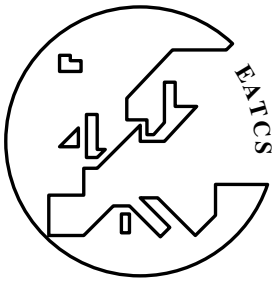
ISSN 0252–9742

Bulletin

of the

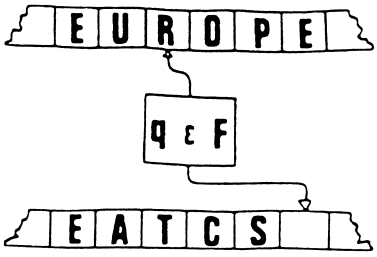
European Association for Theoretical Computer Science

EATCS



Number 136 February 2022

Council of the European Association for Theoretical Computer Science



President:	Artur Czumaj	United Kingdom
Vice Presidents:	Anca Muscholl	France
	Giuseppe F. Italiano	Italy
Treasurer:	Jean-Francois Raskin	Belgium
Bulletin Editor:	Stefan Schmid	Germany

Ivona Bezakova	USA	Anca Muscholl	France
Tiziana Calamoneri	Italy	Luke Ong	UK
Thomas Colcombet	France	Tal Rabin	USA
Artur Czumaj	UK	Eva Rotenberg	Denmark
Javier Esparza	Germany	Maria Serna	Spain
Fabrizio Grandoni	Switzerland	Alexandra Silva	USA
Thore Husfeldt	Sweden, Denmark	Jiri Sgall	Czech Republic
Giuseppe F. Italiano	Italy	Ola Svensson	Switzerland
Fabian Kuhn	Germany	Jukka Suomela	Finland
Slawomir Lasota	Poland	Till Tantau	Germany
Elvira Mayordomo	Spain	Sophie Tison	France
Emanuela Merelli	Italy	Gerhard Wöeginger	The Netherlands

Past Presidents:

Maurice Nivat	(1972–1977)	Mike Paterson	(1977–1979)
Arto Salomaa	(1979–1985)	Grzegorz Rozenberg	(1985–1994)
Wilfred Brauer	(1994–1997)	Josep Díaz	(1997–2002)
Mogens Nielsen	(2002–2006)	Giorgio Ausiello	(2006–2009)
Burkhard Monien	(2009–2012)	Luca Aceto	(2012–2016)
Paul Spirakis	(2016–2020)		

Secretary Office:	Efi Chita	Greece
	Emanuela Merelli	Italy

EATCS Council Members

email addresses

Ivona Bezakova . . . . . ib@cs.rit.edu
Tiziana Calamoneri . . . . . calamo@di.uniroma1.it
Thomas Colcombet . . . . . Thomas.Colcombet@irif.fr
Artur Czumaj . . . . . A.Czumaj@warwick.ac.uk
Javier Esparza . . . . . esparza@in.tum.de
Fabrizio Grandoni . . . . . fabrizio@idsia.ch
Thore Husfeldt . . . . . thore@itu.dk
Giuseppe F. Italiano . . . . . giuseppe.italiano@uniroma2.it
Fabian Kuhn . . . . . kuhn@cs.uni-freiburg.de
Slawomir Lasota . . . . . sl@mimuw.edu.pl
Elvira Mayordomo . . . . . elvira@unizar.es
Emanuela Merelli . . . . . emanuela.merelli@unicam.it
Anca Muscholl . . . . . anca@labri.fr
Luke Ong . . . . . luke.Ong@cs.ox.a.uk
Tal Rabin . . . . . chair.sigact@sigact.acm.org
Jean-Francois Raskin . . . . . jraskin@ulb.ac.be
Eva Rotenberg . . . . . eva@rotenberg.dk
Maria Serna . . . . . mjserna@cs.upc.edu
Stefan Schmid . . . . . stefan.schmid@tu-berlin.de
Alexandra Silva . . . . . alexandra.silva@cornell.edu
Jiri Sgall . . . . . sgall@iuuk.mff.cuni.cz
Ola Svensson . . . . . Ola.Svensson@epfl.ch
Jukka Suomela . . . . . jukka.suomela@aalto.fi
Till Tantau . . . . . tantau@tcs.uni-luebeck.de
Sophie Tison . . . . . sophie.tison@lifl.fr
Gerhard Wöeginger . . . . . g.j.woeginger@math.utwente.nl

Bulletin Editor: Stefan Schmid, Berlin, Germany
Cartoons: DADARA, Amsterdam, The Netherlands

The bulletin is entirely typeset by pdfTEX and ConTEXt in TXfonts.

