Adam Beneš, Petr Chmel (eds.)

Preface

Spring school on Combinatorics has been a traditional meeting organized for more than 40 years for faculty and students participating in the Combinatorial Seminar at Faculty of Mathematics and Physics of the Charles University. It is internationally known and regularly visited by students, postdocs and teachers from our cooperating institutions in the DIMATIA network. As it has been the case for several years, this Spring School is supported by Computer Science Institute (IÚUK) of Charles University, the Department of Applied Mathematics (KAM) and by some of our grants (SVV, UNCE, Progres). This year we are glad we can also acknowledge generous support by the RSJ Foundation.

The Spring Schools are entirely organized and arranged by our students. The topics of talks are selected by supervisors from the Department of Applied Mathematics (KAM) and Computer Science Institute (IÚUK) of Charles University as well as from other participating institutions. In contrast, the talks themselves are almost exclusively given by students, both undergraduate and graduate. 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 is organized in Nečtiny, Hrad Nečtiny (in Plzeň Uplands in western Bohemia) with a great variety of possibilities for outdoor activities.

Robert Šámal, Pavel Veselý
Adam Beneš, Petr Chmel, Barbora Dohnalová, Júlia Križanová

KAM

CSI
IÚUK

RSJ

Contents

Schedule of talks

	9:00	10:00	11:00	12:00
Saturday *slides room*	An Exact Algorithm for the Unanimous Vote Problem Vojtěch Gaďurek 9:00 – 10:00 38	Listing 6-Cycles Katarzyna Szwed 10:10 – 11:10 67	The number of odd spanning trees in the complete graphs Karolína Hylasová 11:20 – 12:20 40	Lunch 12:30 – 13:30
Saturday *whiteboard room*	Kempe equivalence of 4-colorings of graphs on non-orientable surfaces Jakub Štepo 9:00 – 10:00 65	Lonely edges in cubic graphs Matúš Matok 10:10 – 11:10 54		Lunch 12:30 – 13:30
Sunday *slides room*	Differential privacy from axioms Jakub Komárek 9:00 – 10:00 45	On k-colorability of (bull, H)-free graphs Richarlotte Valérà Razafindravola 10:10 – 11:10 59	Sublinear-Time Algorithm for MST-Weight Revisited Filip Filipkowski 11:20 – 12:20 36	Lunch 12:30 – 13:30
Sunday *whiteboard room*	A note on strongly and totally chain intersecting families Rachel Dufau-Sansot 9:00 – 10:00 23	A Simple Analysis of Ranking in General Graphs Samuel Vaško 10:10 – 11:10 74	Playing Snake on a Graph Michal Katrlík 11:20 – 12:20 43	Lunch 12:30 – 13:30
Monday *slides room*	Verifying Two Lines of C with Why3: An Exercise in Program Verification Bartosz Wójcik 9:00 – 10:00 76	Simulating Time With Square-Root Space Jakub Żuchowski 10:10 – 11:10 78	A note on color-bias perfect matchings in hypergraphs Terezie Lejbová 11:20 – 12:20 52	Lunch 12:30 – 13:30
Monday *whiteboard room*	A proof of the 3/4-conjecture for the total domination game Jakub Smolík 9:00 – 10:00 63	Density of rainbow triangles and properly colored K_4's Adam Džavoronok 10:10 – 11:10 29	On the Parameterized Complexity of Diverse SAT Bogdan Błędziński 11:20 – 12:20 16	Lunch 12:30 – 13:30
Tuesday *slides room*	Efficient Catalytic Graph Algorithms Ján Plachý 9:00 – 10:00 58	Dudeney's Dissection Is Optimal Eliška Červenková 10:10 – 11:10 19	Space-Efficient Hierholzer: Eulerian Cycles in $O(m)$ Time and $O(n)$ Space Barbora Dohnalová 11:20 – 12:20 25	Lunch 12:30 – 13:30
Tuesday *whiteboard room*	The induced saturation number for V_3 is linear Wiktor Szymonek 9:00 – 10:00 69	Fair allocation: What if we allow sharing? Hana Salavcová 10:10 – 11:10 61	An Equilateral Triangle of Side $> n$ Cannot be Covered by $n^2 + 1$ Unit Equilateral Triangles Homothetic to it Alica Dományová 11:20 – 12:20 27	Lunch 12:30 – 13:30
Wednesday	Trip Day 9:00 –			
Thursday *slides room*	Cryptography: Proving closeness to Reed-Solomon codes Kristýna Mašková 9:00 – 10:00 7	Cryptography: Proving closeness to Reed-Solomon codes Jan Frnka 10:10 – 11:10 9	Integer Area Dissections of Lattice Polygons via a Non-Abelian Sperner's Lemma Aleksa Džuklevski 11:20 – 12:20 31	Lunch 12:30 – 13:30
Thursday *whiteboard room*	Optimal embeddings of posets in hypercubes Adam Beneš 9:00 – 10:00 14	On the Number of Compositions of Two Polycubes Daniel Čech 10:10 – 11:10 17	Cryptography: Post Quantum Cryptography from Lattices Sejin Park 11:20 – 12:20 11	Lunch 12:30 – 13:30

	18:00	19:00	20:00	21:00
Friday	Arrival – 18:00	Dinner 18:00 – 18:45	Competitive Online Transportation Simplified Igor Januszkiewicz 19:00 – 20:00 41	
Saturday		Dinner 18:00 – 19:00	Shannon Switching game and its generalization Júlia Križanová 19:00 – 20:00 49	
Sunday		Dinner 18:00 – 19:00	Practical aspects of logic & Lean Mirek Olšák 19:00 – 20:00	Pub-quiz 20:00 – 22:00
Monday		Dinner 18:00 – 19:00	Introduction to discharging and density of 5/2-critical graphs Martin Kopřiva 19:00 – 20:00 48	
Tuesday		Dinner 18:00 – 19:00	The Rosenfeld counting method David Mikšaník 19:00 – 20:00 56	
Wednesday	Trip Day – 18:00	Dinner 18:00 – 19:00	Using Hardness vs Randomness to Design Low-Space Algorithms Petr Chmel 19:00 – 20:00 21	
Thursday		Dinner 18:00 – 19:00	Introduction to Quantum Algorithms Roman Edenhofer 19:00 – 20:00 33	

Series Talks

Kristýna Mašková
maskovakristyna25@gmail.com

Presented paper by -

Proving closeness to Reed-Solomon codes
as part of series Cryptography

(https://doi.org/10.4230/LIPIcs.ICALP.2018.14)

Introduction

In modern cryptography, we have many protocols that rely on polynomials of bounded degree. To make these protocols secure, we need a way to check that a massive table of values actually corresponds to a polynomial of the correct degree. The naive way to check that is to interpolate (read all evaluations) and check the degree. However, if the degree is large, this approach can be very inefficient. FRI is an interactive protocol that allows us to verify that a function is close to a degree d polynomial while reading only $\mathcal{O}(\log d)$ values. The goal of this talk is to introduce FRI and show how it achieves this.

Definition 1 *The Reed Solomon code over the field $\mathbb{F}$, evaluation domain $H \subseteq \mathbb{F}$ and degree d is the linear code*

$$\mathsf{RS}[\mathbb{F}, H, d] = \{p : H \to \mathbb{F} \text{ where } p \in \mathbb{F}^{<d}[X]\}.$$

By evaluating a function f at every point in the domain $H = \{h_1, \ldots, h_n\}$, we can naturally view it as a vector of length $n = |H|$ in $\mathbb{F}^n$:

$$\mathrm{vec}(f) = (f(h_1), \ldots, f(h_n)).$$

(Further, we will omit the notation vec and just write f for both the polynomial on H and a vector of its evaluations.)

This allows us to measure the distance between any two functions $f, g : H \to \mathbb{F}$ using the fractional Hamming distance:

$$\Delta(f, g) = \frac{1}{n}|f - g|_0.$$

Definition 2 *For $\delta \in (0, 1)$, we say that a function $f : H \to \mathbb{F}$ is δ-close to $\mathsf{RS}[\mathbb{F}, H, d]$ if there is some polynomial $p \in \mathsf{RS}[\mathbb{F}, H, d]$ s.t. $\Delta(f, p) \leq \delta$. We write $\Delta(f, \mathsf{RS}[\mathbb{F}, H, d]) \leq \delta$.*

We say that a function $f : H \to \mathbb{F}$ is δ-far from $\mathsf{RS}[\mathbb{F}, H, d]$ if for all polynomials $p \in \mathsf{RS}[\mathbb{F}, H, d]$ we have $\Delta(f, p) > \delta$. We write $\Delta(f, \mathsf{RS}[\mathbb{F}, H, d]) > \delta$.

Definition 3 *For a function f define the even and odd parts of f as*

$$f_{even}(X^2) = \frac{f(X) + f(-X)}{2} \quad \text{and} \quad f_{odd}(X^2) = \frac{f(X) - f(-X)}{2X}$$

Using the even and odd parts and an element $r \in \mathbb{F}$ we can define a fold of f as

$$\mathsf{Fold}[f, r] = f_{even}(X) + r \cdot f_{odd}(X) \in \mathbb{F}^{<d/2}[X]$$

Denote $k = \log d$ the number of FRI rounds and $q \in \mathbb{N}$ the security parameter. For evaluation domain H and $i \in [k]$, we write

$$H^{2^i} = \{h^{2^i} \mid h \in H\}.$$

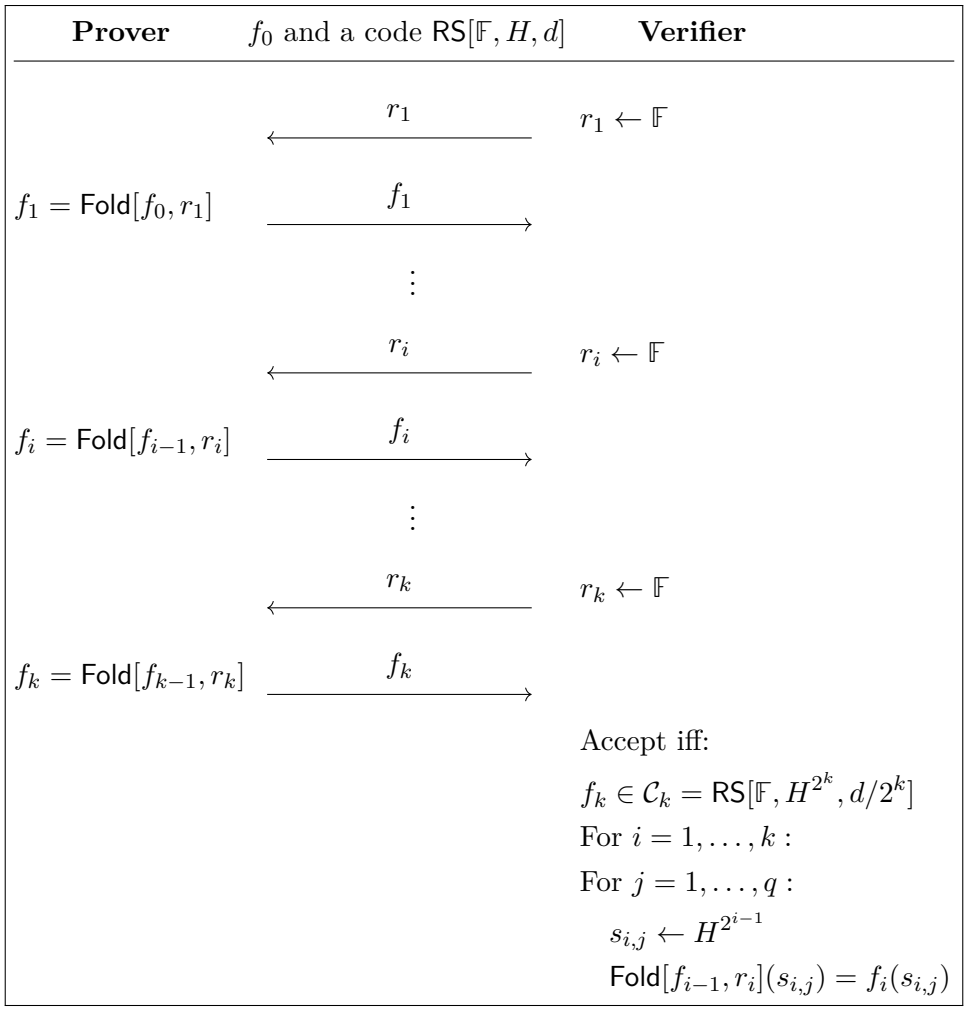


Figure 1: FRI Protocol

Definition 4 (Correctness / Completeness) *The FRI protocol is (perfectly) complete if for every valid function $f \in \mathsf{RS}[\mathbb{F}, H, d]$ the Verifier V always accepts. Formally,*

$$\Pr[\langle P, V(f)\rangle = 1] = 1.$$

Definition 5 (Soundness) *The FRI protocol has soundness error ε if for every function f that is δ-far from $\mathsf{RS}[\mathbb{F}, H, d]$, and for every possible (potentially malicious) Prover strategy P^*, the probability that the Verifier V accepts is at most ε. Formally,*

$$\Pr[\langle P^*, V(f)\rangle = 1] \leq \varepsilon.$$

Jan Frnka
jan.frnka@gmail.com

Presented paper by -

Proving closeness to Reed-Solomon codes *as part of series* Cryptography

(https://eprint.iacr.org/2025/1993.pdf)

Introduction

Having introduced the FRI protocol in the previous talk, we now focus on its formal guarantees. In this talk, we provide a soundness proof for the FRI variant using uncorrelated queries across rounds. We conclude by sketching how this proof extends to the more efficient version with correlated queries.

Preliminaries

Let ω be a primitive n-th root of unity in a finite field $\mathbb{F}$. We denote

$$H = \{1, \omega, \omega^2, \ldots, \omega^{n-1}\}$$

Definition 1 *For $f\colon H \to \mathbb{F}$ define the even and odd part of f as*

$$f_{even}(X^2) = \frac{f(X) + f(-X)}{2} \quad \text{and} \quad f_{odd}(X^2) = \frac{f(X) - f(-X)}{2X}$$

Using the even and odd parts we can define a fold of f as $\mathsf{Fold}[f, r]\colon H^2 \to \mathbb{F}$

$$\mathsf{Fold}[f, r](X) = f_{even}(X) + r \cdot f_{odd}(X).$$

Prover	f_0 and a code $\mathsf{RS}[\mathbb{F}, H, d]$	**Verifier**
	$\xleftarrow{\quad r_1 \quad}$	$r_1 \leftarrow \mathbb{F}$
$f_1 = \mathsf{Fold}[f_0, r_1]$	$\xrightarrow{\quad f_1 \quad}$	
	$\vdots$	
	$\xleftarrow{\quad r_k \quad}$	$r_k \leftarrow \mathbb{F}$
$f_k = \mathsf{Fold}[f_{k-1}, r_k]$	$\xrightarrow{\quad f_k \quad}$	
		Accept iff:
		$f_k \in \mathcal{C}_k = \mathsf{RS}[\mathbb{F}, H^{2^k}, d/2^k]$
		For $i = 1, \ldots, k$:
		For $j = 1, \ldots, q$:
		$s_{i,j} \leftarrow H^{2^{i-1}}$
		$\mathsf{Fold}[f_{i-1}, r_i](s_{i,j}) = f_i(s_{i,j})$

Figure 2: FRI Protocol

Definition 2 (Soundness) *The FRI protocol has soundness error ε if for every function f that is δ-far from $\mathsf{RS}[\mathbb{F}, H, d]$, and for every possible (potentially malicious) Prover strategy P^*, the probability that the Verifier V accepts is at most ε. Formally,*

$$\Pr[\langle P^*, V(f)\rangle = 1] \leq \varepsilon.$$

Fact 3 (Correlated agreement) *Let $\rho = \frac{d}{n}$ be the rate of the Reed-Solomon code and $\delta \in (0, 1 - \sqrt{\rho})$. Then for a function f δ-far from $\mathsf{RS}[\mathbb{F}, H, d]$, the folded function $\mathsf{Fold}[f, r]$ is δ-far from $\mathsf{RS}[\mathbb{F}, H^2, \frac{d}{2}]$, with high probability over the choice of r. Formally, there exists $\varepsilon_{CA} > 0$, s.t. $\forall f$ δ-far from $\mathsf{RS}[\mathbb{F}, H, d]$ we have*

$$\Pr_{r \leftarrow \mathbb{F}}\left[\mathsf{Fold}[f, r] \text{ is } \delta\text{-close to } \mathsf{RS}\left[\mathbb{F}, H^2, \frac{d}{2}\right]\right] < \varepsilon_{CA},$$

where $\varepsilon_{CA} \in \mathcal{O}(\frac{n}{|\mathbb{F}|})$.

Fact 4 (Union Bound) *Let $A_1, A_2, \ldots, A_\ell$ be a collection of events. Then*

$$\Pr\left[\bigcup_{i=1}^{\ell} A_i\right] \leq \sum_{i=1}^{\ell} \Pr[A_i].$$

Fact 5 (AM-GM) *Let $x_1, x_2, \ldots, x_n \geq 0$. Then*

$$\frac{x_1 + x_2 + \cdots + x_n}{n} \geq (x_1 x_2 \cdots x_n)^{1/n}.$$

Fact 6 *For all $x \in \mathbb{R}$,*

$$1 - x \leq e^{-x}.$$

Main theorem

Theorem 7 *Let ρ be the rate of a Reed-Solomon code, $\delta \in (0, 1 - \sqrt{\rho})$, k be the number of FRI rounds and q the number of Verifier queries. Then for ε the soundness error of FRI, we have the following bound*

$$\varepsilon \leq k \cdot \varepsilon_{CA} + e^{-\delta q}.$$

Sejin Park
ilesejin@snu.ac.kr

Post Quantum Cryptography from Lattices
as part of series Cryptography

Introduction

Many cryptographic systems are based on hard mathematical problems such as integer factorization and the discrete logarithm problem. However, these problems can be efficiently solved using quantum algorithms, making such systems vulnerable to quantum computers. To address this, we need cryptographic schemes that remain secure in the presence of quantum adversaries. One promising approach is lattice-based cryptography, which is currently one of the leading candidates for post-quantum cryptography due to its strong security guarantees and practical efficiency. Moreover, lattice-based cryptography enables computations to be performed directly on encrypted data, a property that underlies fully homomorphic encryption. This talk introduces the fundamental techniques of lattice-based cryptography, focusing on the Short Integer Solution (SIS) problem and the Learning With Errors (LWE) problem. Finally, we will explore their applications and examine cryptographic schemes based on these assumptions.

Lattices

Definition 1 *An n-dimensional lattice $\mathcal{L}$ is any subset of $\mathbb{R}_n$ that is both:*

1. *an additive subgroup: $\mathbf{0} \in \mathcal{L}$, and $-\mathbf{x}, \mathbf{x}+\mathbf{y} \in L$ for every $\mathbf{x}, \mathbf{y} \in \mathcal{L}$*
2. *discrete: every $\mathbf{x} \in \mathcal{L}$ has a neighborhood in $\mathbb{R}_n$ in which $\mathbf{x}$ is the only lattice point*

Since a lattice is a finite dimensional vector space, it could be generated by a finite basis. We can define a lattice using the basis vectors.

$$\mathcal{L}(\mathbf{b_1}, \mathbf{b_2}, \ldots, \mathbf{b_n}) = \left\{ \sum_{i \in [n]} c_i \mathbf{b_i} \middle| c_i \in \mathbb{Z} \right\}$$

In this talk, we will only handle integer lattices where all the points on the lattices are integer points.

The Short Integer Solutions Problem

Definition 2 ***(SIS)*** *Let $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ be a uniform randomly sampled matrix. The* ***Short Integer Solution (SIS) problem****, denoted $SIS_{n,q,\beta,m}$, is to find a nonzero integer vector $\mathbf{z} \in \mathbb{Z}_q^m$ such that*

$$\mathbf{Az} \equiv 0 \ (mod\, q) \text{ and } \|\mathbf{z}\| \leq \beta.$$

Ajtai proved the following theorem regarding the hardness of SIS.

Theorem 3 *For any $m = poly(n), \beta > 0, q \geq \beta \cdot poly(n)$, solving $SIS_{n,q,\beta,m}$ is as hard as solving $GapSVP_\gamma$ and $SIVP_\gamma$ on arbitrary n-dimensional lattices for some $\gamma = \beta \cdot poly(n)$.*

There are no known efficient quantum algorithms for solving problems related to the Shortest Vector Problem (SVP), which suggests that SIS remains secure even against quantum adversaries. Note that n is the primary parameter determining the security, while m and q only need to satisfy the

required constraints. β must be significantly smaller than q, but not too small so that it guarantees the existence of a solution.

Observe that multiplication by the matrix A defines a one-way function when restricted to short vectors. Thus, a hash function can be naturally derived from SIS.

Theorem 4 ***(Ajtai Hash Function)*** *Define the hash function H on $\{0,1\}^m$ as follows:*

$$H : \mathbb{Z}_q^{n\times m} \times \{0,1\}^m \to \mathbb{Z}_q^n, H(\mathbf{A}, \mathbf{x}) = \mathbf{A} \cdot \mathbf{x}$$

Then, H is a collision-resistant hash function.

The main advantage of this lattice based hash function is its linearity. This allows the hash value to be updated efficiently as follows. Given two inputs x and x', compute their difference $\Delta x = x' - x$ and compute $y = A\Delta x$. If the change is small, then Δx has only a few nonzero entries, so the multiplication can be computed efficiently. Finally, we obtain: $Ax' = H(A, x) + y$.

