# Spring School on Combinatorics 2006

Jan Hladký, Marek Krčál, Bernard Lidický (eds.)


41
PENSION

## Preface

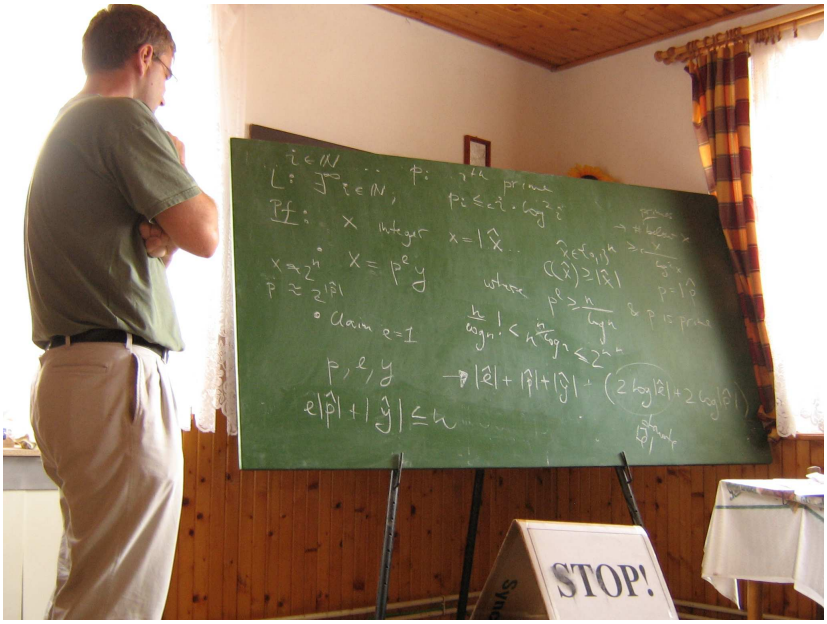
Spring school on Combinatorics is a traditional meeting organized for members of the Combinatorial Seminar at Charles University for nearly 30 years. By now it is internationally well known and it is regularly visited by students, postdocs and teachers from our cooperating institutions in the DIMATIA and COMBSTRU networks. In the years 1999–2001, and again in 2004–2006, the school is supported by ERASMUS–SOCRATES Intensive Programme 503334–IC–1–2002–1–CZ–ERASMUS–IPUC–1 which includes participation of universities from Bonn, Berlin, Bordeaux, Barcelona, Pisa and recently Bergen.

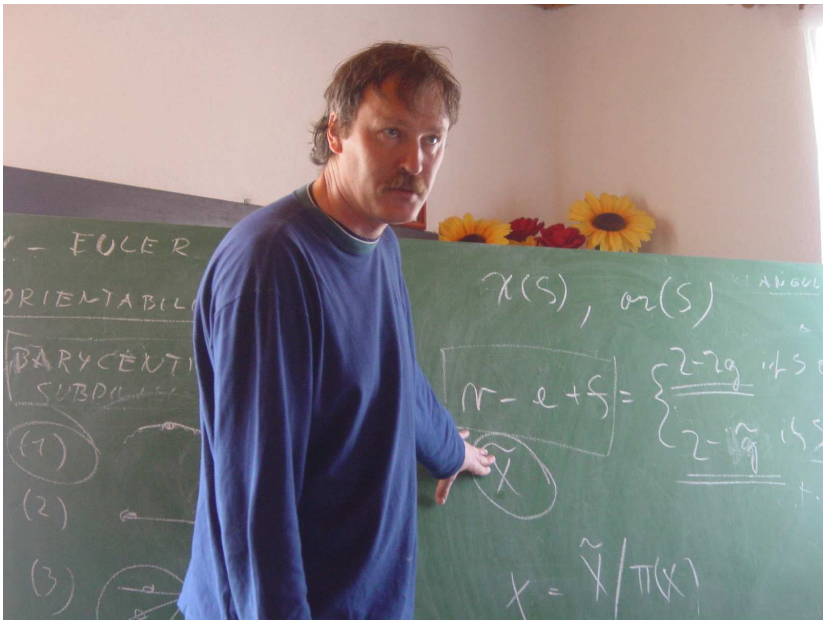
Spring Schools are entirely organized and arranged by undegraduate students. The lectures' subjects are selected by supervisors from the Department of Applied Mathematics (KAM) and Institute for Theoretical Computer Science (ITI) of Charles University as well as from other participating institutions. In contrast, the lectures themselves are almost exclusively given by students, both graduate an undergraduate. This leads to a unique atmosphere of the meeting which helps the students in further studies and their scientific orientation.

This year the Spring School was organized in Borová Lada, a mountain village in Šumava hills with a great variety of possibilities for outdoor activities like hiking a biking. Some of them are mirrored by photos in this volume.

We thank Marek Krčál, Bernard Lidický and Honza Hladký as the main organizers who also completed this volume. We also thank Robert Šámal, Martin Mareš, Ondřej Pangrác, Michal Koucký and other colleagues who took part both in the organization and in the Spring School itself. We hope to meet all this year's participants next year!

Jiří Fiala, Jan Kratochvíl, Jaroslav Nešetřil


STOP!



## Abstracts of all talks (in order of appearance)

**Vít Jelínek**
jelinek@kam.mff.cuni.cz

### Sumsets in semigroups

The aim of my talk is to present a recent combinatorial proof by Nathanson and Ruzsa of a result of Khovanskii on the size of sumsets in an arbitrary abelian semigroup.

Let $S$ be an arbitrary abelian semigroup (i.e., $S$ is a set with a commutative and associative operation '+'). For two sets $A, B \subseteq S$, we write $A+B$ to denote the set of all elements of $S$ that can be obtained by adding an element of $A$ and an element of $B$; in other words, $A + B = \{a + b; a \in A,\, b \in B\}$. Moreover, we write $kA$ for the $k$-fold sum $A + A + \cdots + A$.

In early 1990's, Khovanskii [1,2] proved that for every abelian semigroup $S$ and any two finite sets $A, B \subseteq S$, the size of the set $B + kA$ is equal to a polynomial in $k$, up to finitely many exceptions; formally, there is a polynomial $p$ and a constant $C$ such that for every $k > C$, $|B + kA| = p(k)$. In 2000, Nathanson [3] has proved a more general claim:

**Theorem 1.** For every $\ell + 1$ finite subsets $A_1, A_2, \ldots, A_\ell, B$ of $S$, there is a polynomial $p$ with $\ell$ variables, and a constant $C$ such that for every $\ell$-tuple of integers $k_1, \ldots, k_\ell$, all of them greater than $C$, we have the identity

$$|B + k_1A_1 + k_2A_2 + \cdots + k_\ell A_\ell| = p(k_1, \ldots, k_\ell).$$

In 2002, Nathanson and Ruzsa [4] found an elementary and simple proof of this statement. I will present Nathanson and Ruzsa's proof in the simplified setting where $B = A_1 = \ldots = A_\ell$, and then I will briefly explain how to adapt the arguments to prove the theorem in full generality.

#### References

[1] A. G. Khovanskii: Newton Polyhedron, Hilbert Polynomial and Sums of Finite Sets, *Functional Analysis and Its Applications,* 26:276–281 (1992).

[2] A. G. Khovanskii: Sums of Finite Sets, Orbits of Commutative Semigroups and Hilbert Functions, *Functional Analysis and Its Applications,* 29:102–112 (1995).

[3] M. Nathanson: Growth of sumsets in abelian semigroups, *Semigroup Forum,* 61:149–153 (2000).

[4] M. Nathanson, I. Ruzsa: Polynomial growth of sumsets in abelian semigroups, *Journal de Théorie des Nombres de Bordeaux,* 14:2 (2002).

**Alexandr Kazda**
alexandr.kazda@seznam.cz

Presented paper by Noga Alon, Eval Lubetzky

## Codes and Xor Graph Products

(http://www.math.tau.ac.il/~nogaa/PDFS/xor4.pdf)

The article focuses on studying the *xor products* of graphs. Given the graphs $G$ and $H$, their xor product $G \cdot H$ is the graph with vertex set $V(G) \times V(H)$ (Cartesian product) in which $(v_1, v_2)$ and $(u_1, u_2)$ connected iff exactly one of the following pairs is connected: either $v_1u_1$ in $G$ or $v_2u_2$ in $H$. We can then define $G^n$ by induction. The notion of xor product can be easily generalised to multigraphs (the only new phenomenon introduced by multigraphs is the possibility of loops in a graph). Due to the number of issues regarding xor products that are addressed in the original article, only a part of it will be referred here.

The main points of interest are the asymptotics of independence and clique numbers of $G^n$ (denoted $\alpha$ and $\omega$, respectively) as $n$ tends to infinity. Recall that $\alpha(G)$ is the maximal size of an independent set in $G$ (a set $U \subset V$ such that no two vertices in $U$ are connected) and $\omega(G)$ is the maximal size of a clique in $G$ (a set $W \subset V$ such that every pair of vertices in $W$ is conneced). To be able to compare the speed of growth of these numbers, denote:

$$x_\alpha(G) = \lim_{n\to\infty} \sqrt[n]{\alpha(G^n)} = \sup_n \sqrt[n]{\alpha(G^n)}$$
$$x_\omega(G) = \lim_{n\to\infty} \frac{\omega(G^n)}{n} = \sup_n \frac{\omega(G^n) - 2}{n+1}.$$

The correctness of these definitions stems from the fact that (for $\alpha(G) > 0$) the function $f(n) = \alpha(G^n)$ is super-multiplicative (i.e. $f(n+m) \geq f(n)f(m)$) and $\hat{g}(n) = \omega(G^{n-1}) - 2$ is (for $\omega(G) > 0$) super-additive (i.e. $\hat{g}(n+m) \geq \hat{g}(n) + \hat{g}(m)$).

The following bounds for $x_\alpha$ are established:

**Theorem 1.** *Let $G$ be a multigraph such that $\alpha(G) > 0$. Then $\sqrt{|V(G)|} \leq x_\alpha(G) \leq |V(G)|$.*

**Theorem 2.** *Let $G = (V, E)$ be a multigraph and let $H$ be an induced subgraph on $U \subset V$ satisfying $\alpha(H) > 0$. Then:*

$$x_\alpha(H) \leq x_\alpha(G) \leq x_\alpha(H) + |V| - |U|.$$

**Theorem 3.** *Let $G$ be a loopless nontrivial $d$-regular graph on $n$ vertices, and let $d = \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ denote the eigenvalues of $G$ (that is, the eigenvalues of $G$'s adjacency matrix). Then:*

$$x_\alpha(G) \leq \max(|n - 2d|, 2|\lambda_2|, 2|\lambda_n|).$$

It can be proven that it is $x_\alpha(K_3) = x_\alpha(K_4) = 2$ (the upper bound is a corollary of theorem 3).

Call a function $f : V \to \mathbf{Z}_2^k$ (for some $k \geq 1$) a *proper representation* of $G$ if there is a $b_f \in \{0, 1\}$ such that $\forall u, v \in V(G), uv \in E(G) \Leftrightarrow f(u) \cdot f(v) = b_f$. The *dimension* of a proper representation $f$ is $\dim(f(V))$ in $\mathbf{Z}_2^k$.

The following bounds for $x_\omega$ are established:

**Theorem 4.** *If $G = (V, E)$ is a graph with a proper representation $f$, then $x_\omega(G) \leq \dim(f)$.*

**Theorem 5.** *For every graph $G = (V, E)$ it is $x_\omega(G) \leq |V|$*

**Theorem 6.** *Let $r = p^k$ for some prime $p \geq 3$ and $k \geq 1$. Then:*

$$r - 1 - \frac{r}{r+2} \geq x_\omega(K_r) \geq r - 1.$$

While the precise value of the number $x_\omega(K_3)$ is not known, it must lie between 1.7 (result of a computer search) and 2 (corollary of theorem 6).

A nice application of this theory is counting the maximum possible number $f_3(n)$ of vectors in $\{0, 1, 2\}^n$ such that the Hamming distance between every pair is even (i.e. they differ in an even number of components) and the maximum possible number $g_3(n)$ of vectors from the same space such that the Hamming distance between every pair is odd. It is easy to see that $f_3(n) = \alpha(K_3^n)$ and $g_3(n) = \omega(K_3^n)$. Applying the asymptotic for graph products, we obtain that $f_3$ grows like $2^n$ for large $n$ while $g_3$ surprisingly grows only linearly.

**Kjartan Høie**

kjartan.hoie@ii.uib.no

Presented paper by P. Hell, J. Nešetřil

## A reformulation of Hadwiger's conjecture

(Graphs and homomorphisms p104-106)

Hadwiger's conjecture is regarded by many as one of the most challenging open problems in graph theory and can be viewed as a generalization of the four colour theorem. It states that: *Any graph, G, with chromatic number n contains $K_n$ as a minor*

The conjecture is trivial for $n \leq 2$ and relatively easy for $n = 3$ and $n = 4$. For $n = 5$ and $n = 6$ it is equivalent to the the four colour theorem. But for $n \geq 7$ the conjecture remains open.

I will in my talk present Proposition 3.43 from [Hell, Nesetril: Graphs and homomorphisms p104-106]. (Also [Naserasr, Nigussie: On the reformulation of

Hadwiger's conjecture]). The proposition states that *Hadwiger's conjecture is equivalent to the following statement: Every proper minor closed class of graphs has a maximum.*

I will present necessary background and go through the proof in detail.

**Ondřej Suchý**

ondra@s.cz

Presented paper by Henning Fernau

## Roman domination: A Parameterized Perspective

(Lecture Notes in Computer Science, Volume 3831 / 2006)

We analyze *Roman domination* from a parameterized perspective. More specifically, we prove that this problem is $\mathcal{W}[2]$-complete for general graphs. However, parameterized algorithms are presented for graphs of bounded treewidth and for planar graphs. Moreover, it is shown that a parametric dual of *Roman domination* is in $\mathcal{FPT}$ .

Roman domination is one of the many variants of dominating set problems. It comes with a nice (hi)story: namely, it should reflect the idea of how to secure the Roman Empire by positioning the armies (legions) on the various parts of the Empire in a way that either (1) a specific region $r$ is also the location of at least one army or (2) one region $r'$ neighboring $r$ has two armies, so that $r'$ can afford sending off one army to the region $r$ (in case of an attack) without loosing self-defense capabilities.

**Formally.** *Let us first formally describe the problem.*

To this end, notice that we will use standard notions from graph theory. In whole talk we deal with simple undirected graphs. $N(v)$ is the open neighborhood of vertex $v$, and $N[v] = N(v) \cup v$ is the closed neighborhood. An instance of *Roman domination* (ROMAN) is given by a graph $G = (V, E)$, and the parameter, a positive integer $k$. The question is: Is there a Roman domination function R such that $R(V) := \sum_{x \in V} R(x) \leq k$? Here, a Roman domination function of a graph $G = (V, E)$ is a function $R : V \to \{0, 1, 2\}$ with $\forall v \in V : R(v) = 0 \Rightarrow \exists x \in N(v) : R(x) = 2$. $D_R = R^{-1}(\{1, 2\})$ is then the Roman domination set. The minimum of $R(V)$ over all valid Roman domination functions $R$ is also called the Roman domination number of a given graph.

**Parametrized complexity.** *In the following, we give the necessary background on parameterized complexity:*

A parameterized problem $P$ is a subset of $\Sigma^* \times N$, where $\Sigma$ is a fixed alphabet and $N$ is the set of all non-negative integers. Therefore, each instance of the

parameterized problem $P$ is a pair $(I, k)$, where the second component $k$ is called the parameter. The language $L(P)$ is the set of all YES-instances of $P$. We say that the parameterized problem $P$ is fixed-parameter tractable if there is an algorithm that decides whether an input $(I, k)$ is a member of $L(P)$ in time $f(k)|I|^c$, where $c$ is a fixed constant and $f(k)$ is a function independent of the overall input length $|I|$. The class of all fixed-parameter tractable problems is denoted by $\mathcal{FPT}$ .

