1, 7, 9; 2, 2, 0)                           ( 6, 6, 6; 1, 1, 9)
( 1, 7, 9; 2, 6, 0)    ( 2, 6,10; 3, 3, 1)    ( 6, 6, 6; 1, 3, 3)
( 1, 7, 9; 4, 2, 0)    ( 2, 6,10; 3, 5, 1)    ( 6, 6, 6; 1, 3, 7)
( 1, 7, 9; 4, 6, 0)    ( 2, 6,10; 3, 5,15)    ( 6, 6, 6; 1, 5, 9)
( 3, 3,11; 2, 2, 2)    ( 2, 6,10; 5, 3,15)    ( 6, 6, 6; 1, 7, 7)
( 3, 3,11; 2, 2, 4)    ( 2, 6,10; 5, 5,15)    ( 6, 6, 6; 3, 3, 5)
( 3, 3,11; 2, 2,10)    ( 2, 8, 8; 3, 3, 1)    ( 6, 6, 6; 3, 3, 7)
( 3, 5, 9; 2, 0, 2)    ( 2, 8, 8; 3, 5, 1)    ( 6, 6, 6; 3, 5, 5)
( 3, 5, 9; 2, 4, 0)    ( 2, 8, 8; 5, 5, 1)    ( 6, 6, 6; 5, 5, 5)
( 3, 5, 9; 4, 0, 2)    ( 4, 4,10; 1, 1, 1)
( 3, 5, 9; 4, 2, 0)    ( 4, 4,10; 1, 1, 3)    +++ z = 19 +++
( 3, 5, 9; 4, 4, 2)    ( 4, 4,10; 1, 1, 5)
( 3, 5, 9; 4, 4,12)    ( 4, 4,10; 1, 1, 7)    ( 1, 3,15; 2, 2, 0)
( 3, 5, 9; 4, 6, 0)    ( 4, 4,10; 1, 1, 9)    ( 1, 3,15; 6, 2, 0)
( 3, 7, 7; 2, 2, 2)    ( 4, 4,10; 1, 1,11)    ( 1, 5,13; 2, 2, 0)
( 3, 7, 7; 2, 6, 2)    ( 4, 4,10; 1, 5, 1)    ( 1, 5,13; 2, 4, 0)
( 3, 7, 7; 4, 4, 0)    ( 4, 4,10; 1, 5, 7)    ( 1, 5,13; 4, 2, 0)
( 3, 7, 7; 4, 4,12)    ( 4, 4,10; 1, 5, 9)    ( 1, 5,13; 4, 4, 0)
( 3, 7, 7; 4, 6, 0)    ( 4, 4,10; 3, 1, 9)    ( 1, 5,13; 6, 2, 0)
( 5, 5, 7; 0, 2, 4)    ( 4, 4,10; 3, 3, 3)    ( 1, 5,13; 6, 4, 0)
( 5, 5, 7; 0, 2, 6)    ( 4, 4,10; 3, 3, 7)    ( 1, 7,11; 2, 2, 0)
( 5, 5, 7; 0, 2, 8)    ( 4, 4,10; 3, 3, 9)    ( 1, 7,11; 2, 6, 0)
( 5, 5, 7; 0, 4, 6)    ( 4, 4,10; 3, 7, 7)    ( 1, 9, 9; 2, 2, 0)
( 5, 5, 7; 0, 4,10)    ( 4, 6, 8; 1, 1, 1)    ( 1, 9, 9; 2, 4, 0)
( 5, 5, 7; 2, 0, 2)    ( 4, 6, 8; 1, 1, 3)    ( 1, 9, 9; 2, 6, 0)
( 5, 5, 7; 2, 0, 4)    ( 4, 6, 8; 1, 1,11)    ( 1, 9, 9; 2, 8, 0)
( 5, 5, 7; 2, 0, 6)    ( 4, 6, 8; 1, 5, 3)    ( 1, 9, 9; 4, 4, 0)
( 5, 5, 7; 2, 0, 8)    ( 4, 6, 8; 1, 7, 1)    ( 1, 9, 9; 4, 6, 0)
( 5, 5, 7; 2, 2, 8)    ( 4, 6, 8; 3, 5,13)    ( 3, 3,13; 0, 2, 4)
( 5, 5, 7; 2, 4, 2)    ( 4, 6, 8; 3, 9, 3)    ( 3, 3,13; 0, 2,12)
( 5, 5, 7; 2, 4, 6)    ( 4, 6, 8; 3, 9,13)    ( 3, 3,13; 2, 0, 2)
( 5, 5, 7; 2, 6, 2)    ( 4, 6, 8; 5, 1, 3)    ( 3, 3,13; 2, 0,10)
( 5, 5, 7; 2, 6, 6)    ( 4, 6, 8; 5, 3, 1)    ( 3, 3,13; 2, 2, 8)
```

```
( 3, 3,13; 2, 2,12)    ( 5, 7, 7; 2, 8, 0)    ( 4, 6,10; 3, 3, 3)
( 3, 3,13; 4, 2, 4)    ( 5, 7, 7; 2, 8, 2)    ( 4, 6,10; 3, 5, 3)
( 3, 5,11; 2, 4, 0)    ( 5, 7, 7; 4, 4, 2)    ( 4, 6,10; 3, 5,15)
( 3, 5,11; 2, 4, 2)    ( 5, 7, 7; 4, 6, 4)    ( 4, 6,10; 3, 9,15)
( 3, 5,11; 4, 4, 0)    ( 5, 7, 7; 4, 6,12)    ( 4, 6,10; 5, 1, 1)
( 3, 5,11; 4, 4, 2)    ( 5, 7, 7; 6, 6, 2)    ( 4, 6,10; 5, 3,15)
( 3, 5,11; 4, 4,14)    ( 5, 7, 7; 6, 6, 4)    ( 4, 6,10; 5, 5, 1)
( 3, 5,11; 6, 4, 0)                           ( 4, 6,10; 5, 5,13)
( 3, 7, 9; 2, 0, 2)    +++ z = 20 +++         ( 4, 6,10; 5, 7,13)
( 3, 7, 9; 2, 4, 2)                           ( 4, 6,10; 5, 9, 3)
( 3, 7, 9; 4, 0, 2)    ( 2, 2,16; 3, 1, 3)    ( 4, 6,10; 7, 1, 1)
( 3, 7, 9; 4, 2, 0)    ( 2, 2,16; 3, 1, 9)    ( 4, 6,10; 7, 3,15)
( 3, 7, 9; 4, 4, 0)    ( 2, 4,14; 3, 3, 1)    ( 4, 6,10; 7, 5, 3)
( 3, 7, 9; 4, 4, 2)    ( 2, 4,14; 5, 3,17)    ( 4, 8, 8; 1, 1, 1)
( 3, 7, 9; 4, 4,14)    ( 2, 6,12; 3, 3, 1)    ( 4, 8, 8; 1, 1,13)
( 3, 7, 9; 4, 6, 0)    ( 2, 6,12; 3, 5,17)    ( 4, 8, 8; 1, 7, 3)
( 3, 7, 9; 4, 6, 2)    ( 2, 6,12; 5, 3, 1)    ( 4, 8, 8; 1, 9, 1)
( 3, 7, 9; 4, 8, 0)    ( 2, 6,12; 5, 5,17)    ( 4, 8, 8; 1, 9,13)
( 3, 7, 9; 4, 8,14)    ( 2, 6,12; 7, 3, 1)    ( 4, 8, 8; 3, 3, 1)
( 5, 5, 9; 0, 4, 4)    ( 2, 8,10; 3, 3, 1)    ( 4, 8, 8; 3, 5, 1)
( 5, 5, 9; 0, 4, 6)    ( 2, 8,10; 3, 5, 1)    ( 4, 8, 8; 3, 7,15)
( 5, 5, 9; 0, 4,12)    ( 2, 8,10; 3, 7, 1)    ( 4, 8, 8; 5, 5, 1)
( 5, 5, 9; 2, 0, 2)    ( 2, 8,10; 5, 3,17)    ( 4, 8, 8; 5, 5,13)
( 5, 5, 9; 2, 0,10)    ( 2, 8,10; 5, 5,17)    ( 4, 8, 8; 5, 7, 1)
( 5, 5, 9; 2, 2, 2)    ( 2, 8,10; 5, 7,17)    ( 4, 8, 8; 5, 7, 3)
( 5, 5, 9; 2, 2,10)    ( 4, 4,12; 1, 1, 1)    ( 6, 6, 8; 1, 1, 1)
( 5, 5, 9; 2, 6, 2)    ( 4, 4,12; 1, 1, 5)    ( 6, 6, 8; 1, 1, 3)
( 5, 5, 9; 4, 0, 4)    ( 4, 4,12; 1, 1, 9)    ( 6, 6, 8; 1, 1, 5)
( 5, 5, 9; 4, 0, 8)    ( 4, 4,12; 1, 1,13)    ( 6, 6, 8; 1, 1, 7)
( 5, 5, 9; 4, 4, 4)    ( 4, 4,12; 1, 5, 1)    ( 6, 6, 8; 1, 1, 9)
( 5, 5, 9; 4, 4, 8)    ( 4, 4,12; 1, 5, 5)    ( 6, 6, 8; 1, 1,11)
( 5, 5, 9; 4, 8, 4)    ( 4, 4,12; 3, 1, 5)    ( 6, 6, 8; 1, 3, 7)
( 5, 7, 7; 0, 2, 2)    ( 4, 4,12; 3, 1,11)    ( 6, 6, 8; 1, 3, 9)
( 5, 7, 7; 0, 2,10)    ( 4, 4,12; 3, 3, 3)    ( 6, 6, 8; 1, 5, 5)
( 5, 7, 7; 0, 4,12)    ( 4, 4,12; 3, 3,11)    ( 6, 6, 8; 1, 5,11)
( 5, 7, 7; 2, 2, 4)    ( 4, 4,12; 3, 7, 3)    ( 6, 6, 8; 1, 7, 7)
( 5, 7, 7; 2, 2,10)    ( 4, 6,10; 1, 1, 1)    ( 6, 6, 8; 1, 9, 1)