The Learning with Errors Problem

The Learning with Errors Problem (LWE) was introduced by Regev, and it enabled the construction of encryption schemes using lattices.

Definition 5 ***(LWE instance)*** *Let $\mathbf{s} \in \mathbb{Z}_q^n$ be a secret vector and let χ denote a discrete Gaussian distribution over $\mathbb{Z}_q$ with small width relative to q. An **LWE instance** is sampled from the **LWE distribution** $A_{\mathbf{s},\chi}$ over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ by choosing $\mathbf{a} \in \mathbb{Z}_q^n$ uniformly random, sampling $e \leftarrow \chi$, and outputting $(\mathbf{a}, b \equiv \langle \mathbf{s}, \mathbf{a} \rangle + e \bmod q)$.*

Definition 6 ***(Decision LWE)*** *The decision LWE problem is to distinguish between the following two experiments, given m instances $(\mathbf{a_i}, b_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$:*
Experiment 0: $(\mathbf{a_i}, b_i) \leftarrow A_{\mathbf{s},\chi}$ for a fixed secret $\mathbf{s}$
Experiment 1: $(\mathbf{a_i}, b_i) \xleftarrow{\$} \mathbb{Z}_q^n \times \mathbb{Z}_q$

We can rearrange the entries to make a single linear relation.

$$\mathbf{As} = \mathbf{b} + \mathbf{e}, \mathbf{A} = \begin{bmatrix} \mathbf{a_1}^T \\ \mathbf{a_2}^T \\ \vdots \\ \mathbf{a_m}^T \end{bmatrix}, \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

Then, the decision LWE problem is to distinguish the pair $(\mathbf{A}, \mathbf{b})$ from uniform random.

The following hardness theorem is due to Regev.

Theorem 7 *For $m = poly(n), q \leq 2^{poly(n)}$ and discrete Gaussian error distribution χ with $\alpha q \geq 2\sqrt{n}$, solving the decision LWE problem is at least as hard as solving $GapSVP_\gamma$ and $SIVP_\gamma$ on arbitrary n-dimensional lattices for $\gamma = \bar{O}(n/\alpha)$.*

Therefore, the decision LWE is considered quantum-secure. Note that this is a very strong assumption: not only is recovering the secret $\mathbf{s}$ believed to be hard, but even distinguishing the LWE instances from uniformly random pairs is a hard problem.

Ring-SIS and Ring-LWE

For computational efficiency, the ring variants of the lattice problems were introduced.

Definition 8 *Define the polynomial ring $R = \mathbb{Z}[X]/(f(X))$ where $f(X) = X^{2^k} + 1$. Define a norm $\|\cdot\|$ in the ring R. Define $R_q \coloneqq R/qR = \mathbb{Z}_q[X]/(f(X))$.*

Definition 9 ***(Ring-SIS)*** *A uniformly random vector $\vec{a} \in R_q^m$ is given. The Ring-SIS problem is to find a nonzero vector $\vec{z} \in R^m$ such that $\|\vec{z}\| \leq \beta$ and $\langle \vec{a}, \vec{z} \rangle = 0$.*

Definition 10 ***(Ring-LWE instance)*** *Let $s \in R_q$ be the secret and let χ denote the discrete gaussian distribution. A Ring-LWE instance is sampled from the Ring-LWE distribution $A_{s,\chi}$ over $R_q \times R_q$ by choosing $a \in R_q$ uniformly random, choosing $e \leftarrow \chi$, and outputting $(a, b = a \cdot s + e)$.*

Definition 11 ***(Decision Ring-LWE)*** *The decision Ring-LWE problem is to distinguish between the following two experiments, given m instances $(a_i, b_i) \in R_q \times R_q$.*
Experiment 0: $(a_i, b_i) \leftarrow A_{s,\chi}$ for a fixed secret s
Experiment 1: $(a_i, b_i) \xleftarrow{\$} R_q \times R_q$

It is also been shown that the ring variants of SIS and LWE are secure against quantum adversaries. These ring-based problems can be viewed as structured versions of SIS or LWE. While this additional algebraic structure introduces efficiency benefits—most notably enabling faster arithmetic via polynomial multiplication in the ring—it also leads to significant performance improvements in practical implementations.

Regev Encryption

Using LWE, we can construct a public key encryption scheme, which is called the Regev Encryption.

- **Key Generation:** The private key $\mathbf{s}$ is sampled uniformly from $\mathbb{Z}_q^n$. The public key is sampled from the LWE distribution, which leads to $\text{pk} = (\mathbf{A}, \mathbf{v} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e})$. $(\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n})$
- **Encryption:** Sample a random binary vector $\mathbf{r} \xleftarrow{\$} \{0,1\}^m$ and compute
$$\mathbf{c_0} = \mathbf{r}^T \mathbf{A}, c_1 = \mathbf{r}^T \mathbf{v} + \left\lfloor \frac{q}{2} \right\rceil \cdot x$$
Output $\text{ct} = (\mathbf{c_0}, c_1)$.
- **Decryption:** For the given ciphertext $\text{ct} = (\mathbf{c_0}, c_1)$, compute $\tilde{x} = c_1 - \mathbf{c_0} \cdot \mathbf{s}$ and return 0 if closer to 0, return 1 if closer to $\left\lfloor \frac{q}{2} \right\rceil$.

One can easily derive an analogue of the Regev Encryption based on the Ring-LWE problem. (Put $\text{pk} = (a, b = a \cdot s + e)$)

One advantage of Regev Encryption is that it is additively homomorphic. In other words, if $c_1 = \text{Enc}(m_1), c_2 = \text{Enc}(m_2)$ then $\text{Dec}(c_1 + c_2) = m_1 + m_2$. However, the error term grows with each homomorphic addition, and may eventually flood into the encrypted data. Therefore, only a limited number of additions can be performed while keeping the error within the allowable range.

We can extend this scheme to a fully homomorphic encryption (FHE) scheme by introducing two key techniques: homomorphic multiplication and bootstrapping. Homomorphic multiplication enables arbitrary operations on encrypted data, while bootstrapping reduces the accumulated noise in ciphertexts and effectively refreshes them. There are several well-known FHE schemes, including BGV, TFHE, and CKKS, some of which are already beginning to see practical use in real-world applications.

Bibliography

[1] Chris Peikert et al. *A decade of lattice cryptography.* Foundations and Trends in Theoretical Computer Science, 10(4):283–424, 2016.

Standalone Talks

Adam Beneš

ad.benes@gmail.com

Presented paper by Tomáš Flídr, Maria-Romina Ivan and Seab Jaffe

Optimal Embeddings of Posets in Hypercubes

(https://arxiv.org/abs/2509.26630v1)

Motivation

Every finite poset can be embedded into a Boolean hypercube ordered by inclusion. A natural question is how *low* such an embedding can be placed, and in which hypercube dimension an optimal low embedding first appears.

These questions are captured by two parameters introduced in connection with induced poset saturation: the *hypercube-height* $h^*(P)$ and the *hypercube-width* $w^*(P)$.

The main result of the paper is that every finite poset admits an optimal low embedding in a hypercube of dimension at most its size.

Definitions

Let Q_n denote the Boolean lattice on $[n]$, ordered by inclusion. For integers $h \leq w$, let

$$\binom{[w]}{\leq h}$$

denote the induced subposet of Q_w consisting of all subsets of $[w]$ of size at most h.

Definition 1 *For a finite poset P, the **hypercube-height** $h^*(P)$ is the minimum integer h such that there exists $n \in \mathbb{N}$ for which $\binom{[n]}{\leq h}$ contains an induced copy of P.*

Definition 2 *For a finite poset P, the **hypercube-width** $w^*(P)$ is the minimum integer w such that $\binom{[w]}{\leq h^*(P)}$ contains an induced copy of P.*

Thus, $h^*(P)$ measures how low P can be embedded, while $w^*(P)$ measures how many coordinates are needed to realise such an optimal embedding.

Background

A standard embedding shows

$$h^*(P) \leq |P|.$$

In fact, it is known that

$$h^*(P) \leq |P| - 1.$$

This bound is tight, for example for chains.

The difficult parameter is $w^*(P)$. Previously it was shown that

$$w^*(P) \leq |P| h^*(P),$$

which yields only a quadratic upper bound in general. Bastide, Groenland, Ivan and Johnston conjectured that in fact

$$w^*(P) \leq |P|$$

for every finite poset P.

Main Result

Theorem 3 *For every finite poset P, we have*

$$w^*(P) \leq |P|.$$

This proves the conjecture from the earlier paper.

Two-layered Posets

The paper first proves the result for two-layered posets, where every element is either minimal or maximal. In that case the argument is more concrete: some minimal elements can be replaced by singletons, and maximal elements are adjusted accordingly.

This special case gives useful intuition for the general construction.

Consequences and Questions

The result improves the general upper bound for induced poset saturation obtained by previous methods. In particular, it implies

$$\mathrm{sat}^*(n, P) \leq 2n|P| - 1$$

for sufficiently large n.

The paper ends with several structural questions, for example:

- For which posets do we have
$$h^*(P) = |P| - 1?$$
- Is there a tight general relation between $h^*(P)$ and $w^*(P)$?
- For which posets do we have
$$w^*(P) = |P|?$$

Takeaway

The main message is that an optimally low embedding of a finite poset never needs more than $|P|$ coordinates. The key technical tool is Hall's theorem, used to compress an existing embedding without increasing its height.

Bogdan Błędziński

bb439950@students.mimuw.edu.pl

Presented paper by Neeldhara Misra, Harshil Mittal, Ashutosh Rai

On the Parameterized Complexity of Diverse SAT

(https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ISAAC.2024.50)

Introduction

Sometimes, finding just one solution for a given instance of a problem is not sufficient for us. We want to know something more about the set of solutions or find solutions that are sufficiently different from each other. In diversity variants of the problem, we aim to find diverse solutions. In this talk, it will be presented how to find two solutions for a given instance of Affine-SAT that are dissimilar and see how much more difficult this problem becomes.

Definitions

Definition 1 (Affine formula) *Affine formula is a Boolean formula that is a conjunction of linear equations over $\mathbb{F}_2$. A formula is n-affine if it is affine and each of its equations contains at most n variables. A formula is (n, k)-affine if it is n-affine and every variable appears in at most k equations.*

Definition 2 (Exact Differ Affine-SAT problem) *Given an affine formula Φ and an integer d, decide whether there are two satisfying assignments σ_1, σ_2 of Φ that differ on exactly d variables.*

Definition 3 (Max Differ Affine-SAT problem) *Given an affine formula Φ and an integer d, decide whether there are two satisfying assignments σ_1, σ_2 of Φ that differ on at least d variables.*

Definition 4 (Independent Set on Cubic Graphs problem) *Given an undirected graph G in which every vertex has degree 3, and an integer k, decide whether there exists a subset X of vertices of G such that $|X| = k$ and no two vertices in X are connected with an edge. This problem is known to be* NP*-hard.*

Definition 5 (Exact Even Set problem) *Given a universe U, a family F of subsets of U, and an integer k, decide whether there exists a set $X \subseteq U$ such that $|X| = k$ and for every set $S \in F$ $|X \cap S|$ is even. This problem is known to be W[1]-hard parametrized by k.*

Theorems

Exact Differ Affine-SAT	Max Differ Affine-SAT
Polynomial-time on 2-affine formulas	Polynomial-time on 2-affine formulas
NP-hard, even on (3, 4)-affine formulas	NP-hard, even on 4-affine formulas
W[1]-hard parametrized by d	FPT parametrized by d

Daniel Čech
cech.daniel01@gmail.com
Presented paper by Andrei Asinowski, Gill Barequet, Gil Ben-Shachar, Martha Carolina Osegueda, Günter Rote

On the Number of Compositions of Two Polycubes

(https://doi.org/10.57717/cgt.v3i1.41)

Introduction

Polyominoes are objects similar to Tetris pieces. We will show lower and upper bound on the number of compositions of such objects in two dimensions. Also an algorithm for finding the exact number of such compositions will be introduced. This can be, for example, useful for determining how likely two polymers are to connect in a solution.

Main part

Definition 1 (Polycube) *A d-dimensional polycube is a connected set of cells on the cubical lattice $\mathbb{Z}^d$, where the connectivity is through $(d-1)$-dimensional faces.*

Definition 2 (Polyomino) *A polyomino is a 2-dimensional polycube.*

Definition 3 (Polycube's size) *The size of a polycube is the number of d-dimensional cells it contains.*

Definition 4 (Composition of two polycubes) *A composition of two d-dimensional polycubes is the placement of one of them relative to the other, such that they touch each other (sharing one of more $(d-1)$-dimensional faces) but do not overlap, so that the union of their cell sets is a valid polycube.*

Theorem 5 (Minimum number of compositions in two dimensions)

1. *Any two polyominoes of sizes n_1 and n_2 have $\Omega(\sqrt{n_1+n_2})$ compositions.*
2. *For every two numbers $n_1 \geq 1, n_2 \geq 1$, there is a pair of polyominoes of sizes n_1 and n_2 with $\Theta(\sqrt{n_1+n_2})$ compositions.*

Observation 6 (Maximum number of compositions in two dimensions) *Any two polyominoes of sizes n_1 and n_2 have $O(n_1 n_2)$ compositions.*

Theorem 7 *For every $n \geq 1$, there are two polyominoes, each of size at most n, that have at least*

$$\frac{n^2}{2^{8\cdot\sqrt{\log_2 n}}}$$

compositions.

The near-quadratic bound is achieved via a recursive construction of polyominoes called toothbrushes D_k (dense) and S_k (sparse) of order k, using a degree parameter r.

Lemma 8 *The sizes of D_k and S_k are bounded from above by $O(r^{k+1})$.*

Lemma 9 (Number of compositions lower bound) *There are at least r^{2k} compositions of D_k and S_k.*

Definition 10 (Notch coordinates) *The i-th coordinate of a notch is defined as the index (from 0 to $r-1$) of D_{i-1}'s (or S_{i-1}) copy on the i-stick from its apex.*

Lemma 11 (Bounding boxes claim) *Given a composition of D_k and S_k via the notches N_D and N_S. For $1 \leq i \leq k$, $B(D_i)$ (D_i is the copy in which N_D lies) and $B(S_i)$ do not overlap the rim of the other toothbrush.*

Lemma 12 (Bounding boxes overlapping claim) *Given D_i and S_i of order $i \geq 2$, if their sub-brushes D_{i-1} and S_{i-1} have overlapping bounding boxes, then no other pair ($D_{i-1}{}'$, $S_{i-1}{}'$), where $D_{i-1}{}'$ and $S_{i-1}{}'$ are other copies on the same level, can overlap.*

Definition 13 *Minkowski sum of A and B is defined as $A \oplus B := \{a + b | a \in A, b \in B\}$.*

Theorem 14 *Given two polyominoes, each of size at most n, their number of compositions can be computed in $O(n^2)$ time and $O(n^2)$ space.*

Bibliography

[1] Asinowski, A., Barequet, G., Ben-Shachar, G., Osegueda, M. C., & Rote, G. (2024). On the Number of Compositions of Two Polycubes. Computing in Geometry and Topology, 3(1), 4:1–4:18. https://doi.org/10.57717/cgt.v3i1.41

[2] G. Barequet and R. Barequet, An improved upper bound on the growth constant of polyominoes, Proc. 8th European Conf. on Combinatorics, Graph Theory, and Applications, Bergen, Norway, August-September 2015, Electronic Notes in Discrete Mathematics, 49, 67–172, November 2015, doi:10.1016/j.endm.2015.06.025.

Eliška Červenková
cervenel@kam.mff.cuni.cz

Presented paper by Erik D. Demaine, Tonan Kamata, and Ryuhei Uehara

Dudeney's Dissection Is Optimal

(https://arxiv.org/abs/2412.03865)

Introduction

Dissection is the process of transforming one shape P into another shape P' by cutting P into pieces and re-arranging those pieces to form P' (without overlap). In 1907, Henry Ernest Dudeney demonstrated a four-piece solution to his puzzle "cut any equilateral triangle ... into as few pieces as possible that will fit together and form a perfect square", which today remains perhaps the most famous example of dissection. Over a century later, Demaine, Kamata, and Uehara finally solved Dudeney's puzzle by proving the following theorem:

Theorem 1 *There is no dissection with three or fewer polygonal pieces between a square and an equilateral triangle, when we forbid flipping pieces.*

Cut Graphs, Vertex Classification and Matching Diagrams

Definition 2 *A **cut graph** G^X is a finite geometric graph representing a dissection of a target shape $X \in \{P, P'\}$. Its faces correspond one-to-one with the pieces, vertices $V(G^X)$ correspond to the corners of X and the points along the cut lines having at least one surrounding angle not equal to π, and edges $E(G^X)$ consist of the cut and boundary lines connecting these vertices.*

Vertex classification:

- **Corner vertex:** a boundary vertex corresponding to a corner of the target shape X.
- **Side vertex:** a boundary vertex corresponding to a point along a side of the target shape X.
- **Paired vertex:** an internal vertex where exactly two piece corners meet.
- **Convex vertex:** an internal vertex where three or more piece corners meet, and all interior angles are convex.
- **Reflex vertex:** an internal vertex where three or more piece corners meet and only one of them is reflex.
- **Flat vertex:** an internal vertex where at least two piece corners and one piece side meet.

Definition 3 *An **edge matching diagram** $\mathcal{ED}$ for a pair of cut graphs G^P and $G^{P'}$ is defined as follows: For each $X \in \{P, P'\}$, define the node set $N^X(\mathcal{ED})$ as follows:*

1. *Initialize the node set $N^X(\mathcal{ED})$ as $E(G^X)$.*
2. *For an edge $x \in N^X(\mathcal{ED})$, if both endpoints of x in the cut graph are flat vertices and an angle of π appears on opposite sides of x, then remove x from $N^X(\mathcal{ED})$.*
3. *For each side of a piece x, if there is a flat vertex on x when forming X, add a new corresponding node to $N^X(\mathcal{ED})$.*

A set of links $L(\mathcal{ED})$ between $N^P(\mathcal{ED})$ and $N^{P'}(\mathcal{ED})$ is formed by connecting a link for each side $e \in E(P_1) \cup E(P_2) \cup \cdots \cup E(P_k)$ according to the correspondence between links representing the sides of pieces and nodes representing the edges of the graph.

Definition 4 *A **vertex matching diagram** $\mathcal{VD}$ for G^P and $G^{P'}$ is defined as follows: For each $X \in \{P, P'\}$, the node set $N^X(\mathcal{VD})$ is $V(G^X)$. The set of links $L(\mathcal{VD})$ between $N^P(\mathcal{VD})$ and $N^{P'}(\mathcal{VD})$ is formed by connecting a link for each corner $x \in V(P_1) \cup V(P_2) \cup \cdots \cup V(P_k)$ according to the correspondence between links representing the corners of pieces and nodes representing the vertices of the graph.*

Definition 5 *A **weight** $\omega(x)$ for each node x of $\mathcal{VD}$ is the sum of the interior angles of the corners that gather at the corresponding vertex v.*

Definition 6 ***Divide-in-two** vertices, denoted as $N^X_{dit}(\mathcal{VD}) \subset N^X(\mathcal{VD})$, are the union of (i) side vertices whose adjacent vertices are both corner vertices and (ii) flat vertices whose adjacent vertices are both paired vertices.*
***Buried** vertices, denoted as $N^X_{bur}(\mathcal{VD}) \subset N^X(\mathcal{VD})$, are the set of paired vertices whose adjacent vertices are both non-flat.*
*A path $P = (p_1, \ldots, p_k)$ of $\mathcal{VD}$ is **well-behaved** if $p_2, \ldots, p_{k-1} \in N^X_{dit}(\mathcal{VD}) \cup N^X_{bur}(\mathcal{VD})$.*

Lemma 7 *Let C be a connected component of $\mathcal{VD}$ that contains a boundary node v, and suppose the following conditions hold: Both of the next sides s and s' of v along the boundary correspond to monochromatic nodes in $\mathcal{ED}$. None of the next sides of any other node in C correspond to monochromatic nodes in $\mathcal{ED}$. Then C forms a cycle.*

Geometric Constraints for Square (S) and Triangle (T)

Lemma 8 *In any dissection of T and S, no piece contains two corners of T.*

Lemma 9 *Let P_1, P_2, P_3 be a three-piece dissection of T and S. Then no P_i contains two diagonal corners of S.*

Lemma 10 *Let P_1, P_2, P_3 be a three-piece dissection of T and S. Then we can assume without loss of generality that each P_i is a simple polygon.*

Definition 11 *For a graph G^X and an angle $\theta < \pi$, the **θ-diff** is defined as the number of times θ appears minus the number of times $2\pi - \theta$ appears as an interior angle. We define **cc-diff** as the sum of all θ-diffs for $\theta < \pi$, and **tri-diff** as the difference $\pi/3$-diff $- 2\pi/3$-diff.*

Definition 12 *Let e be a boundary edge of G^S and v, v' be the adjacent vertices of e. (v, e, v') is called a **U-shaped boundary** if both v and v' are corner vertices of degree 1 in $\mathcal{VD}$.*

Main proof

Idea: The proof reduces all possible three-piece dissections to a finite set of topological equivalence classes. By matching these through shared angular invariants, a small number of feasible pairs of square and triangle cut graphs are identified. Ultimately, every case is ruled out by showing that the side lengths required for the pieces lead to geometrically impossible equations, proving the four-piece solution is indeed optimal.