All contributions are to be sent electronically to

bulletin@eatcs.org

and must be prepared in LaTeX 2ε using the class `beatcs.cls` (a version of the standard LaTeX 2ε `article` class). All sources, including figures, and a reference PDF version must be bundled in a ZIP file.

Pictures are accepted in EPS, JPG, PNG, TIFF, MOV or, preferably, in PDF. Photographic reports from conferences must be arranged in ZIP files layed out according to the format described at the Bulletin's web site. Please, consult `http://www.eatcs.org/bulletin/howToSubmit.html`.

We regret we are unfortunately not able to accept submissions in other formats, or indeed submission not *strictly* adhering to the page and font layout set out in `beatcs.cls`. We shall also not be able to include contributions not typeset at camera-ready quality.

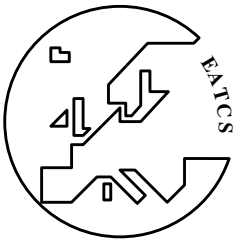
The details can be found at `http://www.eatcs.org/bulletin`, including class files, their documentation, and guidelines to deal with things such as pictures and overfull boxes. When in doubt, email `bulletin@eatcs.org`.

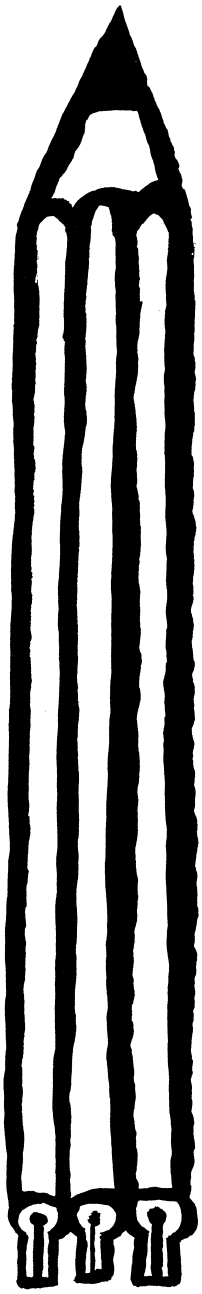
Deadlines for submissions of reports are January, May and September 15th, respectively for the February, June and October issues. Editorial decisions about submitted technical contributions will normally be made in 6/8 weeks. Accepted papers will appear in print as soon as possible thereafter.

The Editor welcomes proposals for surveys, tutorials, and thematic issues of the Bulletin dedicated to currently hot topics, as well as suggestions for new regular sections.

Table of Contents

EATCS Matters





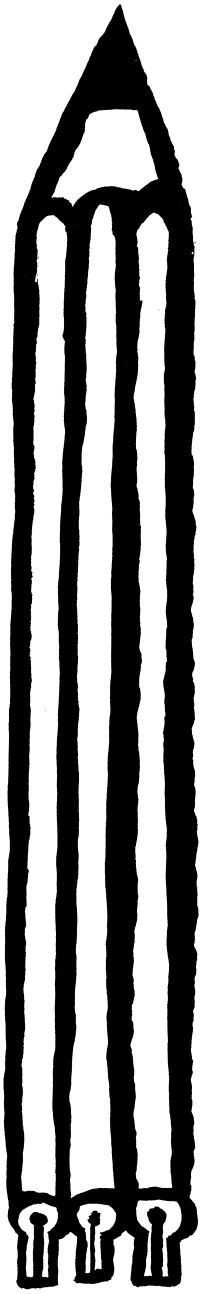
Dear EATCS members,

I hope my letter finds you and your family safe and in good health.

First of all, let me wish you a very happy 2022. I hope that this will be a healthy and fantastic year for all of us, full of great research advances, exciting conferences and workshops. I look forward to working together with all of you in order to continue promoting the development of theoretical computer science.