There is also a hardness theory, most notably, the $\mathcal{W}[t]$ hierarchy, that complements fixed-parameter tractability:

$$\mathcal{FPT} = \mathcal{W}[0] \subseteq \mathcal{W}[1] \subseteq \mathcal{W}[2] \subseteq \ldots$$

It is commonly believed that this hierarchy is strict. Since only the second level $\mathcal{W}[2]$ will be of interest to us in this paper, we will only define that class below. We do this in the "Turing way".

A parameterized reduction is a function $r$ that, for some polynomial $p$ and some function $g$, is computable in time $O(g(k)p(|I|))$ and maps an instance $(I, k)$ of $P$ onto an instance $r(I, k) = (I', k')$ of $P'$ such that $(I, k)$ is a YES-instance of $P$ if and only if $(I', k')$ is a YES-instance of $P'$ and $k' \leq g(k)$. We also say that $P$ reduces to $P'$.

$\mathcal{W}[2]$ can be characterized by the following problem on Turing machines: An instance of *Short multi-tape nondeterministic Turing machine computation* (SMNTMC) is given by a multi-tape nondeterministic Turing machine $M$ (with two-way infinite tapes), an input string $x$, and the parameter, a positive integer $k$. The question is: Is there an accepting computation of $M$ on input $x$ that reaches a final accepting state in at most $k$ steps? More specifically, a parameterized problem is in $\mathcal{W}[2]$ iff it can be reduced with a parameterized reduction to *Short multi-tape nondeterministic Turing machine computation*.

**Lemma.** *Red-blue dominating set, restricted to bipartite graphs is $\mathcal{W}[2]$-hard.*

**Theorem 1.** *Roman domination is $\mathcal{W}[2]$-complete.*

**Theorem 2.** *Planar Roman domination can be solved in $O^*(3.3723^k)$ time, e.g. is in $\mathcal{FPT}$*

**Theorem 3.** *Minimum Roman domination, parameterized by the treewidth $tw(G)$ of the input graph $G$, can be solved in time $O(5^{tw(G)}|V(G)|)$.*

**Another problem.** *Parametric dual:*

Given a graph $G$ and a parameter $k_d$, is there a Roman domination function $R$ such that $|R^{-1}(1)| + 2|R^{-1}(0)| \geq kd$ ?

**Theorem 4.** *Our version of parametric dual of Roman domination allows for a problem kernel of size $(7/6)kd$, measured in terms of vertices. Hence, this problem is in $\mathcal{FPT}$ .*

**Louis Esperet**
esperet@labri.fr

# $(d,1)$-total labelling of sparse graphs

**Abstract**

The $(d,1)$-total number $\lambda_d^T(G)$ of a graph $G$ is the width of the smallest range of integers that suffices to label the vertices and the edges of $G$ so that no two adjacent vertices have the same color, no two incident edges have the same color and the distance between the color of a vertex and the color of any incident edge is at least $d$. This notion was introduced by Havet and Yu in [1]. In this paper, we study the $(d,1)$-total number of sparse graphs and prove that for any $0 < \varepsilon < \frac{1}{2}$, and any positive integer $d$, there exists a constant $C_{d,\varepsilon}$ such that for any $\varepsilon\Delta$-sparse graph $G$ with maximum degree $\Delta$, we have $\lambda_d^T(G) \leq \Delta + C_{d,\varepsilon}$.

## Introduction

In the channel assignment problem, we need to assign frequency bands to transmitters. If two transmitters are too close, interferences will occur if they attempt to transmit on close frequencies. In order to avoid this situation, the channels assigned must be sufficiently far. Moreover, if two transmitters are close but not too close, the channels assigned must still be different. Havet and Yu in [1] and [2] introduced the $(d,1)$-total labelling, defined as follows:

The $(d,1)$-total labelling of a graph $G = (V,E)$ is a function $c : V \cup E \to \mathrm{N}$ verifying:

$(i)$ $\forall (u,v) \in V^2 : uv \in E \Rightarrow c(u) \neq c(v)$

$(ii)$ $\forall (u,v,w) \in V^3 : uv \in E, uw \in E \Rightarrow c(uv) \neq c(uw)$

$(iii)$ $\forall (u,v) \in V^2 : uv \in E \Rightarrow |c(u) - c(uv)| \geq d$

The span of a $(d,1)$-total labelling is the maximum difference between two assigned integers. The $(d,1)$-*total number* of a graph $G$, denoted by $\lambda_d^T(G)$, is the minimum span of a $(d,1)$-total labelling of $G$. Notice that the $(1,1)$-total labelling is the traditional total coloring.

**Conjecture 1.** *Let $G$ be a graph with maximum degree $\Delta$, then $\lambda_d^T(G) \leq \min\{\Delta + 2d - 1, 2\Delta + d - 1\}$.*

Finally, the best known upper bound for general graphs is due to Esperet and Havet [3] who proved :

**Theorem 1.** *Let $G$ be a graph with maximum degree $\Delta$, then $\lambda_d^T(G) \leq \Delta + O(\log \Delta)$.*

In [4], Molloy and Reed proved that the total chromatic number of any graph with maximum degree $\Delta$ is at most $\Delta$ plus an absolute constant. Moreover, in [6], they gave a simpler proof of this result for sparse graphs. In this paper, we generalize their approach to the $(d,1)$-total number of sparse graphs.

A vertex $v$ is called *$\alpha$-sparse* iff $|E(N(v))| \leq \binom{\Delta}{2} - \alpha\Delta$. An $\alpha$-sparse graph is a graph in which all the vertices are $\alpha$-sparse.

Our main result is the following :

**Theorem 2.** *For any $0 < \varepsilon < \frac{1}{2}$, and any positive integer $d$, there exists a constant $C_{d,\varepsilon}$ such that for any $\varepsilon\Delta$-sparse graph $G$ with maximum degree $\Delta$, we have $\lambda_d^T(G) \leq \Delta + C_{d,\varepsilon}$.*

The proof of Theorem 2 is based on a probabilistic approach due to Molloy and Reed. It uses intensively concentration inequalities as well as Lovász Local Lemma. Moreover, we conjecture:

**Conjecture 2.** *For any positive integer $d$, there exists a constant $C_d$, such that for any graph $G$ with maximum degree $\Delta$, we have $\lambda_d^T(G) \leq \Delta + C_d$.*

In next section, we present the procedure used to prove Theorem 2. And then we analyse this procedure. In the following, we will need some probabilistic tools (see Appendix $A$ and [6] for more details).

### Proof of Theorem 2

Since $\lambda_d^T(G) \leq 2\Delta + d - 1$, if we prove that for some $\Delta_0(d, \varepsilon)$ and some $C_{d,\varepsilon}$, any $\varepsilon\Delta$-sparse graph $G$ of maximum degree $\Delta \geq \Delta_0$ verifies $\lambda_d^T(G) \leq \Delta + C_{d,\varepsilon}$, then Theorem 2 will be proved.

Let $\phi$ be a full or partial coloring of $G$. Any edge $e = uv$ such that $|\phi(u) - \phi(e)| < d$ or/and $|\phi(v) - \phi(e)| < d$ is called a *reject edge*. The graph $R$ induced by the reject edges is called the *reject graph*. It will be convenient for us to consider the *reject degree* of a vertex $v$, which is the number of edges $e = uv$ such that $|\phi(u) - \phi(e)| < d$. Observe that $deg_R(v)$ is at most the reject degree of $v$ plus $2d - 1$.

**Sketch of Proof** To prove Theorem 2, we apply the following steps :

*Step* 1. First, we will color the edges by Vizing's Theorem using the colors $\{1, \ldots, \Delta\}$.

*Step* 2. Then we will use the Naive Coloring Procedure to color the vertices with colors $\{1, \ldots, \Delta + 2d - 1\}$. This procedure creates reject edges. However, we can prove that after the procedure, the maximum degree of the reject graph $R$ is a constant $D_{d,\varepsilon}$ which does not depend on $\Delta$.

*Step* 3. Finally, we erase the color of the vertices of $R$ and recolor these vertices greedily with the colors $\{\Delta + 3d - 2, \ldots, \Delta + 3d - 1 + D_{d,\varepsilon}\}$. Taking $C_{d,\varepsilon} = D_{d,\varepsilon} + 3d - 2$, this proves that $\lambda_d^T(G) \leq \Delta + C_{d,\varepsilon}$.

We now present the Naive Coloring Procedure.

**The Naive Coloring Procedure**

For each vertex $v$, we maintain two lists of colors : $L_v$ and $F_v$. $L_v$ is the set of colors which do not appear in the neighborhood of $v$. Initially, $L_v = \{1, \ldots, \Delta + 2d - 1\}$. After iteration $I$ (specified later), $F_v$ will be a set of forbidden colors. Until iteration $I$, $F_v = \emptyset$.

During the Naive Coloring procedure, we will perform $i^*$ (specified later) iterations of the following procedure :

*Step* 1. Assign to each uncolored vertex $v$ a color choosen uniformly at random in $L_v$.

*Step* 2. Uncolor any vertex which receives the same color as a neighbor in this iteration.

*Step* 3.

**Iteration** $i \leq I$. Let $v$ be a vertex having more than $T$ (specified later) neighbors $u$ which are assigned a color $c(u)$ such that $|c(uv) - c(u)| < d$ in this iteration. For any $v$, we uncolor all such neighbors.

**Iteration** $i > I$.

(a) Uncolor any vertex $v$ which receives a color from $F_v$ in this iteration.

(b) Let $v$ be a vertex having more than one neighbor $u$ which is assigned a color such that $|c(uv) - c(u)| < d$ in this iteration. For any $v$, we uncolor all such neighbor.

(c) Let $v$ be a vertex having at least one neighbor $u$ such that $|c(uv) - c(u)| < d$ in this iteration. For any $v$, we place $\{c(vw) - d + 1, \ldots, c(vw), \ldots, c(vw) + d - 1\}$ in $F_w$ for every $w \in N(v)$.

*Step* 4. For any vertex $v$ which retained its color (*i.e.* which was not uncolored during a previous step), we remove $c(v)$ from $L_u$ for any $u \in N(v)$.

After $i^*$ iterations of this procedure, we have a partial coloring of $G$. We then complete this coloring in order to obtain a reject graph $R$ with a bounded maximum degree which does not depend on $\Delta$ (Section 4.3).

### Analysis of the procedure

#### The first iteration

Let $\zeta = \frac{\varepsilon}{2e^3}$. In this subsection, we prove that:

**Claim 1.** *The first iteration produces a partial coloring with bounded reject degree for which every vertex has at least* $\frac{\zeta}{2}\Delta$ *repeated colors in its neighborhood.*

We recall that $\mathcal{C} = \Delta + 2d - 1$ is the initial size of each color list $L_v$. Let $A_v$ be the number of colors $c$ such that at least two neighbors of $v$ receive the color $c$ and all such vertices retain their color during Step 2. Let $B_v$ be the number of neighbors of $v$ which are uncolored at Step 3. Notice that vertices are uncolored at Step 3 regardless of what happened at Step 2. Let $X_v$ be the event that "$A_v < \zeta\Delta$". Let $Y_v$ be the event that "$B_v \geq \frac{\zeta}{2}\Delta$". If no type $X$ event occurs, every vertex has at least $\zeta\Delta$ repeated colors in its neighborhood at the end of Step 2. If no type $Y$ event occurs, less than $\frac{\zeta}{2}\Delta$ vertices are uncolored in each neighborhood. As a consequence, if we show that with positive probability, no type $X$ or $Y$ event occurs, Claim 1. will be proved.

**Claim 2.** $\mathbf{Pr}(X_v) < e^{-\alpha \log^2 \Delta}$*, for a particular constant* $\alpha > 0$*.*

**Claim 3.** $\mathbf{Pr}(Y_v) < e^{-\beta\Delta}$*, for a particular constant* $\beta > 0$*.*

We now use Lovász Local Lemma to prove Claim 1. Each event $X_v$ only depends on the colors assigned to the vertices at distance at most 2 from $v$, and each event $Y_v$ depends on the colors assigned to the vertices at distance at most 3 from $v$. Hence, each event is mutually independent of all but at most $2\Delta^6$ other events. For $\Delta$ sufficiently large, $\mathbf{Pr}(X_v) < \frac{1}{8\Delta^6}$ and $\mathbf{Pr}(Y_v) < \frac{1}{8\Delta^6}$. Using Lovász Local Lemma, this proves that with positive probability no type $X$ or $Y$ event happens. Thus with positive probability, the first iteration produces a partial coloring with bounded reject degree, such that each vertex has at least $\frac{\zeta\Delta}{2}$ repeated colors in its neighborhood.

**The next iterations**

Let $d_i = \left(1 - \frac{1}{4}e^{-\frac{2}{\zeta}}\right)^i \Delta$ and $f_i = \frac{4(2d-1)}{\zeta} \sum_{j=I+1}^{i-1} D_j$. Let $i^*$ be the smallest integer $i$ such that $d_i \leq \sqrt{\Delta}$. Observe that for any $i \leq i^*$, we have $d_i \geq (1 - \frac{1}{4}e^{-\frac{2}{\zeta}})\sqrt{\Delta}$.

**Claim 4.** *At the end of each iteration* $1 \leq i \leq i^*$*, with positive probability every vertex has at most* $d_i$ *uncolored neighbors, and each list* $F_v$ *has size at most* $f_i$*.*

*Proof.* By induction on $i$. ♢

**The final phase**

At this point, we have a partial coloring such that:

- each vertex $v$ has at most $\sqrt{\Delta}$ uncolored neighbors;
- the reject degree of each vertex is at most $IT + 1$;
- each vertex has a list of at least $\frac{\zeta\Delta}{2}$ available colors.

It will be more convenient to use lists of equal sizes. So we arbitrarily delete colors from each list, so that for every uncolored vertex $v$, we have $|L_v| = \frac{\zeta\Delta}{2}$. For each uncolored vertex, we choose a subset of colors from $L_v$ which will be *candidates* for $v$ and we prove that with positive probability, there exists a candidate for each uncolored vertex, such that we can complete our partial coloring of $G$.

A candidate $a$ for $v$ is said to be *good* if:

**Condition 1** for every neighbor $u$ of $v$, $a$ is not candidate for $u$;

**Condition 2** for every neighbor $u$ of $v$, and every neighbor $w$ of $u$, there is no candidate $b$ of $w$ such that $|c(uv) - a| < d$ and $|c(uw) - b| < d$.

If we find a good candidate for every uncolored vertex, Condition 1 ensures that the vertex coloring obtained is proper, and Condition 2 ensures that no reject degree increases by more than one.

**Claim 5.** *There exists a set of candidates* $S_v$ *for each uncolored vertex* $v$*, such*

*that each set contains at least one good candidate.*

We obtain a coloring of $G$ with maximum reject degree at most $IT+2$. So the reject graph $R$ obtained has maximum degree at most $IT+2d+1$. We uncolor the vertices of $R$ and recolor them greedily with the colors $\{\Delta+3d-2,\ldots,\Delta+IT+5d\}$. This final coloring is a $(d,1)$-total labelling of $G$. Since $I$ and $T$ are independant of $\Delta$, we proved that $\lambda_d^T(G)\leq\Delta+C_{d,\varepsilon}$.

**Remark.** *By looking carefully at each inequality during the procedure, we can replace $\Delta+C_{d,\varepsilon}$ by $\Delta+C_\varepsilon d\log d$, where $C_\varepsilon$ is a constant that does not depend on $d$.*

### Further work

Theorem 2 can be transformed into a randomized algorithm, using a powerful technique introduced by Beck [5] and extended to a wide range of applications of the symmetric form of the Local Lemma by Molloy and Reed [7].

### References


[1] F. Havet and M.-L. Yu: $(d,1)$-total labelling of graphs, Technical Report 4650, INRIA, 2002.

[2] F. Havet: $(d,1)$-total labelling of graphs, Workshop Graphs and Algorithms, Dijon (FRANCE), 2003.

[3] L. Esperet: Coloration $(d,1)$-totale et méthode probabiliste, Master's thesis, ENS Lyon and INRIA Sophia Antipolis, 2003.