Bibliography

[1] Erik D. Demaine, Tonan Kamata, and Ryuhei Uehara. *Dudeney's Dissection Is Optimal*. 17th Innovations in Theoretical Computer Science Conference (ITCS 2026).

Petr Chmel
chmel@iuuk.mff.cuni.cz

Presented paper by Edward Pyne, Roei Tell

Using Hardness vs Randomness to Design Low-Space Algorithms

(https://bulletin.eatcs.org/index.php/beatcs/article/view/880/942)

Introducing the hammer

Definition 1 (Randomized computation) *We say $L \in$ BPL if there is a randomized Turing machine M (with one-way access to the random tape) that runs in logspace, always halts in polynomial time, $x \in L$ if $\Pr[M(x) = 1] \geq 2/3$ and $x \notin L$ if $\Pr[M(x) = 1] \leq 1/3$.*

We say $L \in \mathrm{Avg}_{\varepsilon}\mathrm{L}$ if there is a deterministic Turing machine M that runs in logspace, always halts in polynomial time, and $M(x) \in \{0, 1, \perp\}$. Moreover, for every $n \in N$, we have that $\Pr_{x \leftarrow \mathbf{U}_n}[M(x) = \mathbf{1}[x \in L]] \geq 1 - \varepsilon$, and if $M(x) \neq \mathbf{1}[x \in L]$, then $M(x) = \perp$.

Definition 2 (Fooling and distinguishing) *A function (a PRG) $G : \{0,1\}^{\ell} \to \{0,1\}^{m}$ ε-fools a function (a distinguisher) $D : \{0,1\}^{m} \to \{0,1\}$, if $|\Pr_{x \leftarrow \mathbf{U}_m}[D(x) = 1] - \Pr_{s \leftarrow \mathbf{U}_\ell}[D(G(s)) = 1]| \leq \varepsilon$, where $\mathbf{U}_m, \mathbf{U}_\ell$ are uniform distributions on m and ℓ bits respectively.*

In the reverse direction, we say that if $|\Pr_{x \leftarrow \mathbf{U}_m}[D(x) = 1] - \Pr_{s \leftarrow \mathbf{U}_\ell}[D(G(s)) = 1]| \leq \varepsilon$, then D is an ε-distinguisher for G.

Remark 3 (On PRGs) *In essence, we want a function $G : \{0,1\}^{\ell(n)} \to \{0,1\}^{m(n)}$ with $l(n) \ll m(n)$ such that each probabilistic algorithm A running in time $m(n)$, for each n large enough and for each input $x \in \{0,1\}^n$, $|\Pr_{r \leftarrow \mathbf{U}_{m(n)}}[A(x, r) = 1] - \Pr_{s \leftarrow \mathbf{U}_{\ell(n)}}[A(x, G(s)) = 1]| \leq \frac{1}{n}$.*

We can then try all seeds of length $\ell(n)$, and estimate the probability of accepting.

Hammer 4 (Deterministic reconstruction) *Let $f \in \{0,1\}^{T}$ be a string that cannot be computed by circuits of size $s \ll T$. (That is, there is no circuit of size s that, given $j \in [T]$, computes the j-th bit of f.)*

We give f to a deterministic algorithm GEN such that, when given a circuit D, $\mathsf{GEN}^f(D)$ outputs an estimate for $\mathbb{E}[D]$. Either the estimate is good (say, $\pm 1/10$), or the proof of correctness shows that there exists a circuit C with size smaller than s that uses queries to D such that the truth table of C^D is f. (And thus, if f is indeed hard, then the estimate must be good.)

When considering small space algorithms (that is, D represents a log-space algorithm), we can actually have a second probabilistic algorithm REC such that, if the estimate of $\mathsf{GEN}^f(D)$ is bad, then $\mathsf{REC}^f(D)$ efficiently *prints a small circuit C such that C^D computes f.*

Using our hammer

Theorem 5 (Certified derandomization, [3]) *Let $\mathcal{L}_{\mathrm{hard}} \in \mathrm{SPACE}[\mathcal{O}(n)]$. Then for every $\varepsilon > 0$ and every $L \in$ BPL, there is a deterministic logspace algorithm A that on input $x \in \{0,1\}^n$, either*

- *outputs $L(x)$,*
- *or outputs "I cannot produce an output, because the hardness assumption is false", followed by a circuit C of size $2^{\varepsilon n'}$ for $\mathcal{L}_{\mathrm{hard}}$ on inputs of size $n' \in \Theta(\log n)$.*

Theorem 6 (Win-win algorithms, [2]) *There are algorithms A_1, A_2 such that for every directed graph G on n vertices, one of the following holds:*

- *$A_1(G)$ solves s-t connectivity in G in polynomial time and polylogarithmic space,*
- *$A_2(G)$ estimates length-n random walk probabilities in G (up to $\pm 1/\mathrm{poly}(n)$) in nondeterministic logspace.*

Moreover, both algorithms report if they fail to compute the desired answer and do not exceed their resource bound in any case.

Theorem 7 (Hardness of composition, [4]) *Suppose that there is a $\delta > 0$ such that for every polynomial $T(n)$ and constant $\varepsilon > 0$, the following holds. There is a function f computable in time T and space $\mathcal{O}(\log n)$ such that any algorithm computing $f(f(x))$ successfully of ε-fraction of inputs $x \in \{0,1\}^n$. Then,* $\mathrm{BPL} \subseteq \bigcap_{\varepsilon \geq 0} \mathrm{Avg}_\varepsilon \mathrm{L}$.

Some technical details

Definition 8 (Predictor) *Given a distribution $\mathbf{w}$ over $\{0,1\}^M$, we say $P : \{0,1\}^m \to \{0,1\}$ is a* δ-predictor *for $\mathbf{w}$ if there is an interval $I \subseteq [M]$ where $|I| = m$ and a bit $j \in [M] \setminus I$ such that $\Pr_{w \leftarrow \mathbf{w}}[P(w_I) = w_j] \geq \frac{1}{2} + \delta$.*

Definition 9 (Distinguish-to-predict (D2P)) *An ε-distinguish to δ-predict transform for a class of circuits $\mathcal{C}$ is an algorithm that takes as input an M-bit $C \in \mathcal{C}$ and outputs a set of procedures $P_1, \ldots, P_L$, where each P_j is a function $\{0,1\}^i \to \{0,1\}$ for some $i \leq M$ such that for every distribution $\mathbf{w}$ for which C is an ε-distinguisher, there is $j \in [L]$ such that P_j is a δ-predictor for $\mathbf{w}$.*

Definition 10 (Read-Once Branching Program (ROBP)) *An ROBP $B : \{0,1\}^M \to \{0,1\}$ consists of $M+1$ layers, where each layer i is labeled with a distinct input variable x_{j_i}. Each node in layer i has two outgoing edges to layer $i+1$ labeled with 0 and 1. The first layer has a start node s and each node in the last layer is labeled with an output value.*

The ROBP computes a function by starting in s and iterating through the layers while reading the values of each input variable and following the edges labeled with the value to the next layer. The maximal number of nodes in any layer is the width *of an ROBP.*

Theorem 11 (D2P for ROBPs, [1, 3]) *There exists a deterministic logspace $(1/M)$-distinguish to $(1/\mathrm{poly}(M))$-predict transform for ROBPs of width W. The algorithm outputs $\mathrm{poly}(M, W)$ predicotrs, each of which is also an ROBP.*

Bibliography

[1] Dean Doron, Edward Pyne, Roei Tell. *Opening up the distinguisher: A hardness to randomness approach for BPL= L that uses properties of BPL.*
In Proceedings of the 56th Annual ACM Symposium on Theory of Computing, 2024.

[2] Dean Doron, Edward Pyne, Roei Tell, Ryan R. Williams. *When Connectivity Is Hard, Random Walks Are Easy with Non-determinism.*
In Proceedings of the 57th Annual ACM Symposium on Theory of Computing, 2025.

[3] Edward Pyne, Ran Raz, Wei Zhan. *Certified hardness vs. randomness for log-space.*
In 2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS), 2023.

[4] Edward Pyne, Roei Tell. *Composing Low-Space Algorithms.*
Electronic Colloquium on Computational Complexity: TR25-140, 2025.

Rachel Dufau-Sansot
rachel.dufau-sansot9@etu.univ-lorraine.fr

Presented paper by Dániel Gerbner

A note on strongly and totally chain intersecting families

(https://link.springer.com/article/10.1007/s00373-025-02989-4)

Introduction

In this talk, I will present results on the cardinality of some families of subsets of an n-element underlying set. In 2007, Bernáth and Gerbner defined *strongly (p,q)-chain intersecting families* as families that do not contain p-chain $A_1 \subsetneq \cdots \subsetneq A_q$ and q-chain $B_1 \subsetneq \cdots \subsetneq B_q$ with $A_p \cap B_1 = \emptyset$. They were able to get an upper bound on the cardinality of such families for some values of p and q. In the paper I will present, they get an upper bound on all strongly (p,q)-chain intersecting families if n is large enough.

Definitions

We will use the following definitions:

- chain of length k : family of k subsets in pairwise containment relation; $F_1 \subsetneq \cdots \subsetneq F_k$.
- level: family of all the subsets of a given size.
- k-Sperner family: family with no chain of length more than k.
- middle $k-1$ levels: all the level of size more than $\lfloor \frac{n-k+1}{2} \rfloor$ and less than $\lfloor \frac{n+k-1}{2} \rfloor$.
- upper z-level: family of all subsets of size at least $n-z+1$.
- upper $z+\frac{1}{2}$ level: the family made of the upper z-level plus the subsets of size $n-z$ containing a fixed element.

We say that a family $\mathcal{F}$ is *intersecting* is for all $F_1, F_2 \in \mathcal{F}$, $F_1 \cap F_2 \neq \emptyset$.

We say that a family $\mathcal{F}$ is *strongly (p,q)-chain intersecting* if it does not contain p-chain $A_1 \subsetneq \cdots \subsetneq A_q$ and q-chain $B_1 \subsetneq \cdots \subsetneq B_q$ with $A_p \cap B_1 = \emptyset$.

We say that a family $\mathcal{F}$ is *r-complementing chain pair free* if it does not contains subsets $F_1 \subsetneq \cdots \subsetneq F_r$ with their complements.

Results

In 2007, Bernáth and Gerbner stated the following conjecture:

Conjecture 1 *If $\mathcal{F}$ if a strongly (p,q) chain intersecting family, then $|\mathcal{F}| \leq \{|R_1|, |R_2|\}$ with R_1 is the upper $\frac{n+p}{2}$ levels and R_2 is the middle $q-1$ levels.*

They were able to prove that the conjecture holds if $n+p$ is even or if $p \geq q$ or $p = 1$.

In the paper I will present, they have been able to prove the conjecture for n large enough.

Theorem 2 *Let p, q be positive integers and n integer large enough. Then conjecture 1 holds.*

Idea of the proof

The main tool of the proof is the *permutation method*.
Let $\mathcal{F}$ be the family whose cardinality we want to determine.
Let α be a permutation of $[n] = \{1, \ldots, n\}$, and $G \subset [n]$, we have that $\alpha(G) = \{\alpha(x) : x \in G\}$.
For a family $\mathcal{G}$, we have that $\alpha(\mathcal{G}) = \{\alpha(G) : G \in \mathcal{G}\}$.
Let g_i be the number of subsets of size i in $\mathcal{G}$.
We consider this double sum :

$$\sum_{\alpha} \sum_{F \in \mathcal{F} \cap \alpha(\mathcal{G})} \frac{1}{g_{|F|}} \binom{n}{|F|} = \sum_{F \in \mathcal{F}} \sum_{\alpha : F \in \alpha(\mathcal{G})} \frac{1}{g_{|F|}} \binom{n}{|F|}$$

.

We can compute that the right hand side of this equality is equal to $n!|\mathcal{F}|$. If we can get an upper bound on $\sum_{F \in \mathcal{F} \cap \alpha(\mathcal{G})} \frac{1}{g_{|F|}} \binom{n}{|F|}$, we have un upper bound on $|\mathcal{F}|$.

This method work with any family $\mathcal{G}$, but only some of them give sharp bound. We will consider two families:

- The *chain-pair* is a family made of a full chain $\mathcal{A}$ (a chain of length $n + 1$) and the chain consisting of the complements of the members of $\mathcal{A}$.
- The *circle*: We consider $[n]$ cyclically (1 comes after n). An *interval* is a set of consecutive elements. $[n]$ and $\emptyset$ are intervals. The circle is the family consisting of all intervals.

Barbora Dohnalová
bdohnalova@kam.mff.cuni.cz
Presented paper by Ziad Ismaili Alaoui, Detlef Plump, Sebastian Wild

Eulerian Cycles in O(m) Time and O(n) Space.

(https://doi.org/10.48550/arXiv.2508.05251)

Introduction

An *Eulerian cycle* in a graph is a closed walk that traverses every edge exactly once. You might recall that a directed graph is Eulerian if for each vertex, its in-degree is equal to its out-degree, and it is (strongly or weakly) connected. There are many well known polynomial algorithms for finding an Eulerian cycle in an Eulerian graph, for example the Hierholzer's algorithm.

Hierholzer's algorithm proceeds by finding a cycle in the given graph, and if there is any edge not covered by this cycle, it iterates and glues the new cycle to the old one.

The Algorithm

Standard implementations of Hierholzer's algorithm run in linear time and use $\Theta(m \log m)$ bits of working memory. The authors of this paper propose an algorithm visualised in Figure 3, which also runs in linear time but only needs $O(n \log m)$ bits of memory. The idea is to do edge-centric DFS, without remembering all the steps, since it does not matter in which order we add the subcycles.

In the algorithm, they distinguish between several types of edges. We call the edges that have not already been traversed *Black*. When we traverse an edge, we either color it *Red*, if it goes to a previously unvisited vertex, or *Green*, if its end vertex has been visited before. When we backtrack through an edge, we make it *Dashed*.

Notation

We will use the following notation:

- $d^-(v)$: the in-degree of v,
- $d^+(v)$: the out-degree of v,
- $\Gamma^-(v, i)$: the i-th in-neighbour of v,
- $\Gamma^+(v, i)$: the i-th out-neighbour of v,
- $d^-(v, \textsc{Black})$: the number of incoming Black edges of v,
- $d^+(v, \textsc{Black})$: the number of outgoing Black edges of v.

Correctness

For proving correctness of the algorithm, we will need to introduce the notion of *needle vertices*.

Definition 1 (Needle vertex) *Let u denote the current vertex of the traversal. A vertex w is a* needle vertex *if and only if:*

- $w = u$ *and* $d^+(w, \textsc{Black}) = d^-(w, \textsc{Black})$*; or*
- $w \neq u$ *and* $d^+(w, \textsc{Black}) = d^-(w, \textsc{Black}) + 1$.

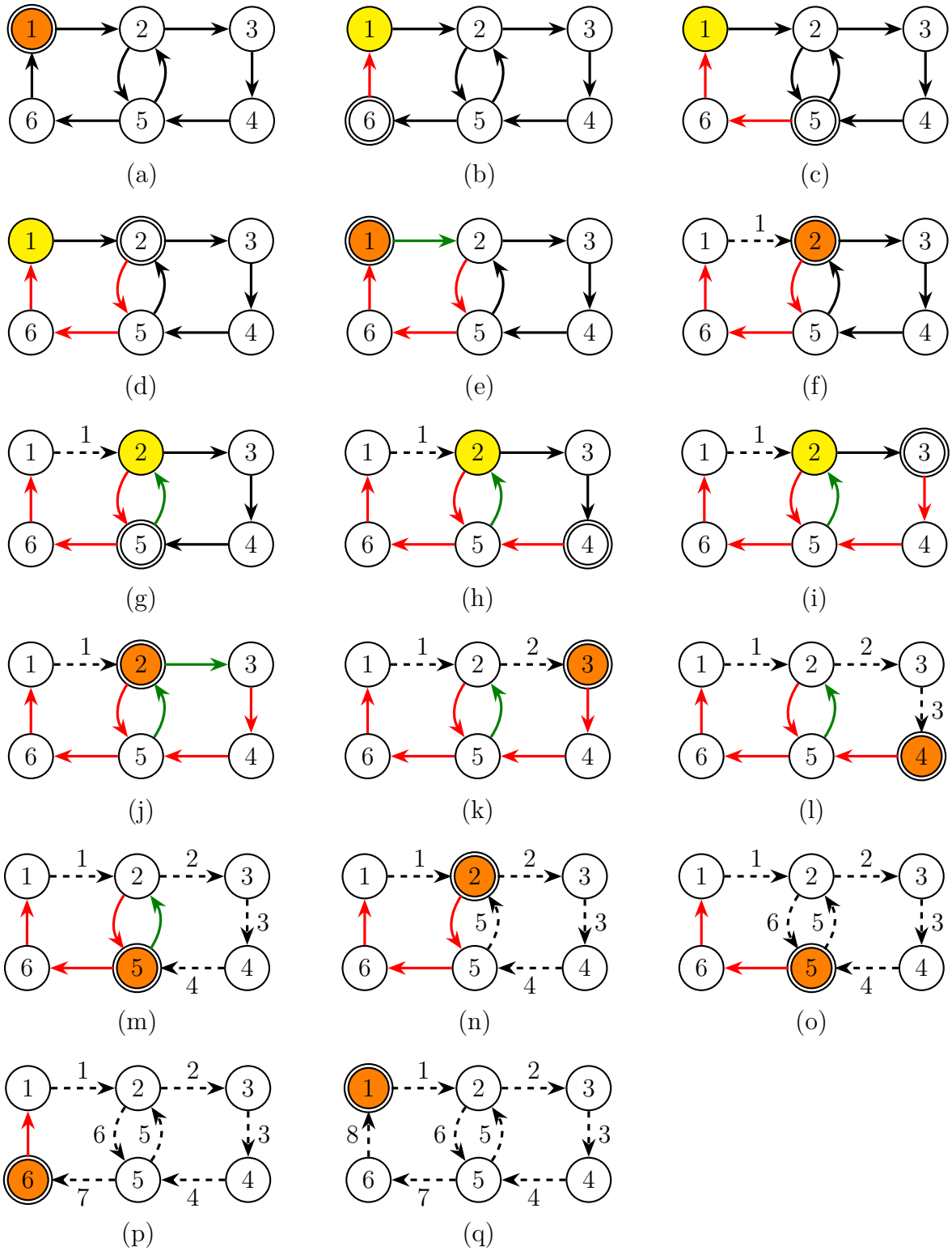


Figure 3: Sample execution of Space-Efficient-Hierholzer. For clarity, Dashed edges are labelled in the order in which they are written. The current vertex is represented by double borders. Needle vertices of the first type are colored orange, and needle vertices of the second type are colored yellow.

Alica Dományová

alica@domany.sk

Presented paper by Jineon Baek, Seewoo Lee

An Equilateral Triangle of Side $< n$ Cannot be Covered by $n^2 + 1$ Unit Equilateral Triangles Homothetic to it

(https://www.tandfonline.com/doi/full/10.1080/00029890.2024.2416882)

Introduction

John Conway and Alexander Soifer showed that $n^2 + 2$ unit equilateral triangles can cover an equilateral triangle T of side $> n$. Their paper was famously short as an attempt to set the world record for the shortest math paper ever. They also conjectured that it is not possible to cover T with $n^2 + 1$ unit triangles. This remains open. Jineon Baek and Seewoo Lee proved that if the $n^2 + 1$ unit triangles have sides parallel to the sides of T, then they cannot cover T.

We will show that the title of this talk is not lying. Actually, we will prove an even more general statement from the paper (Theorem 5).