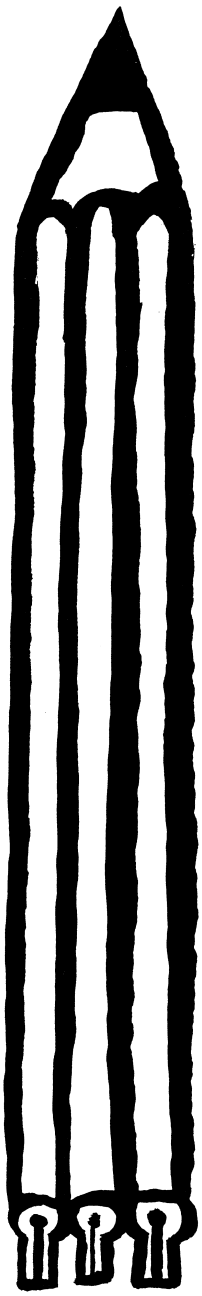
We do not know yet what to expect from this year, but I am optimistically looking forward to 2022. While the coronavirus pandemic will still have a major impact on our lives and our scientific activities, I hope that we are slowly getting back to a safe and healthy environment. Therefore I am looking forward to see not only more fantastic research, but also I hope to see more scientific activities in the community, and even if some of them still focusing on online participation, but many involving meetings and conferences.

I am especially looking forward to attending the 49th EATCS International Colloquium on Automata, Languages, and Programming (ICALP 2022), the EATCS flagship conference that will be held in Paris, France, July 4-8, 2022 (https://icalp2022.irif.fr/). After two years of running ICALP fully online, in 2022 the event will be organized in a hybrid fashion: while we expect most of participants to attend the conference in person, online participation will be made possible. The PC chairs are David Woodruff



(track A) and Mikołaj Bojańczyk (track B), and the conference chair is Thomas Colcombet. ICALP 2022 will feature six fantastic invited speakers: Albert Atserias (Universitat Politècnica de Catalunya), Constantinos Daskalakis (MIT), Leslie Ann Goldberg (Oxford), Madhu Sudan (Harvard, USA), Stéphan Thomassé (l'Ecole Normale Supérieure de Lyon), and Santosh Vempala (Georgia Tech). I hope that many of you have submitted your best work to ICALP 2022 and I expect to see a great scientific program, to be selected by the PC in mid April. As usual, ICALP will be preceded by a series of workshops, which will take place on July 4.

ICALP 2022 will be the occasion to celebrate the 50th anniversary of both the first ICALP and of EATCS - with arguably, year 1972 marking the birth of European theoretical computer science. The first ICALP conference was organized by Maurice Nivat in July 1972 in Rocquencourt, Paris. It is fascinating to see now the 45 papers published at ICALP 1972 (not all in English, with a few papers in German and over a dozen in French!), with so different focus than what we see nowadays: 23 papers on automata and formal languages, 12 in theory of programming, and 11 in computational complexity. As for EATCS, the efforts to establish a Europe-centered scientific organization in Theoretical Computer Science led to the submission to EEC and to Belgian authorities of legal documents to create EATCS on June 24, 1972; this is the official date of the constitution of EATCS (or as it was then officially called Association Européenne d'Informatique Théorique (AEIT), *or in*



English, European Association for Theoretical Computer Science (EATCS)*). A royal decree by the King of Belgium from September 4, 1972 officially created EATCS and approved its statute. The named founders of EATCS were: Giorgio Ausiello (Italy), Jaco de Bakker (The Netherlands), Maurice Nivat (France), Mike Paterson (UK), Manfred Paul (Federal Republic of Germany), Michel Sintzoff (Belgium), and Leo Verbeek (The Netherlands). I hope to see many of you joining us in these celebrations of the 50th anniversaries of ICALP and of EATCS during the ICALP 2022 conference in Paris!*

Also, please allow me to remind you about three EATCS affiliated conferences that will take place in summer and fall this year: the 47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022 in Vienna, https://www.ac.tuwien.ac.at/mfcs2022/ - also celebrating its 50th anniversary!), the 30th Annual European Symposium on Algorithms (ESA 2022), and the 35th International Symposium on Distributed Computing (DISC 2022).