[4] M. Molloy and B. Reed: A bound on the total chromatic number, Combinatorica, 1998.

[5] J. Beck: An Algorithmic Approach to the Lovász Local Lemma I, Random Structures & Algorithms, 1991.

[6] M. Molloy and B. Reed: Graph coloring and the probabilistic method, Vol. 23 of *Algorithms and combinatorics*, Springer-Verlag, 2002.

[7] M. Molloy and B. Reed: Further Algorithmic Aspects of the Local Lemma, Proceedings of the 30th ACM Symposium on Theory of Computing, 1998.



**Jiří Fink**

jirka.fink@matfyz.cz


Presented paper by Stephan Brandt

**On the structure of graphs with bounded clique number**

(Combinatorica, Springer-Verlag)

In a talk, a structural result for maximal $K_r$-free graphs is proven, which proides a simple proof of thte Andrasfai-Erdos-Sos Theorem, saying that every $K_r$-free graph with minimum degree $\delta>(1-\frac{1}{r-4/3})n$ is $(r-1)$-colorable.

The crucial substructure is that of a 5-wheel-like graph $W_{r,k}$. This is a graph consisting of two clique $Q_1$, $Q_2$ of order $r-2$, which intersect in exactly $k$ vertices, where $0 \leq k < r-2$, together with a vertex v, adjacent to all vertices of $Q_1$ and $Q_2$ and an edge $w_1w_2$, where $w_1$ is adjacent to the vertices of $Q_1$ and $w_2$ is adjacent to the vertices of $Q_2$.

**Proposition.** *Let $G$ be a maximal $K_r$-free graph. Then $G$ is either complete $(r-1)$-partite, or $G$ contains a 5-wheel-like subgraph.*

**Proposition.** *Let $G$ be a $K_r$-free graph containing a 5-wheel-like subgraph $W_{r,k}$ with top $v$ and bottom $w_1w_2$. If $\delta(G) > \frac{2r+k-4}{2r+k-1}n$ then $k < r-3$ and $G$ contains $W_{r,k+1}$, having top $v$ and bottom $w_1w_2$.*


**Arnaud Labourel**

labourel@labri.fr

Presented paper by Nicolas Bonichon, Cyril Gavoille and Arnaud Labourel


## Short Labels by Traversal and Jumping


### Abstract

In this paper, we propose an efficient implicit representation of caterpillars and binary trees with $n$ vertices. Our schemes, called *Traversal & Jumping*, assign to vertices of the tree distinct labels of $\log_2 n + O(1)$ bits, and support constant time adjacency queries between any two vertices by using only their labels. Moreover, all the labels can be constructed in $O(n)$ time.


### Introduction

**Related works** The two basic ways of representing a graph are adjacency matrices and adjacency lists. The latter representation is space efficient for sparse graphs, but adjacency queries require searching in the list, whereas matrices allow fast queries to the price of a super-linear space. Another technique, called *implicit representation* or *adjacency labeling scheme*, consists in assigning unique labels to each vertex such that adjacency queries can be computed alone from the labels of the two involved vertices without any extra information source. The goal is to minimize the maximum length of a label associated with a vertex while keeping fast adjacency queries.

Adjacency labeling schemes, introduced by [Breuer66,BF67], have been investigated by [KNR88,KNR92]. They construct for several families of graphs adjacency labeling schemes with $O(\log n)$-bit labels like labels of size $2\lceil \log n \rceil$ bits for tree. This result has been improved in a non trivial way by [AKM01] to $1.5 \log n + O(\log \log n)$ bits, and more recently to $\log n + O(\log^* n)$ bits

[AR02b], leaving open the question of whether trees enjoy a labeling scheme with $\log n + O(1)$ bit labels.

In this paper we present adjacency labeling schemes for caterpillars (i.e., a tree whose nonleaf vertices induce a path), and binary trees with $n$ vertices. Both schemes assign distinct labels of $\log n + O(1)$ bits, and support constant time adjacency queries. Moreover, all the labels can be constructed in $O(n)$ time. We observe that the recursive scheme of [AR02b] for general trees does not simplify for caterpillars or binary trees. The worst-case label length remains $\log n + O(\log^* n)$ and the adjacency query time $\Omega(\log^* n)$.

**Outline of techniques**

Roughly speaking, the Traversal & Jumping technique consists in:

S electing a suitable traversal of the tree (or of the graph);

A ssociating with each vertex $x$ some information $C(x)$;

P erforming the traversal and assign the labels with increasing but non necessarily consecutive numbers to the vertices.

Intuitively, the adjacency test between $x$ and $y$ is done on the basis of $C(x)$ and $C(y)$. Actually, the jumps achieved in Step 3 are done by selecting an interval associated with each vertex in which its label must be. It is important to note that the intervals are ordered in the same way as the corresponding vertices in the traversal. Moreover, all vertex intervals must be disjoint. The position of the label of $x$ in its interval is tuned in order to encode $C(x)$ in the label in a self-extracting way. In general, the information $C(x)$ determines the intervals length of all the neighbours of $x$ which are after in the traversal. The main difficulty is to design the minimal information $C(x)$ and to tune the jumps, i.e., the interval length. The maximum label length is simply determined by the value of the last label assigned during the traversal.

**Preliminaries**

We assume a RAM model of computation with $\Omega(\log n)$-bit words. In this model, standard arithmetic operations on words of $O(\log n)$ bits can be done in constant time. Given a binary string $A$, we denote by $|A|$ its length, and for a binary string $B$, $A \circ B$ denotes the concatenation of $A$ followed by $B$. Given an $x \in \mathrm{N}$, we denote by $\mathrm{bin}(x)$ its standard binary representation. A *code* is a set of words, and a code is *suffix-free* if no words of the code is the ending of another one. A basic property of suffix-free codes it that they can be composed, by the concatenation of a fixed number of fields, to form new suffix-free codes. A simple suffix-free code is defined by $\mathrm{code}_0(x) = 1 \circ 0^x$, where $0^x$ is the binary string composed of $x$ zeros. This code extends to more succinct codes defined recursively by $\mathrm{code}_{i+1}(x) = \mathrm{bin}(x) \circ \mathrm{code}_i(|\mathrm{bin}(x)| - 1)$ for every $i \geq 0$. It is easy to check that, for every $i \geq 0$, $\mathrm{code}_i$ is suffix-free. E.g., $\mathrm{code}_0(5) = 100000$, $\mathrm{code}_1(5) = 101\,100$, and $\mathrm{code}_2(5) = 101\,10\,10$. If a word $w$ has $\mathrm{code}_i(x)$ as suffix, then $x$ can be extracted from $w$ in $O(i)$ time. In this paper, we will essentially use $\mathrm{code}_i$ for $i \in \{0, 1, 2\}$. We check that for every $x \in \mathrm{N}$, $|\mathrm{code}_0(x)| = x + 1$,

$|\text{code}_1(x)| = 2\lfloor \lg x \rfloor + 2$, and $|\text{code}_2(x)| = \lfloor \lg x \rfloor + 2\lfloor \lg \lfloor \lg x \rfloor \rfloor + 3$. We can find a word having the wanted suffix by attributing to the vertex a interval of values of enough lenght.

**Claim.** *Let $w$ be a word, and $z$ an integer. One can compute in constant time an integer $x \in [z, z + 2^{|w|})$ such that $w$ is a suffix of* $\text{bin}(x)$.

### Caterpillars

A leaf is a vertex of degree one, and an inner vertex is a nonleaf vertex. A tree is a *caterpillar* if the subgraph induced by its inner vertices is a path.

**Theorem 1.** *The family of caterpillars with $n$ vertices enjoys an adjacency labeling scheme with labels of length at most $\lceil \log n \rceil + 6$ bits, supporting constant time adjacency query. Moreover, all the labels can be constructed in $O(n)$ time.*

Consider a caterpillar $G$ of $n$ vertices. We denote by $X = \{x_1, \ldots, x_k\}$ the inner vertices of $G$ (ordered along the path). For every $i$, let $Y_i = \{y_{i,1}, \ldots, y_{i,d_i}\}$ be the set of leaves attached to $x_i$, with $d_i = 0$ if $Y_i = \emptyset$. The traversal used in our scheme is a prefix traversal of the caterpillar rooted at $x_1$ where the vertices of $Y_i$ are traversed before the vertex $x_{i+1}$. According to this traversal, the inner vertex $x_i$ stores information necessary to determine the adjacency with the vertices of $Y_i \cup \{x_{i+1}\}$. The leaves do not store any specific information in their label. With each inner vertex $x_i$, we associate an interval $I(x_i)$ of length $p_i$, for some suitable integer $p_i$, in which its label $\ell(x_i)$ must be. The integer $p_i$ must be large enough for $x_i$ to store $p_{i+1}$. To achieve that property, we impose that $\forall i, p_i$ is a power of two ($p_i = 2^{t_i}$). So $p_i$ need only to store $t_{i+1} = \log p_{i+1}$. Moreover, we impose that $p_i$ is greater than $d_i$ in order to store the leaf of $x_i$ in an interval of size $p_i$. The information encoded by $x_i$ is the ordered pair $(t_i, t_{i+1})$. To encode this information, we propose the following suffix-free code:

$$C(x_i) \;=\; \text{code}_0(t_i - |\text{code}_1(t_{i+1})|) \circ \text{code}_1(t_{i+1}) \;.$$

We place the leafs of $x_i$ in a interval $I'(x_i)$ of size $p_i = 2^{t_i}$ placed consecutively to $\ell(x_i)$. Then, we put the interval $I(x_{i+1})$ assigned to $x_{i+1}$ right after the interval.

Now, we must define the adjacency function between $x$ and $y$. Without loss of generality, we can assume that $\ell(x) < \ell(y)$ and that $x = x_i$ is an internal node. $x$ adjacent to $y$ if and only if $\ell(y) \in (\ell(x_i), \ell(x_i) + p_i + p_{i+1}]$. Clearly, this test can be computed in constant time from the labels because $x_i$ store $t_i$ and $t_{i+1}$ using $code_0$ and $code_1$ and so can compute $p_i = 2^{t_i}$ and $p_{i+1} = 2^{t_{i+1}}$.

It remains us to show that the maximum value of the labels are in $O(n)$. Due to the lack of the place, we can not give a complete proof. The important fact used in the proof is that, the vertex $x_i$ encode $t_{i+1} = \log p_{i+1}$ and not $p_{i+1}$ using $code_1$ with $2|bin(t_{i+1})|$ bits. So, $I(x_i)$ is of size $2^{2|bin(t_{i+1})|} = O(\log^2 p_i)$ and after some calculation, we show that the greater label have value in $O(n)$.

### Binary trees

A tree is binary if it is rooted and each inner vertex has at most two children.

**Theorem 2.** *The family of binary trees with $n$ vertices enjoys an adjacency labeling scheme with labels of length at most $\log n + O(1)$ bits, supporting constant time adjacency query. Moreover, all the labels can be constructed in $O(n)$ time.*

*Proof.* We use the same method as for caterpillar. We choose the prefix traversal in which the children are traversed by growing weight, i.e., the size of the subtree rooted at the child. Each inner vertex stores the size of the interval of its both sons and the distance in labels to the interval of its right son. These informations are enough to compute adjacency. By using good encoding and approximation of information, we can obtain labels of length $\log n + O(1)$ bits. Due to the lack of space, we cannot give a complete proof of this theorem in this extended abstract. The reader may refer to [BCA06] for a complete proof. ♢

### Conclusion

The unsolved *implicit graph representation conjecture* of [KNR88,KNR92] asks whether every hereditary family of graphs with $2^{O(n \log n)}$ labeled graphs of $n$ vertices enjoys a $O(\log n)$-bit adjacency labeling scheme. This is motivated by the fact that every family with at least $2^{cn \log n}$ labeled graphs of $n$ vertices requires adjacency labels of at least $c \log n$ bits.

Our schemes suggest that, at least for trees, labels of $\log n + O(1)$ bits may be possible. Therefore, we propose to prove or to disprove the following:

*Every hereditary family of graphs with at most $n!2^{O(n)} = 2^{n \log n + O(n)}$ labeled graphs of $n$ vertices enjoys an adjacency labeling scheme with labels of $\log n + O(1)$ bits.*

We observe that several well-known families of graphs are concerned by this proposition: trees, planar graphs, bounded treewidth graphs, graphs of bounded genus, graphs excluding a fixed minor (cf. [NRTW05] for counting such graphs). Proving the latter conjecture appears to be hard, e.g., the best upper bound for planar graphs is only $3 \log n + O(\log^* n)$.

**Michal Koucký**
koucky@math.cas.cz

## A Brief Introduction to Kolmogorov Complexity

### Introduction

The set of all finite binary strings is denoted by $\{0,1\}^*$. For $x \in \{0,1\}^*$, $|x|$ denotes the length of $x$.


eulerian, Δ-free
(Ehrenfeucht, Haase, Harju, Rozenberg)
3-colorability
both P, S(P) are polynomial
both NPC
polynomial
is NPC


Algorithm and Lower Bound
General Facts
Computing Competitive Ratio, Lower Bound
Algorithm
Analysis
Suppose that algorithm fails, i.e., there is job $j$, $p_j$
Let $J = S_1 \cup j$. Ordered as they appeared on inp
Let $T_1 \dots T_m$ be the end times of last $m$ jobs
Then $\sum_{i=1} p_i = \sum_{i=1}^{m} s_i T_{m-i+1}$ is sufficient.
Let $I'$ be
Then $C_{max}$
We have: $T_i \geq C_{max}(I_m$
So it suffices $\sum_{i=1} p_i \geq R \sum$
Let $q_1, \dots, q_m$ be sizes of last $m$
sum of remaining jobs. Let $O_i = C_{max}$
satisfy (4) up to some scale. Thus the abo
Ebenlendr, Javer, Sgall
Preemptive Onlin

*Kolmogorov complexity* tries to answer the fundamental question: "What is a random object?" Consider which of the following (decimal) strings seem to be random?

33333333333

31415926535

84354279521

Most people would rule out the first one to be random, and they could agree that the remaining two are random. Indeed, most statisticians would agree that the latter two are random as they pass essentially all possible statistical tests. Yet, the second sequence consists of the first eleven digits of $\pi$. The third one is taken really at random.

From the perspective of probability, all three strings have the same probability of being chosen when we take a string of eleven digits fully at random namely, each of them has probability $10^{-11}$. Hence, they all are equally likely to be obtained by a random process. So probability does not really explain the intuitive notion of *randomness*.

Imagine that we would extend our strings to one million digits. Then the first string would became a million times the digit three, the second one would be the first million digits of $\pi$ and the last one would be 84354279521.... In fact it would take us thousand of pages to describe the last one. There is no pattern in it. It is really random.

Hence, the notion of randomness is connected to patterns in strings and to a way how we can describe them. The first two strings in our example have very short descriptions (few words) whereas the last string has very long description as it lacks any regularity. The longer the necessary description of a string the more randomness is in the string. This intuition leads to the following definition of *Kolmogorov complexity* of a string $x \in \{0,1\}^*$: the Kolmogorov complexity of $x$ is the length of the shortest description of $x$. Of course the length of the description depends on the language we use for our description—we can use Czech or French or English....

We make it formal as follows. Let $\phi : \{0,1\}^* \to \{0,1\}^*$ be a *partial recursive function*. (A partial recursive function is any function $f$ for which there is a program that takes an input $y$ and produces output $f(y)$. $f(y)$ may not be defined for some $y$ and the program may not halt on such $y$'s or to produce any output.)

**Definition.** For a string $x \in \{0,1\}^*$, the Kolmogorov complexity of $x$ with respect to $\phi$ is

$$C_\phi(x) = \min\{|p|,\ p \in \{0,1\}^* \ \&\ \phi(p) = x\}.$$

Let us consider several examples. If $\phi_1$ is the identity function $\phi_1(x) = x$ then $C_{\phi_1}(x) = |x|$. If $\phi_2(0) = 10111001011110011010110111$ and $\phi_2(1x) = x$ then $C_{\phi_2}(10111001011110011010110111) = 1$ and $C_{\phi_1}(x) = |x| + 1$ for all other strings $x$. So Kolmogorov complexity depends a lot on the chosen *descriptive language* $\phi$. Luckily, the following Invariance Theorem brings some order into this chaos.