( 5, 7, 7; 2, 6, 0)    ( 4, 6,10; 1, 1,13)    ( 6, 6, 8; 1, 9, 7)
( 5, 7, 7; 2, 6, 4)    ( 4, 6,10; 1, 7, 1)    ( 6, 6, 8; 1, 9, 9)
```

```
( 6, 6, 8; 3, 1, 9)
( 6, 6, 8; 3, 3, 9)
( 6, 6, 8; 3, 5, 7)
( 6, 6, 8; 3, 7, 3)
( 6, 6, 8; 3, 7, 7)
( 6, 6, 8; 3,11, 3)
( 6, 6, 8; 3,11, 5)
( 6, 6, 8; 3,11, 7)
( 6, 6, 8; 3,11, 9)
( 6, 6, 8; 5, 1, 7)
( 6, 6, 8; 5, 3, 5)
( 6, 6, 8; 5, 3, 7)
( 6, 6, 8; 5, 5, 7)
( 6, 6, 8; 5,11, 7)

+++ z = 21 +++

( 1, 1,19; 2, 0, 2)
( 1, 1,19; 2, 0, 6)
( 1, 3,17; 2, 2, 0)
( 1, 3,17; 4, 2, 0)
( 1, 3,17; 8, 2, 0)
( 1, 5,15; 2, 2, 0)
( 1, 5,15; 2, 4, 0)
( 1, 5,15; 6, 2, 0)
( 1, 5,15; 6, 4, 0)
( 1, 7,13; 2, 2, 0)
( 1, 7,13; 2, 6, 0)
( 1, 7,13; 4, 2, 0)
( 1, 7,13; 4, 6, 0)
( 1, 7,13; 6, 2, 0)
( 1, 7,13; 6, 6, 0)
( 1, 9,11; 2, 2, 0)
( 1, 9,11; 2, 4, 0)
( 1, 9,11; 2, 6, 0)
( 1, 9,11; 2, 8, 0)
( 3, 3,15; 2, 0, 4)
( 3, 3,15; 2, 2, 2)
( 3, 3,15; 2, 2, 6)
( 3, 3,15; 2, 2,14)
( 3, 5,13; 2, 0, 2)
( 3, 5,13; 2, 4, 0)
( 3, 5,13; 4, 2, 0)
( 3, 5,13; 4, 4,16)
( 3, 5,13; 4, 6, 0)
( 3, 5,13; 6, 4, 0)
( 3, 5,13; 8, 4, 2)
( 3, 7,11; 2, 2, 2)
( 3, 7,11; 2, 6, 2)
( 3, 7,11; 4, 2, 2)
( 3, 7,11; 4, 4, 0)
( 3, 7,11; 4, 4,16)
( 3, 7,11; 4, 6, 0)
( 3, 7,11; 4, 6, 2)
( 3, 7,11; 4, 8,16)
( 3, 7,11; 6, 2, 2)
( 3, 7,11; 6, 4, 0)
( 3, 7,11; 6, 6, 0)
( 3, 9, 9; 0, 2, 2)
( 3, 9, 9; 0, 4, 2)
( 3, 9, 9; 2, 4, 0)
( 3, 9, 9; 2, 8, 0)
( 3, 9, 9; 4, 4, 2)
( 3, 9, 9; 4, 4,16)
( 3, 9, 9; 4, 6, 2)
( 3, 9, 9; 4, 8, 2)
( 3, 9, 9; 6, 6, 2)
( 5, 5,11; 0, 2, 4)
( 5, 5,11; 0, 2,12)
( 5, 5,11; 0, 4,10)
( 5, 5,11; 0, 4,14)
( 5, 5,11; 2, 0, 2)
( 5, 5,11; 2, 0, 6)
( 5, 5,11; 2, 0, 8)
( 5, 5,11; 2, 0,10)
( 5, 5,11; 2, 0,12)
( 5, 5,11; 2, 2, 8)
( 5, 5,11; 2, 2,12)
( 5, 5,11; 2, 4, 2)
( 5, 5,11; 2, 4, 6)
( 5, 5,11; 2, 4,10)
( 5, 5,11; 2, 6, 2)
( 5, 5,11; 2, 6, 8)
( 5, 5,11; 2, 6,10)
( 5, 5,11; 4, 0,10)
( 5, 5,11; 4, 2, 4)
( 5, 5,11; 4, 2, 8)
( 5, 5,11; 4, 4, 8)
( 5, 5,11; 4, 4,10)
( 5, 5,11; 4, 8, 4)
( 5, 5,11; 4, 8, 8)
( 5, 7, 9; 0, 2, 2)
( 5, 7, 9; 0, 2, 4)
( 5, 7, 9; 0, 2,12)
( 5, 7, 9; 0, 4,14)
( 5, 7, 9; 0, 6, 4)
( 5, 7, 9; 2, 0, 2)
( 5, 7, 9; 2, 0,12)
( 5, 7, 9; 2, 2, 4)
( 5, 7, 9; 2, 2,12)
( 5, 7, 9; 2, 4, 4)
( 5, 7, 9; 2, 6, 0)
( 5, 7, 9; 2, 8, 0)
( 5, 7, 9; 2, 8, 2)
( 5, 7, 9; 4, 0, 2)
( 5, 7, 9; 4, 0,14)
( 5, 7, 9; 4, 2, 2)
( 5, 7, 9; 4, 6, 0)
( 5, 7, 9; 4, 6, 2)
( 5, 7, 9; 4, 6, 4)
( 5, 7, 9; 4, 6,14)
( 5, 7, 9; 4,10, 4)
( 5, 7, 9; 4,10,14)
( 5, 7, 9; 6, 0, 4)
( 5, 7, 9; 6, 4, 0)
( 5, 7, 9; 6, 4, 4)
( 5, 7, 9; 6, 6, 0)
```

```
( 5, 7, 9; 6, 6,12)   ( 7, 7, 7; 0, 2,10)   ( 7, 7, 7; 2, 2,10)
( 5, 7, 9; 6, 6,14)   ( 7, 7, 7; 0, 4, 4)   ( 7, 7, 7; 2, 6,10)
( 5, 7, 9; 6, 8, 2)   ( 7, 7, 7; 0, 6, 6)   ( 7, 7, 7; 2, 8, 8)
( 5, 7, 9; 6, 8,12)   ( 7, 7, 7; 2, 2, 2)   ( 7, 7, 7; 4, 4, 4)
( 5, 7, 9; 6,10,12)   ( 7, 7, 7; 2, 2, 4)   ( 7, 7, 7; 4, 4, 8)
( 5, 7, 9; 6,10,14)   ( 7, 7, 7; 2, 2, 6)   ( 7, 7, 7; 4, 6, 6)
( 7, 7, 7; 0, 2, 2)   ( 7, 7, 7; 2, 2, 8)   ( 7, 7, 7; 6, 6, 6)
```

Acta Univ. M. Belii
Math no. 10(2003), pp. 45–51

PROFESSOR ANTON DEKRÉT'S SEVENTIETH BIRTHDAY

Rudolf Zimka

In September 2002 Professor RNDr. Anton Dekrét, DrSc., a distinquished Slovak mathematician celebrated with his friends his seventieth birthady.

Professor Anton Dekrét was born in a small settlement Dobroč, close to Čierny Balog. After finishing the basic school in Dobroč he was educated at Brezno (1943 - 1947) and Banská Bystrica Gymnaziums (1947 - 1951). He completed his further education at the Comenius University in Bratislava at the Faculty of Natural Sciences, specializing in the field of mathematics – projective geometry in 1955. After graduation his first work was as a teacher of mathematics at a secondary school (6 years) and then as a university teacher of mathematics at the Pedagogical College in Martin (1 year). From 1962 to 1973 he worked at the College of Transport and Communications in Žilina. In 1974 he moved to Zvolen where an Institute of Applied Mathematics was to be found at the College of Forest and Wood. Since 1998 Professor Dekrét has been working at the Department of Informatics of the Faculty of Natural Sciences at the Matthiae Belii University in Banská Bystrica.

Professor Dekrét has been from his graduation up to the present day constantly engaged in the teaching profession. His lectures and textbooks are known for accuracy, comprehension and clarity; therefore they are very popular with students. Professor Dekrét has been always trying to present mathematical knowledge to his students in a way for them to utilize it in a practical manner. Success in this approach can be clearly noted by the hundreds of his students who have taken up important posts in Slovak society.