But there will be some more exciting theory conferences taking place in the summer, where I hope to see strong in-person attendance. The 54th ACM Symposium on Theory of Computing (STOC 2022, http://acm-stoc.org/stoc2022/) will be held this year in Europe, in Rome, Italy, June 20-24, 2022. Another flagship theory conference, the 37th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS, https://lics.siglog.org/lics22/) will be held this year at the Technion in Haifa, Israel, August 2-5, 2022, as part of the Federated Logic Conference (FLOC 2022).

I expect all these conferences to bring a large in-person attendance and to stimulate fantastic research advances in theory.

As usual, let me close this letter by reminding you that you are always most welcome to send me your comments, criticisms and suggestions for improving the impact of the EATCS on the Theoretical Computer Science community at president@eatcs.org. We will consider all your suggestions and criticisms carefully.

I look forward to seeing many of you around, at ICALP in Paris, or during other conferences or workshops that I hope to attend this spring and summer and fall, or maybe only online, and to discussing ways of improving the impact of the EATCS within the theoretical computer science community.

Artur Czumaj
University of Warwick, UK
President of EATCS
president@eatcs.org

February 2022

Salomaa Prize in Automata Theory, Formal Languages and related topics

The University of Turku and the Developments in Language Theory Symposium established the Salomaa prize in order to honor outstanding achievments in the area of formal language theory. The previous prize winners are:

2018 Jean-Éric Pin (Paris Diderot University, Paris, France)

2019 Artur Jeż (University of Wrocław, Wrocław, Poland)

2020 Joël Ouaknine (Max Planck Institute, Saarbrücken, Germany) and James Worrell (University of Oxford, Oxford, UK)

All member of the formal language community are kindly invited to send nominations by email to the chair of the prize committee 2022:

Juraj Hromkovič, ETH Zurich, juraj.hromkovic@inf.ethz.ch

The guidelines for the nomination letter are found at:

https://math.utu.fi/salomaaprize/guidelines

The prize consists of a diploma and 2000 Euros and will be awarded at DLT 2022.

Institutional Sponsors

CTI, Computer Technology Institute & Press "Diophantus"
Patras, Greece

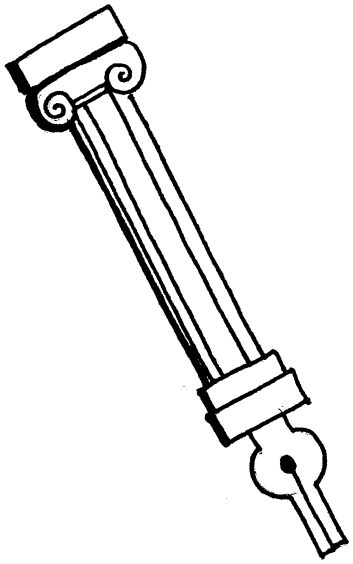
CWI, Centum Wiskunde & Informatica
Amsterdam, The Netherlands

MADALGO, Center for Massive Data Algorithmics
Aarhus, Denmark

Microsoft Research Cambridge
Cambridge, United Kingdom

Springer-Verlag
Heidelberg, Germany

EATCS
Columns



The Interview Column

BY

Chen Avin and Stefan Schmid

Ben Gurion University, Israel and TU Berlin, Germany
{chenavin,schmiste}@gmail.com

KNOW THE PERSON BEHIND THE PAPERS

Today: Keren Censor-Hillel

Bio: *Keren Censor-Hillel is an Associate Professor in the Department of Computer Science at the Technion. She completed her PhD thesis, for which she received the Principles of Distributed Computing Doctoral Dissertation Award, in 2010, and joined the Technion in 2013 after being a Simons Postdoctoral Fellow at MIT. Censor-Hillel received an Alon Fellowship, awarded by the Israeli Academy of Science, in 2013. She is a recipient of a 2016 Krill Prize given by The Wolf Foundation and a 2018 Henry Taub Prize for Academic Excellence. Her research is supported by grants from the ISF, NSF-BSF, and an ERC Starting Grant by the European Commission.*

We ask all interviewees to share a photo with us. Can you please tell us a little bit more about the photo you shared?