**Theorem 1.** *There exists a partial recursive function $U$ so that for any other partial recursive function $\phi$ there is a constant $c > 0$ such that*

$$C_U(x) \leq C_\phi(x) + c$$

*for all strings $x$.*

A machine $U$ satisfying the preceding theorem is in some sense minimal among all machines, and we will call it *universal*.

*Proof.* The proof is quite simple. Let $\phi_0, \phi_1, \phi_2, \ldots$ be an enumeration of all partial recursive functions. (Every p.r.f. can be associated with a program that computes it and that program can be uniquely mapped to a (possibly huge) integer.) Let $\langle x, y\rangle : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ be some simple to compute one-to-one mapping, e.g., $\langle x, y\rangle = 0^{|x|}1xy$. Then $U$ is defined as follows: On input $w$, decode $w$ into $i$ and $p$ such that $w = \langle i, p\rangle$ and run $\phi_i$ on input $p$. If $\phi_i(p)$ stops then output whatever $\phi_i$ had output.

It is easy to verify that such a $U$ is partial recursive and that it satisfies our theorem. ♢

So from now on we fix some machine $U$ which satisfies the Invariance Theorem and we will consider the Kolmogorov complexity of $x$ to be $C_U(x)$. We will write $C(x)$ instead of $C_U(x)$ from now on.

We are ready to define a random string.

**Definition.** A string $x$ is Kolmogorov random if $C(x) \geq |x|$.

This definition is not void as there is a Kolmogorov random string of every length: there are $2^n - 1$ descriptions of length less than $n$ but there are $2^n$ strings of length $n$. Let us consider couple of strings and their Kolmogorov complexity:

1. $0^n$ has Kolmogorov complexity $\log n + O(1)$ as we only need to specify the integer $n$ and a short program that will reconstruct $0^n$ from $n$.
2. The sequence of the first $n$ digits of $\pi$ has Kolmogorov complexity $\log n + O(1)$. The reason is the same as above. (Just download a program for $\pi$ from Internet.)
3. There is a Kolmogorov random string $x$ with $C(x) \geq n$. See above.
4. What about some string of complexity about $\sqrt{n}$? Sure, there is one. Consider $y \in \{0,1\}^{\sqrt{n}}$ that is Kolmogorov random. Then $x = y0^{n-\sqrt{n}}$ has Kolmogorov complexity about $\sqrt{n} + O(1)$. Why? If it would have a

description much shorter than $|y|$ we could describe $y$ using such description: first produce $x$ and then output only its first $\sqrt{|x|}$ bits. On the other hand, description of $y$ is a good description of $x$: produce $y$ and then append $|y|^2 - |y|$ zeroes. So there are strings of essentially all possible complexities.

5. Every string of length $n$ has Kolmogorov complexity at most $n + O(1)$. Why?
6. How many ones and zeroes has a Kolmogorov random string of length $n$? About a half, but exactly? There are $\binom{n}{n/2}$ strings which have the same number of ones and zeroes. Call the set of such strings $S^n_{n/2}$. These string can be easily identified and enumerated by a program. Hence, given $n$ and $i$, we can find the $i$-th string $s_i$ in $S^n_{n/2}$ and output it. Thus, $C(s_i)$ is at most the size of a description of $i$, plus the size of a description of $n$, plus some constant for the program described above. $|S^n_{n/2}| < \binom{n}{n/2} < c2^n/\sqrt{n}$ for some constant $c$, hence to specify $i$ we only need at most $n - 1/2 \log n + \log c$ bits. $n$ does not really have to be specified as it can be deduced from the length of the description of $i$. Thus, all strings $s_i \in S^n_{n/2}$ have Kolmogorov complexity at most $n - 1/2 \log n + O(1)$. It turns out that Kolmogorov random strings of length $n$ have $n/2 \pm c\sqrt{n}$ zeroes. By Chernoff bound there are relatively few strings of length $n$ that have the number of ones farther from $n/2$ than $c\sqrt{n}$, and by extending the argument above there are also relatively few strings that have the number of ones closer to $n/2$ than $c\sqrt{n}$. Since these strings are few and easy to identify, they have small Kolmogorov complexity. (In fact, the deviation from $n/2$ in the number of ones have to be Kolmogorov random by itself.) The following proposition generalizes this argument.

**Proposition.** *Let $A$ be a recursive (recursively enumerable) set and $n$ be an integer. Let $A_n = A \cap \{0,1\}^n$. For all strings $x$ in $A_n$ it holds, $C(x) \leq \log |A_n| + 2 \log n + O(1)$.*

Often the term $2 \log n$ can be omitted as $n$ can be deduced from the length of the description.

*Proof.* The proof is straightforward. Since $A$ is recursive (recursively enumerable) we can design a program that given $i$ and $n$ prints the $i$-th string of $A_n$ in some enumeration. Hence, all strings in $A_n$ can be described by giving $i$, $n$ and the program for enumerating $A$. The description of $i$, $n$ and the program has to be concatenated into one string in such a way that $i$, $n$ and the program can be recovered from the string. One can use the pairing function from the proof of Theorem 1 for doing that. The factor two in the logarithmic term comes from there. ♢

It is useful to note that the set of strings that are *not* Kolmogorov random is recursively enumerable—given a string $x$ we can run all programs of length shorter than $x$ in parallel and see if any one of them ever outputs $x$. If that

happens we accept $x$.

This brings us to the fact that the number of strings of length $n$ that are Kolmogorov random is Kolmogorov random by itself. It is about $2^n/c$ for some constant $c > 1$. If that were not the case, we could find all strings of length $n$ that are not Kolmogorov random, and then print the first one which should be random. Program for such a computation would only need to know the number of non-random strings of length $n$. The number of non-random strings is $2^n$ minus the number of random strings, i.e., we can easily compute one from the other one. Since the above program prints out a Kolmogorov random string, both the numbers of random and non-random strings must require close to $n$ bits to specify. Hence, they are both about $2^n/c$.

**Proposition.** *It is uncomputable (undecidable) whether a string is Kolmogorov random.*

We have seen that non-random strings are recursively enumerable. This proposition thus implies that Kolmogorov random strings are not recursively enumerable as otherwise we could decide about a string whether it is Kolmogorov random or not.

*Proof.* We give two proofs. The first one is very simple, the second one is more complex but it shows that deciding Kolmogorov randomness is as hard as deciding the Halting Problem.

1. Assume we can decide whether a string is Kolmogorov random by some program $P$. We can then specify the lexicographically first Kolmogorov random string of length $n$ using $\log n + O(1)$ bits: run program $P$ on all strings of length $n$ in the lexicographical order until you find a string that is Kolmogorov random; print out the Kolmogorov random string. This only requires to specify the program $P$ and $n$. Hence, no such $P$ can exist.

2. Define the Halting Problem by $H = \{x;\ \text{program } x \text{ halts on the input } 0\}$. Assume we can decide which strings are Kolmogorov random by some program $P$. We can then decide for any string $x$ whether the program $x$ halts on the input 0 or not as follows: Let $n = |x|$. Using $P$ decide for each string $y$ of length $2n$ whether it is Kolmogorov random or not. For each string $y$ that is not random find some program $p_y$ of length less than $n$ that prints it out. Let $t_y$ be the number of steps that it takes to $p_y$ to output $y$. Set $t_x = \max_y t_y$. Run $x$ on the input 0 for $t_x$ steps and if it accepts within $t_x$ steps then output $x \in H$ otherwise output $x \notin H$.

The reason why the above program would decide $H$ correctly is that if $x \in H$ but the running time of $x$ on the input 0 is more than $t_x$ then the actual running time of $x$ can be used as an upper bound for the running time of all $p_y$'s. As the running time of $x$ can be specified using $n + O(1)$ bits (namely by specifying $x$) we could specify all non-random strings of length $2n$ using only $n + 2\log n + O(1)$ bits. (Run all programs of length less than $2n$ for $t_x$ steps and see what they output.) Hence, we could describe the lexicographically first

Kolmogorov random string of length $2n$ using only $n+2\log n+O(1)$ bits. Thus the running time of $x$ must be smaller than $t_x$.

Since our program could correctly decide the Halting Problem, Kolmogorov randomness of strings must be undecidable. ♢

## Applications

We give here several applications of Kolmogorov complexity.

### Graph labelings

We start with an example related to the talk of Arnaud Labourel on graph labelings. For a (finite) class of graphs $\mathcal{G}$ a *labeling scheme of label length* $\ell$ is a function $A : \{0,1\}^\ell \times \{0,1\}^\ell \to \{0,1\}$ together with a labeling $l_G : V(G) \to \{0,1\}^\ell$ of every graph $G \in \mathcal{G}$ so that for all $x, y \in V(G)$, $(x,y) \in E(G)$ iff $A(l(x), l(y)) = 1$. We have already seen in the talk of Arnaud that the class of all the trees on $n$ vertices has labeling scheme with labels of length $\log n + O(1)$. The natural question is how large labels are needed to label the class of all the graphs on $n$ vertices. We claim that this length is $n/2 + O(\log n)$.

First, we show that labels of length $n/2 + \log n$ are sufficient. This is due to Jiří Sgall. Each vertex is going to be labeled by its vertex number plus a bit-vector of length $n/2$ which specifies to which of the next $n/2$ vertices under a cyclic ordering of vertices the vertex is connected. Given two vertex labels at least one of the labels contains the required adjacency information.

Using Kolmogorov complexity we want to show that $n/2$ bits are needed. First notice that by exhaustive search we can actually find the best labeling scheme for graphs on $n$ vertices. In fact we can write a program that will find it. This program will produce the function $A$. Each graph $G$ on $n$ vertices can be fully described by listing labels of its vertices $l_G(1), l_G(2), \ldots, l_G(n)$ in the optimal labeling scheme. Such a description requires $\ell n$ bits. Hence, every graph can be described by $\ell n + 2\log n + O(1)$ bits, by providing the vertex labels, $n$ and the program to compute $A$. On the other hand, a graph on $n$ vertices may contain $\binom{n}{2} = n(n-1)/2$ different edges. Hence, there are at least $2^{n^2/2-n}$ different graphs on $n$ vertices and each of them is uniquely described by a description of length $\ell n + 2\log n + O(1)$. Thus $\ell n + 2\log n + O(1) \geq n^2/2 - n$, i.e., $\ell \geq n/2 - 2$, for $n$ large enough. Thus a labeling scheme for graphs on $n$ vertices requires labels of length about $n/2$.

### Prime Number Theorem

We provide another application of Kolmogorov complexity to number theory. Let $p_i$ denote the $i$-th prime number. We will show the following theorem:

**Theorem 2.** *There is a constant $c$ such that for infinitely many $i$, $p_i < c \cdot i \cdot \log^2 i$.*

This theorem is a weak version of the usual Prime Number Theorem that $p_i / i \ln i \to 1$ as $i \to \infty$.

*Proof.* For a positive integer $n$ let $1\widehat{n}$ be its binary representation with $\widehat{n} \in \{0,1\}^*$. Clearly $2^{|\widehat{n}|} \leq n < 2^{|\widehat{n}|+1}$. Fix a large enough integer $x$ with Kolmogorov random $\widehat{x}$. We will make several observations regarding $x$.

1. $x = p^e y$, for some $p, e$ and $y$, where $p$ is a prime and $p^e > \log x / \log\log x$. If all maximal prime-power factors of $x$ were at most $\log x / \log\log x$, then $x \leq (\log x / \log\log x)! < \log x^{\log x / \log\log x} = x$.

2. $e = 1$. Note, $2^{e|\widehat{p}|} \cdot 2^{|\widehat{y}|} < p^e y = x$. Hence,

$$e|\widehat{p}| + |\widehat{y}| \leq |\widehat{x}| \leq C(\widehat{x}).$$

At the same time, $x$ can be specified by giving $e$, $p$, and $y$. Hence, $\widehat{x}$ can be given by some encoding of $\widehat{e}$, $\widehat{p}$ and $\widehat{y}$ into one binary string so that we would be able to tell apart all three of them. The pairing function used in the proof of Theorem 1 is too inefficient for our purposes. We can use the following pairing function: $\langle u, v\rangle = l_u 01 uv$, where $l_u$ is the binary representation of $|u|$ in which each digit is doubled. Thus $|\langle u, v\rangle| \leq |u| + |v| + 2\log|u| + 2$. Using this pairing function we can describe $\widehat{e}$, $\widehat{p}$ and $\widehat{y}$ to obtain:

$$C(\widehat{x}) \leq |\widehat{e}| + |\widehat{p}| + |\widehat{y}| + 2\log|\widehat{e}| + 2\log|\widehat{p}| + O(1).$$

But this implies that $e = 1$. (If $e > 1$ is small then $p$ must be large and hence $(e-1)|\widehat{p}|$ outweighs $|\widehat{e}| + 2\log|\widehat{e}| + 2\log|\widehat{p}| + O(1)$. If $e$ is large then $(e-1)|\widehat{p}|$ also outweighs the additional terms.)

¿From 1. and 2. we can deduce that $x = py$ for some prime $p > \log x / \log\log x$. Let $i$ be such that $p = p_i$. Prime $p_i$ can be described by giving its index $i$ plus a short program that will reconstruct $p_i$ from $i$. Hence,

$$C(\widehat{x}) \leq |\widehat{i}| + |\widehat{y}| + 2\log|\widehat{i}| + O(1).$$

Together with the above lower bound on $C(\widehat{x})$ we get

$$|\widehat{p_i}| \leq |\widehat{i}| + 2\log|\widehat{i}| + O(1).$$

Using the relationship between $n$ and $\widehat{n}$, we conclude $p_i \leq c \cdot i \log^2 i$, for some constant $c$ independent of $i$. ¿From the fact that this is true for arbitrarily large constant $x$ and $p_i > \log x / \log\log x$ we conclude the theorem. ♢

**Gödel Incompleteness Theorem**

Let $T$ be a sound logical theory over a countable language with recursively enumerable axioms. If $T$ is rich enough to describe computation of Turing machines then for some constant $c_T$ and integer $x$, the formula "$C(\overline{x}) \geq \overline{c_T}$" is true but unprovable, where $\overline{x}$ is the constant describing $x$. (This is a $\Pi_1$ formula saying that for all programs $p$ smaller than $\overline{c_T}$ and for all computations

$\tau$, if $\tau$ is a computation of $p$ then the output of $\tau$ is not $\overline{x}$.) If for all $x$ and all $c_T$ such formula were provable whenever it would be true then by enumerating. all proofs, for given $c_T$ we could find the first *large* $x$ with $C(x) \geq c_T$. But if we choose $c_T$ with succinct representation (very low Kolmogorov complexity), then we will be able to produce $x$ of high Kolmogorov complexity merely from the description of $c_T$, the description of $T$ and some small program. Of course, that is impossible. So "$C(\overline{x}) \geq \overline{c_T}$" cannot be provable for any large enough Kolmogorov non-random $c_T$ and any $x$ although it is true for many $x$ and $c_T$.

**Universal search procedure**

The problem SAT= $\{\psi;\ \psi$ is a satisfiable Boolean formula$\}$ is a well known NP-complete problem. A related problem is SAT-search where given a satisfiable Boolean formula $\psi$ we want to output an assignment $a$ to $\psi$ such that $a$ satisfies $\psi$. The computational complexities of SAT and SAT-search are closely related. If SAT has an efficient algorithm then SAT-search has one as well: perform a binary search for a satisfying assignment of $\psi$ by choosing the assignment bit by bit. On the other hand if SAT-search has an efficient algorithm (and we know its running time) then SAT has an efficient algorithm as well: run algorithm for SAT-search on $\psi$ and if it produces an assignment within its allowed running time and the assignment satisfies $\psi$ then $\psi$ belongs to SAT. We will present an (almost) optimal algorithm for SAT-search. We will need the following definition.