The beginning of his scientific work can be traced back to his time at the College of Transport and Communications in Žilina. There he took an active part at the seminar on differential geometry at the Faculty of Technology, closely cooperating with Professor Hejný. He also formed a working relationship with the scientists at the Brno's school of differential geometry, represented especially by Professors Kolář and Kowalski. Having moved to Zvolen Professor Dekrét led seminars on differential geometry at the Department of Mathematics and Projective Geometry at the Faculty of Wood Technology. Also he intensified the collaboration with his colleagues in Brno especially with Professor Kolář and started scientific cooperation with the profile departments at the Faculty of Wood Technology. By his hard working, strong-willed-mind and enthusiasm he gained much respect, winning over

2000 Mathematics Subject Classification. 01A70.
Key words and phrases. Anton Dekrét, jubilee.
Received 1. 11. 2002; Accepted 16. 4. 2003

many of his colleagues into scientific work. The scientific activity at the Department of Mathematics and Projective Geometry and the cooperation with the profile departments achieved, under the influence of the activities of Professor Dekrét, a higher dimension.

The main fields of the scientific interest of Professor Dekrét are differential geometry of fibre spaces, applications of mathematics at modelling processes connected with the diffusion of water in wood. During last few years he has also devoted himself to the questions of utilizing differential geometry in robotics and control theory. The results he gained in differential geometry can be divided into following three groups:

(1) Contributions to the theory of prolonged functors (properties of the canonical form and properties of the main connections on non-holonomic and semi-holonomic prolongations of the main fibre spaces);
(2) Contributions to the theory of generalized connections on fibre spaces (coherences of various tensor fields with connections, covariant derivative of geometric objects) and especially on iterated tangent fibrations;
(3) The list of connections naturally deduced from some geometrical objects (connections constructed from (1,1) - forms on a tangent fibration, from vector fields on a tangent fibration and on a general fibre space, from differential equations of higher order on differential manifolds and their relations with Lagrangian and Hamiltonian formalism).

The results achieved in wood technology are the product of his cooperation with wood experts. The results deal particularly with modelling tensorial character of water motion and tension arising in wood during its drying process and methods of the calculation of diffusion coefficient. Professor Dekrét has published so far 44 scientific papers on differential geometry and 42 scientific papers on the application of mathematics in wood technology an in robotics.

Professor Dekrét does not limit his work only to the university. His activities in the Union of Slovak Mathematicians and Physicists, of which he has been a member since his university graduation, are also very significant. He has carried out various roles over the years including chairmanship of its branch in Zvolen. It is greatly to his credit that this branch was very active and successful. Professor Dekrét took part in the preparation, organization and leading many summer and winter schools and seminars on differential geometry along with many other meetings of mathematicians, especially from Žilina, Zvolen and Banská Bystrica. Partakers of these meetings remember them with pleasure as Professor Dekrét has always the skill to join successfully mathematical programs with sport and social activities. Participants could feel fully the truth of the proverb: "In a healthy body there is sound mind". Professor Dekrét is an excellent champion of this proverb. Not only in mathematics but also in favourite sport activities like mountain and cross-country-ski tours considerably prevail his excellent mind and physical condition. The Union of Slovak Mathematicians and Physicists has appreciated the meritorious work of Professor Dekrét by electing him an honorary member of the Union of Slovak Mathematicians and Physicists.

Tonko, allow me on behalf of all your friends, colleagues and students to wish you in the following years good health, a peaceful mind, numerous further valuable results in mathematics, complimented by many nice walks and tours in the beautiful Slovak nature.

The list of the publications of Professor RNDr. Anton Dekrét, DrSc.

Scientific papers on differential geometry

[1] Dekrét A.: *Poznámka o T-páre komplexov*, Matematický časopis, 1697, 48-54.

[2] Dekrét A.: *Poznámka k hyperbolickému páru komplexov priamok v P_3*, Acta Facultatis RN UC, Mathematica XV, 1967, 45-52.

[3] Dekrét A.: *Poznámka k transverzálnemu zobrazeniu medzi plochami v P_3*, Sborník prací VšD a Výskumného ústavu dopravního, Žilina, 1968, č. 15, 57-67.

[4] Dekrét A.: *K teórii T-páru komplexov priamok v P_3*, Matematický časopis 19, 1969, No 3, Žilina, 192-201.

[5] Dekrét A.: *Poznámka k distribúcii variet v P_n*, Sborník prací VšD a Výskumného ústavu dopravního, 1970, 34, 9-20.

[6] Dekrét A.: *Note on the theory of T-pair of manifolds in the projective space P_n*, Matematický časopis 20, 1970, No 1, 38-48.

[7] Dekrét A.: *On a pair of manifolds with connection*, Matematický časopis 22, 1972, No 2, 97-107.

[8] Dekrét A.: *On canonical forms on non-holonomic and semi-holonomic prolongations on principal fibre bundles*, Czechoslovak Math. Journal 22, 1972, Praha, 653-662.

[9] Dekrét A.: *On a pair of connections on a principal fibre bundle*, Matematický časopis 24, 1974, No 1, 59-68.

[10] Dekrét A.: *The coordinate form of the composition of non-holonomic jets*, Práce a štúdie VšD v Žiline, séria matematicko-fyzikálna, č. 1, 1974, 7-11.

[11] Dekrét A.: *On the connections on the prolongations of principal fibre bundles*, Matematický časopis 24, 1975, No 3, 293-302.

[12] Dekrét A.: *On horizontal structure on differentiable manifold*, Mathematica Slovaca 27, 1977, No 1, 25-32.

[13] Dekrét A.: *On general connections and covariant derivative on fibre bundles*, Proceeding of Czechoslov. - GDR - Polish scientific School in Diff. Geometry, Sci. Comm., Pol. Ac. of Sc., 1977, 40-55.

[14] Dekrét A.: *Horizontal structures on fibre manifolds*, Mathematica Slovaca 27, 1977, No 3, 257-265.

[15] Dekrét A.: *O konexiách na fibrovaných priestoroch s grupoidom operátorov*, Zborník vedeckých prác DF VSLD Zvolen, Alfa 1979, 347-352.

[16] Dekrét A.: *On bilinear structures on differentiable manifolds*, Archivum math., Fac. sci. nat. UJEP Brunensis, XV, 1979, 193-202.

[17] Dekrét A.: *On forms and connections on fibre bundles*, Časopis pro pěstování matematiky, roč. 105, 1980, Praha, 73-80.

[18] Dekrét A.: *On higher order point singularities of some geometric object fields*, Mathematica Slovaca, 300, 1980, No 2, 133-138.

[19] Dekrét A.: *On quasi-Riemanian fiber manifolds*, Czechoslovak. Math. Journal, 31, 1981, 229-240.

[20] Dekrét A.: *Prolongation of natural bundles*, Mathematica Slovaca 35, 1985, No 3, 243-249.

[21] Dekrét A.: *On quasi-jets*, Časopis pro pěstování matematiky, 111, 1986, Praha, 345-352.

[22] Dekrét A.: *On connections on the second iterated tangent bundle*, Archivum Mathematicum, Vol. 23, No 4, 1987, 215-230.

[23] Dekrét A.: *On quasi-jets and connections*, Proceedings of the Conference on Differential Geometry and its Applications, Brno, 1987, pp. 91-100.

[24] Dekrét A.: *Mechanical structures and connections*, Proceedings of the Conference on Differential Geometry and Applications, Dubrovnik, 1988, Novi Sad 1989, pp. 121-132.

[25] Dekrét A., Vosmanská G.: *Natural transformations of 2-quasi-jets*, Archivum Mathematicum, Tom 276, 1991, pp. 155-160.

[26] Dekrét A.: *Vector fields and connections on fibred manifolds*, Supplemento di Rendiconti del Circulo Matematico di Palermo, Serie 2-num. 22, 1989, 25-34.

[27] Dekrét A.: *Vector fields and connections on TM*, Časopis pro pěstování matematiky, 115, 1990, No 4, 360-367.

[28] Dekrét A.: *Ordinary differential equations on manifolds and connections*, Proceedings of Conferencie on Differential Geometry and its Applications, Brno, 1989, Word scientific, Singapore, 1990, pp. 27-32.

[29] Dekrét A.: *Connections deduced from mechanical system of second order*, Proceedings of Colloquium on Differential Geometry in Eger, 1989, Math. Soc. Bolyai, Budapest, 1992, pp. 201-212.

[30] Dekrét A.: *A property of connections of mechanical systems of higher order*, Mathematica Slovaca, 43, 1993, No 1, pp. 77-87.

[31] Dekrét A.: *Natural connections of 1-forms on tangent bundles*, Proceedings of Conference on Differential Geometry and its Applications, Silesian University, Opava, 1993, pp. 265-271.

[32] Dekrét A.: *(1,1)-forms and Connections on a vector bundle TM*, Acta Univ. M. Belii, Math., 1994, pp. 3-8.

[33] Dekrét A.: *On almost complex structures on fibre bundles*, Acta Univesity M. Belii, Math. No 3, 1995, pp. 3-8.

[34] Dekrét A.: *Almost complex structures and connections on TM*, Proceedings of Conference on Differential Geometry and its Applications, Brno, 1995, Masaryk. university, 1996, pp. 133-140.

[35] Dekrét A.: *On non vertical linear (1,1)-tensor fields and connections on tangent bundles*, Acta University M. Belii Banská Bystrica, Math. No 4, 1996, pp. 17-23.

[36] Dekrét A.: *On skew 2-projectable almost complex structures on TM*, Archivum Mathematicum, Tom 34, 1998, pp. 285-293.