Theorems and Definitions

Theorem 1 (Conway and Soifer) *$n^2 + 2$ unit equilateral triangles can cover an equilateral triangle T of side $n + \varepsilon$ for a sufficiently small $\varepsilon > 0$.*

Conjecture 2 (Conway and Soifer) *$n^2 + 1$ unit equilateral triangles cannot cover an equilateral triangle T of side $n + \varepsilon$ for any $\varepsilon > 0$.*

Theorem 3 *If T is an equilateral triangle of side $> n$, then $n^2 + 1$ unit equilateral triangles homothetic to T cannot cover T.*

Definition 4 *An H-triangle (a shorthand notation for a horizontal triangle) is a triangle with one side parallel to the x-axis. For any H-triangle T, its base is the length of the side l parallel to the x-axis, and its height is the distance between l and the vertex of T which is not on l.*

Theorem 5 *Let X be any union of n H-triangles of base b and height h with disjoint interiors. Then X cannot be covered by $n + 1$ H-triangles of base less than b and height less than h.*

Corollary 6 *The minimum number of unit equilateral H-triangles required to cover an equilateral H-triangle of side $n + \varepsilon$ with a sufficiently small $\varepsilon > 0$ is $n^2 + 2$. Also, the minimum number of unit equilateral H-triangles required to cover a trigon made of n equilateral H-triangles of side $1 + \varepsilon$ with a sufficiently small $\varepsilon > 0$ is $n + 2$.*

Theorem 7 *The largest value of $\varepsilon > 0$ such that the equilateral H-triangle T of side $n + \varepsilon$ can be covered by $n^2 + 2$ equilateral H-triangles is $\varepsilon = \frac{1}{n+1}$.*

Theorem 8 *The largest value of $\varepsilon > 0$ such that the equilateral H-triangle T of side $n + \varepsilon$ can be covered by $n^2 + 3$ equilateral H-triangles is $\varepsilon = \frac{1}{n}$.*

Proof of Theorem 5

Definition 9 *For every H-triangle T, define its y-coordinate y_T as the y-coordinate of the horizontal side of T.*

Definition 10 *For every H-triangle T and $t \in \mathbb{R}$, define $\tilde{f}_T(t)$ as the length of the segment of the line $y = t$ covered by T unless $t = y_T$ and the line contains the base. For $t = y_T$, choose the value of*

$\tilde{f}_T(y_T)$ so that $\tilde{f}_T$ is right-continuous: the base of T if T is pointed upwards, and 0 if T is pointed downwards. Define $f_T : [0,1) \to \mathbb{R}$ as the function

$$f_T(t) = \sum_{n \in \mathbb{Z}} \tilde{f}_T(t+n).$$

Corollary 11 *If an H-triangle T of base 1 and height 1 is pointed downwards, then $f_T(t) = \{t - y_T\}$, and if T is pointed upwards, then $f_T(t) = 1 - \{t - y_T\}$.*

Definition 12 *Define $\mathcal{T}$ as the abelian group generated by all functions $t \mapsto \{t-a\}$ and $t \mapsto 1 - \{t-a\}$ with $a \in [0,1)$.*

Lemma 13 *Any function $f : [0,1) \to \mathbb{R}$ in $\mathcal{T}$ has the following properties.*

1. *f is right-continuous.*
2. *f is differentiable everywhere except for a finite number of points, and the derivative is always equal to a fixed constant $a \in \mathbb{Z}$.*
3. *For all $s, t \in [0,1)$, the value $f(t) - f(s)$ is equal to $a(t-s)$ modulo 1.*
4. *The integral $\int f$ is equal to $\frac{b}{2}$ for some $b \in \mathbb{Z}$ where $b - a$ is divisible by 2.*

Lemma 14 *Let $f : [0,1) \to \mathbb{R}$ be any function in $\mathcal{T}$ such that $\int f = \frac{1}{2}$ and $f(t) \geq 0$ for every $t \in [0,1)$. Then there is a positive odd integer a and some $c \in [0,1)$ such that f is either $f(t) = \{at + c\}$ or $f(t) = 1 - \{at + c\}$.*

Adam Džavoronok
adam.dzavoronok@gmail.com

Presented paper by J. Balogh, P. Bradshaw, R. Garcia, B. Lidický

Density of rainbow triangles and properly colored K_4's

(https://arxiv.org/abs/2511.21061)

Introduction

In this talk, we will try to sketch proofs of the following results:

Theorem 1 *Let $G = (V, E)$ be a simple graph, and colour the edges of G with red, green, and blue. Denote R, G, B as the number of red, green, and blue edges, respectively, and T as the number of rainbow triangles in G. Then, $T^2 \leq 2RGB$.*

Theorem 2 *Let G be a graph with R red edges, G green edges, and B blue edges, and suppose that G has K properly coloured K_4's. Then, $K \leq \frac{1}{4}(RGB)^{\frac{2}{3}}$. Furthermore, if $K =\leq \frac{1}{4}(RGB)^{\frac{2}{3}} > 0$, then G is obtained from a balanced blowup of a properly coloured K_4, possibly by adding a set of isolated vertices.*

Besides being fairly natural questions on their own, these problems are related to the so called *joint* problem, about which I would like to talk if time permits. Our main tool, which I would like to present, will be the so called *flag algebras*.

Informal Crash Course to Flag Algebras

Flag algebras, introduced by Razborov, are a framework for proving inequalities between subgraph densities in large discrete structures. They are especially useful in extremal combinatorics, where one wants to bound the density of a target configuration based on the densities of simpler configurations.

Definition 1 (Density). Let H and G be finite structures of the same kind, with $v(H)$ denoting the number of vertices of H. The *density* of H in G is

$$p(H, G) = \frac{\left|\{X \subseteq V(G) : |X| = v(H),\ G[X] \cong H\}\right|}{\binom{v(G)}{v(H)}}.$$

Thus $p(H, G)$ is the probability that a uniformly chosen $v(H)$-subset of vertices of G induces a copy of H.

Definition 2 (Convergent sequence). A sequence $(G_n)_{n\geq 1}$ with $v(G_n) \to \infty$ is *convergent* if, for every fixed finite H, the limit

$$\lim_{n\to\infty} p(H, G_n)$$

exists.

The basic philosophy is that extremal questions should first be studied on convergent sequences of large graphs. In that setting, the relevant information is encoded by the limiting densities of all finite configurations.

Definition 3 (Limit object). A convergent sequence determines a function

$$\varphi(H) = \lim_{n\to\infty} p(H, G_n)$$

on finite structures H. In the language of flag algebras, φ extends to a positive homomorphism on an algebra $\mathcal{A}$. In practice, one may think of $\varphi(H)$ simply as the asymptotic density of H.

The algebra $\mathcal{A}$. Start with the vector space $\mathbb{R}\mathcal{F}$ of formal linear combinations of finite 3-edge-colored graphs up to isomorphism. We quotient by the relations

$$F = \sum_{F' \in \mathcal{F}_{v(F)+1}} p(F, F')\, F',$$

which encode the consistency of induced densities under one-vertex extension. The product is defined by

$$F_1 \cdot F_2 = \sum_H p(F_1, F_2; H)\, H,$$

where $p(F_1, F_2; H)$ is the probability that a random partition of $V(H)$ into parts of sizes $v(F_1)$ and $v(F_2)$ induces F_1 and F_2. After extending bilinearly, this turns the quotient into a commutative algebra $\mathcal{A}$. Its homomorphisms $\varphi : \mathcal{A} \to \mathbb{R}$ are precisely limit objects coming from convergent sequences of large graphs.

Many counting arguments in extremal graph theory are rooted at a chosen vertex, edge, or small substructure. Flag algebras formalise this by allowing some vertices to be labelled.

Definition 4 (Type). A *type* σ is a finite structure in which all vertices are labelled by $\{1, 2, \ldots, \ell\}$ for some $\ell \geq 0$.

Definition 5 (Flag). A *σ-flag* is a finite structure F together with an injective map

$$\theta : [\ell] \hookrightarrow V(F)$$

such that the labelled vertices induce a copy of the type σ. Isomorphisms of flags are required to preserve the labels.

The point is that flags record local statistics conditioned on a fixed rooted configuration. For example, if σ is a labelled edge, then a σ-flag may represent a triangle extending that edge in a specified way. For unlabelled structures, this reduces to the familiar idea that the product of two densities can be expanded as a linear combination of the densities of larger configurations.

Definition 7 (Averaging operator). There is a linear map

$$[\![\cdot]\!] : \mathcal{A}^\sigma \to \mathcal{A}$$

called the *averaging* (or *unlabeling*) operator. It forgets the labels and averages over all choices of the root. Thus, rooted identities become unrooted density inequalities after applying $[\![\cdot]\!]$.

Proposition 8 (Flag algebra Cauchy–Schwarz). For every type σ and every $f, g \in \mathcal{A}^\sigma$,

$$[\![f^2]\!]\,[\![g^2]\!] \geq [\![fg]\!]^2.$$

Bibliography

[1] A. A. Razborov, *Flag algebras*, Journal of Symbolic Logic **72** (2007), 1239–1282.

Aleksa Džuklevski

dzuklevski.aleksa@gmail.com

Presented paper by Aaron Abrams & Jamie Pommersheim

Integer Area Dissections of Lattice Polygons via a Non-Abelian Sperner's Lemma

(https://doi.org/10.1080/00029890.2025.2530372)

Introduction

The main question we will answer in this talk is the following: Which convex polygons can be dissected into lattice triangles of area 1? We present a full characterization of such polygons using a "non-abelian" version of the famous Sperner's lemma.

The main theorem we will prove is the following one:

Theorem 1 *For any convex lattice polygon P (inscribed in the integer lattice), the following are equivalent.*

(1) P can be dissected into lattice triangles of area 1;

(2) P can be dissected into lattice triangles of integer area;

(3) P can be diagonally dissected into lattice triangles of integer area;

(4) The boundary word of P is contractible.

The proof of the theorem will have several steps, and we introduce the terminology necessary for each step below.

Steps of the proof

Triangles are enough (1) $\Leftrightarrow$ (2)

Theorem 2 *Every lattice triangle P with (positive) integer area can be dissected into lattice triangles of area 1.*

Coloring by parity

The *parity* of the lattice point is one of the four pairs:

$$A = (\text{even, even}),\ B = (\text{odd, even}),\ C = (\text{odd, odd}),\ D = (\text{even, odd}),$$

according to the parity of its coordinates. We will think of A, B, C, D as colors.

Proposition 3 (Parity proposition) *The area of a lattice triangle P is an integer iff two of its vertices have the same color.*

Corollary 4 *Three collinear lattice points in the plane cannot all have different colors.*

• Recall that the area of $\triangle P$ is given by:

$$\text{Area}(P) = \frac{1}{2}\begin{vmatrix} 1 & 1 & 1 \\ x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \end{vmatrix}$$

• *diagonal dissection* - dissection of P without any new vertices added.

• By parity proposition, for dd we only need to know the colors of the vertices of P.

Abstractifying geometric polygons

• A *combinatorial n-gon* is an abstract cycle graph G with n vertices; any 3-element set of G is a triangle, and an *abstract triangulation* is an arbitrary collection of triangles.

• since we are interested in triangulations of planar polygons, we require that the triangles fit together to form a disk;

Definition 5 (Boundary word) *Let P be a lattice polygon, with vertices colored according to parity. The* boundary word *of P is the cyclic word* Word*(P) obtained by recording the colors of the vertices in counterclockwise order around P.*

Definition 6 (Contracting step, contractible) *Let w be a cyclic word $(X_1, X_2, .., X_n)$. As usual, indices are taken modulo $n \geq 2$.*

Contracting step*: if letters X_{i-1}, X_i, X_{i+1} are not all distinct, then delete X_i from w.*

The cyclic word w is contractible *if there is a sequence of contracting steps starting with w that results in a word (X) consisting of a single letter.*

• If $\triangle$ is a colored triangle, then we call $\triangle$ *good* if two vertices of $\triangle$ have the same color. *Good dissection* of G is a diagonal dissection of G into good triangles.

Proposition 7 ((3) $\Leftrightarrow$ (4)) *A colored polygon G has a good dissection iff* Word*(G) is contractible.*

Non-abelian Sperner

• A colored triangle is called tricolor if its vertices all have different colors.

Theorem 8 (Non-abelian Sperner) *Let T be a colored triangulation of a combinatorial polygon. If* Word*(P) is not contractible, then T contains a tricolor triangle.*

Together with Proposition 7 this gives:

Theorem 9 (Non-abelian Sperner) *Let G be a colored poolygon. Then there exists a colored triangulation of G with no tricolor triangles iff* Word*(P) is contractible. Moreover, if such a triangulation exists, then one exists with no interior vertices.*

Finally, we have a theorem that completes all the equivalences of the main theorem:

Theorem 10 ((2) $\Leftrightarrow$ (4)) *Let P be a lattice polygon with corners colored according to their parity. If* Word*(P) is not contractible, then P has no integral dissection.*

Linear time recognition

The following lemma tells us that no matter how we do the contractions, the result will be the same:

Lemma 11 (Diamond lemma, word version) *Let w be a cyclic word and let w' be obtained from w by a contracting step. Then w is contractible iff w' is contractible.*

Proposition 12 *There is an algorithm that takes as input a cyclic word w of length n and decides in time linear in n whether w is contractible.*

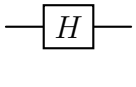
Roman Edenhofer
edenhofer@irif.fr

Introduction to Quantum Algorithms

Qubits, Unitaries and Measurements

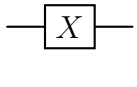
- (Single-qubit state) A *single-qubit state* is a vector
$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle = \begin{pmatrix}\alpha\\ \beta\end{pmatrix} \in \mathbb{C}^2 \quad \text{with } |\alpha|^2 + |\beta|^2 = 1.$$
- (Computational basis states) We write
$$|0\rangle = e_0 = \begin{pmatrix}1\\ 0\end{pmatrix}, \qquad |1\rangle = e_1 = \begin{pmatrix}0\\ 1\end{pmatrix}.$$
and more generally, for $i \in \{0,1\}^n$, $|i\rangle = e_i$ and $\langle i| = e_i^T$.
- (Multi-qubit states) An n-qubit state lives in the tensor product space $(\mathbb{C}^2)^{\otimes n} \cong \mathbb{C}^{2^n}$. It is a vector
$$|\psi\rangle = \sum_{x\in\{0,1\}^n} \alpha_x\,|x\rangle = \sum_{x\in\{0,1\}^n} \alpha_{x_1,\ldots,x_n}\,|x_1\rangle \otimes \cdots \otimes |x_n\rangle \quad \text{with} \sum_{x\in\{0,1\}^n} |\alpha_x|^2 = 1.$$
- (Measurement of an n-qubit state) Measuring  an n-qubit state yields outcome $x \in \{0,1\}^n$ with probability $|\alpha_x|^2$. After the measurement, the state collapses to $|x\rangle$.
- (Unitary) A matrix $U \in \mathbb{C}^{d\times d}$ is called *unitary* if $U^\dagger U = UU^\dagger = I$.

Hadamard H



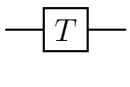
$$= \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}$$

Pauli-X



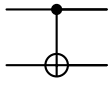
$$= \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}$$

T gate



$$= \begin{pmatrix}1 & 0\\ 0 & e^{i\pi/4}\end{pmatrix}$$

Controlled-X (CNOT)



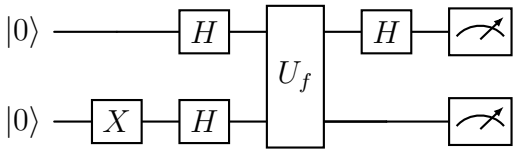
$$= \begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 1 & 0\end{pmatrix}$$

- (Universal gate set) The gate set $\mathcal{G} = \{H, X, T, CX\}$ is universal, meaning every n-qubit unitary can be approximated by a sequence of gates from $\mathcal{G}$.

Deutsch's Algorithm

We wish to determine whether a given boolean function $f : \{0,1\} \to \{0,1\}$ is constant.

- Classical Oracle: $O_f(x)$ returns $f(x)$.
- Quantum Oracle: O_f defined by $O_f\,|x\rangle\,|y\rangle = |x\rangle\,|y \oplus f(x)\rangle$.



Classically, we need two oracle queries, quantumly, we need only one via Deutsch's circuit:

Grover's Algorithm

We wish to solve the *Unstructured Search Problem*: Let $N = 2^n$, and suppose we are given a string

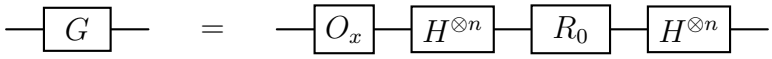
$$x = x_0 x_1 \cdots x_{N-1} \in \{0,1\}^N$$

with the promise that there exists exactly one index $i \in \{0, \ldots, N-1\}$ such that $x_i = 1$, while for all $j \neq i$, we have $x_j = 0$. The task is to output the unique index i such that $x_i = 1$.

- Classical Oracle: $O_x(j)$ returns x_j.
- Quantum Oracle: O_x defined by $O_x |j\rangle = (-1)^{x_j} |j\rangle$.

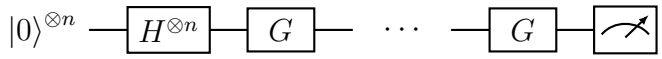
Classically, we need $\Theta(N)$ oracle queries to find i, quantumly, we need only $\Theta(\sqrt{N})$ queries via Grover's Algorithm.

The *Grover operator* is defined via:



where R_0 is the reflection around the state $|0\rangle^{\otimes n}$.

The full algorithm is given via:

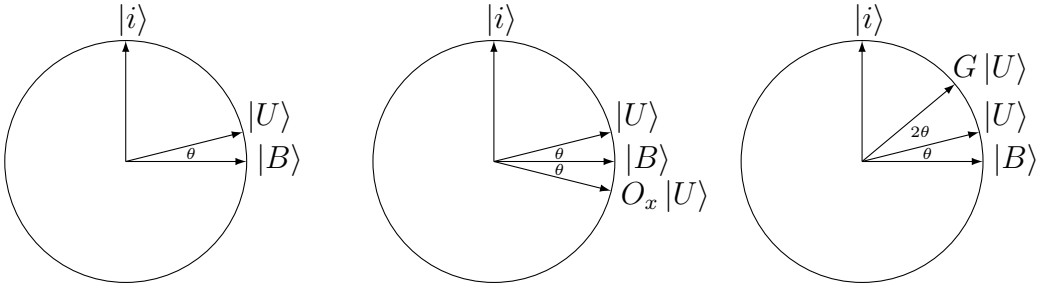


Claim 1 *With $k = O(\sqrt{N})$ repetitions of the Grover operator G, the algorithm above ends in state $|i\rangle$ with high probability.*

After the first layer of Hadamard gates the sate is

$$|U\rangle = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} |j\rangle = \sin(\theta) |i\rangle + \cos(\theta) |B\rangle$$

where $|B\rangle = \frac{1}{\sqrt{N-1}} \sum_{j \neq i} |j\rangle$ and $\theta = \arcsin(1/\sqrt{N})$. Each iteration of the Grover operator corresponds to a rotation with angle 2θ in the two-dimensional space spanned by $|i\rangle$ and $|B\rangle$.



After k iterations of the Grover operator, the state is

$$G^k |U\rangle = \sin((2k+1)\theta) |i\rangle + \cos((2k+1)\theta) |B\rangle .$$

Let $k^* = \frac{\pi}{4\theta} - \frac{1}{2}$. Choosing, $k = \lfloor k^* \rceil$, we find $k = O(\sqrt{N})$ since $\theta \geq \sin(\theta) = 1/\sqrt{N}$. The final measurement will return state $|i\rangle$ with probability:

$$\sin^2((2k+1)\theta) = \sin^2(\frac{\pi}{2} + 2(k^* - k)\theta) = \cos^2(2(k^* - k)\theta) \geq \cos^2(\theta) \geq 1 - \theta^2 \approx 1 - \frac{1}{N}.$$

Filip Filipkowski
filip.filipkowski.stud@pw.edu.pl
Presented paper by G.Patlin, J. Van Den Brandt
Sublinear-Time Algorithm for MST-Weight Revisited
(https://kam.mff.cuni.cz/~spring/media/papers/5/sublinear_mst.pdf)

Introduction

The main contribution of the paper is a new randomized sublinear-time algorithm for approximating weight of Minimal Spanning Tree of a weighted graph. The algorithm makes local randomized probes into the graph by using a carefully modified version of Prim's algorithm: start from a uniformly random vertex, stop the exploration early according to a random threshold, and return the heaviest edge added so far. The central insight is that this returned value is distributed like the weight of a uniformly random edge of the MST, up to a controlled truncation effect in the fast version. As a consequence, averaging many independent samples yields a $(1+\varepsilon)$-approximation of the MST weight in expected sublinear time. The techniques further imply a perfect sampler for sampling an edge of the MST uniformly at random in just $O^{\sim}(d)$ expected time.

Main Theorem and lemmas

Theorem 1 (Main theorem) *Let G be a weighted graph with average degree d, and let W denote the ratio between the maximum and minimum edge weight. For every $\varepsilon > 0$, there is a randomized algorithm that returns a $(1+\varepsilon)$-approximation of the MST weight of G using*

$$O\Big(Wd\varepsilon^{-2}\log(W/\varepsilon)\Big)$$

queries in expectation.

Lemma 2 (Exact sampling lemma) *Consider the randomized modified Prim procedure with a sufficiently large truncation parameter T. Then for every weight value w, the probability that the procedure returns w is exactly*

$$\frac{\#\{\textit{MST edges of weight } w\}}{n},$$

and the probability of returning 0 is $1/n$. In particular,

$$\mathbb{E}[\textsc{SampleMSTEdge}_T(G)] = \frac{\mathrm{MST}(G)}{n}.$$

Corollary 3 (Uniform MST-edge sampling) *The exact sampling lemma implies that the same randomized Prim procedure can be used to sample MST edge weights exactly. If the MST is unique, then repeated execution yields a uniformly random edge of the MST.*

Lemma 4 (Truncation bound) *If the sampler is run with a smaller cutoff $T = O(W/\varepsilon)$, then its expectation changes by only a small multiplicative factor. Hence the truncated sampler remains sufficiently accurate for approximation of MST weight.*

Lemma 5 (Variance bound) *The output of the truncated sampler has bounded variance. Therefore, averaging*

$$O(W/\varepsilon^2)$$

independent samples is enough to obtain a $(1+\varepsilon)$-approximation with constant success probability.

Lemma 6 (Cost of one sample) *A single execution of the randomized truncated Prim procedure inspects only*

$$O(d \log(W/\varepsilon))$$

edges in expectation.

Vojtěch Gaďurek
dlaza@kam.mff.cuni.cz

Presented paper by Feyza Duman Keles et al.

An Exact Algorithm for the Unanimous Vote Problem

(https://epubs.siam.org/doi/10.1137/1.9781611978964.12)

Problem Definition

The Unanimous Vote problem asks us to find the ordering that minimizes the expected number of flips. Consider n independent, biased coins, each with a known probability of heads $p_i \in [0, 1]$. We flip each coin once, in that assigned order, until we have observed both a head and a tail, or flipped all coins.

Cost Function

The cost of an ordering a is the expected number of flipped coins required to determine whether or not a vote is unanimous. Let $z_j^1(a) = \prod_{i=1}^{j-1} p_{a(i)}$ be the probability that the first $j-1$ coins are heads in ordering a, and let $z_j^0(a) = \prod_{i=1}^{j-1} \overline{p}_{a(i)}$ be the probability that the first $j-1$ coins are tails.