Let $\langle \cdot, \cdot \rangle$ be a pairing function. Levin defines the time-bounded Kolmogorov complexity of a string $x$ relative to a string $y$ by:

$$C_t(x|y) = \min\{|p| + \log t,\ p \in \{0,1\}^* \ \&\ U(\langle p, y\rangle) = x \text{ in } t \text{ steps}\}.$$

The algorithm for SAT-search works as follows: on input formula $\psi$, for $i = 1, 2, \ldots$ try all strings $a$ with $C_t(a|\psi) = i$, and see if any of them satisfies $\psi$. If yes, output such an $a$.

We leave implementation details of this algorithm to the interested reader. If $p$ is the optimal algorithm for SAT-search and $t$ is its running time on formula $\psi$ then the satisfying assignment for $\psi$ will be found in time about $2^{|p|}t^2$ by our algorithm. Hence, our algorithm for SAT-search is at most quadratically slower than the best algorithm for SAT-search. The only thing that stands in our way towards $\$1,000,000$ is that we do not have a good estimate on the running time of our SAT-search algorithm.

**Shuang Wang**
Shuang.Wang@student.uib.no
Presented paper by Shuang Wang, Fredrik Manne and Qin Xin

## Better Group Testing for Consecutive Defectives

Group testing was originally introduced as a potential approach to economical mass blood testing [1]. However, due to its basic nature, group testing technique has been applied to many computer science subjects: complexity theory, computational geometry and learning models among others. It has also been used in multiaccess communication and information coding, and recently, in clone library screening.

The general group testing problem refers to the task of distinguishing at most $d$ defective items within set $A$ of cardinality $n$. A group testing algorithm is formed by a sequence of pooling tests. In each test a particular subset, a *pool*, of set $A$ is used. The outcome of the test is *positive* if and only if there is at least one defective item in the pool, and it is *negative* otherwise. In our research, it is assumed that the items in set $A$ are ordered from 1 to $n$. A number of different group testing methods have been used in the past. One of them is *adaptive group testing* in which the choice of the items forming a pool at some stage of an algorithm is based on the outcome of previous tests. In *nonadaptive group testing*, the content of every single pool is determined prior to the execution of the algorithm.

In this paper, we study the special group testing such that all defective items are consecutive. For the adaptive case, we propose a simple algorithm to solve this problem with at most $\lceil \log \frac{n}{d} \rceil + 2\lceil \log d \rceil + 3$ tests, which also improves the currently best known upper bound $\lceil \log(n \cdot d) \rceil + 6$ due to Juan and Chang in [2]. For the nonadaptive case, we give an alternative version of the algorithm proposed by Colbourn in [3], which improves the currently best known upper bound from $\lceil \log \frac{n}{d-1} \rceil + 2 \cdot d + 1$ to $\lceil \log \frac{n}{d} \rceil + d + 3$.

### References

[1] R. Dorfman, The detection of defective members of large populations, *Ann. Math. Statist.*, 14 (1943), pp. 436–440.

[2] J.S. Juan and G.J. Chang, Consecutive Group Testing, *2002 International Conference on Graph Theory and Combinatorics and the Second Cross-Strait Conference on Graph Theory and Combinatorics*.

[3] C.J. Colbourn, Group Testing for Consecutive Positives, *Annals of Combinatorics* 3 (1999), pp. 37–41.

**Zdeněk Dvořák**
rakdver@atrey.karlin.mff.cuni.cz

Presented paper by H. Cohn, R. Kleinberg, B. Szegedy, C. Umans

## Group-theoretic Algorithms for Matrix Multiplication

A group-theoretic approach to bounding the exponent in the time complexity of matrix multiplication is presented. This approach is fundamentally different from the traditional ones based on the original Strassen's algorithm (relating to it in a similar way as the DFT-based multiplication of polynomials relates to the divide-and-conquer approach), but surprisingly, exactly the same bounds are obtained based on the same algebraic structures (the resulting algorithms however are not at all alike). Also, two hypotheses that would imply an $2+o(1)$ bound on the exponent will be presented.

**Gábor Hegedüs**
guest05@kam.mff.cuni.cz

## On Ramsey numbers

The Ramsey number $R(k,\ell)$ is the smallest integer $n$ such that in any two-coloring of the edges of a complete greaph on $n$ vertices $K_n$ by red and blue, either there is a red $K_k$ (i.e. a complete subgraph on $k$ vertices all of whose edges are colored red) or there is a blue $K_\ell$. In [6] F. P. Ramsey showed that $R(k,\ell)$ is finite for any two integers $k$ and $\ell$. P. Erdős in [1] obtained by probabilistic arguments the following non–constructive lower bound for the diagonal Ramsey numbers $R(k,k)$:

**Theorem 1.** *If* $\binom{n}{k} \cdot 2^{1-\binom{k}{2}} < 1$, *then* $R(k,k) > n$. *Thus* $R(k,k) > \lfloor 2^{k/2} \rfloor$ *for all* $k \geq 3$.

One of the striking applications of the Frankl–Wilson theorem [4] for prime moduli was an explicit construction of graphs of size $\exp(c \log^2 k / \log\log k)$ without homogeneous complete subgraph $K_k$. These are the largest explicit Ramsey–graphs known to date. V. Grolmusz in [5] gave an alternative construction of explicit Ramsey graphs of the same logarithmic order of magnitude. This construction is easily extandable to the case of several colors.

In this lecture we give a new conjectured lower bound for the diagonal Ramsey numbers $R(k,k)$.

P. Erdős and G. Szekeres proved the following Theorem in [3].

**Theorem.** *Let* $k > 0$ *be a positive integer. There exists a point set* $\mathcal{H} \subseteq \mathcal{R}^2$, $|\mathcal{H}| = 2^{k-2}$ *in general position such that* $\mathcal{H}$ *does not contain a* $k$*-point convex*

*independent set.*

We use this beautiful construction to make plausible the following conjecture.

**Conjecture.** *Let $k > 2$ be a fix integer. Then $R(k,k) > 2^{k-2}$.*

### References


[1] P. Erdős, Some Remarks on the Theory of Graphs, *Bulletin of the American Mathematical Society*, **53** 292–294 (1947)

[2] P. Erdős, G. Szekeres, A combinatorial problem in geometry, *Compositio Math.* Vol. 2 463–470 (1935)

[3] P. Erdős, G. Szekeres, On some extremum problems in elementary geometry, *Ann. Univ. Sci. Budapest. Eötvös Sect. Math.* **3–4** 53–62 (1961)

[4] P. Frankl, R. M. Wilson, Intersection theorems with geometric consequences, *Combinatorica* **1** 357–368 (1981)

[5] V. Grolmusz, Superpolynomial size set-systems with restricted intersections mod 6 and explicit Ramsey graphs. *Combinatorica*, **20**, 73–88 (2000)

[6] F. P. Ramsey, On a problem of formal logic, *Proc. London Math. Soc.* **30** (2), 264–286 (1929)



**Dušan Daniel**

daniel@savbb.sk


## Snarks

Snarks are a nontrivial cubic graphs whose edges cannot be properly colored by three colors. Snarks creat an inportant class of graphs, because a lot of problems of graph theory is possible to reduce on snarks. Classically, 'nontrivial' of snarks is interpreted, like a condition that the girth of snark is at least five and cyclical connectivity is at least four. For notions of nontrivial of snarks are important the operations of **reductions and decompositions**. Under **reduction** of snark $K$ we think of removing of set of vertices, those that $K - v$ is a graph whose edges cannot be properly colored by three colors. **Decomposition** of snark is an operation, which splits a snark on pair of small snarks. The simplest reductions important to deal with are 2, 3, and 4-reductions and talking about decomposition are 4 and 5 decomposition. With decomposition is closely connected 4 and 5 product, we are going to focus on it. We also mention about 6 decomposition. In general, our thinkings lead to finding of set of prime number snarks, it means snarks, which are not possible to reduce or split.

**Marek Sterzik**
marek_sterzik@volny.cz

## Evasiveness of Graph Properties I

In series of three talks we shall introduce the notion of evasiveness and show some results concerning AKR conjecture.

The first talk we give basic definitions, show some evasiveness results using so-called adversary argument and group theory. In the second talk we introduce topological approach, deviced by Kahn et al. In the third talk the topological approach is used to show evasiveness in some special cases.

For a given boolean function $f$ of $n$ variables we will ask how many variables we have to know to determine the value of the function. For an algorithm computing the function $f$ one can construct a decision tree. The nodes of the tree represent variables, which the algorithm asks for and the leaves are represent the possible outputs of the algorithm. The algorithm begins in the root of the tree and in each node we decide if we go to the left or right subtree (depending on the value of the variable) We will also ask for the minimal depth of any decision tree for the function $f$ (which is called $D(f)$).

A boolean function of $n$ variables is *evasive*, if $D(f)$ is equal to $n$. In other words we must ask for all variables that we can be sure, we know the value of $f$.

Boolean functions can represent properties of some structures, which are encoded in the values of variables. The structures can be specially graphs. Each variable then corresponds to an edge of the graph and means if the edge is present. A boolean function $f$ is a *graph property*, if the value of $f$ is invariant under graph isomorphism.

A graph property $f$ is *monotone*, if adding edges preserves the property.

The allready mentioned Aanderaa-Karp-Rosenberg conjecture says, that every monotone non-trivial graph property is evasive.

A boolean function $f$ of $n$ variables is *weekly-symetric*, if there exist any transitive permutation group $G$ (a subgroup of $S_n$) such that the value of $f$ does not change if we permute the variables in $f$ with any permutation of $G$. Transitive group means, that for any $i, j \in \{1, 2, ..., n\}$ there exists a permutation $\gamma$ such that $\gamma(i) = j$.

One can generalize the AKR conjecture saying every monotone weakly-symetric function $f$ is evasive.

The main result presented in the talk is to prove that the generalized AKR conjecture holds for functions with $n$ variables, where $n$ is prime.Note, that this says nothing about the AKR conjecture, because only the $K_3$ graph has prime number of edges.

The proof is divided into three parts. In the first part we show, that if the value

$$\mu(f) = \sum_{x \in \{0,1\}^n} f(x)(-1)^{|x|}$$

is non-zero, then $f$ is evasive ($|x|$ means the number of ones in the vector). In the second part we will show, that $n$ divides the size of $G$ and following the Cauchy's prime theorem there exists an element $\gamma$ of order $n$. Since $n$ is prime, $\gamma$ also consists of one single cycle. In the third part we use the permutation $\gamma$ to show that all boolean vectors $x$, which are not equal to $\overline{0}, \overline{1}$ can be divided into equivalent classes $V_k$ of size $n$. Since for two equivalent vectors $x$, $y$ holds the equality $f(x)(-1)^{|x|} = f(y)(-1)^{|y|}$, we can calculate $\mu(f)$ as

$$\mu(f) = \sum_k n f(x)(-1)^{|x|} + f(\overline{1})(-1)^n$$

which is also congrunet to $\pm 1$ modulo $n$ and cannot be zero. So $f$ is evasive.

Using more sophisticated group-theoretical argument, one can show similar statement for $n$ being a prime power.

**Rudolf Stolař**
riv@email.cz

Presented paper by David R. Wood

## Drawing a Graph in a Hypercube

A $d$-dimensional hypercube drawing of a graph represents the vertices by distinct points in $0, 1^d$, such that the line-segments representing the edges do not cross. We study lower and upper bounds on the minimum number of dimensions in hypercube drawing of a given graph. This parameter turns out to be related to Sidon sets and antimagic injections. We show that for $n$-vertex $m$-edge graphs with degeneracy d is the minimum volume of hypercube drawing at most $2n + 2dm$ and for $n$-vertex graphs with bandwidth (resp. pathwidth) $k$ is the minimum volume of hypercube drawing at most $4k(2n - 1)$ (resp. $(16 + o(1))kn$).

**Dana Bartošová**
dadik@email.cz

## Evasiveness of Graph Properties II

We show the link between monotone boolean functions and topological objects and their properties, namely simplicial complexes, contractibility and fixed point property.

First we define an abstract simplicial complex

**Definition 1.** *A simplicial complex is a finite collection $K$ of sets such that*
*(1)* $\forall X \in K, Y \subseteq X \Rightarrow Y \in K$ *and*
*(2)* $K \neq \emptyset$.
*The sets in $K$ are called (abstract) simplices The elements of all sets in $K$ are called vertices of $K$.*

and corresponding geometric realization

**Definition 2.** *Let $V = \{p_0, \ldots, p_s\}$ be a finite set of $s+1$ affinely independent in a linear normed space. The convex hull $\sigma$ of $V$ $\{\sum_{v \in V} \alpha_v v : \sum \alpha_v = 1, \alpha_v \geq 0\}$ is called the (closed) $s$-simplex. Elements of $V$ are called vertices of $S$.*

**Definition 3.** *The convex hull of an arbitrary subset of vertices of the simplex $\sigma$ is called face of $\sigma$.*

**Definition 4.** *A collection $K = \{S_1, S_2, \ldots, S_n\}$ is said to form a geometric simplicial complex if*
*(1)* $\forall S_i, T$ *a face of* $S_i \Rightarrow T \in K$,
*(2)* $\forall S_i, S_j \in K, S_i \cap S_j \neq \emptyset \Rightarrow S_i \cap S_j$ *is a face of both* $S_i$ *and* $S_j$

It's legal to mix the abstract simplicial complex with its geometric realization.

A topological property, contractibility, will help us in proves of evasiveness of graph properties.

**Definition 5.** *A geomteric simplicial complex $K$ is contractible, if there exists a continuous mapping $H : K \times [0,1] \to K$ such that $H(x,0) = x$ and $H(x,1) = p_0$ for some $p_0 \in K$.*

For a general simplicial complex $K$ it is undecidable whether $K$ is contractible.

In following lemma we prove that if two special subcomplexes of a complex $K$ are contractible, then $K$ is contractible:

**Lemma 1.** *If for some $v \in K$, $K \setminus v = \{X \in K : v \notin X\}$ and $K/v = \{X \in K : v \notin X, X \cup \{v\} \in K\}$ are contractible, then $K$ is contractible.*

We come to the connection between simplicial complexes and monotone functions:

A monotone boolean function $f \neq 1$ gives a simplicial complex $K_f = \{S \subseteq \{1, \ldots, n\} : f(x^S) = 0\}$, where $(x^S)_i$ is 1 for $i \in S$ and 0 otherwise. Naturally we also get: $K_{f|_{x_i=0}} = \{S \subseteq \{1, \ldots, i-1, i+1, \ldots, n\} : S \in K_f\} = K_f \setminus i$ and $K_{f|_{x_i=1}} = \{S \subseteq \{1, \ldots, i-1, i+1, \ldots, n\} : S \cup \{i \in K_f\} = k_f/i$.

Now we can easily prove (by induction) a characterization of monotone non-evasive function which are not constantly 1:

**Lemma 2 (Kahn-Saks-Sturtevant).** *If $f \neq 1$ is a non-evasive monotone function, then $K_f$ is contractible.*

From fixed point theory we know that every continuous mapping from a con-

tractible polyhedron into itself has a fixed point. We are interested in fixed points of simplicial mappings between two simplicial complexes:

**Simplicial mapping.** *Let $K$ and $L$ be two abstract simplicial complexes. A simplicial mapping is a mapping $f : V(K) \to V(L)$ that maps simplices to simplices, i.e. $f(X) \in L$ whenever $X \in K$.*

For our purposes we consider a one-to-one simplicial mapping $f : V(K) \to V(K)$ and we identify all its fixed points as the convex combinations of centers of gravity of those faces of K that are the cycles of the permutation $f$.

**Eva Ondráčková**
efa@atrey.karlin.mff.cuni.cz

## Seidel's switching and $H$-free graphs

**Definition.** *Let $G$ be a graph. Then the Seidel's switch of a vertex subset $A \subseteq V_G$ is called $S(G, A)$ and*

$$S(G, A) = (V_G, E_G \triangle \{xy : x \in A,\ y \in V_G \setminus A\}).$$

**Definition.** *We say that two graphs $G$ and $H$ are switching equivalent (denoted by $G \sim H$) if there is a set $A \subseteq V_G$ such that $S(G, A)$ is isomorphic to $H$. The set*

$$[G] = \{S(G, A) : A \subseteq V_G\}$$

*is called the switching class of $G$.*

Note that $\sim$ is an equivalence relation on graphs, and switching classes are the equivalence classes of $\sim$ for graphs on a fixed set of vertices $V_G$ (not considering isomorphism), as shown by Seidel [5].