[37] Dekrét A.: *Connections induced by (1,1)-tensor fields on cotangent bundles*, Math. Bohemica, 123, 1998, No 3, 317-331.

[38] Dekrét A.: *Lagrangeans on a manifold with a (1,1)-tensor field*, Acta University M. Belii Banská Bystrica, Math. No 6, 1998, pp. 3-13.

[39] Dekrét A.: *On (1,1)-tensor fields on symplectic manifolds*, Archivum Mathematicum, Tom 35, 1999, No 4, 329-336.

[40] Dekrét A., Bakša J.: *Poznámka ku grafike plôch v E_3*, Zborník príspevkov z konferencie s medzinárodnou účasťou „Informatika a informačné technológie", Univer. M. Belii, 1999, pp. 187-191.

[41] Dekrét A., Bakša J.: *Tangential prolongation of surfaces in E_3-clasification of parametric nets*, Acta UMB, Math. No 8, 2000, pp. 3-9.

[42] Dekrét A.: *On applications of the tangential prolongation of vector fields*, Acta oeconomica No 6, Ekonomická fakulta UMB, Banská Bystrica, 2000, pp. 3-9.

[43] Dekrét A., Bakša J.: *Applications of line objects in robotics*, Acta Univ. M. Belii, **9**(2002), pp. 29–42.

[44] Dektrét A.: *On vertical accessibility of control systems on fibre manifolds*, Studies of the University of Zilina, Mathematical Series, Vol. 15 (2002), 1-10.

Scientific papers on the application of mathematics in wood technology

[1] Dekrét A.: *Poznámka k metodike výpočtu tepelných strát v potrubí*, Zborník vedeckých prác DF VšLD Zvolen, ALFA 1977, ss. 115-123.

[2] Dekrét A., Reginač L.: *Anizotropia dreva vzhľadom na jeho dendrometrickú štruktúru*, Zborník vedeckých prác DF VšLD Zvolen, ALFA 1979, ss. 79-96.

[3] Dekrét A., Husárik F., Reginář L.: *Poznámka k výpočtu kubatúry výrezov*, Zborník vedeckých prác DF VšLD Zvolen, 1979, ss. 113-125.

[4] Dekrét A., Babiak M., Reginář L.: *O aproximácii cylindrickej ortotropie priamočiarou ortotropiou*, Zborník z vedeckej konferencie „25 rokov VšLD vo Zvolene", VšLD Zvolen, 1977, ss. 55-65.

[5] Dekrét A., Kurjatko S.: *Tenzoriálnosť difúzie vody v prírodnom dreve*, Zborník vedeckých prác DF VšLD vo Zvolene, ALFA 1980, ss. 171-176.

[6] Kurjatko S., Dekrét A.: *Preverenie tenzorového charakteru difúzie vlhkosti v dreve*, Zborník z vedeckej konferencie „3. Meždunarodnyj simpozium - Fundamentaľnye isledovanija drevesiny", Dresden, 1980, 126-131.

[7] Požgaj A., Dekrét A.: *Model deformácie dreva*, Zborník z vedeckej konferecnie „3. Meždunarodnyj simpozium - Fundamentatľnye isledovanija drevesiny", Dresden, 1980, ss. 203-212.

[8] Dekrét A., Kurjatko S.: *Overenie tenzorového charakteru difúzie vodných pár v smrekovom dreve*, Drevársky výskum, 1981, XXVI, Zväzok 2, ss. 57-63.

[9] Dekrét A., Kurjatko S.: *Povrchové podmienky pri použití 2. Fickovho zákona*, Zborník z medzinárodnej vedeckej konferencie VšLD, Zvolen, 1982, ss. 35-42.

[10] Trebula P., Dekrét A.: *Vákuové sušenie agátového dreva*, Zborník z medzinárodnej vedeckej konferencie VšLD, Zvolen, 1982, ss. 266-275.

[11] Trebula P., Dekrét A.: *Vakuumtrocknung von Robiniennholzes*, Holzindustrie, 4, 1983, 115-116.

[12] Trebula P., Dekrét A.: *Vákuové sušenie hrabového dreva s konvekčným ohrevom*, Drevársky výskum, roč. 28, No 3-4, 1983, Bratislava, ss. 63-73.

[13] Trebula P., Dekrét A.: *Vákuové sušenie agátového dreva s konvekčným ohrevom*, Zborník prác konferencie „Sušenie dreva, vývoj sušiarenských zariadení, riadenie procesov dreva z hľadiska energetickej náročnosti, SDVU, Bratislava, 1983, 182-187.

[14] Trebula P., Dekrét A.: *Vakuumtrocknung von Hainbuchenholz*, Holztechnologie, Wissenschaftlich-technische Zeitschrift fur die holzverarbeitende Industrie, 1984, Januar/Februar, 20-22.

[15] Dekrét A., Kurjatko S.: *O povrchových podmienkach pri difúzii vodných pár v dreve*, Drevársky výskum, zv. 2, 1986, ss, 69-80.

[16] Dekrét A., Trebula P.: *Vákuové sušenie smrekového dreva s konvekčným ohrevom*, Zborník vedeckých prác DF VšLD, 1985/86, ss. 223-231.

[17] Trebula P., Dekrét A.: *Vákuové sušenie hrabového dreva s kontaktným ohrevom*, Drevársky výskum, zväzok 110, 1986, SDVU Bratislava, 47-59.

[18] Trebula P., Dekrét A.: *Modeli na optimana na vakumno izsušavane na bičenii materialii s kontaktno nagrevane*, Naučno-techničeska konferencija, Plovdiv, 1986.

[19] Trebula P., Dekrét A.: *Technologiczne cechy podcisnieviovego suszenia i modelovanie tego procesu*, Przemysl drzevny, No 11, 1987, 19-21.

[20] Trebula P., Dekrét A.: *Vákuové sušenie reziva*, Vedecké aktuality VšLD Zvolen, 1987, ss. 74.

[21] Dekrét A., Trebula P.: *Stanovenie vnútorných napätí pri jednosmernom sušení dreva*, Zborník referátov medzinárodnej vedeckej konferencie, Sekcia 4, VšLD Zvolen, 1987, ss. 81-86.

[22] Dekrét A., Tebula P.: *On a model of internal stresses in drying wood*, Zborník referátov vedeckého seminára „Modelovanie-kvalita, ekonomika procesov sušenia dreva", VšLD Zvolen - TU Drážďany, Zvolen 1988, ss. 36-42.

[23] Dekrét A., Trebula P.: *Napätia v procese sušenia dreva*, Vedecké pedagogické aktuality 5, 1989, VšLD Zvolen, ss. 61.

[24] Dekrét A.: *On the diffusion coefficient in wood*, Proceedings of the symposium on „ The latest knowledge in the sphere of structure an physics of wood", VšLD Zvolen 1990, pp. 199-202.

[25] Dekrét A., Vacek V.: *Poznámka k Stammovej teórii viazanej vody v dreve*, Zborník vedeckých prác DF VšLD vo Zvolene, ALFA, 1991, ss. 51-60.

[26] Dekrét A., Trebula P.: *Modelovanie objemových síl pre napätia vznikajúce v sušenom dreve*, Zborník 8. medzinárodnej sušiarenskej konferencie, Karlove Vary, 1991, ss. 99-102.

[27] Dekrét A., Vacek V.: *Odhad a predikcia napätí v procese sušenia dreva*, Zborník z medzinárodnej konferencie „ Les, drevo, ekológia", sekcia „Wood structure properties", TU Zvolen, 1992, pp. 261-252.

[28] Dekrét A., Bodnár F.: *Estimations of internal stresse in drying wood by variational principles*, Proceedings of international conference „Vacuum drying of wood 93", TU Zvolen, 1993, pp. 92-99.

[29] Dekrét A., Bodnár F.: *On internal stresses in drying wood*, Internationales Wissenschaftliches Symposium „Theoriie und Praxis des Vakuum Schnittholztrocknung", TU Zvolen, 1993, ss. 61-66.

[30] Dekrét A.: *Estimations of the average moisture content and temperature of wood in the vacuum drier with radiant heating*, Internationales Wissenschaftliches Symposium „Theoriie und Praxis des Vakuum Schnittholztrocknung", TU Zvolen, 1993, ss. 22-25.

[31] Dekrét A., Bodnár F.: *Modeling of stresses in drying wood*, Proceedings of the 2-d international symposium „Wood structure and properties", 1994, TU Zvolen, 1995, pp. 91-96.

[32] Dekrét A., Vacek V.: *On moisture content distribution in drying board*, Proceedings of the international Conference „Vacuum drying of wood 95", TU Zvolen, 1995, pp. 62-65.

[33] Dekrét A., Vacek V.: *Another method of the estimation of internal stresses in drying wood*, Proceedings of the international Conference „Vacuum drying of wood 95", TU Zvolen, 1995, pp. 66-72.

[34] Dekrét A.: *Možnosti modelovania vzťahu „napätie-deformácia" v sušenom dreve*, Zborník 1. Sympózia „Interakcia dreva s rôznymi formami energie", TU Zvolen, 1996, pp. 115-119.

[35] Dekrét A., Trebula P.: *Niektoré teoretické problémy sušenia dreva*, Vedecké štúdie, 1997, TU Zvolen, 1997, ss. 58.