The exact expected cost of an ordering a is expressed as:

$$cost(a) = 1 + \sum_{j \geq 2} (z_j^0(a) + z_j^1(a))$$

State Biases and Blocks

Definition 1 *An ordering a is **0-biased** at position j if $z_j^0(a) > z_j^1(a)$. It is **1-biased** at position j if $z_j^0(a) < z_j^1(a)$.*

Positions in the ordering can be partitioned into blocks according to their biases. Each block is a maximal set of contiguous positions of the same bias.

Structural Theorems

Theorem 2 (Monotonicity Within a Block) *Consider any optimal ordering a with block sequence $[B_1, \ldots, B_q]$. Within each 0-biased block, the coins have decreasing probability of heads. Within each 1-biased block, they have increasing probability of heads.*

Theorem 3 (Monotonicity Across Blocks) *For all $1 \leq k < q$, if B_k is 0-biased, then it only contains coins with probability of heads $> 1/2$. If B_k is 1-biased, then it only contains coins with heads probability $< 1/2$.*

Corollary 4 *There exists an optimal ordering where the coins in blocks $[B_1, \ldots, B_{q-1}]$ can be partitioned into a set S_+ (with probabilities $> 1/2$) and a set S_- (with probabilities $< 1/2$). The ordering a restricted to the coins in S_+ are sorted by decreasing probability of heads. The ordering a restricted to the coins in S_- are sorted by increasing probability of heads.*

Theorem 5 (Almost Greedy Optimality) *There is an optimal ordering a where the following holds except for the position of the last coin of B_{q-1}:*

- *If a is 0-biased at position x, then for all positions $x' > x$, $p_{a(x)} \geq p_{a(x')}$.*
- *If a is 1-biased at position x, then for all positions $x' > x$, $p_{a(x)} \leq p_{a(x')}$.*

Algorithmic Complexity

Theorem 6 *There is an exact algorithm (Modified Greedy Algorithm) that solves the Unanimous Vote problem and runs in time* $O(n \log n)$.

Karolína Hylasová

khylas@fav.zcu.cz

Presented paper by Yong-De Feng, Yawen Chen, Baoyindureng Wu

The number of odd spanning trees in the complete graphs

(https://doi.org/10.1016/j.dam.2025.12.043)

Introduction

An odd graph is a graph G for which every $v \in V(G)$ satisfies $d_G(v) \equiv 1 \pmod 2$. An odd spanning tree T of G is a spanning tree such that $d_G(v) \equiv 1 \pmod 2$ for all $v \in V(T)$. It is known that every complete graph K_n of even order has an odd spanning tree. In this talk, we establish the exact number of labeled odd spanning trees in K_n. By employing the classical Prüfer sequence and constructing the generating function, we prove that the number of labeled odd spanning trees in K_n is given by $\frac{1}{2^n}\binom{n}{k}\sum_{k=0}^{n}(2k-n)^{n-2}$ (where n is even).

Igor Januszkiewicz
igorjanuszkiewicz997@gmail.com
Presented paper by Stephen Arndt, Benjamin Moseley, Kirk Pruhs, Marc Uetz
Competitive Online Transportation Simplified
(https://arxiv.org/abs/2508.08381)

Introduction

In the online transportation problem m parking garages are placed in a metric space. Each garage i has a positive integer capacity c_i. At each integer time t, $1 \leq t \leq k$ a car arrives at some location in the metric space. The online algorithm must irreversibly assign the car to a parking garage that is not full. The cost of the assignment is the distance from the car's location to the assigned parking garage. The goal is to minimize the total cost of the assignments.

In [1] it was conjectured that:

Conjecture 1 *There is a* $(2m-1)$*-competitive deterministic algorithm for the online transportation problem.*

The paper proves the following result:

Theorem 2 *There is a* $(8m-7)$*-competitive deterministic algorithm for the online transportation problem.*

Proof outline

First, we define a algorithm for a relaxation of the problem, called Iterant Car Problem, where cars can move to other garages after being assigned. Then we define a algorithm $\mathcal{A}$ for a special case of the problem with *power-of-two tree metric* T. After that we generalize the algorithm for any metric space to obtain algorithm $\mathcal{B}$. Finally, we convert $\mathcal{B}$ to a algorithm $\mathcal{C}$ for the original problem.

Definitions and notation

- Power-of-two tree metric - a metric space that can be represented as a tree. One garage is located at each vertex. Each edge is has a weight of the form 2^j for $j \in [0, n]$. The distance between two garages is the sum of the weights of the edges on the unique path between them.
- $T_j(i)$ - level j tree for garage $i \in [m]$, that is the portion of T reachable from garage i without traversing any edges with weight 2^j or greater.
- $W_j(i)$ - arbitrary depth-first-search walk/tour of $T_j(i)$ that starts at garage i and traverses each edge of $T_j(i)$ once in each direction.
- $\mathrm{Opt}(Z, Y)$ - optimal matching cost for instance Z in metric space Y.
- Bottleneck cost - if Y is a tree metric, we define the bottleneck cost for parking a car arriving at garage i in garage j as the weight of the maximum-weight edge on the path between i, j in the tree Y.
- $\mathrm{BottleOpt}(Z, Y)$ - optimal bottleneck matching cost for instance Z in metric space Y where Y is a tree metric.

Bibliography

[1] Bala Kalyanasundaram and Kirk Pruhs. On-line network optimization problems. In Amos Fiat and Gerhard J. Woeginger (eds.), *Online Algorithms: The State of the Art*, pages 268–280. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998. https://doi.org/10.1007/BFb0029573

Michal Katrlík

katrlikmichal@gmail.com

Presented paper by Denise Graafsma, Bodo Manthey, and Alexander Skopalik

Playing Snake on a Graph

(https://arxiv.org/abs/2506.21281)

Introduction

The original game of snake, as known from phones, is played on a grid. We consider a generalization of the game of Snake to arbitrary graphs as a 2 player game between the snake and the apple placer. The game is called the game of Snake. Deciding which player wins on a given graph is called the *snake problem*.

Definition 1 *A graph on which the snake player wins are called snake-winnable.*

Rules of the game of Snake

The game is played between two players, the *apple placer*, and the *snake*. It is played on graph G.

Definition 2 (Body of the snake) *The body of the snake of length l in time t is a simple path of length $l-1$ in G, denoted by $S^t = (s_1^t, s_2^t, ..., s_l^t)$. With s_1^t being called the head of the snake, s_l^t the tail of the snake and the vertices $(s_2^t, ..., s_{l-1}^t)$ the main body of the snake.*

1. ***Initialization*** - At the start of the game, the apple placer places the snake with body of length one into a vertex. Then the apple placer also places an apple into any other vertex.

2. ***Snake's turn*** - In every turn, the snake can move to any vertex v that its head is adjacent to that is not occupied by its main body. Meaning the change in the snake's body on turn $t+1$ when moving to vertex v not containing the apple is from $S^t = (s_1^t, s_2^t, ..., s_l^t)$ to $S^{t+1} = (v, s_1^t, s_2^t, ..., s_{l-1}^t)$. If v was the vertex containing the apple the change is from $S^t = (s_1^t, s_2^t, ..., s_l^t)$ to $S^{t+1} = (v, s_1^t, s_2^t, ..., s_l^t)$.

3. ***Apple placer's turn*** - When the snake eats an apple, the apple placer places a new apple into a vertex not occupied by the snake.

4. ***Game end***- The snake wins the game if its body contains all the vertices of G. The apple placer wins the game if the snake can not make any move or $S^k = S^t$ for some $k \neq t$.

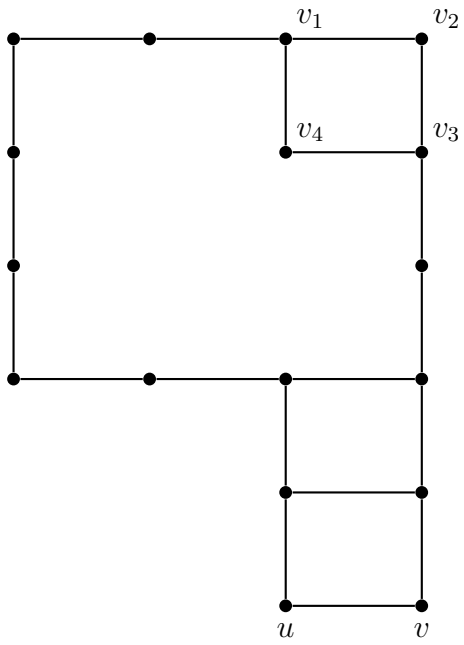
Results

Observation 3 *A graph containing a Hamiltonian cycle is snake-winnable.*

Observation 4 *A graph not containing an Hamiltonian path is not snake-winnable.*

Definition 5 (Theta graph) *A $\theta(p, q, r)$ is a graph made of two vertices joined by three internally vertex disjoint paths of lengths p, q and r.*

Theorem 6 *A bipartite graph $G = (V, E)$ with $|V|$ odd, is snake-winnable iff $\theta(|V|-3, 2, 2)$ is a subgraph of G.*



Gadget used in the proof of Theorem 7

Theorem 7 *The snake problem, even restricted to grid graphs is* NP*-hard.*

Definition 8 *The girth of the graph G, denoted by $g(G)$, is the length of the shortest cycle in G. In case of forests, the girth is defined as infinite.*

Lemma 9 *Let C be a cycle of graph G that contains the snake and l the current length of the snake. If the head of the snake leaves C and returns to it after going through m vertices in the rest of the graph, then $g(G) \leq |C| - l + 2m + 2$*

Theorem 10 *Let G be a graph not containing a Hamiltonian cycle. If $g(G) > 6$ G is not snake-winnable.*

Theorem 11 *Let G be a graph and v its cut vertex. Then G is snake winnable iff G_1 and G_2 are the only two connected components of $G - v$, $|G_1| = |G_2| \geq 2$ and $G_1 + v$ and $G_2 + v$ are both complete.*

Jakub Komárek
komaja@email.cz

Presented paper by Guy Blanc, William Pires, Toniann Pitassi

Differential privacy from axioms

(https://arxiv.org/abs/2511.21876)

Introduction

Differential privacy is an accepted notion of privacy, widely used both in theory and in practice. It has many nice properties, such as strong composition, which allows combining multiple algorithms satisfying differential privacy with only a small loss in privacy.

The definition of differential privacy is really strong: even if an adversary has control over the entire dataset except for one individual, the adversary cannot learn much about that individual from the output of the algorithm. However, in the common setting where all data points are drawn from the same unknown distribution, the attacker is not nearly as powerful. Thus, we might suspect that there might exist a weaker average-case definition of privacy that still works sufficiently well in this statistical setting and that yields better guarantees than the worst-case definition of differential privacy.

This paper shows that this is not the case: any privacy definition that satisfies a set of four reasonable axioms is in a sense equivalent to differential privacy, at the cost of at most polynomially increasing the number of needed samples.

Differential Privacy

Definition 1 ((ε, δ)-Differential Privacy) *A randomized algorithm $\mathcal{M} : X^n \to Y$ is (ε, δ)-DP if, for every $S, S' \in X^n$ differing in only one of the n coordinates and $Y' \subseteq Y$,*

$$\Pr[\mathcal{M}(S) \in Y'] \leq e^{\varepsilon} \cdot \Pr[\mathcal{M}(S') \in Y'] + \delta.$$

Theorem 2 (Advanced composition, [1]) *For all $\varepsilon, \delta, \delta' \geq 0$, the class of (ε, δ)-differentially private mechanisms satisfies $(\varepsilon', k\delta + \delta')$-differential privacy under k-fold adaptive composition for:*

$$\varepsilon' = \sqrt{2k \ln(1/\delta')}\, \varepsilon + k\varepsilon(e^{\varepsilon} - 1).$$

Definitions

Definition 3 (Statistical task) *A statistical task is defined by a set of distributions $\mathscr{D}$ over data domain X, an output space Y and a mapping $\mathcal{T}$ from distributions $\mathcal{D} \in \mathscr{D}$ to valid responses $\mathcal{T}(\mathcal{D}) \subseteq Y$. An algorithm $\mathcal{M} : X^n \to Y$ solves $\mathcal{T}$ with failure probability β if, for all $\mathcal{D} \in \mathscr{D}$,*

$$\Pr_{\boldsymbol{S} \sim \mathcal{D}^n}[\mathcal{M}(\boldsymbol{S}) \in \mathcal{T}(\mathcal{D})] \geq 1 - \beta.$$

Definition 4 (Equivalent algorithm) *We say an algorithm $\mathcal{M}' : X^m \to Y$ is (β, β')-equivalent to $\mathcal{M} : X^n \to Y$ if, any statistical task that $\mathcal{M}$ solves with failure probability β, $\mathcal{M}'$ solves with failure probability β'.*

Definition 5 (Privacy measure) *A* privacy measure *is a mapping $\mathcal{P}$ from (possibly randomized) algorithms $\mathcal{M} : X^n \to Y$ to their level of privacy, parametrized as a number on $\mathbb{R}_{\geq 0}$. We adopt the convention that a lower values for $\mathcal{P}(\mathcal{M})$ indicate that $\mathcal{M}$ is more private. We say that $\mathcal{M}$ is $\mathcal{P}$-private if $\mathcal{P}(\mathcal{M}) \leq 1$.*

Definition 6 (Blatantly non-private) *A mechanism $\mathcal{M} : X^n \to Y$ is* blatantly non-private *if there is a "high-entropy" distribution $\mathcal{D}$ (formally $\mathcal{D}(x) \leq 1/(100n^2)$ for all $x \in X$) and adversary A mapping mechanism outputs $y \in Y$ to datasets $S' \in X^n$ for which*[1]

$$\mathop{\mathbb{E}}_{\substack{\boldsymbol{S}\sim\mathcal{D}^n \\ \boldsymbol{S}'\leftarrow A(\mathcal{M}(\boldsymbol{S}))}} [\sum_{x\in\boldsymbol{S}} \mathbb{1}[x \in \boldsymbol{S}']] \geq 0.9n.$$

Axioms

Axiom 1 (Preprocessing maintains privacy) *We say a privacy measure $\mathcal{P}$ satisfies the* preprocessing axiom *if the following is true.*

1. ***Reordering the input maintains privacy:*** *For any algorithm $\mathcal{M} : X^n \to Y$ and permutation $\pi : [n] \to [n]$, defining*
$$\mathcal{M} \circ \pi(S) \coloneqq \mathcal{M}(S_{\pi(1)}, \ldots, S_{\pi(n)}),$$
we have that $\mathcal{P}(\mathcal{M} \circ \pi) \leq \mathcal{P}(\mathcal{M})$.

2. ***Remapping the domain maintains privacy:*** *For any mapping $\sigma : X \to X$ and algorithm $\mathcal{M} : X^n \to Y$, defining*
$$\mathcal{M} \circ \sigma(S) \coloneqq \mathcal{M}(\sigma(S_1), \ldots, \sigma(S_n)),$$
we have that $\mathcal{P}(\mathcal{M} \circ \sigma) \leq \mathcal{P}(\mathcal{M})$.

Axiom 2 (Prohibits blatant non-privacy) *We say a privacy measure $\mathcal{P}$ satisfies the* prohibits blatant non-privacy *axiom if any $\mathcal{P}$-private algorithm is not blatantly non-private.*

Axiom 3 (Strong composition) *For $c < 1$, we say a privacy measure $\mathcal{P}$ satisfies c-*strong composition *if for any algorithms $\mathcal{M}^1, \ldots, \mathcal{M}^\ell : X^n \to Y$ all satisfying $\mathcal{P}(\mathcal{M}^i) \leq \varepsilon$ and*

$$\varepsilon' \coloneqq \tilde{O}(\varepsilon \cdot \ell^c) = O(\varepsilon \cdot \ell^c \cdot \mathrm{polylog}(n)),$$

if $\varepsilon' \leq 1$, then the composed algorithm $\mathcal{M}' : X^n \to Y^\ell$ that takes in a sample $S \in X^n$ and outputs the ℓ responses $(\mathcal{M}^1(S), \ldots, \mathcal{M}^\ell(S))$ is $\mathcal{P}$-private.

Axiom 4 (Linear scalability) *We say a privacy measure $\mathcal{P}$ satisfies* linear scaling*, if for some polynomial $p : \mathbb{R}^2 \to \mathbb{R}$, any $\mathcal{P}$-private algorithm $\mathcal{M} : X^n \to Y$, any failure probability $\beta > 0$, and any large enough $k \geq p(n, 1/\beta)$, there exists a $(\beta, \beta' \coloneqq O(\beta))$-equivalent algorithm $\mathcal{M}'$ taking in $m \coloneqq kn$ samples that satisfies $\mathcal{P}(\mathcal{M}') \leq O(1/k)$.*

Results

Theorem 7 (Axioms imply differential privacy) *Let $\mathcal{P}$ be any privacy measure that satisfies Axioms 1 to 4 and $\mathcal{M} : X^n \to Y$ be any algorithm that is $\mathcal{P}$-private. For any $\varepsilon, \delta > 0$ and $m \coloneqq \mathrm{poly}(n, 1/\varepsilon, \log(1/\delta))$ there is an (ε,δ)-DP algorithm $\mathcal{M}' : X^m \to Y$ that is equivalent (as in Definition 4) to $\mathcal{M}$.*

Theorem 8 (DP satisfies axioms) *Approximate differential privacy (Definition 1) satisfies Axioms 1 to 4.*

Theorem 9 (Minimality of axioms) *Theorem 7 does not hold if $\mathcal{P}$ is allowed to not satisfy any one of Axioms 1 to 4.*

Bibliography

[1] This constant of 0.9 could be replaced with any $c < 1$.

[1] Cynthia Dwork and Aaron Roth. *The Algorithmic Foundations of Differential Privacy*. FNT in Theoretical Computer Science 9, 2013.

Martin Kopřiva
mkopriva@kma.zcu.cz

Presented paper by Zdeněk Dvořák and Luke Postle

Density of $\frac{5}{2}$-critical graphs

(https://arxiv.org/pdf/1411.6668)

Introduction

In the first part of this talk, we will introduce the discharging method and the concept of reducible configurations using simple illustrative examples. These examples are intended to build intuition and highlight the core ideas behind the technique. In the second part, we will apply the gained insights to the paper *Density of $\frac{5}{2}$-critical graphs* and explain how these methods are used in its main results.

A graph G is $\frac{5}{2}$-critical if G has no circular $\frac{5}{2}$-coloring (or equivalently, homomorphism to C_5), but every proper subgraph of G has one. We prove that every $\frac{5}{2}$-critical graph on $n \geq 4$ vertices has at least $\frac{5n-2}{4}$ edges, and list all $\frac{5}{2}$-critical graphs achieving this bound. This implies that every planar or projective-planar graph of girth at least 10 is $\frac{5}{2}$-colorable.

The main result of the paper is the following theorem, which characterizes 5/2-critical graphs. It shows that any such graph has bounded potential, providing a strong structural restriction crucial for the overall argument.

Theorem 1 *If G is a $\frac{5}{2}$-critical graph distinct from C_3, then $p(G) \leq 2$.*

In fact, the authors prove a stronger statement. They show that the bound in Theorem 1 is tight by giving a complete characterization of all $\frac{5}{2}$-critical graphs whose potential reaches the extremal value. In particular, they identify an explicit finite list (see Fig. 4) of exceptional graphs with a potential exactly equal to 2, and prove that any other $\frac{5}{2}$-critical graph avoiding these configurations must have a strictly smaller potential.

Theorem 2 *If G is a $\frac{5}{2}$-critical graph distinct from C_3, E_1 and E_2, then $p(G) \leq 1$.*

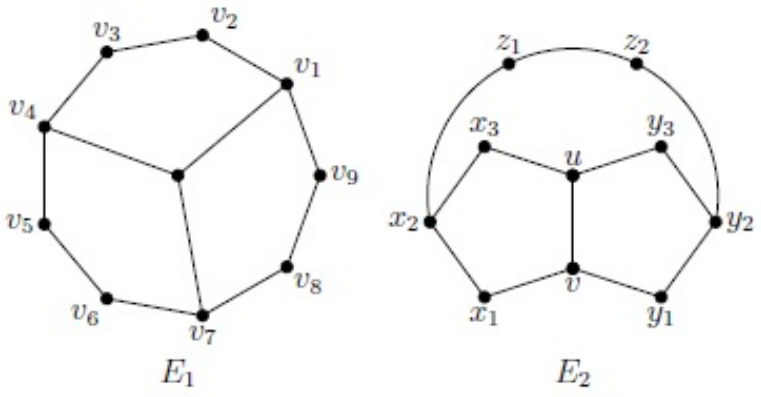


Figure 4: Exceptional graphs.

Bibliography

[1] Zdeněk Dvořák and Luke Postle: Density of $\frac{5}{2}$-critical graphs. *Combinatorica* 37, 863–886 (2017).

Júlia Križanová
krizanova@kam.mff.cuni.cz

Shannon Switching game and its generalization

Introduction

The Shannon switching game is a two-player game on a network in which one player, called *Short*, tries to preserve or create connectivity, while the other player, called *Cut*, tries to destroy it. In its classical form, the game is played on a graph with one distinguished edge $e^* = st$. A major contribution of Lehman was to reformulate the game in the language of matroids and to characterize exactly when Short has a winning strategy. This matroidal viewpoint also suggests a natural strategy for Short and serves as a starting point for further generalizations.