Let $P$ be a graph property. We define the problem $S(P)$ as follows: determine whether a given graph $G$ is switching-equivalent to a graph possessing the property $P$.

For a fixed graph $H$, we consider the property "being $H$-free". Polynomial-time decision algorithms are known for S("being $H$-free") if $H$ has at most three vertices or is isomorphic to a $P_4$. The algorithm for $K_2$ or $I_2$ has been found by Hage et al. [1], the one for $K_{1,2}$ or $K_2 + K_1$ is due to Kratochvíl et al. [4]. Hayward [2] and independently Hage at al. [1] found an algorithm for $P_3$ or $I_3$; the result is a core of the polynomial-time algorithm for recognizing $P_3$-structures of graphs. The case of $P_4$ has been solved by Hertz [3] in connection to perfect switching classes. We show that a polynomial-time algorithm exists even if $H$ is isomorphic to a claw $K_{1,3}$.

**Lemma.** *Let $G$ be a graph and $A \subseteq V_G$. Then $S(G, A)$ is claw-free if and only if for every four-vertex induced subgraph $H$ of $G$ that is switching-equivalent to a claw the following is true:*

*$|V(H) \cap A|$ is odd if $H$ is a claw,*

*$|V(H) \cap A|$ is even if $H$ is a not claw.*

**Theorem.** *Given a graph $G$, we can in polynomial time find a set $A \subseteq V_G$ such that $S(G, A)$ is claw-free, or find out that no such $A$ exists.*

**Problem.** *Is there a graph $H$ such that the problem S("being $H$-free") is NP-complete?*

### References


[1] J. Hage, T. Harju, E. Welzl: *Euler graphs, triangle-free graphs and bipartite graphs in switching classes*, in: Proceedings ICGT 2002, LNCS 2505, Springer-Verlag (2002), pp. 148–160.

[2] R. B. Hayward: *Recognizing $P_3$-structure: A switching approach*, J. Combin. Th. Ser. B **66** (1996), pp. 247–262.

[3] A. Hertz: *On perfect switching classes*, Discrete Appl. Math. **94** (1999), pp. 3–7.

[4] J. Kratochvíl, J. Nešetřil, O. Zýka: *On the computational complexity of Seidel's switching*, Proc. 4th Czech. Symp., Prachatice 1990, Ann. Discrete Math. **51** (1992), pp. 161–166.

[5] J. J. Seidel: *A survey of two-graphs*, Teorie combinatorie, Atti Conv. Lincei, Vol 17, Academia Nazionale dei Lincei, Rome (1973), pp. 481–511.



**Jan Herman**

hermitko@email.cz


Presented paper by M. Bordewicha, K. T. Huberb, C. Semplec

## Identifying phylogenetic trees


(Discrete Mathematics 300 (2005) pages 30 – 43)


For a finite set $X$, an *X-tree* $\mathcal{T} = (T; \phi)$ is an ordered pair consisting of a tree T, with vertex set $V$ say, and a map $\phi : X \to V$ with the property that, for all $v \in V$ with degree at most two, $v \in \phi(X)$. An X-tree is also called a *semi-labelled tree*.

A *character* on $X$ is a function $\chi$ from a non-empty subset $X\prime$ of $X$ into a set $C$ of character states. If $|C| = 2$, then $\chi$ is a two-state character. Let $\pi(\chi)$ denote the partition of $X\prime$ corresponding to $\{\chi^{-}1(\alpha) : \alpha \in \mathcal{C}\}$.

Let $\chi$ be a character on $X$ and let $\mathcal{T} = (T; \phi)$ be an X-tree. We say that $\mathcal{T}$ *displays* $\phi$ if there is a subset $E$ of edges of $T$ such that, for all $A, B \in \pi(\chi)$ with

$A \neq B$, there exists two connected components of the graph obtained from $T$ by deleting the edges in $E$ with $\chi(A)$ being a subset of the vertex set of one component and $\chi(B)$ being a subset of the vertex set of the other component. More generally, $\mathcal{T}$ *displays a collection* $\mathcal{C}$ of characters on $X$ if $\mathcal{T}$ displays each character in $\mathcal{C}$, in which case $\mathcal{C}$ is *compatible*. For a compatible collection $\mathcal{C}$ of characters on $X$, we say that $\mathcal{C}$ *infers* a character $\chi$ if every X-tree that displays $\mathcal{C}$ also displays $\chi$.

Associated with each edge $e$ of an X-tree $\mathcal{T} = (T; \phi)$ is an *X-split*; that is, a bipartition of $X$ into the label sets of the two connected components of $\mathcal{T} - e = (T - e; \phi)$. An X-tree $\mathcal{T}'$ is a *refinement* of $\mathcal{T}$ if every X-split of $\mathcal{T}$ is an X-split of $\mathcal{T}'$. We say that $\mathcal{C}$ *identifies* an X-tree $\mathcal{T}$ if $\mathcal{T}$ displays $\mathcal{C}$ and every X-tree $\mathcal{T}'$ that displays $\mathcal{C}$ is a refinement of $\mathcal{T}$.

**Definition.** *Let $\mathcal{C}$ be a collection of characters on $X$ and let $\mathcal{T} = (T; \phi)$ be an X-tree. Let $X', X' \in X$. The set of vertices in the minimal subtree of $\mathcal{T}$ that connects the vertices of $\phi(X)$ is denoted by $\mathcal{T}(X)$. The partition intersection graph of $\mathcal{C}$, denoted $int(\mathcal{C})$, is the graph that has vertex set $V(\mathcal{C}) = \bigcup_{\chi \in \mathcal{C}} \{(\chi, A) : A \in \pi(X)\}$ and an edge joining $(\chi_1, A)$ and $(\chi_2, B)$ if $A \cap B$ is non-empty.*

**Definition.** *Let $\mathcal{T} = (T; \phi)$ be an X-tree and let $e = \{u1, u2\}$ be an edge of $T$. Then $e$ is strongly distinguished by a character $\chi$ on $X$, if there exist $A_1$ and $A_2$ in $\pi(\chi)$ such that, for each $i \in \{1, 2\}$, the following hold:*

*(i) $\phi(A_i)$ is a subset of the vertex set of the component of $T - e$ containing $u_i$*

*(ii) the vertex set of each component of $T - u_i$ , except for the one containing the other end vertex of $e$, contains an element of $\phi(A_i)$*

*(iii) $\phi^{-}1(u_i)$ is a subset of $A_i$*

*We say $\mathcal{T}$ is strongly distinguished by a collection $\mathcal{C}$ of characters if every edge of $T$ is strongly distinguished by some character in $\mathcal{C}$.*

**Definition.** *For a collection $\mathcal{C}$ of characters on $X$, we let $\mathcal{G}(\mathcal{C})$ denote the set of graphs $\mathcal{G}(\mathcal{C}) = \{G :$ there is an X-tree $\mathcal{T}$ displaying $\mathcal{C}$ with $G = int(\mathcal{C}, \mathcal{T})\}$.*

A useful partial order on $\mathcal{G}(\mathcal{C})$ is obtained by setting, for all $G_1, G_2 \in \mathcal{G}(\mathcal{C})$, $G_1 \leq G_2$ if the edge set of $G_1$ is a subset of the edge set of $G_2$.

**Theorem.** *Let $\mathcal{C}$ be a collection of characters on $X$. Then $\mathcal{C}$ identifies an X-tree if and only if the following conditions hold:*

*(i) there is an X-tree that displays $\mathcal{C}$ and, for every edge $e$ of this tree, there is a character on $X$ inferred by $\mathcal{C}$ that strongly distinguishes $e$; and*

*(ii) there is a unique maximal element in $\mathcal{G}(\mathcal{C})$.*

*Moreover, if $\mathcal{C}$ identifies an X-tree $\mathcal{T}$, then $\mathcal{T}$ satisfies the properties in (i) and $int(\mathcal{C}, \mathcal{T})$ is the unique maximal element of $\mathcal{G}(\mathcal{C})$.*


The firefighter problem
(Joint work with
Our problem:
Given a (finite) set S
fighters per turn,
can the fire be contained?

**Robert Babilon**
babilon@kam.mff.cuni.cz

Presented paper by Terence Tao and Van Vu

## Determinant of Random $\pm 1$ Matrices

(http://arxiv.org/abs/math/0411095)

Let $M_n$ denote a random $n \times n$ matrix with entries $+1$ and $-1$. Random means that each entry of $M_n$ is $+1$ or $-1$ with the probability $1/2$, and is independent on other entries. We will focus on the study of the determinant of $M_n$, particularly on the expected value of $|\det(M_n)|$, and on the probability that the matrix $M_n$ is singular.

It is not difficult to see that $0 \leq |\det(M_n)| \leq (\sqrt{n})^n = n^{n/2}$. The lower bound holds for singular matrices, and the upper bound holds for so called Hadamard matrices (i.e. $\pm 1$ matrices with orthogonal rows). There is a conjecture that the typical value of $|\det(M_n)|$ should be of the order of $\sqrt{n!} = e^{-n+o(n)}n^{n/2}$. However, even proving that $M_n$ is typically regular (i.e. with at least constant probabilty) is a non-trivial task. This was first proved by Komlós in 1967.

The problem of determining the asymptotic behavior of $Prob(\det(M_n)) = 0$ precisely is a notorious open problem. Since a matrix $M_n$ with two identical or opposite rows (or columns) is necessarily singular, it is easy to see that $Prob(\det(M_n) = 0) \geq (1+o(1))n^2 2^{1-n}$. It has often been conjectured that this is the dominant source of singularity. Prior to the presented paper, the best result is due to Kahn, Komlós, and Szemerédi (1995): $Prob(\det(M_n) = 0) = (0.999 + o(1))^n$. In the presented paper the result is improved to the following:

$$Prob(\det(M_n) = 0) = (0.938 + o(1))^n.$$

Let us turn back to the determinants. It is an easy observation that $\det(M_n)$ is divisible by $2^{n-1}$. Thus the Komlós result implies that $Prob(|\det(M_n)| \geq 2^{n-1}) = 1 - o(1)$. Surprisingly, this is the best estimate of previous to the presented paper. This result is much improved to the following:

$$Prob(|\det(M_n)| \geq \sqrt{n!} \cdot \exp(-o(n^{1/2+\varepsilon})) = 1 - o(1) \quad \text{for any } \varepsilon > 0.$$

**Martin Pergel**
perm@kam.mff.cuni.cz

Presented paper by Carsten Thomassen

## Plane Cubic Graphs with Prescribed Face Areas

(Combinatorics, Probability and Computing 1 (1992) p. 371 – 381)

If $G$ is a plane, cubic graph, then $G$ has a drawing such that each edge is

a straight line segment and each bounded face has any prescribed area. The special case where all areas are the same proves a conjecture of G. Ringel, who gave an example of a plane triangulation that cannot be drawn in this way.

**Basic definitions:** For a plane graph $G$ a *redrawing* of $G$ is a plane graph $G'$ such that each edge of $G'$ is a straight line segment and such that $G'$ is the image of $G$ under a homeomorphism of the euclidean plane. If $\{G_n\}_{n=1}^{\infty}$ is a sequence of redrawings of $G$ such that for any $x$, vertex of $G$, the sequence $x_n$ of corresponding vertices converge to a point $x'$, then the union of straight line segments $x'y'$ (where $\{x, y\}$ is an edge of $G$) is called the limit of $G_n$.

$H$ is a *modification* of $G$ if it is a redrawing of $G$ that has the same number of faces.

If $H$ is a modification of $G$, then, for each positive real number $\varepsilon > 0$ there exists a redrawing $G'$ of $G$, such that for each vertex $x$ of $G$ the vertex $x'$ in $G'$ corresponding to $x$ has Euclidean distance $< \varepsilon$. Such a redrawing will be called an *$\varepsilon$-replacement* of $H$.

**Note:** Our definitions require *the modification of a graph* to have the same number of faces, but for larger face (e. g. hexagon) some part of this face may collapse (e. g. to a four-cycle with one edge sticked to one its vertex). Under these assumptions we prove the following theorem.

**Main result:** Let $G$ be any cubic plane graph. Then there exists a modification $H$ of $G$ such that every face is bounded by a triangle of any prescribed area. Moreover, for every $\varepsilon > 0$ there exists an $\varepsilon$-replacement of $H$ that is a redrawing of $G$ and in which every face has the prescribed area.

**Jan Hladký**
hladk@seznam.cz

## Evasiveness of Graph Properties III

We exhibit evasiveness of some families of monotone functions using topological approach, which was introduced by Kahn et al.

Proofs of all the results presented in the talk are based on a crucial lemma about contractibility of a simplicial complex associated with a monotone non-evasive function. The proof of the lemma was shown in a previous talk given by Dana Bartošová.

**Lemma 1.** *Let $f : \{0,1\}^n \to \{0,1\}$ be a monotone non-evasive function. Then the associated complex $\mathcal{K}_f$ is contractible.*

By a *bipartite property* we mean a function whose input variables are indicator variables of the edges of $K_{m,n}$, such that the values of the function are invariant under relabeling the vertices in each of the the parts of $K_{m,n}$.

In the talk we give complete proofs of the following two theorems.

**Theorem 2.** *Let $f : \{0,1\}^n \to \{0,1\}$ be a nontrivial monotone property which is invariant under a cyclic permutation of the input variables. Then $f$ is evasive.*

**Theorem 3.** *Let $f : \{0,1\}^{mn} \to \{0,1\}$ be a nontrivial monotone bipartite function. Then $f$ is evasive.*

Both proofs are based on Lemma 1. We assume that we have a non-evasive function satisfying the assumptions. We pick a permutation $\psi$ of the inputs. Then $\psi$ is a simplicial mapping on the vertices of $\mathcal{K}_f$. Since $\mathcal{K}_f$ is conctractible, the geometric extension of $\psi$ has, by Lefschetz theorem, a fixed point. This fact leads directly to contradiction in the case of Theorem 2. In the case of Theorem 3, one gets a contradiction computing Euler characteristics of the corresponding complex $\mathcal{H}$ of the fixed points of $\psi$. On one hand, the Euler characteristics of $\mathcal{H}$ should be, by Hopf Index Formula, $-1$, on the other hand, it is easy to describe $\mathcal{H}$ explicitly and to compute that the Euler characteristics must be an even number.

We show a sketch of the proof of AKR-conjecture for graphs with number of vertices being a prime power.

**Theorem 4.** *Let $f$ be a nontrivial monotone graph property on a graph with prime power number of vertices. Then $f$ is evasive.*

Using Theorem 3 and Theorem 4 we prove that AKR-conjecture is true up to a factor.

**Theorem 5.** *There is a constant $c > 0$, such that every nontrivial monotone graph property $f$ on a graph in $n$ vertices, the decision complexity of $f$ is at least $D(f) > c\binom{n}{2}$.*

In the end of the talk we discuss briefly latest results about evasiveness of subgraph containment which are due to Chakrabarti, Khot and Shi.

## References


[1] A. Chakrabarti, S. Khot and Y. Shi. Evasiveness of Subgraph Containment and Related Properties. *Lecture Notes in Computer Science* Volume 2010, 110–126 (2002).

[2] J. Kahn, M. Saks and D. Sturtevant. A topological approach to evasiveness. *Combinatorica* Volume 4, 297–306 (1984).

[3] A. Yao. Monotone bipartite graph properties are evasive. *SIAM J. on Computing* Volume 17, 517–520 (1988).

**Bernard Lidický**
bernard@matfyz.cz

Presented paper by Rowan Davies, Gordon F. Royle

## Tabu Search and Football Pool Problem

(Discrete Applied Mathematics 74 (1997) p217 – 228)

Tabu search is a general purpose aproximation algorithm for finding best configuration from all possible configurations. The solved problem may be written like:

Minimize $c(x)$ where $x \in X$ and $X$ is set of all possible configurations.