[36] Trebula P., Klement I., Dekrét A.: *Beach wood drying with the microwave heating*, Acta Facultatis Xylologiae Zvolen, XXXIX, 1997, pp. 87-93.

[37] Dekrét: *Poznámka o modelovaníniektorých javov v sušenej doske*, Zborník z

medzinárodnej vedeckej konferencie „Les-Drevo-Životné prostredie 97", ss. 139-144.

[38] Trebula P., Dekrét A.: *K niektorým problémom kontaktného sušenia*, Vedecké štúdie TU Zvolen, 12/1998/ A, TU Zvolen, 1999, pp. 56.

[39] Dekrét A., Trebula P.: *Poznámka k teplotným podmienkam v dreve pri kontaktnom sušení*, Acta Facultatis Xylologiae, TU Zvolen, XXXX, 1998, ss. 43-49.

[40] Dekrét A., Trebula P.: *O jednom odhade závislosti difúzneho koeficienta na vlhkosti a teplote*, Acta Facultatis Xylologiae, XXXXI, 1999, TU Zvolen, pp. 59-65.

[41] Dekrét A., Pajtašová M.: *Elastická zložka deformácie v sušenom dreve*, 2. vedecké sympózium Interakcia dreva s rôznymi formami energie, TU Zvolen, 2000, pp. 33-36.

[42] Klement I., Trebula P., Dektrét A., Kačík F.: *Vplyv teplotných a vlhkostných polív hrabovom dreve na jeho fzyiká lne a chemické zmeny.* Vedecké štúdie, Zvolen TU 10/2001/A, ss. 59.

Dept. of Applied Informatics; Faculty of Economics; Matej Bel University; Tajovského 10; SK–975 90 Banská Bystrica; Slovak Republic
E-mail: zimka@ef.umb.sk

Acta Univ. M. Belii
Math no. 10(2003), pp. 53–64

THE CLASSIFICATION OF PROGRAMMING ENVIRONMENTS

Ján Kollár, Peter Václavík and Jaroslav Porubän

Abstract. A process functional paradigm prevents the use of assignments in programs, at the same time providing full power of both functional and imperative languages to a programmer. $\mathcal{PFL}$ – an experimental process functional language, originally developed as a programming language, seems to be promising to integrate the implementation requirements for any language aimed to von Neumann computer architectures. As we hope, $\mathcal{PFL}$ may serve as a unified implementation language in the future. That is why the formalized definitions of environments presented in this paper are useful and constructive for further development of $\mathcal{PFL}$ as a minimal superset of programming languages currently being used in a practice. In particular, we will classify the environments dividing them into two basic categories – external and internal environments, that may be defined in any scope of a program. Then we extend the notion of explicit and implicit environments to object and modular environments. Finally, we formulate the requirements for safe programming, which prevents the use of undefined values in programs.

Introduction

In the past, except well known imperative languages such as Pascal, C, Modula and Ada we have analyzed the languages that combines imperative and functional paradigms to exploit the benefits of both – the ability to manipulate the state preserving at the same time functional semantics, such as SML [13], Scheme [1], Clean [2] and Haskell [12,17,18], that either exploit the environments explicitly or hide them to a programmer [20,21]. Our aims were strongly practical – to develop a programming language that integrates the ideas promising to correct programming [14,15] and predicting the behavior of the systems [4,9] statically.

$\mathcal{PFL}$ – an experimental process functional language [5,6,7,8,9,10,11,19] was originally developed as a general purpose programming language based on a semantically minimal superset of known programming paradigms, such as imperative and functional, modular and object-oriented, deterministic and non-deterministic, sequential and parallel, message passing and shared memory, etc. At the same time, the goal was to provide a minimal set of syntactic constructs to a programmer.

2000 Mathematics Subject Classification. 68N15, 68N18.

Key words and phrases. Programming paradigms, applicative programming, process functional programming, explicit and implicit environments, object and modular environments, environmental application, referential transparency, side effects.

This work was supported by VEGA Grant No.1/8134/01 – Binding the Process Functional Language to MPI.

Received 10. 12. 2002; Accepted 16. 4. 2003

Syntactically, $\mathcal{PFL}$ is an extended subset of Haskell. Semantically, $\mathcal{PFL}$ integrates both functional and imperative programming paradigms, not however in monadic manner [20,21].

There are no assignments in $\mathcal{PFL}$ expressions provided to a user. All imperative actions are performed by applications of processes. On the other hand, all what is done is seen. It means that the state can be monitored even using graphic interface strongly bound to the source specification, visualizing the flow of data in environments. We do not want to hide the imperative actions to a user making them functional; we just want to separate them from functional grains, to provide the transparent and strong feedback to a user about mapping a problem to target architecture.

At present time, it seems that $\mathcal{PFL}$ is promising to be used as a unified implementation bridge between specification languages and computer architectures, since of its high abstraction coming out from the functional basis from one side, and its ability to affect computer architecture resources directly from the other side. Using $\mathcal{PFL}$, neither a target imperative language, such as C in pure functional languages able to manipulate state [17], nor a core language as for constrained functional languages [16] is required. Instead of that, the machine code can be generated directly.

However, $\mathcal{PFL}$ as a unified implementation language needs more detailed formalization of effects that arise from the state manipulation, since they affect the function of computation. This paper is not devoted to reduction strategies and/or the restrictions that must be taken into account when deterministic computation is considered. In this paper we concentrate just on the classification of environments manipulated in $\mathcal{PFL}$ that may be found in programming languages in general, and we will show, how they are affected, using process functional paradigm, which is materialized in $\mathcal{PFL}$.

We will define explicit and implicit environments first. Then we will use them to define object and modular environments. For the purpose of understanding the definitions of environments, we introduce both basics of process functional paradigm as well as a brief overview of $\mathcal{PFL}$ language constructs, related however just to the definitions of environments, less to the semantics of the language or its implementation.

As a result, we formulate the requirements for safe programming with respect to further development of $\mathcal{PFL}$. The definitions of programming environments introduced in this paper are useful, since they show the danger coming out from an undisciplined programming using imperative languages, and they are constructive, since they show the way, in which the use of undefined values can be prevented, still preserving full power of imperative languages. As we hope, this is solved systematically an transparently using process functional paradigm. On the other hand, integrating functional and imperative programming paradigms into the process functional paradigm is just the first step towards removing the gap between the specification and implementation.

Process Functional Paradigm

The form of a single argument "pure" function f in $\mathcal{PFL}$ script is as follows

$$f :: T_x \to T_e$$
$$f\ x = e$$

where the first line contains the type definition (TD), (sometimes called type signature) for f, expressing that f is a mapping from the values of the type T_x to the values of the type T_e. The second line contains the definition (D) of the function f; f is defined for each argument x by the expression e. Since a $\mathcal{PFL}$ function is just a specific process, which has no environment variable in TD, we will use a "pair" $PD = (TD; D)$ to designate a process definition PD including function definition. Then, whenever appropriate, instead of two lines of the definition above we will write PD in the form

$$PD = (f :: T_x \to T_e;\ f\ x = e)$$

Further, to minimize the space, we will use the form

$$PD;\ \ldots ; PD;$$

in which semicolons represent invisible newline characters causing all PD's above are indented to the same column given by first PD in the sequence.

The static semantics for the application $(f\ m)$ is expressed by application rule, as follows:

(1)
$$\frac{f :: T_x \to T_e \qquad m : T_x}{f\ m :\ T_e}$$

According to (1), provided that f is of the type $T_x \to T_e$, and expression m is of the type T_x, then f is applied correctly to m, and then the application $(f\ m)$ is of the type T_e.

The dynamic semantics of the application $(f\ m)$ using semantic function $\mathcal{E}val$ and lambda form $(\lambda x.\ e)$ for f is as follows,

(2)
$$\mathcal{E}val[\![\ (\lambda x.\ e)\ m\ ]\!] = e[m/x]$$

where $e[m/x]$ (the value of application) is the expression e in which each occurrence of the lambda variable x is substituted by the expression m, or by the value of the expression m, depending on the reduction strategy.

The notion of variables in a purely functional language is mathematical; it means that all variables, such as lambda variable x or function f are values.

On the other hand, a variable in imperative languages is a memory cell used to store values. Process functional paradigm integrates both meanings using the theory of mutable abstract types [3]: process functional variable is a cell v containing (defined) data value $m \neq \perp$ (then we write $v[m]$), or undefined value $\perp$ ($v[\perp]$). At the same time, an environment variable v is an overloaded mapping, as follows.

(3)
$$v :\ \widetilde{T} \to T$$

where T is a data type and $\widetilde{T} = T \cup ()$ where $()$ is the unit type.

The value of the application $(v\ m')$ depends on the argument m' as well as on the value m having been stored in the environment variable before.

According to definition 1, if $(m : ())$, then $(v\ m')$ is the access.

Definition 1.

$$(1.1)\qquad \frac{v[m] \vdash v : \widetilde{T} \to T \qquad m' : \ ()}{v\ m' : \ T \vdash m' : T}$$

where $(v\ m') = m'$.

$$(1.2)\qquad \frac{v[\bot] \vdash v : \widetilde{T} \to T \qquad m' : \ ()}{v\ m' : \ \Omega \vdash \bot : \Omega}$$

where $(v\ m') = \bot$.