Definitions and Theorems

Definition 1 (Matroid) *A* matroid *is a pair $M = (E, \mathcal{C})$, where E is a finite non-empty set and $\mathcal{C} \subseteq 2^E$ is a family of subsets of E such that*

1. $\emptyset \notin \mathcal{C}$,
2. *if $C', C'' \in \mathcal{C}$ and $C' \subseteq C''$, then $C' = C''$,*
3. *if $C', C'' \in \mathcal{C}$, $e \in C' \cap C''$, and $C' \neq C''$, then there exists $C \in \mathcal{C}$ such that $C \subseteq C' \cup C'' - \{e\}$.*

The elements of $\mathcal{C}$ are called circuits*, and the elements of E are called* branches*.*

Definition 2 (Span) *Let $M = (E, \mathcal{C})$ be a matroid. For a set $A \subseteq E$, its* span *is*

$$\mathrm{sp}(A) = \{\, e \in E \mid e \in A \text{ or } \exists C \in \mathcal{C} \text{ such that } e \in C \subseteq A \cup \{e\} \,\}.$$

If $e \in \mathrm{sp}(A)$, we say that A spans the branch e.

Lemma 3 (Basic properties of span) *Let $M = (E, \mathcal{C})$ be a matroid, let $X, Y \subseteq E$, and let $e \in E$. Then:*

1. $X \subseteq \mathrm{sp}(X)$;
2. *$e \in \mathrm{sp}(X) \setminus X$ if and only if there exists a circuit $C \in \mathcal{C}$ such that $C - X = \{e\}$;*
3. *if $X \subseteq Y$, then $\mathrm{sp}(X) \subseteq \mathrm{sp}(Y)$;*
4. $\mathrm{sp}(X) = \mathrm{sp}(\mathrm{sp}(X))$;
5. *if $X \subseteq \mathrm{sp}(Y)$, then $\mathrm{sp}(X) \subseteq \mathrm{sp}(Y)$.*

Lemma 4 (Exchange Lemma) *Let $x \in E$ be a branch and let $X, Y \subseteq E$ be sets of branches such that $x \in X - Y$ and $\mathrm{sp}(Y) \subseteq \mathrm{sp}(X)$. Then either $\mathrm{sp}(Y) \subseteq \mathrm{sp}(X - \{x\})$, or there exists a branch $y \in Y - X$ such that $\mathrm{sp}(Y) \subseteq \mathrm{sp}\Big((X \cup \{y\}) - \{x\}\Big)$.*

Lemma 5 (Exchange outside the common span) *Let $X, Y \subseteq E$ be sets of branches, and let $x, d \in E$ such that $x \in X - Y, d \notin \mathrm{sp}(X \cap Y), d \in \mathrm{sp}(X) = \mathrm{sp}(Y)$. Then there exists a branch $y \in Y - \mathrm{sp}(X \cap Y)$ such that $\mathrm{sp}\Big((X \cup \{y\}) - \{x\}\Big) = \mathrm{sp}(Y)$.*

Definition 6 (Matroidal Shannon switching game) *Let $M = (E, \mathcal{C})$ be a matroid, and let $e^* \in E$ be a distinguished branch. The* Shannon switching game *on M is the two-player game in which Short and Cut alternately tag branches from $E \setminus \{e^*\}$. Branches tagged by Short are colored red, while branches tagged by Cut are colored blue.*

Short wins if the set of red branches spans e^*, that is, if there exists $S \subseteq E \setminus \{e^*\}$ such that $e^* \in \mathrm{sp}(S)$. Equivalently, Short wins if there exists a circuit $C \in \mathcal{C}$ such that $e^* \in C \subseteq S \cup \{e^*\}$. The objective of Cut is to prevent this.

Definition 7 (Types of Shannon switching games) *A Shannon switching game is said to be*

- short *if Short, when playing second, has a winning strategy against any strategy of Cut;*
- cut *if Cut, when playing second, has a winning strategy against any strategy of Short;*
- neutral *if the player moving first has a winning strategy.*

Theorem 8 (Lehman's main theorem) *Consider the Shannon switching game played on a matroid $M = (E, \mathcal{C})$ with distinguished branch e^*. The game is short if and only if there exist sets $A, B \subseteq E$ such that*

1. *$e^* \notin A \cup B$,*
2. *$A \cap B = \emptyset$,*
3. *$e^* \in \mathrm{sp}(A) = \mathrm{sp}(B)$.*

Further Generalization

Definition 9 ((k, ℓ)-game) *A (k, ℓ)-game is a variant of the Shannon game in which, in each round, Short chooses k edges and Cut chooses ℓ edges. The classical Shannon switching game is the special case $k = \ell = 1$.*

A natural question is whether a (k, k)-game is short exactly when there exist $k+1$ disjoint spanning sets.

Definition 10 (Shannon game with uniform timers) *Let $G = (V, E)$ be a finite undirected graph and let $e^* = st \in E$ be the distinguished edge. Fix an integer $P \geq 1$, called the* period timer. *Initially all edges are black.*

In each round:

1. *Cut colors one black or blue edge blue,*
2. *Short colors one black or red edge red,*
3. *all positive timers are decreased by one, and an edge whose timer reaches zero becomes black again.*

Whenever an edge is colored red or blue, its timer is reset to P.

This model generalizes the classical Shannon game by allowing previously colored edges to expire and become available again. The classical game is recovered when the timers are infinite.

Bibliography

[1] Alfred Lehman. *A Solution of the Shannon Switching Game*. Journal of The Society for Industrial and Applied Mathematics, 1964.

Terezie Lejbová
lejbovat@fav.zcu.cz

Presented paper by József Balogh, Andrew Treglown, Camila Zárate-Guerén

A note on color-bias perfect matchings in hypergraphs

(https://doi.org/10.48550/arXiv.2401.17073)

Introduction

This paper focuses on the existence of perfect matchings with a significant color imbalance in uniform hypergraphs. While classical results focus on minimum degree conditions guaranteeing a perfect matching, we consider a stronger requirement: the matching should contain substantially more edges of one color than expected in an arbitrary edge-coloring.

Given an r-edge-colored k-uniform hypergraph, a natural question is how large the minimum l-degree must be to ensure the existence of a color-biased perfect matching. It turns out that for all $2 \leq l < k$, the required degree threshold is asymptotically the same as that guaranteeing a perfect matching, regardless of the coloring. This stands in contrast to the graph case, where stronger conditions are necessary.

Definitions

Definition 1 *Hypergraph is a pair $H = (V, E)$, where V is a set of vertices and E is a set of subsets of V called hyperedges.*

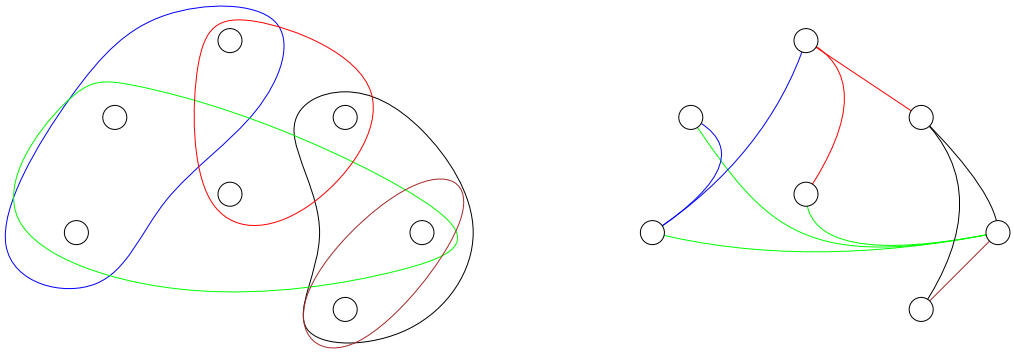


Figure 5: Two hypergraph representations.

Definition 2 *A perfect matching in a hypergraph H is a collection of vertex-disjoint edges of H which covers the vertex set $V(H)$.*

Definition 3 *Perfect matching is called color-bias if it contains significantly more than half of its edges in one color (due to arbitrary 2-edge-coloring).*

Definition 4 *Given a k-uniform hypergraph H and an l-element vertex set $S \subseteq V(H)$, we define $d_H(S)$ to be the number of edges containing S (where $l \in [k+1]$).*

Definition 5 *The minimum l-degree $\delta_l(H)$ of H is the minimum of $d_H(S)$ over all l-element sets of vertices in H.*

Definition 6 *Suppose that $l, k, n \in \mathbb{N}$ such that $l \leq k-1$ and k divides n. Let $m_l(k, n)$ denote the smallest integer m such that every k-uniform hypergraph H on n vertices with $\delta_l(H) \geq m$ contains a perfect matching.*

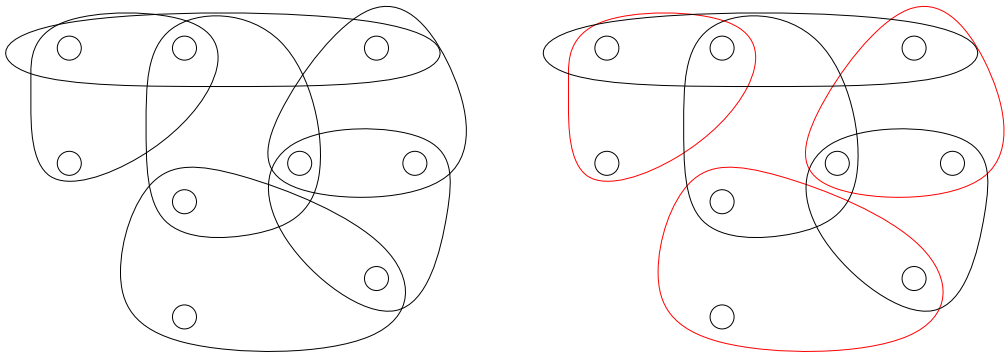


Figure 6: Perfect matching in 3-uniform hypergraph.

Definition 7 *Given integers $1 \leq l < k$, let $\mathcal{C}_{k,l}$ be the set of all $c > 0$ such that $m_l(k,n) \leq c\binom{n}{k-l}$ for a sufficiently large $n \in k\mathbb{N}$. Set $c_{k,l}$ to be the infimum of $\mathcal{C}_{k,l}$.*

Main result

Theorem 8 *Let $k, l, r \in \mathbb{N}$ where $2 \leq l \leq k$ and $r \geq 2$. Given any $\eta > 0$ where $c_{k,l} + \eta < 1$, there exist an $n_0 \in \mathbb{N}$ such that the following holds:*

Let H be a k-uniform hypergraph on $n \geq n_0$ vertices, where $n \in k\mathbb{N}$. If $\delta_l(H) \geq (c_{k,l} + \eta)\binom{n}{k-l}$, then given any r-coloring of $E(H)$, there is a perfect matching in H with at least $\frac{n}{rk} + \frac{\eta n}{8r(r-1)k^k(k^2+k)}$ edges of the same color.

Bibliography

[1] József Balogh, Andrew Treglown, Camila Zárate-Guerén: A note on color-bias perfect matchings in hypergraphs (2024), arXiv:2401.17073

Matúš Matok
matus.matok@fmph.uniba.sk

Lonely edges in cubic graphs

The notion of *perfect matching*, i.e., a 1-regular spanning subgraph—a set of disjoint edges covering all vertices of a given graph—is a well studied subject in graph theory. Already in 1891, Petersen proved that each edge of a bridgeless cubic graph can be covered by a perfect matching. We say that an edge e is *matching double covered* if there exist two distinct perfect matchings containing e. Otherwise, e is said to be *lonely*. If all edges of a graph are matching double covered, so is the graph. A graph G is *Klee* if is the complete graph on 4 vertices or it is obtained from another Klee graph by the operation of replacing a vertex with a triangle. All Klee graphs are therefore 3-connected and planar.

Lemma 1 *[1] Let G be a cubic graph. G is uniquelly 3-edge colorable if and only if G is Klee.*

Let T be a rooted oriented tree (with arcs oriented towards the leaves) with two types of nodes : *vertex nodes* and *edge nodes*. The root node is an edge node. Every edge node has two children which are vertex nodes representing its endvertices – the edge u_iu_j has children u_i and u_j. Every vertex node has at most one child and, if it has one, it is an edge node. A rooted oriented tree with all these properties will be called *special*.

Let T be a special rooted oriented tree. For a node s, let $T[s]$ denote the sub-tree of T rooted at s – the rooted tree induced by all the nodes x such that there is an oriented path from s to x in T. Let $E(T[s])$ and $V(T[s])$ denote the set of edges and vertices corresponding to edge nodes and vertex nodes of $T[s]$, respectively.

Definition 2 *Let G be a Klee graph, let $e = uv$ be an edge of G, and let M be a perfect matching containing e. A special rooted oriented tree T is a* decomposition tree *of G with respect to e if $V(T_e^G) = (V(G) \setminus \{u, v\}) \cup (M \setminus \{e\})$ and, in addition, for every edge node f (including the root node), the subgraph $G[V(T[f])]$ of G induced by the vertices of $T[f]$ is connected, moreover, the edge f is its unique bridge.*

Theorem 3 *Let G be a Klee graph, and let $e = uv$ be an edge of G. Then we can decide whether e is a lonely edge, and moreover,*

- *if e is lonely, then there exists a decomposition tree T_e^G with certificates at all internal vertex-nodes, and*
- *if e is not lonely, then there exists a perfect matching M containing e distinct from the canonical one.*

Let $v \in V(G)$ be a vertex. We say that v is *ℓ-preserving* if $\ell \in L$ is lonely in $G \uparrow v$ or $v \in \ell$. In this second case we will abuse the notation and rename e to ℓ so that contents of L are consistent.

Lemma 4 *Let G be a Klee graph other than K_4. Let $G' = G \downarrow xyz$ be a Klee graph obtained from G by reducing a triangle xyz into an e-preserving vertex p, where $e = st \in E(G')$ is a lonely edge. If $p = s$ then the following are true.*

- $x, y, z \in V(G)$ *are e-preserving,*
- $t \in V(G)$ *is not e-preserving, and*
- *without loss of generality* $yz \in E(G)$ *is lonely (assuming* $xt \in E(G)$*).*

If $p \notin st$ *then the following then without loss of generality assume that* $xp' \in E(G)$ *and* $pp' \in M_e$ *where* M_e *is the color class of* e*. Then the following are true.*

- x *is not e-preserving and*
- y, z *are e-preserving.*

Lemma 5 *If a Klee graph has two adjacent lonely edges, then it is a Klee ladder.*

Lemma 6 *Let* G *be a Klee graph. Let* e_1 *and* e_2 *be two lonely edges of the same color. Then* G *has only two triangles, and each of* e_1 *and* e_2 *belongs to a triangle. (not both the same trivially). Moreover, by removing the lonely edges the graph becomes bipartite. Moreover, the bipartition is defined by A and B (which vertex is black for which lonely edge).*

Lemma 7 *Let* G *be a Klee graph. Let* e_1 *and* e_2 *be two lonely edges of two different colors. Then* e_1 *and* e_2 *are at distance 2, moreover, the graph is an "alternating ladder of ladders".*

Bibliography

[1] Thomas George Fowler: Unique coloring of planar graphs. `https://api.semanticscholar.org/CorpusID:126324586`

[2] Jan Goedgebeur and Davide Mattiolo and Giuseppe Mazzuoccolo and Jarne Renders and Isaak H. Wolf : Cubic graphs with edges in exactly one perfect `https://arxiv.org/abs/2402.08538`

[3] Kardoš, František and Máčajová, Edita and Zerafa, Jean Paul: Three-Cuts are a Charm: Acyclicity in 3-Connected Cubic Graphs. `http://dx.doi.org/10.1007/s00493-024-00126-y`

David Mikšaník
miksanik@iuuk.mff.cuni.cz

The Rosenfeld counting method

Introduction

The Rosenfeld counting method is a method for proving the existence of objects by way of clever counting, introduced by Rosenfeld in 2020 [3]. It turns out that in many cases the Rosenfeld counting method can be substituted for the Lovász Local Lemma (and/or the so-called Entropy compression method), which often yield slightly better results and simpler proofs. However, it is worth noting that neither method seems to be superior. Another advantage of the Rosenfeld counting method is that we prove "for free" not only the existence of one such object, but exponentially many.

We demonstrate this phenomenon by presenting a short proof by Martinsson of the celebrated Johansson-Molloy theorem: triangle-free graphs of maximum degree Δ can be colored with at most $(1 + o(1))\Delta/\ln \Delta$ colors (see Theorem 3).

Coloring triangle-free graphs

Before stating the result precisely, let us recall two well-known definitions.

Definition 1 *The* chromatic number *of a graph G is the smallest number $\chi(G)$ of colors required to (properly) color the vertices of G.*

Definition 2 *A graph G is said to be* triangle-free *if it does not contain a triangle (i.e., a complete graph on three vertices) as a subgraph.*

Theorem 3 *For every $\varepsilon > 0$, there exists $\Delta = \Delta(\varepsilon)$ such that every triangle-free graph G with maximum degree $\Delta(G) \geq \Delta$ has*

$$\chi(G) \leq (1 + \varepsilon)\frac{\Delta(G)}{\ln \Delta(G)}.$$

This theorem was proved by Molloy in 2019 [2], improving Johansson's earlier result [1] by a constant factor. The Molloy's bound on the chromatic number is tight up to a factor of at most two. Hurley and Pirot reproved and generalized this result using the Rosenfeld counting method [4], obtaining a considerably simpler proof. As a consequence, they obtained an exponential estimation on the number of colorings, which is also tight up to a factor of at most two.

In this talk, we present the proof of Theorem 3 by Martinsson [5], which is essentially the Hurley–Pirot proof, streamlined to establish "only" Theorem 3. The emphasis will be on demonstrating the Rosenfeld counting method.

Bibliography

[1] A. Johansson. *Asymptotic choice number for triangle-free graphs.* Technical Report 91-5, DIMACS, 1996.

[2] M. Molloy. *The list chromatic number of graphs with small clique number.* Journal of Combinatorial Theory, Series B, vol. 134, pp. 164–284, 2019.

[3] M. Rosenfeld. *Another Approach to Non-Repetitive Colorings of Graphs of Bounded Degree.* The Electronic Journal of Combinatorics, vol. 27(3), 2020.

[4] E. Hurley and F. Pirot. *Colouring locally sparse graphs with the first moment method.* https://arxiv.org/abs/2109.15215

[5] A. Martinsson. *A simplified proof of the Johansson-Molloy Theorem using the Rosenfeld counting method.* https://arxiv.org/abs/2111.06214

Ján Plachý
janci@kam.mff.cuni.cz

Presented paper by James Cook and Edward Pyne

Efficient Catalytic Graph Algorithms

(https://doi.org/10.4230/LIPIcs.ITCS.2026.43)

Introduction

In the catalytic model, an algorithm is given read and write access to an ordinary working tape, and a *catalytic* tape which has to be restored to it's original content at the end of the computation. In the *catalytic logspace* (**CL**) decision class, the working tape is of size $\mathcal{O}(\log n)$, while the catalytic tape is of polynomial size. It has been shown [1] that graph problems of $s \to t$ connectivity[2] and estimation of random walks[3] are in **CL**. However, the former work focused on the theoretical properties of the problems, with runtimes of the proposed algorithms bounded by large polynomials.

Efficient catalytic algorithms

In this talk, we present three simple, fast and practically implementable catalytic algorithms for the given problems on a graph G with n nodes and m edges:

1. a deterministic algorithm for $s \to t$ connectivity running in $\tilde{\mathcal{O}}(n^3 m)$ time and $\tilde{\mathcal{O}}(n^2)$ catalytic space[4], based on lifting G to a layered graph on $n+1$ layers,
2. a randomized improvement of the algorithm for $s \to t$ connectivity running in $\tilde{\mathcal{O}}(nm)$ time and $\tilde{\mathcal{O}}(n)$ catalytic space, being a first use of randomness in catalytic computing, which can be made *locally revertible* in polylog time at a cost of using $\tilde{\mathcal{O}}(n^3)$ catalytic space,
3. an algorithm for random walk estimation running in $\tilde{\mathcal{O}}(mT^2/\varepsilon)$ time, $\mathcal{O}(\log(mT/\varepsilon))$ workspace and $\tilde{\mathcal{O}}(nT \cdot \log(m/\varepsilon))$ catalytic space for a given error ε.

Definition 1 *An algorithm is* locally revertible *in time t if at any point during it's execution, the algorithm can be paused and queried on an index i of the initial configuration τ of the catalytic tape, returning τ_i in time t and being able to continue the execution.*

In order to use arithmetic over an arbitrary modulus q required by the randomized algorithm, the l-bit catalytic registers might need to be shifted by a random value.

Fact 2 *Choosing the largest d such that $qd \leq 2l$, applying a uniformly random shift $\beta \in [2l]$ to an l-bit register with initial configuration τ results in a value $(\tau + \beta) \mod 2l$, which is* valid *for q, (i.e. is smaller than qd) with probability $\geq 1 - 1/(d+1)$.*

Bibliography

[1] Harry Buhrman, Richard Cleve, Michal Kouckỳ, Bruno Loff, and Florian Speelman. *Computing with a full memory: catalytic space.* In Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing, STOC '14, pages 857–866, New York, NY, USA, 2014. Association for Computing Machinery. doi:10.1145/2591796.2591874.

[2] Whether vertices s and t are connected in a given simple directed graph.

[3] Estimating the probability a T-step random walk starting at vertex s ends at vertex t, within ε additive error.

[4] $\tilde{\mathcal{O}}$ ignores polylog factors.