Tabu search requires some neighbour structure on $X$. The algorithm starts in any (possibly random) configuration and move from one configuration to neighbour configuration with obvious goal finding the optimal solution.

Simple greedy algorithm check all neighbours and choose the best neighbour. Unfortunatelly this way of choosig will stop in first local minimum instead of global minimum. Any neighborhood search based algorith must solve case of local minimas somehow.

Tabu search remebers a list of few last configurations and new moves may be done only to configurations not on list. Those configurations are tabu. The name of algorith comes form this "tabu list".

Code for tabu search may look this way:

```
p := some magic constant;
x := some configuration;
list := empty;
while (we want to continue)
{
   y := neighbour of x with minimal c(x) that isn't in list;
   add(list, y);
   if (size(list) > p) remove_oldest(list);
   x := y;
}
```

It may be also a good idea to remember best reached point in whole algorithm. Because we may find the optimal solution but continue to other configurations.

**Definition 1.** *For graph $G(V, E)$ and $x \in V$ we define $N(x) = \{x\} \cup \{y \in V : (x, y) \in E\}$. For $X \subset V$ we define $N(X) = \bigcup_{x \in X} N(x)$.*

**Definition 2.** *For graph $G(V, E)$ any $D \subseteq V$ is called dominating set if $V = N(D)$*

In order to use tabu search to find dominating set of graph $G$ we need to define

set of configuration $S$ for tabu search and $c(x)$ for all $x \in S$. We put $S = 2^V$ and $\forall x \in S : c(x) = |x| + |V\ N(x)|$. We say that $x, y \in S$ are neighbours if they differ only one vertex. WLOG $|x| = |y| - 1$ and $x \subset y$.

**David Hartman**

hartman@kam.mff.cuni.cz

Presented paper by B. Martin and F. Madelaine

## Towards a trichotomy for quantified H-coloring.

(Lecture Notes in Computer Science, July. Swansea, CiE 2006 (preprint))

The convexity of constraint satisfaction problem is still open question. It is mostly represented by *dichotomy conjecture*, that states that every constrant satisfaction problem is either tractable or NP. In this paper the authors concentrate on the restriction of the quantified constraint satisfaction problem to graphs and investigate its convexity. This restriction helps them to use graphic properties rather than algebraic method as usual. The definition of *quantified $H$-coloring problem* is motivated in *$H$-coloring problem* established and proofed by Hell and Nešetřil. The H-coloring problem is a generalization of graph coloring using homomorphisms, where homomorphism is a vertex mapping from one graph to another that preserving edges. Than the $H$-coloring problem can be defined as: For a given graph $H$ if you take a graph $G$ as an input, you accept it if and only if there is a homomorphism from $G$ to $H$.

The quantified $H$-coloring problem is introduced by definition of a two player game. Let $\mathcal{G}$ be the *$n$-paritioned graph* consisting of graphs $G$ and a partition $\{U_1, X_2, U_3, X_4, \ldots, U_{2n+1}, X_{2n+2}\}$ of $V(G)$. The *$(\mathcal{G}, H)$-game* is a two player game where opponent plays universal partitions $U_i$ and the proponent plays existential partitions $X_i$. The partitions are alternate in ascendenting order, until all partitions have been played. For each vertex in partition $U_{2i+1}$ opponent chooses a vertex in $H$ (define function $f_{U_{2i+1}} : U_{2i+1} \to V(H)$) and for each vertex in $X_{2i}$ proponent chooses a vertex in $H$ (define function $f_{X_{2i}} : X_{2i} \to V(H)$). Proponent wins if, and only if, the function $f = f_{U_1} \cup f_{X_2} \cup \ldots f_{X_{2n+2}}$ is a homomorphism from $G$ to $H$. This homomorphism is called as *alternating homomorphism* from $n$-paritioned graph $\mathcal{G}$ to $H$ and write it as $G \overset{alt}{\to} H$. The quantified $H$-coloring problem is than defined as the decision problem that should decide whether for the input partitioned graph ( $G$) holds $G \overset{alt}{\to} H$.

Hell and Nešetřil proofed that the class of $H$-coloring problems has a dichotomy. The problem from this class is tractable if $H$ is bipartite and NP otherwise. This conclusion motivates authors to find a trichotomy in the class of quantified $H$-coloring problems. This trichotomy is expressed by final theorem of the paper.

**Theorem.** *Let $H$ be a graph with at most one cycle. The quantified $H$-coloring problem exhibiths a trichotomy.*

*I f $H$ is bipartite then the quantified $H$-coloring problem is tractable.*

*I f $H$ is not bipartite and not conected then the quantified $H$-coloring problem is $NP$-complete.*

*I f $H$ is not bipartite connected then the quantified $H$-coloring problem is $PSpace$-complete.*

**Jan Hubička**

hubicka@kam.mff.cuni.cz

Presented paper by S. Szeider

## Homomorphisms of conjunctive normal forms

(Discrete Applied Mathematics 130 (2003) 351-365)

Consider propositional formulas in conjunctive normal form represented as set of clauses and clauses represented as sets of literals (*literal* is either variable $v$ or negation of variable $\overline{v}$).

Let $H$ and $F$ be formulas and $\varphi$ map from the literals of $H$ to the literals of $F$. We call $\varphi$ *homomorphism* if it preserves complements and clauses, i.e. $\varphi(\overline{l}) = \overline{\varphi(l)}$ for every literal $l$ of $H$, and $\{\varphi(l); l \in C\} \in F$ for every clause $C$ of $H$.

Homomorphisms preserve unsatisfiability (i.e. if there is a homomorphism $H$ to $F$, then unsatisfiability of $H$ implies unsatisfiability of $F$).

Similarly to graph homomorphisms, the homomorphisms on formulas imply an quasiorder and each equivalency class has up to isomorphism unique representative called *core*. The core of formula can be used as reduced form of original formula. Deciding whether formula is an core is however NP-complete.

A notion of *proof by homomorphism* can be defined in following manner. Assume that $H$ is an unsatisfiable formula and, based on the specific nature of $H$, its unsatisfiability can be established in polynomial time. Given different formula $F$ and homomorphism from $H$ to $F$ it follows that $F$ is unsatisfiable too and the property "being homomorphism" can be verified in polynomial time. The triple $(H, \varphi, F)$ can be considered as proof on the unsatisfiability of $F$.

We will consider sets $\Gamma$ of unsatisfiable formulas such that for every unsatisfiable formula $F$, there exists $H \in \Gamma$ and homomorphism $\varphi$ from $H$ to $F$ (i. e. $\Gamma$ is *homomorphically complete*); $\Gamma$ can be recognized in polynomial time (i. e., $\Gamma$ is *tractable*).

Given such an $\Gamma$, then $\Pi_\Gamma = \{(H, \varphi, F); H \in \Gamma$ and $\varphi$ is homomorphism from $H$ to $F\}$ is proof system.

Let $MU(k)$ denote the set of minimally unsatisfiable formulas (i.e. unsatisfiable formulas where removing single clause makes it satisfiable) for which the umber of clauses exceeds the number of variables exactly by $k$. While it is computionaly hard problem to recognize minimally unsatisfiable formulas, formulas in $MU(k)$ can be recognized in polynomial time for every fixed $k \geq 1$.

Main result of the paper is the homomorphic completeness of $MU(1)$.

Given two proof systems $\Pi$ and $\Pi'$, we say that $\Pi'$ *p-simulates* $\Pi$ if every proof $x \in \Pi$ can be transformed into proof $x' \in \Pi'$ in polynomial time such that $x$ and $x'$ prove the same formula. If $\Pi$ and $\Pi'$ $p$-simulate each other, then we say that they are $p$-equivalent The efficiency of (propositional) proof system is closely related to the NP=co-NP question; this is also the main motivation for studying the relative complexity of proof systems.

It is shown that every proof in $(H, \varphi, F) \in \Pi_{MU(1)}$ can be transformed into a tree resolution proof of $F$ in polynomial time, so the $\Pi_{MU(1)}$ and tree resolution proofs are $p$-equivalent. For fixed $k \geq 1$, the set $MU(\leq k)$ of all minimally unsatisfiable formulas for which the number of clauses exceeds the number of variables by at most $k$ is homomorphically complete. Since $MU(1) \subseteq MU(\leq k)$ it is conceivable that for $\Pi_{MU(\leq k)}$ is stronger than $\Pi_{MU(1)}$. It is shown, however, that $\Pi_{MU(\leq k)}$ and $\Pi_{MU(1)}$ are $p$-equivalent.

**Radovan Šesták**
radofan@gmail.com

Presented paper by Noga Alon, Vera Asodi

## Tracing a single user

Let $g(n, r)$ be the maximum possible cardinality of a family $F$ of subsets of $\{1, 2, ..., n\}$so that given a union of at most $r$ members of $F$, one can identify at least one of these members. Study of this function is motivated by questions in molecular biology and in this paper we show that $g(n, r) = 2^{\Theta(\frac{n}{r})}$. The upper bound has been solved by Csuros and Ruszinko. Probabilistic proof of the lower bound is given and later derandomised algorithm, based on the proof, is presented for explicit construction of family of subsets of size $2^{\frac{n}{20r}}$

**Definition 1.** *Let $[n] = \{1, 2, ..., n\}$, and $F \subseteq 2^{[n]}$ be a family of subsets of $[n]$. Such set $F$ is called r-single-user tracing superimposed (r-SUT) if for all choices of $F_1, .., F_k \subseteq F$ with $1 \leq |F| \leq r$, $\bigcup_{A \in F_1} A = \bigcup_{A \in F_2} A = ... = \bigcup_{A \in F_k} A \implies \bigcap_{i=1}^{k} F_i \neq \emptyset$.*

**Theorem 1.** *For any $r \geq 2$and $n \geq 20r$, there exists an r-SUT family of subsets of $[n]$ of size at least $2^{\frac{n}{20r}}$.*

Fix $r \geq 2$and $n \geq 20r$. Let $m = 2^{n/20r}$, and let $p = 1/r$. Choose a family $F = \{F_1, ..., F_k\}$of sebsets of $[n]$ at random, where the subsets $F_i$are chosen independently as follows. Every $x \in [n]$ is chosen to be in $F_i$independently with probability $p$. We next show that with positive probability the family $F$ is r-SUT. Thus, wer have to show that, with positive probability, for all choices of $F_1, ..., F_k \subseteq F$ such that $1 \leq |F_i| \leq r$ for all $1 \leq i \leq k$ and $\bigcap_{i=1}^{k} F_i = \emptyset$, the unions $\bigcup_{A \in F_i} A$ for $1 \leq i \leq k$ are not equal. We consider two different casees, according to the size of $\bigcup_{i=1}^{k} F_i$. Proposition 2 deals with the case $|\bigcup_{i=1}^{k} F_i| < 2r$, Propositions 3 with $|\bigcup_{i=1}^{k} F_i| \geq 2r$, and in Propositioin 4 we combine the above to complete the proof of Theorem 1.

**Proposition 2.** *The following holds with probability greater than $\frac{1}{2}$. For all $s < 2r$, and for all distinct $A_1, A_2, ..., A_s \in F$, there exists an element $x \in [n]$ that belongs to exactly one of the sets $A_i$, $1 \leq i \leq s$.*

**Proposition 3.** *The following holds with probability greater than $\frac{1}{2}$. For all $t \leq r$, and for all distinct $A_1, ..., A_r, B_1, ..., B_l \in F$, $\bigcup_{i=1}^{r} A_i \not\subseteq \bigcup_{i=1}^{l} B_i$.*

**Proposition 4.** *Any family that satisfies the properties in Propositions 2 and 3 is r-SUT. Therefore, with positive probability, the random family $F$ is r-SUT, and hence $\frac{\log g(n,r)}{n} \geq \frac{1}{20r}$.*

Explicit Construction takes time $m^{O(r)}$, is based on derandomisation of previous proofs and combines the method of conditional expectations with the known constructions of small sample spaces supporting $2r$-wise independent random variables.


**David Howard**

dmh@math.gatech.edu


## Strange Combinatorial Connections

This talk is based on ideas from two papers. The first is by S. Felsner and W. T. Trotter (*Colorings of Diagrams of Interval Orders and $\alpha$-sequences of Sets*, Discrete Mathematics, 144 (1995), 23-31) and the second is by W. T. Trotter (*New Perspectives on Interval Orders and Interval Graphs*, Surveys in Combinatorics, R. A. Bailey, ed., London Mathematical Society Lecture Note Series 241 (1997), 237–286).

I call a poset $P = (X, P)$ an *interval order* if there there exists a function $I$ mapping each $x \in X$ to a closed interval $I(x) = [a_x, b_x]$ of $\mathcal{R}$, with the property that for all $x, y \in X$, $x < y$ in $P$ iff $b_x < a_y$ in $\mathcal{R}$. Let $t$ be a positive integer and $S = (S_0, S_1, \ldots, S_h)$ be a sequence of sets. $S$ is an *$\alpha$-sequence* if $S_1 \not\subseteq S_0$ and $S_j - (S_i \cup S_{i-1}) \neq \emptyset$, for all $i, j$ with $1 \leq i < j \leq h$. Let $t$ be a positive

integer and let $n = 2^t$. A list $(A_1, A_2, \ldots, A_t)$ of the subsets of $\{1, 2, \ldots, t\}$ is called a *monotone hamiltonian path* in the $t$-cube if $A_1 = \emptyset$, and for $1 < i < t$ and $S \subseteq A_i$, then $S = A_j$ for some $j \in \{1, 2, ..., i+1\}$.

This talk examines three different questions. First, for each $t \geq 1$, what is the largest integer $h = C(t)$ so that whenever P is an interval order of height $h$, the chromatic number of the diagram of P is at most $t$? Second, what is the longest $\alpha$-sequence where each set in the sequence is a subset of $\{1, 2, ..., t\}$? Finally, do monotone hamiltonian paths always exist in the $t$-cube? Though none of these answers are known, it turns out all three are related. The first two questions have the same answer, and it can be shown that $C(t) \leq 2^{t-1} + \lfloor \frac{t-1}{2} \rfloor$. The final question is true if and only if this inequality is an equality. While I will not prove here that these questions are equivalent, I will present concepts fundamental to the proof, as well as motivation for why these hamiltonian paths might or might not exist.

**Paul Raff**
praff@math.rutgers.edu

## The Firefighter Problem in the Two-Dimensional Grid

Consider the following dynamic graph theory problem: a graph $G$, a finite subset $S \subseteq G$ and a function $f : \mathrm{N} \to \mathrm{N}$ are given. The set $S$ is thought of as vertices that are initially on fire. Each vertex on the graph has three possible attributes: on fire, protected, or neither. Once a vertex is on fire or protected it stays that was permanently. A game is played where at each turn the player is given $f(i)$ firefighters to be placed at vertices that are neither on fire nor defended. Once a fighter is placed on a vertex, that vertex is defended. Afterwards, the fire spreads out one level, meaning the vertices

$$\{v \in G \mid v \text{ is adjacent to a vertex on fire and is not defended } \}$$

become on fire. There are two ways the game ends:

Each vertex of $G$ eventually becomes either on fire or protected.

There are unprotected vertices of $G$ which never become on fire. If the second condition is attained, then we say that "$f$ contains the fire in $G$ starting at $S$". The general question of whether a fire can be contained has been shown to be NP-complete.

In this talk, I will focus the problem to where $G$ is the two-dimensional infinite grid, meaning

$$V(G) = \mathrm{Z}^2$$
$$E(G) = \{\{(x, y), (x', y')\} \mid |x - x'| + |y - y'| = 1\}$$

Previously, it was known that the function $f$ defined by $f(t) = 1$ for all $t$ cannot contain the fire, whereas the function $f$ defined by $f(t) = 2$ for all $t$ can contain the fire. I will discuss improvements made by myself and Kah Loon Ng from DIMACS which culminates in the following (which will be made more specific):

**Theorem.** *If the "average" value of $f$ is more than 1.5, then $f$ contains the first in $G$ starting at any $S$.*

**Petr Golovach**
pagolovach@yahoo.com

## Generalized domination in chordal graphs

Let $G$ be a graph with vertex set $V(G)$ and edge set $E(G)$. The open neighborhood of a vertex is denoted by $N(u) = \{v\colon (u, v) \in E(G)\}$.