According to definition 2, if $(m' : T)$, then $(v\ m')$ is the update of environment variable.

Definition 2.

$$(2.1)\qquad \frac{v[m] \vdash v : \widetilde{T} \to T \qquad m' : \ T}{v\ m' : \ T \vdash v[m'] \vdash m' : T}$$

where the state transition is $v[m] \Rightarrow v[m']$ and $(v\ m') = m'$.

$$(2.2)\qquad \frac{v[\bot] \vdash v : \widetilde{T} \to T \qquad m' : \ T}{v\ m' : \ T \vdash v[m'] \vdash m' : T}$$

where the state transition is $v[\bot] \Rightarrow v[m']$ and $(v\ m') = m'$.

Clearly, case (1.1) is the access of a data value, case (1.2) is the unwanted access of undefined value (prevented by the type checking), case (2.1) is the update called modification and case (2.2) is the update called initialization of the environment variable by the value m'. Hence, well-typed cases are (1.1), (2.1) and (2.2).

The dynamic semantics of the application $(v\ m')$ can be easily derived from the definitions 1 and 2. Using $\mathcal{E}val$, it is as follows:

Definition 3.

$$\begin{array}{ll}
(1.1) & \mathcal{E}val[\![\ v\ m'\ ]\!]\ v[m] = \ \mathcal{E}val[\![\ m\ ]\!]\ v[m], \quad \text{if } m' : () \\
(1.2) & \mathcal{E}val[\![\ v\ m'\ ]\!]\ v[\bot] = \ \mathcal{E}val[\![\ \bot\ ]\!]\ v[\bot], \quad \text{if } m' : () \\
(2.1) & \mathcal{E}val[\![\ v\ m'\ ]\!]\ v[m] = \ \mathcal{E}val[\![\ m'\ ]\!]\ v[m'], \quad \text{if } m' : T \\
(2.2) & \mathcal{E}val[\![\ v\ m'\ ]\!]\ v[\bot] = \ \mathcal{E}val[\![\ m'\ ]\!]\ v[m'], \quad \text{if } m' : T \\
 & \mathcal{E}val[\![\ \ m\ \ ]\!]\ v[m] = \ \mathcal{E}val[\![\ m\ ]\!] \\
 & \mathcal{E}val[\![\ \ \bot\ \ ]\!]\ v[\bot] = \ \bot
\end{array}$$

Since $(v\ m')$ may affect the state of computation by the side effect, this application is rather environmental than functional.

In $\mathcal{PFL}$ however, it is possible to perform environmental applications just indirectly via applications of processes. This approach prevents an undisciplined use of assignments, since there is no decision left to a programmer where to apply an environment variable in an expressions. It means that no environment variable occurs in source $\mathcal{PFL}$ expressions. Instead of that, environment variables are introduced in processes type definitions. This guarantees the systematic and disciplined use of hidden assignments in programs.

As shown below, it is the matter of the translation, to "remove" environment variables from type definitions and to "bring" them into expressions.

We will use mathematical form for $\mathcal{PFL}$ constructs, not numbering them. In this form we use $\rightarrow$, and $\Rightarrow$ instead of `->` and `=>`. We also use array curly brackets in the form $\{\!|$ and $|\!\}$ instead of { and } to precede the confusion with set brackets.

A $\mathcal{PFL}$ program consists of a set of multi-argument process definitions PD and main expression e in global scope $s = 0$, as follows,

$$\begin{array}{l}\textbf{program } M \textbf{ where}\\ PD;\ \ldots\ ; PD;\\ \textbf{main} = e\end{array}$$

where PD is a pair consisting of a type definition TD and the definition D, $PD = (TD; D)$

The form of PD is as follows.

$$\begin{array}{lll} f :: \overline{T_1} \rightarrow \ldots \rightarrow \overline{T_n} & \rightarrow \widetilde{T} & \\ f\ \ p_1\ \ \ldots\ \ p_n & = & e \\ & & \textbf{where} \\ & & \quad PD;\ \ldots\ ; PD; \end{array}$$

where PD's that follow keyword **where** are local processes definitions.

Using BNF, the syntax of type expressions is as follows:

$$\begin{array}{lll} \overline{T} & ::= & v\ T \mid v\ \{\!|R|\!\}\ T \mid \widetilde{T} \\ \widetilde{T} & ::= & () \mid T \\ T & ::= & a \mid T^P \mid T \rightarrow\ T \mid \{\!|R|\!\} \rightarrow T \mid T^D\ T_1 \ldots T_r \mid Cl\ T_1 \ldots T_u \\ \{\!|R|\!\} & ::= & \{\!|T_1^R, \ldots, T_d^R|\!\}, d \geq 1 \end{array}$$

where $r, u \geq 0$, v is an environment variable, $\{\!|R|\!\}$ is n-dimensional range, () is unit type, a is a type variable, T^P is primitive type $(Char, Int, Float)$, $T \rightarrow\ T$ is function type, $\{\!|R|\!\} \rightarrow T$ is array type, T^D is algebraic data type, Cl is a class name and T_i^R are range types – enumerated algebraic types, characters, integers, or their finite subranges in the form $c_i^L\ .\ c_i^U$, corresponding to lower, and upper bounds of an array, such that $c_i^L, c_i^U : T_i^R$.

Attention must be paid to the type expressions $v\ T$ and $v\ \{\!|R|\!\}\ T$, since they are just syntactic shortcuts, which serve during compilation

- to derive the environment variable types $v : \widetilde{T} \rightarrow T$ and $v : \{\!|R|\!\} \rightarrow \widetilde{T} \rightarrow T$, transforming both to T in target process f type definition, and
- transforming each expression as follows:

– Provided that m is a process argument of source type $v\ T$, then it is translated into the form of environmental application $(v\ m)$, of the type T.

– Provided that $(\{\!|e^R|\!\}m)$, (e^R is an index expression) is a process argument argument of source type $v\ \{\!|R|\!\}\ T$, then it is translated into the form $((v\ \{\!|e^R|\!\})\ m)$ which is of the type T again. Here $(v\ \{\!|e^R|\!\})$ selects a cell representing the item of an array, and then this cell – being the environment variable – is applied to m.

According to the transformation above, the source $\mathcal{PFL}$ script comprising environment variables just in type definitions is translated to an intermediate form that comprises the environment variables just in expressions.

Formal parameters p_i are either in the form of simple lambda variable x or in the form of patterns $(C\ p_1\ \ldots p_m)$ or $x@(C\ p_1\ \ldots p_m)$, such that $m \geq 0$, and C is a constructor of algebraic type. The form $x@(C\ p_1\ \ldots p_m)$ is extremely useful, since it allows updating the items of the structure $(C\ p_1\ \ldots p_m)$ in place returning its incoming value x.

$\mathcal{PFL}$ algebraic types are defined using data definitions, in the form as follows:

$$\begin{aligned}\textbf{data}\ T^D\ a_1 \ldots a_u = {} & C_1\ T_{1,1} \ldots T_{1,n_1} \\ & \mid C_2\ T_{2,1} \ldots T_{2,n_2} \\ & \ldots\ldots\ldots\ldots \\ & \mid C_m\ T_{m,1} \ldots T_{m,n_m}\end{aligned}$$

where a_k are type variables, $T_{i,j}$ are type expressions and C_i are constructors of algebraic type T^D. For our purposes it is sufficient to consider just product types $(m = 1)$ an then we may write the definition above in the form

$$\textbf{data}\ T^D\ a_1 \ldots a_u = C\ T_1 \ldots T_n$$

Provided that T^D is defined, and expressions $m_1, \ldots, m_n$ are such that $m_1 : T_1$, $\ldots$, $m_n : T_n$ holds, then

$$(C\ m_1 \ldots m_n)$$

is used in an expression to construct n-tuple of items $m_1, \ldots, m_n$, such that it is of monotype $T^D\ T_1^M \ldots T_u^M$, $T^D\ T_1^M \ldots T_u^M \subset T^D\ a_1 \ldots a_u$.

A dynamic array in $\mathcal{PFL}$ is created by an expression, called array creator, as follows.

$$\{\!|R^F|\!\} \to m$$

used in an expression, where $\{\!|R^F|\!\}$ is a finite subrange, $\{\!|R^F|\!\} \subseteq \{\!|R|\!\}$ and $m : \widetilde{T}$. If $m : T$, then $\{\!|R^F|\!\} \to m$ is the array of items, each initialized to m.

$\mathcal{PFL}$ type synonyms TS are defined using type definitions, in the form as follows

$$\textbf{type}\ TS\ a_1 \ldots a_u = \overline{T}$$

that use is appropriate especially when an environment variable v is shared by different processes, but also when a variable is associated with a memory address (for example 177746), such as follows

$$\textbf{type}\ TS\ a_1 \ldots a_u = v\ T\ \textbf{at}\ \#177746$$

Abstract type is implemented using class definition and corresponding set of instance definitions, in the form as follows,

$$\textbf{class } Cl\ a_1 \ldots a_u \textbf{ where } TD;\ \ldots ; TD;$$

$$\textbf{instance } Cl\ T_1 \ldots T_u \textbf{ where } D;\ \ldots ; D;$$

such that for each instance of a class Cl holds $(Cl\ T_1 \ldots T_n) \subset (Cl\ a_1 \ldots a_n)$. If a class is monomorphic, then it contains no type variables a_i $(u = 0)$ and then it is possible to define just one instance for this class.