Richarlotte Valérà Razafindravola
valera@fav.zcu.cz
Presented paper by Nadzieja Hodur, Monika Pilśniak, Magdalena Prorok, Ingo Schiermeyer

On k-colorability of *(bull,H)*-free graphs

(https://doi.org/10.1016/j.disc.2026.115054)

Introduction

The paper investigates the vertex coloring problem within graph classes defined by forbidden induced subgraphs, focusing in particular on graphs that exclude the bull graph and an additional graph H.

Defintions

Definition 1 *A **bull** in a graph is the subgraph obtained from K_3 by attaching two pendant edges to two distinct vertices of the triangle.*

Definition 2 *Given a non-trivial graph G, we say that G is **perfect** if $\chi(G') = \omega(G')$ for every induced subgraph G' of G, where $\chi(G')$ and $\omega(G')$ denote the chromatic number of G' and the clique number of G', respectively.*

Definition 3 *For an induced cycle C_p with $p \geq 2$ let $C_p[k_1, k_2, \ldots, k_p]$ denote the **clique expansion** of an induced cycle C_p , where its vertices $v_1, v_2, \ldots, v_p$ are replaced by complete graphs K_{k_i} for $1 \leq i \leq p$and additional edges between all pairs of vertices from consecutive cliques.*

Definition 4 *A **spindle graph** G is a graph of order $3p + 1$, $p \geq 1$, which contains a cycle $C : u_0u_1 \ldots u_{3p}u_0$, where $\{u_{3i-2}, u_{3i-1}, u_{3i+1}, u_{3i+2}\} = N_G(u_{3i})$ and $\{u_{3i-3}, u_{3i}\} = N_G(u_{3i-1}) \cap N_G(u_{3i-2})$ for each $i \in [p]$.*

Definition 5 *A **hole** in a graph G is an induced cycle of length at least 4, and an **antihole** in G is an induced subgraph whose complement is a cycle of length at least 4.*

Theorems

Theorem 6 *Let $n \geq 1$, $k \geq 3$, and let G be a clique expansion $C_{2n+1}[k_1, \ldots, k_{2n+1}]$. Then G is k-colorable if and only if the following two conditions are satisfied (all indices are taken modulo $2n + 1$):*

1. $\forall i \in [2n + 1],\ k_i + k_{i+1} \leq k$;
2. $\sum_{i=1}^{2n+1} k_i \leq nk$.

Theorem 7 *Let G be a connected (bull, claw)-free graph and $i \geq 1$. Then*

1. *G contains $\overline{C_7} \oplus K_1$, or*
2. *G contains $C_5 \oplus K_2$, or*
3. *G contains $M_4 \oplus K_1$ or $M_7 \oplus K_1$, or*
4. *G contains $C_{2i+1}[2, 2, \ldots, 2, 1, 3, 1, 3, \ldots, 1]$ or $C_{2i+1}[2, 2, \ldots, 2, 1]$,or*
5. *G contains an induced cycle C_5 and $|V(G)| > 8$, or*

6. *G is 4-colorable.*

Theorem 8 *Let G be a connected (bull, chair, C_5)-free graph and $i \geq 1$. Then*

1. *G contains $\overline{C_7} \oplus K_1$, or*
2. *G contains $C_{2i+1} \oplus K_2$, or*
3. *G contains $M_{3i+1} \oplus K_1$, or*
4. *G contains $C_{2i+1}[2, 2, \ldots, 2, 1, 3, 1, 3, \ldots, 1]$ or $C_{2i+1}[2, 2, \ldots, 2, 1]$, or*
5. *G is 4-colorable.*

Theorem 9 *Let G be a connected (bull, claw, C_5)-free graph. Then*

1. *G contains K_6, or*
2. *G contains $C_7 \oplus K_2$, or*
3. *G contains $C_9 \oplus K_1$, or*
4. *$\alpha(G) = 2$ and $|V(G)| \geq 11$, or*
5. *$\alpha(G) = 2$ and $\Delta(G) \geq 9$, or*
6. *G is a clique expansion $C_{2n+1}[k_1, \ldots, k_{2n+1}]$ with $k_1 + \cdots + k_{2n+1} - 5n > 0$, or*
7. *G is 5-colorable.*

Bibliography

[1] Nadzieja Hodur, Monika Pilśniak, Magdalena Prorok, Ingo Schiermeyer: On k-colorability of *(bull, H)*-free graphs
https://doi.org/10.1016/j.disc.2026.115054

Hana Salavcová
salavcovah@gmail.com

Fair allocation: What if we allow sharing?

Introduction

In many allocation problems, goods can be shared among multiple agents rather than assigned exclusively. This naturally leads to fair allocation models with sharing, where each good may be allocated to several agents, possibly at some cost.

We focus on fairness captured by the maximin share (MMS) benchmark. While MMS is a central notion in fair allocation, it may fail to exist in the classical setting where each good is assigned to a single agent, and only approximate guarantees are known.

We show that limited sharing can restore exact MMS guarantees in certain regimes and improve approximation bounds [1].

Sharing Model

Definition 1 (Instance) *An* instance *of fair division problem is a tuple* (N, M, v, k, c)*, where*

- $N = \{1, \dots, n\}$ *is a* ***set of agents****,*
- M *is a finite* ***set of goods****,*
- *each agent* $i \in N$ *has an additive* ***valuation***
$$v_i : 2^M \to \mathbb{R}_{\geq 0}, \quad v_i(S) = \sum_{g \in S} v_i(\{g\}), \text{ for all } S \subseteq M,$$
- $k \in \mathbb{N}$ *is the* ***sharing bound****, the maximum number of agents that can share a good,*
- $c_{i,g}(t) \in [0, 1]$ *is the* ***cost*** *incurred by agent* i *when good* g *is shared among* t *agents.*

Definition 2 (k-sharing Allocation) *A* k-sharing allocation *is a tuple* $A = (A_1, \dots, A_n)$*, where* $A_i \subseteq M$*, such that each good* $g \in M$ *is assigned to at most* k *agents, i.e.,* $|\{i \in N : g \in A_i\}| \leq k$*, and we allocate all the goods, i.e.,* $\bigcup_{i \in N} A_i = M$*.*

Definition 3 (Utility) *Given a* k*-sharing allocation* A*, the* utility *of agent* i *is*
$$u_i(A) = \sum_{g \in A_i} (1 - c_{i,g}(|N_g(A)|)) \cdot v_i(\{g\}),$$
where $N_g(A) = \{i \in N : g \in A_i\}$*.*

We consider the following cost models:

- **Cost-free sharing:** $c_{i,g}(t) = 0$ for all t,
- **Equal-share model:** $c_{i,g}(t) = 1 - \frac{1}{t}$ for all t,
- **Generous model:** each agent receives at least her equal-share value, i.e., $c_{i,g}(t) \leq 1 - \frac{1}{t}$.

Fairness Notions

Definition 4 (Proportionality (PROP)) *A k-sharing allocation A satisfies* proportionality *if for every agent $i \in N$:*

$$u_i(A) \geq \frac{1}{n} v_i(M).$$

Definition 5 (Maximin Share (MMS)) *A k-sharing allocation A is* MMS-fair *if*

$$u_i(A) \geq \mathrm{MMS}_i^n(M) \quad \textit{for all } i \in N,$$

where the maximin share of agent i is

$$\mathrm{MMS}_i^n(M) := \max_{(B_1,\ldots,B_n)\in\mathcal{A}^n(M)} \min_{j\in[n]} v_i(B_j),$$

and $\mathcal{A}^n(M)$ denotes the set of all possible n-partitions of the set of goods M.

Sharing Results

Theorem 6 (Akrami et al. [2]) *In the cost-free model, if $k \geq 2$, there exists a k-sharing allocation that guarantees each agent her MMS value.*

Observation 7 (Sharing among all agents) *In the generous model, if $k = n$, assigning every good to all agents guarantees each agent her MMS value.*

Theorem 8 (Sharing among half of the agents) *In the generous cost model, if $k \geq n/2$, there exists a k-sharing allocation that guarantees each agent her MMS value (up to parity effects).*

Theorem 9 (Shared Bag-Filling) *There exists a polynomial-time algorithm that computes a k-sharing allocation achieving:*

$$(1 - C)(k - 1)\text{-}MMS,$$

and exact MMS whenever $(1 - C)(k - 1) \geq 1$, where C is the maximum cost.

Bibliography

[1] Hana Salavcová, Martin Černý, and Arpita Biswas. Maximin Share Guarantees via Limited Cost-Sensitive Sharing. *arXiv preprint arXiv:2602.20541*, 2026. https://arxiv.org/abs/2602.20541

[2] Hannaneh Akrami, Alon Eden, Michal Feldman, Amos Fiat, and Yoav Gal-Tzur. Fair Division via Resource Augmentation. *arXiv preprint arXiv:2502.09377*, 2025. https://arxiv.org/abs/2502.09377

Jakub Smolík
smolikj@matfyz.cz

Presented paper by Julien Portier and Leo V. Versteegen

Proof of the 3/4-conjecture for the total domination game

(https://doi.org/10.1137/23M1551584)

Introduction

Let $G = (V, E)$ be a graph without isolated vertices. A vertex v is *totally dominated* by a set $A \subseteq V$ if it has a neighbour in A. The total domination game on G is played by 2 players, Dominator and Staller, who alternate in selecting vertices such that each newly selected vertex increases the number of vertices that are totally dominated by the set of selected vertices A. The game stops when A totally dominates every vertex of G. Dominator's aim is to minimize $|A|$, while Staller wants to maximize it. The *game total domination number* $\gamma_{tg}(G)$ is the number of vertices in the resulting set when Dominator starts the game and both players play optimally. Henning, Klavžar, and Rall [1] proved that if G has no isolated vertices or edges and $|V| = n$, then $\gamma_{tg}(G) \leq \frac{4}{5}n$, and they conjectured that in fact $\gamma_{tg}(G) \leq \frac{3}{4}n$. Portier and Versteegen [2] confirmed this conjecture.

Main idea of the proof

We describe a strategy for Dominator (D) that achieves this. Note that if v has not been played and all neighbours of v are already totally dominated, then v is unplayable. Intuitively, D wants the sum of the numbers of totally dominated vertices and unplayable vertices to grow as quickly as possible. Whether it is easier to totally dominate many new vertices or to make many of them unplayable changes over the course of the game. For this reason, the strategy of D consists of a number of different *phases* that follow each other linearly and come with separate sets of instructions for D.

Describing the state of the game

As observers of the game, we will keep some pieces of data that will help us analyze it. Vertices have a color: they are either *white* or *black*, and we may mark them as *depleted* or *dependent*.

- Every vertex starts **white**, and it may be colored **black** only once it has been totally dominated. Once colored **black**, we cannot change its color back to **white**.
- A vertex is only allowed to be marked as **depleted** if it has not been played so far and has no **white** neighbours. In particular, all **depleted** vertices are unplayable.
- A vertex is only allowed to be **dependent** if it has ≤ 1 **white** neighbours.
- A vertex is not allowed to be marked both as **depleted** and **dependent**, and if it has been marked in any way, we cannot remove the mark.

We use the following notation, where t denotes a point in our analysis after t moves have been played, always assuming that D has played according to our instructions.

- $\beta(t)$ = the number of black vertices
- $\delta(t)$ = the number of depleted vertices

- $\lambda(t)$ = the number of leaves that have been played or marked as depleted
- $\sigma(t)$ = the number of *unplayed* dependent vertices
- $\nu(t)$ = the number of undominated neighbours of dependent vertices
- $\chi(t) = \sigma(t) - \nu(t)$

Observation 1 *β, δ, λ and χ are all non-decreasing in t.*

Simplifications

Suppose for contradiction that the 3/4-conjecture is false, and fix a counter-example G that is minimal with respect to the number of its edges. The *parent* $p(x)$ of a leaf x is its unique neighbour.

Lemma 2 *No parent in G has more than one leaf.*

Lemma 3 *Let u be a parent in G with a leaf w and another neighbour v. Then $G - uv$ has an isolated edge.*

Definition 4 *A leaf x has* type A *if no neighbour of $p(x)$ is a parent, and it has* type B *otherwise.*

Observation 5 *If x is a leaf of type A, then $p(x)$ has exactly one other neighbour z, and z is neither a leaf nor a parent. We say that z is the* grandparent *of both x and $p(x)$.*

Phases of the game

There will be 6 phases, and for each $i \in [6]$, t_i is the last move before the end of phase i, and we let $t_0 = 0$. Denote by $T_i := t_i - t_{i-1}$ the number of moves in phase i. Each phase ends once it is Dominator's turn and a certain configuration for white vertices is no longer possible.

1. No grandparent is adjacent to two or more white parents, and no parent is adjacent to one or more white parents.
2. No grandparent is adjacent to a white parent and at least three white vertices overall.

During phase 3, we will decide after each move of Staller whether or not to set a reaction flag. A necessary condition for the phase to end is that the reaction flag is not set.

3. There exists no parent such that both its leaf and grandparent are white, and there exists no grandparent that is adjacent to a white parent and a second white vertex.
4. No vertex has three or more white neighbours.
5. No vertex has two white neighbours.
6. The last phase ends once no undominated vertex is left.

Bibliography

[1] Michael A. Henning, Sandi Klavžar, and Douglas F. Rall. *The 4/5 upper bound on the game total domination number*. Combinatorica, 37(2), 223-251, 2017.

[2] Julien Portier and Leo V. Versteegen. *Proof of the 3/4-conjecture for the total domination game*. SIAM Journal on Discrete Mathematics, 39(1), 1-18, 2025.

Jakub Štepo
jakub.stepo@gmail.com

Presented paper by Naoki Matsumoto

Kempe equivalence of 4-colorings of graphs on non-orientable surfaces

(https://www.sciencedirect.com/science/article/pii/S0012365X26000312)

Introduction

Let k be a fixed natural number. If we have a k-colouring of a graph G, a *Kempe change* is a natural way of constructing more k-colourings. Such a change is performed by looking at one component of the subgraph induced by two colour classes and swapping the two colours in this component. As a strengthening of colourability, we may then consider the question whether all k-colourings of G are *Kempe-equivalent*; that is, reachable from one to another by a sequence of Kempe changes. Such problems are best studied in a geometric setting, for example for planar graphs.

Observation 1 *For any k, all k-colourings of a bipartite graph are Kempe-equivalent.*

Theorem 2 (Meyniel) *All 5-colourings of a planar graph are Kempe-equivalent.*

Note 3 *The above is not true for 3- or 4-colourings.*

Theorem 4 (Fisk) *All 4-colourings of an Eulerian triangulation of the sphere are Kempe-equivalent.*

Note 5 *The above is not true for any other orientable surface. It is also not true for sphere triangulations in general.*

Theorem 6 (Mohar) *All 4-colourings of a 3-colourable planar graph are Kempe-equivalent.*

Main result

Theorem 7 *All 4-colourings of a 3-colourable triangulation of the projective plane or the Klein bottle are Kempe-equivalent.*

Theorem 8 *There is a 4-colourable graph embeddable in the projective plane for which that does not hold. Also, there is a 3-colourable triangulation of the non-orientable surface of genus 3 for which that does not hold.*

The proof

We shall employ the following terminology. For colours i, j, we speak of (i, j)-*edges* (which connect those two colours), (i, j)-*cycles* (made of (i, j)-edges), (Kempe) (i, j)-*changes*. A cycle on a non-orientable surface may be either *1-sided* or *2-sided*, depending on whether its neighbourhood is a Möbius strip. It may also be either *separating* (i.e., it splits the surface into two parts) or not. A *facial path* is a sequence of faces $F_0, F_1, \dots F_n$ such that F_i and F_{i+1} either coincide or share an edge – in the latter case, we say that the path *passes* the edge. For a 4-colouring of a triangulation, we call an edge e *singular* if the third vertices of the triangles with e have the same colour.

We consider a specific triangulation T_0 of the torus, on Figure 1.

Lemma 9 *If a triangulation T of a surface has both a 4-colouring f and a 3-colouring g, then there is a simplicial map $f \times g : T \to T_0$ such that for each vertex v, $(f \times g)(v) = f(v)g(v)$.*

Lemma 10 *For faces A, A' of T_0 as in the figure, let $\mathcal{P}$ be a facial path from A to A' which passes no $(1, 2)$-edges. Then there is a $\pi \in S_4$ such that $\mathcal{P}$ passes both $(\pi(1), \pi(2))$- and $(\pi(3), \pi(4))$-edges an even, non-zero number of times.*

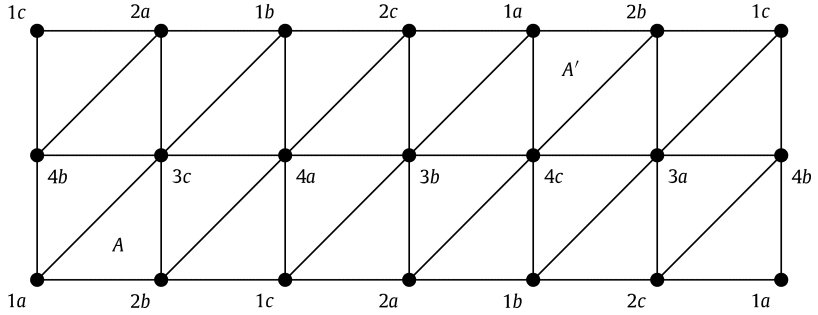


Figure 7: The triangulation T_0

For contradiction, we consider the counterexample with the least number of non-singular edges in a 4-colouring. There, let H be the subgraph induced by the non-singular (i, j)-edges for some i, j.

Proposition 11 *The graph H is Eulerian.*

Proposition 12 *Every cycle in H is non-separating.*

Proposition 13 *If C is a 1-sided cycle in H, it is not a component of H.*

Katarzyna Szwed
katarzyna.szwed.stud@pw.edu.pl
Presented paper by Ce Jin, Virginia Vassilevska Williams and Renfei Zhou

Listing 6-cycles

(https://epubs.siam.org/doi/10.1137/1.9781611977936.3)

Introduction

The enumeration of small subgraphs, such as triangles, 4-cycles and small cliques, is a well-studied and important task in algorithmic graph theory. The authors of this paper propose a simple algorithm for listing t (non-induced) 6-cycles in an n-node input graph in $\tilde{O}(n^2 + t)$ time (where $\tilde{O}(f)$ denotes $O(f \cdot \text{polylog}(f))$). This result is close to the fastest known 6-cycle detection algorithm by Yuster and Zwick (1997), which works in $O(n^2)$ time.

The proposed algorithm uses a similar idea to the one in the folklore 4-cycle listing algorithm (which works in $O(n^2+t)$ time), where a 4-cycle a — b_1 — c, a — b_2 — c is made by pasting together paths a — b_1 — c and a — b_2 — c where $b_1 \neq b_2$.

Here we form a 6-cycle a — b_1 — c_1 — d, a — b_2 — c_2 — d from paths a — b_1 — c_1 — d and a — b_2 — c_2 — d where $b_1 \neq b_2$ and $c_1 \neq c_2$. The new challenge is to avoid wasting time on cases $b_1 = b_2$ and $c_1 = c_2$, which do not result in valid 6-cycles.

Main result

Theorem 1 (Main result, Listing t 6-Cycles) *There is a deterministic algorithm that lists t copies of C_6 in an n-node undirected simple graph in $\tilde{O}(n^2 + t)$ time.*

Theorem 2 (Listing all 6-Cycles) *There is a deterministic algorithm that lists all 6-cycles of the n-node input graph G in $O((n^2 + t)\log(n))$ time, where t denotes the total number of 6-cycles in G.*

Lemma 3 *Given a 4-partite undirected graph $G = (V, E)$ where $V = A \sqcup B \sqcup C \sqcup D$, we can list all 6-cycles whose 6 nodes are in A, B, C, D, C, B respectively in the order they appear on the cycle, in $O(n^2 + t)$ total time, where t denotes the total number of 6-cycles in G.*

Proof Sketch of Lemma 3. The algorithm consists of two stages: (i) Preprocessing several tables and (ii) reporting 6-cycles based on the data from these tables. Lemma 4 will be used to bound the table sizes and, with that, the time complexity of the algorithm.

Lowercase letters a, b, c, d will denote nodes in A, B, C, D respectively, unless otherwise stated. $N_x := \{u \in V : (u, x) \in E\}$ denotes the set of neighbors of $x \in V$. In the algorithm we define the following tables:

- $N_{a,c} := N_a \cap N_c \cap B$ (the set of common neighbors $b \in B$ of a and c).
- $N_{b,d} := N_b \cap N_d \cap C$ (the set of common neighbors $c \in C$ of b and d).

- Let $P_{a,d}$ be the set of $b \in N_a \cap B$ such that $|N_{b,d}| \geq 2$. In other words, it is the set of b that can form the shape $a \text{ —— } b \begin{array}{l} \diagup \\ \diagdown \end{array} \begin{array}{l} c_1 \\ c_2 \end{array} \begin{array}{l} \diagdown \\ \diagup \end{array} d$.

- Let $Q_{a,d}$ be the set of $c \in N_d \cap C$ such that $|N_{a,c}| \geq 2$. In other words, it is the set of c that can form the shape $a \begin{array}{l} \diagup \\ \diagdown \end{array} \begin{array}{l} b_1 \\ b_2 \end{array} \begin{array}{l} \diagdown \\ \diagup \end{array} c \text{ —— } d$.

- Let $R_{a,d}$ be the set of edges $(b, c) \in E \cap ((B \cap N_a) \times (C \cap N_d))$ such that $|N_{a,c}| = |N_{b,d}| = 1$. That is, b and c for which exists the path $a \text{ —— } b \text{ —— } c \text{ —— } d$ without any possible *replacements* of b and c. Meaning, there are no paths $a \text{ —— } b' \text{ —— } c \text{ —— } d$ where $b' \neq b$ or $a \text{ —— } b \text{ —— } c' \text{ —— } d$ where $c' \neq c$.