Let $\sigma, \rho$ be a pair of sets of nonnegative integers. A set of vertices of a graph $G$ is called a $(\sigma, \rho)$-dominating if for every vertex $v \in S$ $|S \cap N(v)| \in \sigma$, and for every $v \notin S$ $|S \cap N(v)| \in \rho$.

The concept of $(\sigma, \rho)$-dominating set was introduced by J.A.Telle (see [2, 3]) as generalization of some known notions (see Table 1 for examples).

| $\sigma$ | $\rho$ | $(\sigma, \rho)$-dominating set |
|---|---|---|
| $\mathrm{N}_0$ | N | dominating set |
| $\{0\}$ | $\mathrm{N}_0$ | independent set |
| $\{0\}$ | $\{1\}$ | 1-perfect code |
| $\{0\}$ | $\{0, 1\}$ | strong stable set |
| $\{0\}$ | N | independent dominating set |
| $\{1\}$ | $\{1\}$ | total perfect dominating set |

**Table 1.** *Examples of $(\sigma, \rho)$-dominating sets,* N *is the set of positive integers,* $\mathrm{N}_0$ *is the set of nonnegative integers.*

We are interested in the complexity of the problem of existence of $(\sigma, \rho)$-dominating set, which will be denoted $\exists(\sigma, \rho)$-domination problem.

It can be easily seen that if $0 \in \rho$ then $\exists(\sigma, \rho)$-domination problem has trivial solution $S = \emptyset$. So we suppose that $0 \notin \rho$. Also we suppose that $\sigma$ and $\rho$ are fixed finite sets, and complexity of $\exists(\sigma, \rho)$-domination problem is investigated for chordal graphs.

**Theorem 1.** *Let $\sigma, \rho$ be finite sets of nonnegative integers, $0 \notin \rho$. If there is a chordal graph with at least two different $(\sigma, \rho)$-dominating sets then $\exists(\sigma, \rho)$-domination problem is NP-complete for chordal graphs.*

**Theorem 2.** *Let $\sigma, \rho$ be finite sets of nonnegative integers, $0 \notin \rho$. If for every chordal graph there is no more than one $(\sigma, \rho)$-dominating set then $\exists(\sigma, \rho)$-domination problem can be solved polynomially for chordal graphs.*

These two theorems give motivation for the following problem: *for what finite sets of nonnegative integers $\sigma$ and $\rho$, $0 \notin \rho$, every chordal graph contains no more than one $(\sigma, \rho)$-dominating set?*

Main intention of our talk is to propose for consideration this problem.

At present there are only few results. In particular there are following bounds (see [1]).

Let $p = \max \sigma$, and $q = \min \rho$.

**Proposition 1.** *If $q \leq p + 1$ then there is a chordal graph with at least two $(\sigma, \rho)$-dominating sets.*

**Proposition 2.** *If $2p + 2 \leq q$ then every chordal graph contains no more than one $(\sigma, \rho)$-dominating set.*

Both bounds are tight. If $\sigma = \{p\}$ and $p + 2 \leq q$ then every chordal graph contains no more than one $(\sigma, \rho)$-dominating set. And if $\sigma$ contains two consecutive integers $i, i + 1$ and $q \leq 2p + 1$ then there is a chordal graph with at least two $(\sigma, \rho)$-dominating sets.

Also the problem investigated for special case $\sigma = \{0, p\}$.

**Proposition 3.** *If $\sigma = \{0, p\}$, $p > 0$ then every chordal graph has no more than one $\{\sigma, \rho\}$-dominating set if and only if $p + 3 \leq q$.*

It would be interesting to receive complete solution even for some special cases. For example for $\sigma = \{p_1, p_2\}$.

Also there is a finite procedure which tests existence of chordal graphs with at least two $\{\sigma, \rho\}$-dominating sets for given finite sets $\sigma$ and $\rho$, $0 \notin \rho$. Unfortunately this procedure is very uneffective. So there is another question: *can existence of chordal graphs with at least two $\{\sigma, \rho\}$-dominating sets be tested by effective algorithm?*

## References


[1] Kratochvíl J., Manuel P. and Miller M., Generalized domination in chordal graphs, Nordic Journal of Computing 2 (1995), 41–50.

[2] Telle J.A., Complexity of domination-type problems in graphs, Nordic Journal of Computing 1 (1994), 157–171.

[3] Telle J.A., Vertex partitioning problems: characterization, complexity and algorithms on partial k-trees, PhD tesisis, Department of Computer Science, Universiy of Oregon, Eugene, 1994.

**Dirk Schlatter**
schlatter@informatik.hu-berlin.de

Presented paper by S. Gerke, D. Schlatter, A. Steger, and A. Taraz

## The random planar graph process

A constrained random graph process $(P_{n,t})_{t=0}^{N}$ is a random graph process which is equipped with an additional acceptance test: after we have randomly chosen the edge to be inserted, we check whether the present graph together with this edge preserves a certain (usually structural) property. If so, we take it, otherwise we reject it (and never look at it again).

In this talk, our requirement is *planarity*, and we are mainly interested in the *evolution* of this constrained random process. It will become crucial to understand how the following two different parametrizations of the process are related. The first one, $P_{n,t_0}$, denotes the random planar graph obtained after $t_0$ edges have been considered. $P_{n,m=m_0}$, on the other hand, describes the random planar graph after $m_0$ edges have been accepted. As edges between vertices in different components are always accepted, it is obvious that $T_{n,t} \subseteq P_{n,t} \subseteq G_{n,t}$ for all $t = 0, \ldots, N$. Thus, after the connectivity threshold for $G_{n,t}$ —which lies at $t = n \log n/2$— $P_{n,t}$ must have at least $n-1$ edges with high probability. The following theorem, which states that we have to consider $\Omega(n^2)$ edges before $(1+\epsilon)n$ edges have been accepted, may thus seem somewhat surprising.

**Theorem.** *For every $\epsilon > 0$, there exists $\delta > 0$ such that*

$$\Pr[e\left(P_{n,\delta n^2}\right) \geq (1+\epsilon)n] < e^{-n}.$$

The uniform model of random planar graphs has found considerable attention in the literature over the past decade [1][2][3][4][5][6][8][9]. Recently, Giménez and Noy [7] gave rather precise asymptotic expressions for both the number of simple labelled planar graphs with $n$ vertices and $dn$ edges, and the number of those which are connected. These results yield an analytic expression for the probability that a uniform random planar graph with $dn$ edges is connected. As it turns out, this probability is bounded away from 0 and 1 for every $1 < d < 3$. From the first theorem above, we can immediately infer that this is not true for $P_{n,m=dn}$.

**Theorem.** *For every $1 < d < 3$,*

$$\Pr[P_{n,m=dn} \text{ is connected}] \longrightarrow 1 \text{ as } n \longrightarrow \infty.$$

Gerke, McDiarmid, Steger, and Weißl [4] have shown the following result about the containment of a fixed planar graph $H$ in a graph $\hat{P}_{n,m=dn}$ which is chosen uniformly at random from the class of all simple labelled planar graphs with $n$ vertices and $dn$ edges:

$\Pr[\hat{P}_{n,m=dn}$ contains at most $\alpha n$ pairwise vertex-disjoint copies of $H] < e^{-\alpha n}$,

for every $1 < d < 3$ and a positive constant $\alpha = \alpha(H, d)$.

In this respect, the two models do agree: the following analogue is our second main result.

**Theorem.** *Let $H$ be a planar graph. For every $1 < d < 3$, there exists $\alpha = \alpha(H, d) > 0$ such that*

$$\Pr[P_{n,m=dn} \text{ has at most } \alpha n \text{ pairwise vertex-disjoint copies of } H\,] < e^{-\alpha n}.$$

## References


[1] N. Bonichon, C. Gavoille, and N. Hanusse, *An information-theoretic upper bound of planar graphs using triangulations*, Symposium of Theoretical Aspects of Computer Science (STACS'03), Springer LNCS 2607, 499-510, 2003.

[2] A. Denise, M. Vasconcellos, D.J.A. Welsh, *The random planar graph*, Congr. Numer. **113**, 61-79, 1996.

[3] S. Gerke and C. McDiarmid, *On the number of edges in a random planar graph*, Combinatorics, Probability, and Computing **13**, 165-183, 2004.

[4] S. Gerke, C. McDiarmid, A. Steger, and A. Weißl, *Random planar graphs with n nodes and a fixed number of edges*, Proceedings of the 16th ACM-SIAM Symposium on Discrete Algorithms (SODA'05), 999-1007, 2005.

[5] S. Gerke, D. Schlatter, A. Steger, and A. Taraz, *The random planar graph process*, submitted.

[6] O. Giménez and M. Noy, *The number of planar graphs and properties of random planar graphs*, Proceedings of the International Conference on the Analysis of Algorithms (AofA'05), 147-156, 2005.

[7] O. Giménez and M. Noy, *Asymptotic enumeration and limit laws of planar graphs*, arXiv math.CO/0501269.

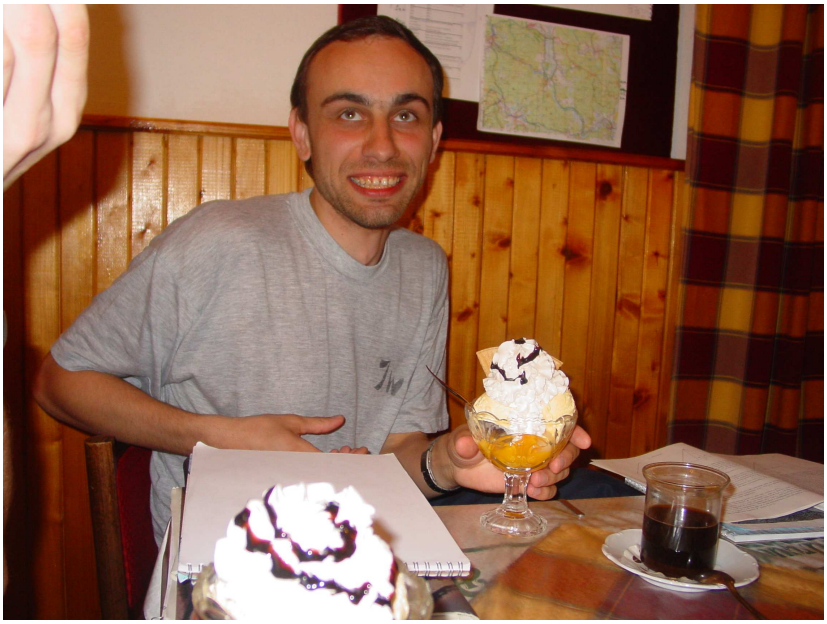
[8] C. McDiarmid, A. Steger, and D.J.A. Welsh, *Random planar graphs*, J. Combin. Theory Ser. B 93, 187-205, 2005.

[9] D. Osthus, H.J. Prömel, and A. Taraz, *On random planar graphs, the number of planar graphs and their triangulations*, J. Combin. Theory Ser. B 88, 119-134, 2003.


THEOREM
PROOF.

## Index of Speakers

## Index of Talk Titles



## List of All Participants


Robert Babilon
Charles University, Prague
Czech Republic
babilon@kam.mff.cuni.cz

Martin Bálek
Charles University, Prague
Czech Republic
balek@kam.mff.cuni.cz

Dana Bartošová
Charles University, Prague
Czech Republic
dadik@email.cz

Michael Behrisch
Humboldt University of Berlin
Germany
behrisch@informatik.hu-berlin.de

Marc Camara
Technical University of Catalonia, Barcelona
Spain
Marc.Camara@upc.edu

Dušan Daniel
Slovak Academy of Sciences, Banská Bystrica
Slovakia
daniel@savbb.sk

Zdeněk Dvořák
Charles University, Prague
Czech Republic
ook@ucw.cz

Tomáš Ebenlendr
Charles University, Prague
Czech Republic
ebik@artax.karlin.mff.cuni.cz

Louis Esperet
University of Bordeaux
France
esperet@labri.fr

Jiří Fiala
Charles University, Prague
Czech Republic
fiala@kam.mff.cuni.cz

Jiří Fink
Charles University, Prague
Czech Republic
jirka.fink@matfyz.cz

Peter Golovach
Charles University, Prague
Czech Republic
pagolovach@yahoo.com

David Hartman
Charles University, Prague
Czech Republic
hartman@kam.mff.cuni.cz

Gábor Hegedüs
Charles University, Prague
Czech Republic
guest05@kam.mff.cuni.cz

Jan Herman
Charles University, Prague
Czech Republic
hermitko@email.cz

Jan Hladký
Charles University, Prague
Czech Republic
hladk@seznam.cz

David Howard
Georgia Institute of Technology, Atlanta
USA
dmh@math.gatech.edu

Kjartan Høie
University of Bergen
Norway
kjartan.hoie@ii.uib.no

Jan Hubička
Charles University, Prague
Czech Republic
hubicka@kam.mff.cuni.cz

Vít Jelínek
Charles University, Prague
Czech Republic
jelinek@kam.mff.cuni.cz

Alexandr Kazda
Charles University, Prague
Czech Republic
alexak@atrey.karlin.mff.cuni.cz

Michal Koucký
Academy of Sciences of the Czech Republic, Prague
Czech Republic
koucky@math.cas.cz

Jan Kratochvíl
Charles University, Prague
Czech Republic
honza@kam.mff.cuni.cz

Marek Krčál
Charles University, Prague
Czech Republic
marek.krcal@seznam.cz

Petr Lopez Kučera
Czech University of Agriculture, Prague
Czech Republic
kucera@pef.czu.cz

Arnaud Labourel
University of Bordeaux
France
labourel@labri.fr

Bernard Lidický
Charles University, Prague
Czech Republic
bernard@matfyz.cz

Peter Liske
Humboldt University of Berlin
Germany
liske@informatik.hu-berlin.de

Anna de Mier
Charles University, Prague
Czech Republic
demier@kam.mff.cuni.cz

Amanda Montejano
Technical University of Catalonia, Barcelona
Spain
amanda@mat.upc.es

Jordi Moragas
Technical University of Catalonia, Barcelona
Spain
jmoragas@mat.upc.es

Roman Nedĕla
Slovak Academy of Sciences, Bansk Bystrica
Slovakia
nedela@savbb.sk

Eva Ondráčková
Charles University, Prague
Czech Republic
efa@matfyz.cz

Martin Pergel
Charles University, Prague
Czech Republic
perm@kam.mff.cuni.cz

Paul Raff
Rutgers University, New Brunswick
USA
praff@math.rutgers.edu

Juanjo Rue
Technical University of Catalonia, Barcelona
Spain
Juanjo.rue@gmail.com

Robert Šámal
Charles University, Prague
Czech Republic
samal@kam.mff.cuni.cz

Dirk Schlatter
Humboldt University of Berlin
Germany
schlatter@informatik.hu-berlin.de

Jiří Sgall
Academy of Sciences of the Czech Republic, Prague
Czech Republic
sgall@math.cas.cz

Petr Škovroň
Charles University, Prague
Czech Republic
xofon@pikomat.mff.cuni.cz

Matěj Stehlík
Charles University, Prague
Czech Republic
matej@matem.unam.mx

Marek Sterzik
Charles University, Prague
Czech Republic
marek_sterzik@volny.cz

Rudolf Stolař
Charles University, Prague
Czech Republic
riv@email.cz

Ondřej Suchý
Charles University, Prague
Czech Republic
ondra@s.cz

Radovan Šesták
Charles University, Prague
Czech Republic
radofan@gmail.com

Shuang Wang
University of Bergen
Norway
shuang.wang@student.uib.no

Ondřej Zajíček
Charles University, Prague
Czech Republic
santiago@mail.cz

Mariano Zelke
Humboldt University of Berlin
Germany
zelke@informatik.hu-berlin.de

More information about the Spring School can be found at
`http://kam.mff.cuni.cz/~spring/`