Provided that TD's in a class definition contains at least one environment variable, i.e. this environment is not empty, new object for an instance of this class is created using the type expression in expressions as follows

$$Cl\ T_1 \ldots T_u$$

Then, if m is an expression of the type $Cl\ T_1 \ldots T_u$ and a process f is defined by D in **instance** $Cl\ T_1 \ldots T_u$, then f is selected using expression

$$(m \Rightarrow f)$$

which, when applied, affects the environment given by m. It may be noticed that if a class environment is empty, then it exists just one object for each instance, hence, in this case we have no opportunity for object programming using such a class.

It is easy to see, that modularity may be implemented in a similar manner. Except a main module, formed by program the additional set of modules can be defined each of them in the form

$$\textbf{module } M \textbf{ where } PD;\ \ldots ; PD;$$

Then a process f defined in module M is accessible in other modules using the expression

$$(M \Rightarrow f)$$

Let the global scope in each module and each object is the same as the global scope in the program, i.e. $s = 0$, and local scopes are such that $s > 0$. Provided that $PD = (TD; D)$ such that $v \in TD \wedge f \in D$, then both f and v are in the same scope. The names of all processes in the same scope s must be unique, and any environment variable name v in this scope must differ from the names of processes. On the other hand, $v \in TD_1 \wedge v \in TD_2$ means sharing this variable by processes defined by D_1 and D_2, which is allowed.

The scope boundary, as well as the visibility of lambda and pattern variables and process names is the same as in Pascal; if f is in the scope s then its formal parameters (lambda and pattern variables) and the names of local processes (as well as its body) are in the scope $s + 1$; if a name used in a scope s_3 is introduced in scopes s_1 and s_2 $(s_1 < s_2 < s_3)$ not however in scopes s, $s_2 < s \leq s_3$, then the used name is such that introduced in the scope s_2.

To prevent the obscure notation, we will consider just single argument processes f and g, such that f is defined in a scope s and g is defined in a scope $s',\ s' \geq s$.

Considering a relation between environments in scopes s and s' in a program, we distinct two essential kinds of environments:

- Explicit environment – formed introducing new names for environment variables that are different from the names used in patterns
- Implicit environment – formed using names introduced in patterns.

This means, that an environment variable in the scope s may belong to the explicit environment $E^{(s)}$, only if it does not belong to the implicit environment $I^{(s)}$.

An explicit environment for values is defined according to definition 4.

Definition 4. For all processes f in the scope s, defined by

$$PD = (f ::\ v\ T \to \widetilde{T_f};\ f\ x\ =\ e_f)$$

where T is primitive monotype or function polytype or dynamic array polytype, it holds:

- Lambda variable x is *the value* of v accessed in e_f by x, where $x : T$.
- If $\{v :\ \widetilde{T} \to T\} \not\subseteq I^{(s)}$ then $\{v :\ \widetilde{T} \to T\} \subseteq E^{(s)}$.
- Then, provided that g is a process defined by

  $$D = (g\ y\ =\ e_g)$$

  in the scope s', $s' \geq (s-1)$, such that f is accessible in e_g, an environment variable v is accessed/updated in e_g by application $(f\ m)$, translated to $f\ (v\ m)$, where v is *the address* and $m :\ \widetilde{T}$.

An explicit environment for static arrays is defined according to the definition 5.

Definition 5. For all processes f in the scope s, defined by

$$PD = (f ::\ v\ \{\!|R^F|\!\}\ T \to \widetilde{T_f};\ f\ x\ =\ e_f)$$

where T is polytype, $\{\!|R^F|\!\}$ is finite subrange $\{\!|R^F|\!\} \subseteq \{\!|R|\!\}$, it holds:

- Lambda variable x is *the value* of $v\ \{\!|e^R|\!\}$ ($\{\!|e^R|\!\}$-th item of the array v), $\{\!|e^R|\!\} \in \{\!|R^F|\!\}$, accessed in e_f by x, where $x : T$.
- If $\{v : \{\!|R|\!\} \to\ \widetilde{T} \to T\} \not\subseteq I^{(s)}$ then $\{v : \{\!|R^F|\!\} \to\ \widetilde{T} \to T\} \subseteq E^{(s)}$
- Then, provided that g is a function or a process defined by

  $$D = (g\ y\ =\ e_g)$$

  in the scope s', $s' \geq (s-1)$, such that f is accessible in e_g, and each environment variable – the $\{\!|e^R|\!\}$-th item of the array v is accessed/updated in e_g by application $f\ (\{\!|e^R|\!\}\ m)$, translated to $f\ (v\ \{\!|e_R|\!\}\ m)$, v is *the address*, where $m :\ \widetilde{T}$.

An explicit environment for algebraic structured data is defined according to the definition 6.

Definition 6. For all processes f in the scope s, defined by

$$PD = (f :: \ v \ T^D \rightarrow \widetilde{T_f}; \ f \ p \ = \ e_f)$$

where $p ::= x \mid x@(C \ x_1 \ \ldots x_n) \mid (C \ x_1 \ \ldots x_n)$, T^D is algebraic monotype, C is a constructor $(C :: \ T_1 \rightarrow \cdots \rightarrow T_n \rightarrow T^D)$, it holds:

- Lambda variable x is *the address* of v. Then x, x_1, ... , x_n, are used in e_f as *values*, where $x : T$, $x_1 : T_1$, ... , $x_n : T_n$.
- If $\{v : \widetilde{T^D} \rightarrow T^D\} \not\subseteq I^{(s)}$ then $\{v : \widetilde{T^D} \rightarrow T^D\} \subseteq E^{(s)}$.
- Then, provided that g is a function or a process defined by

  $$D = (g \ y \ = \ e_g)$$

  in the scope s', $s' \geq (s-1)$, such that f is accessible, an environment variable v is accessed/updated in e_g by application $f \ (C \ m_1 \ \ldots \ m_n)$, translated to $f \ (v \ (C \ m_1 \ \ldots \ m_n))$, v is *the address*, where $m_i : \ \widetilde{T_i}$, for $i = 1 \ldots n$.

The extension to algebraic polytype is straightforward – by the substitution of T^D by type application $(T^D \ T_1^a \ldots T_u^a)$, where T_u^a are type expressions comprising type variables.

According to the definition 5, static arrays belong to an explicit environment and they cannot be used in a higher order manner. According to the definition 6, static data structures such as Pascal records in global memory or on the stack are manipulated.

Opposite to the explicit environment, the implicit environment is defined considering that the items of data structures and arrays are values, but at the same time they are cells, used as implicit environment variables for local processes. Since non-structured values are represented by lambda variables that are values, not cells, it follows that they cannot form an implicit environment.

An implicit environment formed by a lambda variable of a dynamic array type is defined according to definition 7.

Definition 7.

Let f is a process or function in the scope s, defined by

$$D = (f \ x \ = \ e_f)$$

(type definition is out of interest here), such that $x : \{\!|R|\!\} \rightarrow T$, where T is a polytype, and R is infinite range.

Let us consider the application $(f \ (\{\!|R^F|\!\} \rightarrow m))$, such that $R^F \subseteq R$ and $m : T$. Then it holds:

- Lambda variable x is a dynamic array value and $x\{\!|e^R|\!\}$ the array item value, both accessible in e_f, $x : \{\!|R^F|\!\} \rightarrow T$ and $x\{\!|e^R|\!\} : T$.
- $\{x : \{\!|R|\!\} \rightarrow \widetilde{T} \rightarrow T\} \subseteq I^{(s')}$, $s' > s$, provided that $\exists \ g$ in the scope s', defined by

  $$PD = (g :: \ x \ \{\!|\!\}\ \_ \rightarrow \widetilde{T_g}; \ g \ y \ = \ e_g)$$

  where $x \ \{\!|\!\}\ \_ = x \ \{\!|R|\!\} \ T$.
- Then, $\{\!|e^R|\!\}$-th item of a dynamic array x is accessible/updatable in scopes s'', $s'' \geq s'$, if g is accessible, by the application $g \ (\{\!|e^R|\!\} \ m')$, translated to $g \ (x \ \{\!|e^R|\!\} \ m')$, where x is *the value*, and $m' : \ \widetilde{T}$.

A possible type definition for a dynamic array argument of f in the definition 7 is $(\{|R|\} \to T)$, or $v\ (\{|R|\} \to T)$, where v is an explicit environment variable, and $v\ (\{|\} \to \_)$, if v is an implicit environment variable.

An implicit environment formed by a lambda variable of an algebraic structured data type is defined according to definition 8.

Definition 8.

Let f is a process or a function in the scope s, defined by

$$D = (f\ (C\ x_1\ \ldots x_n)\ =\ e) \quad \text{or} \quad D = (f\ x@(C\ x_1\ \ldots x_n)\ =\ e)$$

such that $x : T^D\ T_1^a \ldots T_u^a$, $x_1 : T_1, \ldots, x_n : T_n$, $x : T$, $C : T_1 \to \cdots \to T_n \to T^D\ T_1^a \ldots T_u^a$, T^D is an algebraic data polytype and $T_1, \ldots, T_n$ are polytypes.