Lemma 4 *If node $a \in A$ satisfies $\sum_{c \in C} |N_{a,c}| \geq 100n + k$, then G contains at least k 6-cycles in which a is the only node in A.*

Wiktor Szymonek
szymonekwiktor@gmail.com

Presented paper by James Brownlie and Sean Jaffe

The Induced Saturation Number for $\mathcal{V}_3$ is Linear

(https://arxiv.org/abs/2507.08353v1)

Introduction

The relatively young concept of poset saturation, while analogous to better understood saturation for graphs, remains hard to analyze (particularly in its induced version). A major conjecture in this area states that for any poset $\mathcal{P}$, $sat^*(n, \mathcal{P})$ is either bounded or grows linearly. So far, known results show that in the general case, the induced saturation number is either bounded or at least $2\sqrt{n}$. Linear growth, however, has been shown for a small number of simple posets.

The paper presented in this talk focuses on $\mathcal{V}_3$ and its symmetrical counterpart Λ_3. These posets remained the two simplest examples with an unknown order of growth of their induced saturation numbers. The presented proof that $sat^*(n, \mathcal{V}_3) \geq \frac{n}{2}$ establishes linear order of growth for these posets.

Preliminaries

Definition 1 (Induced copy) *We say that a poset $(\mathcal{Q}, \leq')$ contains an induced copy of a poset $(\mathcal{P}, \leq)$ if there exists an injective function $f : \mathcal{P} \to \mathcal{Q}$ such that $(\mathcal{P}, \leq)$ and $(f(\mathcal{P}), \leq')$ are isomorphic.*

Definition 2 (Induced saturation) *Given a fixed poset $\mathcal{P}$, a family $\mathcal{F}$ of subsets of $[n]$ is $\mathcal{P}$-saturated if $\mathcal{F}$ does not contain an induced copy of $\mathcal{P}$, but $\mathcal{F} \cup \{S\}$ contains an induced copy of $\mathcal{P}$ for all $S \notin \mathcal{F}$. The induced saturation number of $\mathcal{P}$, denoted $sat^*(n, \mathcal{P})$, is the minimum size of a $\mathcal{P}$-saturated family in the Boolean lattice $\mathcal{P}([n])$.*

Definition 3 (Posets $\mathcal{V}_3$ and Λ_3) *The poset $\mathcal{V}_3$ is the four-element poset with one minimal element and three incomparable maximal elements. Its dual poset Λ_3 consists of one maximal element above three incomparable minimal elements.*

Results

The main theorem of the paper provides a linear lower bound for induced saturation number of $\mathcal{V}_3$:

Theorem 4 $sat^*(n, \mathcal{V}_3) \geq \frac{n}{2}$.

By symmetry, the same applies to Λ_3:

Corollary 5 $sat^*(n, \Lambda_3) \geq \frac{n}{2}$.

Knowing that a union of two full chains in $\mathcal{P}([n])$, which intersect at exactly $\emptyset$ and $[n]$, is $\mathcal{V}_3$-saturated and of size $2n$ gives us $sat^*(n, \mathcal{V}_3) \leq 2n$. Thus, we can say that $sat^*(n, \mathcal{V}_3) = \Theta(n)$ (the same applies to Λ_3).

Combining lower bounds for $\mathcal{V}_3$ and Λ_3 with Proposition 5 from [1] we can generalize results for a wider class of posets constructed by the linear sum operation (where $\mathcal{P} * \mathcal{Q}$ denotes the poset formed by placing $\mathcal{P}$ entirely on top of $\mathcal{Q}$):

Corollary 6 *Let $\mathcal{P}$ be any non-empty poset that does not have a unique maximal element. Then $sat^*(n, \mathcal{A}_3 * \mathcal{P}) \geq sat^*(n, \mathcal{V}_3) \geq \frac{n}{2}$, where $\mathcal{A}_3$ is the antichain of size 3. Likewise, if $\mathcal{P}$ is a non-empty poset that does not have a unique minimal element then $sat^*(n, \mathcal{P} * \mathcal{A}_3) \geq sat^*(n, \Lambda_3) \geq \frac{n}{2}$.*

Bibliography

[1] Maria-Romina Ivan and Sean Jaffe. *Gluing Posets and the Dichotomy of Poset Saturation Numbers.* arXiv:2503.12223 (2025).
https://arxiv.org/abs/2503.12223

Filip Úradník
uradnik@kam.mff.cuni.cz

Reliability of Networks

Introduction

In this talk, we investigate how to solve the maximum flow problem in networks with unreliable edge capacities. We assume that each edge can *fail* randomly, either fully (meaning its capacity becomes zero), or partially (meaning its capacity decreases). We also assume that the edge failures are independent of each other. Our aim is to analyze the maximum flow, specifically computing the probability that at least d units of flow can still pass through the network.

Reliability

Let us fix the following notation. We are given a network $N = (G, s, t, c)$, where G is a digraph, s, t are the source and target, and $c : E \to \mathbb{N}$ are the capacities. For each $e \in E$, let $p_e : \{0, \ldots, c(e)\} \to [0, 1]$ be a probability distribution. We define a *random (sub)network* of N as follows.

Definition 1 *A* random subnetwork *of N according to the probabilities $(p_e)_{e \in E}$ is a network denoted as $\mathcal{N}$, defined on the same graph G with the capacity of each edge chosen independently from $\{0, \ldots, c(e)\}$ according to the probability distribution p_e.*

Our aim is to solve the so-called *d-reliability* problem, which can be stated as follows.

Definition 2 (d-reliability) *Let $d \in \mathbb{N}$. The d-reliability of $\mathcal{N}$ is*

$$R^d(\mathcal{N}) = \Pr[\textit{size of a maximum flow in } \mathcal{N} \textit{ is at least } d].$$

Unfortunately, it has been known for a long time that this is a hard problem. In fact, Valiant even included a special case of this problem (with $d = 1$) in his so-called 'Original 13' #P-complete problems.

In this talk, we investigate the complexity results in this area, and our main goal will be to prove the following theorem.

Theorem 3 *Let k be the treewidth of G. Then the d-reliability of $\mathcal{N}$ can be computed in*

$$\mathcal{O}(f(k, d) \cdot n)$$

for some function f independent of n.

In the language of parameterized complexity, we show that the d-reliability problem is fixed-parameter tractable when parameterized by the treewidth k and the demand d. In other words, if d and k are constant, the problem is solvable in $\mathcal{O}(n)$.

Throughout our talk, we will use the notation $N_a \sim \mathcal{N}$ to mean a model of $\mathcal{N}$, where $a \leq c$ is the reduced capacity of the edges.

Treewidth

While this talk will probably be best appreciated by people familiar with treewidth, I will attempt to build some intuition around it, as we will only need some basic properties. The basic idea is to try to overlay a tree over the vertices of the graph as follows.

Definition 4 (Tree decomposition) *A tree decomposition of G is a tuple (T, β), where $T = (V_T, E_T)$ is a rooted tree (with root r) and $\beta : V_T \to 2^V$ is the* bag function *satisfying*

1. *for all $v \in V$, there is a node of the tree $t \in V_T$ such that $v \in \beta(t)$,*
2. *for all $uv \in E$, there is a node of the tree $t \in V_T$ such that $u, v \in \beta(t)$,*
3. *for all $v \in V$, the nodes containing v in their bags form a connected subtree; or in other words, the following graph is connected:*
$$T[\{t;\ v \in \beta(t)\}].$$

We want to make T match the graph G in some sense *as close as possible*. If G is itself a tree, its decomposition can simply have each bag of size 2—each bag covers one edge. In general, we want a tree decomposition which minimizes the maximum size of a bag, or its *width*.

Definition 5 (Width) *The* width *of a decomposition (T, β) is*
$$w(T, \beta) = \max_{t \in V_T} |\beta(t)|.$$
The treewidth *of a graph G is* $\mathrm{tw}(G) = \min_{T,\beta} w(T, \beta)$.

The idea is that the treewidth of a graph is in some sense a measure of how much the graph resembles a tree. Much like many problems are hard in general and easy for trees, it turns out that many problems are still easy on graphs of constant treewidth. There are in general many tree decompositions of the optimal width. We will work with a so-called *nice* tree decomposition, defined as follows.

Definition 6 *A tree decomposition is* nice, *if each node $t \in V_T$ is one of the following types:*

1. *a* leaf node *with zero children,*
2. *a* join node *with exactly two children t_1, t_2 such that*
$$\beta(t) = \beta(t_1) = \beta(t_2),$$
3. *an* introduce node *introducing an element $w \in V$ with exactly one child t' such that*
$$\beta(t) = \beta(t') \cup \{w\},$$
4. *a* forget node *forgetting an element $w \in \beta(t')$ with exactly one child t' such that*
$$\beta(t) = \beta(t') \setminus \{w\}.$$

It is known that a nice tree decomposition of the optimum width always exists, and has $\mathcal{O}(\mathrm{tw}(G) \cdot n)$ nodes. We can also find it efficiently.

External Capacity

The basic idea of the proof is to ignore the edges which are *below* a bag $\beta(t)$, thinking of them only as means to transfer capacity between vertices of $\beta(t)$.

Definition 7 *We denote as E_t the edges within $\beta(t)$, and E_{-t} edges whose endpoints are in some bag $\beta(t')$ below t, but not in $\beta(t)$.*

The meaning of the following definition will be explored further in the talk, but I want to write it formally here for ease of reference:

Definition 8 (External flow, capacity) *Let N_a be a model of $\mathcal{N}$ and let $t \in V_T$. An* external flow under t *is a tuple $(f_{u,v};\ uv \in E_t)$, such that $f_{u,v}$ routes some flow from u to v, meaning it satisfies*

1. *for each $uv \in E_t$, $f_{u,v} : E_{-t} \to \mathbb{N}$, and for all $w \in V$ different from u, v, which is either in $\beta(t)$, or below t,*
$$\sum_{e \in \delta^+_{-t} w} f_{u,v}(e) = \sum_{e \in \delta^-_{-t} w} f_{u,v}(e),$$
2. *for each edge $e \in E$ below t,*
$$\sum_{uv \in E_t} f_{u,v}(e) \leq a(e),$$
3. *for each vertex $w \in V$ below t,*
$$\sum_{uv \in E_t} \sum_{e \in \delta^+_{-t} w} f_{u,v}(e) \leq d.$$
4. *for each vertex $w \in \beta(t)$ and $uv \in E_t$ such that $u, v \neq w$,*
$$\sum_{e \in \delta^+_{-t} w} f_{u,v}(e) = 0.$$

A function $x : E_t \to [d+1]$ is an external capacity of t in N_a, *if there exists an external flow under t $(f_{u,v};\ uv \in E_t)$, such that for each $uv \in E_t$,*
$$x(uv) = \sum_{e \in \delta^-_{-t}(u)} f_{u,v}(e) - \sum_{e \in \delta^+_{-t}(u)} f_{u,v}(e) \leq d.$$

Bibliography

[1] Valiant, Leslie G. *The Complexity of Enumeration and Reliability Problems.* SIAM Journal on Computing. Vol. 8, no. 3, pp. 410–421, 1979.

[2] Kloks, Ton. *Treewidth, Computations and Approximations.* Vol. 842. Springer. Lecture Notes in Computer Science, 1994.

[3] J. S. Lin, C. C. Jane, and J. Yuan, *On reliability evaluation of a capacitated-flow network in terms of minimal pathsets.* Networks, vol. 25, pp. 131–138, 1995.

Samuel Vaško
samuel-mff@vasko.cc

Presented paper by Derakhshan, Roghani, Yu, Saneian

A Simple Analysis of Ranking in General Graphs

(https://arxiv.org/abs/2511.08801)

Introduction

In this work, we study a randomized greedy matching algorithm called Ranking for general graphs. This algorithm was first introduced in the seminal work of Karp, Vazirani, and Vazirani in 1990 for online bipartite matching with one-sided vertex arrivals and was subsequently extended to general graphs and bipartite graphs with random vertex arrivals. The algorithm belongs to a class of matching algorithms called vertex-iterative (VI) randomized greedy matching algorithms. These algorithms first draw a permutation π over the vertices uniformly at random. They then iterate over the vertices in the order of π, matching each vertex to one of its available neighbors (if any) according to a specific neighbor selection policy. In Ranking for general graphs, this policy selects the first unmatched neighbor in the order given by the same permutation π used for the initial iteration.

Proposition 1.1 ([1, Lemma 1]). Let $\alpha > 0$ and M be a maximal matching of G such that $|M| \leq (1/2 + \alpha) \cdot \mu(G)$. Then, at least $(1/2 - 3\alpha) \cdot \mu(G)$ edges of M are in disjoint, length-three augmenting paths.

Combinatorial Tools. We use the following two well-known bounds on the binomial coefficient.

Proposition 1.2 ([2, Chapter 3]). Let $a_1, a_2, \ldots, a_n$ be n positive integers such that $\sum_{i=1}^{n} a_i = m$. Then, for a positive integer x, $\sum_{i=1}^{n} \binom{a_i}{x}$ is minimized when $\{a_1, \ldots, a_n\} = \{\lfloor m/n \rfloor, \ldots, \lceil m/n \rceil\}$.

Proposition 1.3 ([3, Chapter 11]). Let n and αn be two positive integers and $1/n \leq \alpha \leq 1/2$. Then, we have

$$\binom{n}{\alpha n} \leq 2^{nH(\alpha)}$$

where

$$H(\alpha) = -\alpha \log_2(\alpha) - (1 - \alpha) \log_2(1 - \alpha).$$

Definition 2.1 (k-wasteful independent set (k-WIS)). Given matching R the output of Ranking, a subset of $2k$ vertices I is a k-wasteful independent set, iff vertices in I are end-points of k length-three augmenting paths in $R \oplus OPT$. This also implies that I is an independent set of G since Ranking outputs a maximal matching and vertices in I are left unmatched in R. Figure 1 shows an example of a 5-WIS.

Lemma 2.2 (Lower Bound on the Expected Number of k-WIS). If the approximation ratio of Ranking is at most $1/2 + c$ for $0 \leq c \leq 1/6$, then the expected number of k-WIS in the output of Ranking is at least one, when $k = (1/2 - 3c) \cdot n/2$.

Claim 2.3. There exist at most $\binom{n/2}{2k} \cdot 3^k$ different k-WIS for all possible outputs of Ranking on input graph G.

Claim 2.4. Let $I = \{v_1, v_2, \ldots, v_{2k}\}$ be a set of independent vertices in G. Then, the vertices in I form a k-WIS with probability at most $1/2^{2k}$ in the output of RANKING.

Lemma 2.5 (Upper Bound on the Expected Number of k-WIS). The expected number of k-WIS in the output of RANKING is at most $\binom{n/2}{2k} \cdot (3/4)^k$.

Theorem 2.6. The expected approximation ratio of RANKING is at least 0.505 in general graphs.

Bibliography

[1] Christian Konrad, Frédéric Magniez, and Claire Mathieu. *Maximum matching in semi-streaming with few passes.* In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques – APPROX 2012 and RANDOM 2012*, Lecture Notes in Computer Science, vol. 7408, pages 231–242. Springer, 2012.

[2] Albert W. Marshall, Ingram Olkin, and Barry C. Arnold. *Inequalities: Theory of Majorization and Its Applications*. 1979.

[3] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, 2006.

Bartosz Wójcik
bartosz.wojcik2.stud@pw.edu.pl

Presented paper by Jean-Christophe Filliâtre

Verifying Two Lines of C with Why3: An Exercise in Program Verification

(https://doi.org/10.1007/978-3-642-27705-4_8)

Introduction

Formal verification of even short, algorithms, if sufficiently complex, can prove challenging. In this talk we will prove the correctness of a 2-line solution to the n-Queens problem with the Why3 platform. This will serve as an exercise to showcase the difficulties and problems of formaly checking a programs correctness.

Problem formulation

Given $n \in \mathbb{N}$, in how many ways n queens can be placed on a $n \times n$ chessboard, such that no two queens attack each other.

2-line C solution

The following code by Marcel van Kervinc seems to accept a number and output the number of solutions to the n-Queens problem:

```
t(a,b,c){int d=0,e=a&~b&~c,f=1;if(a)for(f=0;d=(e-=d)&-e;f+=t(a-d,(b+d)*2,(
c+d)/2));return f;}main(q){scanf("%d",&q);printf("%d\n",t(~(~0<<q),0,0));}
```

After deobfuscating the code we get the following:

```
int t(int a, int b, int c) {
  int d, e=a&~b&~c, f=1;
  if (a)
  for (f=0; d=e&-e; e-=d)
    f += t(a-d,(b+d)*2,(c+d)/2);
  return f;
}
int queens(int n) {
  return t(~(~0<<n),0,0);
}

int main() {
  int n;
  scanf("%d", &n);
  printf("%d\n", queens(n));
}
```

Algorithm verification

Our goal will be to translate the C code into the WhyML language for formal verification. For clarity the C code needs to be first translated into pseudocode:

Procedure t (set a, set b, set c)
$f \leftarrow 1$
if $a \neq \emptyset$ **then**
  $e \leftarrow (a \setminus b) \setminus c$
  $f \leftarrow 0$
  while $e \neq \emptyset$ **do**
    $d \leftarrow minimal\_element(e)$
    $f \leftarrow f + t(a \setminus d, successor(b \cup \{d\}), predecessor(c \cup \{d\}))$
    $e \leftarrow e\ d$
  end while
end if
return f

Now we can implement the same procedure in Why3:

```
let rec t (a b c: set int) =
  if not (is empty a) then begin
  let e = ref (diff (diff a b) c) in
  let f = ref 0 in
  while not (is empty !e) do
    let d = min elt !e in
    f := !f + t (remove d a) (succ (add d b)) (pred (add d c));
    e := remove d !e
  done;
  !f
end else
  1

let queens (q: int) =
  t (below q) empty empty
```

Bibliography

[1] Jean-Christophe Filliâtre *Verifying Two Lines of C with Why3: An Exercise in Program Verification.*

Jakub Żuchowski
jakub.zuchowski.stud@pw.edu.pl
Presented paper by Ryan Williams

Simulating Time With Square-Root Space

(https://arxiv.org/abs/2502.17779)

Introduction

A classical question in complexity theory asks how much memory is needed to simulate a long computation. If a multitape Turing machine runs in time $t(n)$, can one always simulate it using much less than $t(n)$ space? This is part of the broader study of time–space tradeoffs and of the relationship between time-bounded and space-bounded computation.

In the paper *Simulating Time With Square-Root Space*, Ryan Williams proves that every multitape Turing machine running in time $t(n) \geq n$ can be simulated in

$$\mathcal{O}\left(\sqrt{t(n)\log t(n)}\right)$$

space. This significantly improves the previously known general upper bound of $\mathcal{O}(t(n)/\log t(n))$ due to Hopcroft, Paul, and Valiant from 1975. The result is striking because it applies to completely general computations, not only special models or algorithms, and was widely believed to be impossible before this work.

Main idea

The proof is not based on a direct step-by-step simulation. Instead, the computation is reorganized so that simulating the original machine reduces to evaluating an implicitly defined instance of the *Tree Evaluation* problem. One then applies a recent space-efficient algorithm for Tree Evaluation by Cook and Mertz (STOC 2024) as a blackbox. Once this reduction is in place, the $\mathcal{O}(\sqrt{t \log t})$ space bound drops out almost automatically from the Cook-Mertz result.

At a high level, the computation is partitioned into time and tape blocks, organized into a tree structure where each node evaluates a short segment of the computation given its children's outputs. Because the tree can be navigated implicitly without storing it explicitly, only square-root space is needed to recover the final answer.

Why this is interesting

The theorem gives a new unconditional connection between time and space complexity, improving a bound that had stood for fifty years. The proof is conceptually clean: most of the technical heavy lifting is delegated to Cook and Mertz, leaving a reduction that is both short and illuminating.

Besides the bound itself, the result opens new directions toward separating P from PSPACE: if the simulation could be pushed further to $\mathcal{O}(t(n)^{\varepsilon})$ space for all $\varepsilon > 0$, that separation would follow.

Bibliography

[1] Ryan Williams. *Simulating Time With Square-Root Space*. arXiv preprint arXiv:2502.17779, 2025.

[2] Władysław Homenda and Witold Pedrycz. *Automata Theory and Formal Languages*. De Gruyter, 2022.

List of participants

Adam Beneš

Bogdan Błędziński

Daniel Čech

Eliška Červenková

Petr Chmel

Rachel Dufau-Sansot

Barbora Dohnalová

Alica Dományová

Adam Džavoronok

Aleksa Džuklevski

Roman Edenhofer

Filip Filipkowski

Jan Frnka

Vojtěch Gaďurek

Milan Hladík

Pavel Hubáček

Karolína Hylasová

Igor Januszkiewicz

Michal Katrlík

Jakub Komárek

Michal Koucký

Martin Kopřiva

Júlia Križanová

Terezie Lejbová

Aleksander Łukasiewicz

Kristýna Mašková

Matúš Matok

David Mikšaník

Miroslav Olšák

Sejin Park

Ján Plachý

Richarlotte Valérà Razafindravola

Hana Salavcová

Robert Šámal

Jakub Smolík

Jakub Štepo

Katarzyna Szwed

Wiktor Szymonek

Martin Tancer

Filip Úradník

Samuel Vaško

Pavel Veselý

Bartosz Wójcik

Jakub Żuchowski