Suppose the application $f\ (C\ m_1\ \ldots\ m_n)$, where $m_i : T_i$. Then it holds:

- Lambda variable x is the data value and $x_1, \ldots, x_n$ are its item values, accessible in e_f, $x : T^D\ T_1^a \ldots T_u^a$, and $x_i : T_i$, for all $x_i \in \{x_1, \ldots, x_n\}$.
- $\{x_i : \widetilde{T_i} \to T_i\} \subseteq I^{(s')}$, $s' > s$, provided that $\exists\ g$ in the scope s', defined by

$$PD = (g :: \ x_i\ \_ \to \widetilde{T_g};\ g\ y\ =\ e_g)$$

  where $(x_i\ \_) = (x_i\ T_i)$.
- Then a structured data item x_i is accessible/updatable in e_f (including scopes s'', $s'' \geq s'$), if g is accessible, by the application $g\ m_i'$, translated to $g\ (x_i\ m_i')$, where x_i is *the address*, and $m_i' : \widetilde{T_i}$.

Definition 9.

Object environment is the explicit environment $E^{(0)}$, defined by a class definition in global scope ($s = 0$), and allocated by class application $Cl\ T_1 \ldots T_u$. Its size is dependent on argument types $T_1, \ldots, T_u$. Different objects have mutually disjunctive environments. All object environments are also disjunctive with explicit and implicit environments of a program and modules.

Definition 10.

Since a program is a main module of a modular sequential language, it is sufficient to consider just a set of modules $M_1, \ldots, M_p$, one of them being a program.

Let $E_{M_i}^{(0)}$ is an explicit environment in the global scope $s = 0$ defined by type definitions of global processes in module M_i. Let $(M_j \Rightarrow f_k)$ is a process f_k imported from module M_j being applied in module M_i. Then $\mathbf{I}_{M_i}^{(0)}$ is a set of subsets of explicit environments $E_{M_j}^{(0)}$ formed by environment variables affected by all imported processes, which is defined as follows.

$$\mathbf{I}_{M_i}^{(0)} = \{S \mid S \subseteq E_{M_j}^{(0)},\ v \in S \iff v \in TD, (TD(M_j \Rightarrow f_k), D(M_j \Rightarrow f_k)) \in M_j\}$$

where $TD(M_j \Rightarrow f_k)$ is type definition, and $D(M_j \Rightarrow f_k)$ is the definition of a process f_k in a module M_j.

Then modular environment for a module M_i is defined by the unification of all subsets of explicit environments affected by all imported processes applied in M_i and the explicit environment in global scope of module M_i.

$$\bigcup_{\forall(M_j \Rightarrow f_k) \in M_i} \mathbf{I}_{M_i}^{(0)} \cup E_{M_i}^{(0)}$$

Clearly, modular environments may be disjunctive, i.e. non-overlapping, but also shared.

Conclusion

In this paper, programming environments are defined in terms of $\mathcal{PFL}$ – a process functional language. We introduced the essence of process functional paradigm – programming by application of processes instead of assignments providing the ability for imperative semantics of $\mathcal{PFL}$ programs being described seemingly by a purely functional source script, since no environmental applications occur in source expressions. This approach, from one point of view, does not disqualify the language to the role of a purely functional but still macro language that affects architecture resources indirectly, using an underlying imperative target language. On the other hand, defining the environments in its abstracted nature, we may see a danger of the use of undefined values, inherent to each imperative language.

Without doubt, the implicit environments are initialized to defined values, since algebraic data are constructed by $(C\ m_1 \dots m_n)$ and arrays are created by $(\{\!|R^F|\!\} \rightarrow m)$, using m_i and m that cannot be of unit type $()$. To preserve the safeness in the use of defined values of arrays we must guarantee that all items are defined also for arrays created using loop comprehensions [8] that are over the scope of this paper.

However, since explicit environments are not initialized as may be seen from their definitions, it is quite possible to use the undefined values when applying processes to expressions of unit type. Therefore, to guarantee the use of defined values in programs, all explicit environment variables, except those associated with a memory addresses we must think about their initialization. But this approach decreases the run-time efficiency, especially for local explicit environments ($s > 0$). Moreover, since modular environment for a module is formed by unification of global explicit environment of this module and all subsets of explicit environments affected by imported processes from other modules, the modular environment may be shared. Then, the initialization of shared environments possesses the question, which of potential initializations occurring in the different modules determines the initial state. Clearly, this question has no satisfactory answer for parallel modules, since multiple initialization of shared environments in parallel would result to non-deterministic initial state. On the other hand, it is reasonable to initialize object environments, since they are disjunctive. (Notice, that we do not need static methods, such as in C here, since our "static" processes are defined in program modules.)

At this point it may be seen the advantage of process functional approach: Since $\mathcal{PFL}$ programs are expressions, the use of undefined explicit environment variables are identifiable during the type checking, providing strong feedback to a user about incorrectness, since they are related to arguments of unit types used improperly in applications of processes.

Concluding, the safe programming is such that uses either initialized environments or provide source-to-target feedback about the use of undefined values that yield potential incorrect execution of programs.

References

[1] Abelson H., et al, *Revised Report on the Algorithmic Language Scheme, In: Higher-Order and Symbolic Computation* **11** (1998), 7–105.

[2] Achten P., and R. Plasmeijer, *The implementation of interactive local state transition systems in Clean, In: Koopman et al. (Ed.): IFL'99* **LNCS 1868** (1999), 115–130.

[3] Hudak P., *"Mutable abstract datatypes – or – How to have your state and munge it too", Yale University, Department of Computer Science, Research Report YALEU/DCS/RR-914* (December 1992; rev. May 1993).

[4] Hudák Š., and K. Teliopoulos, *Formal Specification and Time Analysis of Systems, Proc. of International Scientific Conf. ECI'98, Košice-Herľany, Slovakia* (October 8–9 1998), 7–12.

[5] Kollár J., *Process Functional Programming, Proc. ISM'99, Rožnov pod Radhoštěm, Czech Republic* (April 27–29, 1999), 41–48.

[6] Kollár J., *PFL Expressions for Imperative Control Structures, Proc. Scient. Conf. CEI'99, Herľany, Slovakia* (October 14–15, 1999), 23–28.

[7] Kollár J., *Control-driven Data Flow, Journal of Electrical Engineering, No.3–4* **51** (2000), 67–74.

[8] Kollár J., *Comprehending Loops in a Process Functional Programming Language, Computers and Artificial Intelligence* **19** (2000), 373–388.

[9] Kollár, J., and Porubän, J., *Static Evaluation of Process Functional Programs. Analele Universitatii din Oradea, Proc. of the 6'th Scientific Conference with International participation EMES'01 Engineering of Modern Electric Systems, Felix-Spa, Oradea, Romania* (May 24–26, 2001), 93–98.

[10] Kollár, J., Porubän, J., Václavík, P., and Vidiščak, M., *Lazy State Evaluation of Process Functional Programs, ISM'02, Proceedings of the Conference "Information Systems Modelling", Rožnov pod Radhoštěm, Czech Republic* (April 22–24, 2002), 165–172.

[11] Kollár, J., *Imperative Expressions using Implicit Environments. Proc of ECI'2002, the 5-th Int. Scient. Conf. on Electronic Computers and Informatics, Herľany, Slovakia* (Oct. 10-11, 2002,), 86–91.

[12] Launchbury J., and S.L. Peyton Jones, *"Lazy Functional State Threads", Computing Science Department, Glasgow University, 1994, 17 pages.*

[13] Milner R., Mads Tofte, Robert Harper, and David MacQueen, *The Definition of Standard ML - Revised*, The MIT Press, May 1997.

[14] Novitzká, V., *Systems for Deriving Correct Implementations, Proc. ISM'99, Rožnov pod Radhoštěm, Czech Republic* (April 27–29, 1999), 201–207.

[15] Novitzká, V., *Formal Foundations of Correct Programming*, elfa, Košice, Slovakia, 1999.

[16] Paralič, M., *Mobile Agents Based on Concurrent Constraint Programming, Joint Modular Languages Conference, JMLC 2000, Zurich, Switzerland. In: Lecture Notes in Computer Science* **1897** (September 6–8, 2000), 62–75.

[17] Peyton Jones S.L., and P. Wadler, *Imperative functional programming, In 20th Annual Symposium on Principles of Programming Languages, Charleston, South Carolina* (January 1993), 71–84.

[18] Peyton Jones S.L., and John Hughes [editors], *"Report on the Programming Language Haskell 98 – A Non-strict, Purely Functional Language"* (February 1999), 163 pages.

[19] Václavík, P., and Porubän, J., *Object Oriented Approach in Process Functional Language. Proc of ECI'2002, the 5-th Int. Scient. Conf. on Electronic Computers and Informatics, Herľany, Slovakia* (Oct. 10-11, 2002), 92–96.

[20] Wadler P., *The essence of functional programming, In 19th Annual Symposium on Principles of Programming Languages, Santa Fe, New Mexico (draft)* (January 1992), 23 pages.

[21] Wadler P., *The marriage of effects and monads, In ACM SIGPLAN International Conference on Functional Programming* (1998), ACM Press, 63–74.

Department of Computers and Informatics; Technical University of Košice; Letná 9; SK–041 20 Košice; Slovakia
E-mail: `Jan.Kollar@tuke.sk`