Keren: In the photo you can see me hiking. A keen-sighted observer may see some desert surrounding me, which is my favorite hiking area in Israel. I love outdoor activities and sports, and try to find time for them on a daily basis. I also

Credit: Photo by photographer Yasmin Lahav.

share another photo, taken in my office, which is another place I hang out in quite a lot (well, at least in pre-COVID times, when this photo was taken).

Can you please tell us something about you that probably most of the readers of your papers don't know?

Keren: Since I have been working mostly from home lately (as many of us have), if you are reading a paper that I wrote starting March 2020, it is likely that there was a cat on my lap when I was typing into the paper's tex file.

Is there a paper which influenced you particularly, and which you recommend other community members to read?

Keren: The PODC 2006 paper "Computing separable functions via gossip" by Damon Mosk-Aoyama and Devavrat Shah was what got me interested in distributed graph algorithms. I read it in order to present it in a seminar-style class that I took as a graduate student. The paper shows how to compute separable functions in a distributed setting in a very neat way. The topic and paper intrigued me and got me into this research domain.

Is there a paper of your own you like to recommend the readers to study? What is the story behind this paper?

Keren: One of my highest-impact papers is "Algebraic methods in the congested clique", co-authored with Petteri Kaski, Janne H. Korhonen, Christoph Lenzen, Ami Paz, and Jukka Suomela. This paper gives distributed algorithms for matrix multiplication and shows their applications to distance computations and subgraph finding problems. There has been great progress in these fundamental research areas, and at the same time there are still some intriguing open questions, which I welcome everyone to join us in attacking.

When (or where) is your most productive working time (or place)?

Keren: I can work anywhere and even in noisy places, but to be most productive I do need relatively long chunks of uninterrupted time, as my context switching skills could use some improvement. Those who know me know that I am very strict in separating my working and non-working hours.

What do you do when you get stuck with a research problem? How do you deal with failures?

Keren: Failures are the opposite of getting stuck: Failures are inherent for any (research) progress. In many cases, a failure is a huge lead for a direction that doesn't work, and could come along with some insight that has a potential of being useful later. Getting stuck is a different story and can be much more discouraging. It usually makes me talk about the problem with anyone who is willing to listen, and I get back to it once in a while to see if a fresh look into it could bring some new insights.

Is there a nice anecdote from your career you like to share with our readers?

Keren: I started out as a Math undergrad and for some reason I thought that I would not enjoy computer science. Toward the end of my studies I took some TCS courses and those completely changed my view of computer science and I enrolled as a CS graduate student. I am very happy that I have this mathematical education, and I find it to be an important part of my scientific background.

Do you have any advice for young researchers? In what should they invest time, what should they avoid?

Keren: I'd suggest to complement one's research with additional scientific and academic activities. Although such activities require time, they are usually fun, and they almost always lead to new opportunities.

What are the most important features you look for when searching for graduate students?

Keren: I look for students who demonstrate creative thinking and good scientific writing skills, and with whom I enjoy the conversations. I typically take on

students only after some initial research period, so that we can coordinate expectations.

Do you see a main challenge or opportunity for theoretical computer scientists for the near future?

Keren: Opportunities are abundant, as theoretical computer science is about the foundations. The challenge is to stay relevant in order to maintain this importance of the field, by revisiting our lines of research once in a while.

How was your research affected by the pandemic? How do you think it will affect us as a community?

Keren: The pandemic brought many hardships and among those it has been having a substantial negative impact on my research. Still, I consider myself lucky, and I think that for some, perhaps especially the more academically younger researchers, the effect could be worse. I hope that the steps that we have been taking as a community in order to keep these researchers in the loop are helpful.

Please complete the following sentences?

- *My favorite movie is...* The Princess Bride. What can I say, I grew up in the 80's...
- *Being a researcher...* provides me with skills that are useful in other aspects of life. Patience, for example.
- *My first research discovery...* was in the area of coding theory. It is a very different area from distributed computing which I study now, but the thrill of new findings is ever the same.

The Theory Blogs Column

by

Luca Trevisan

Bocconi University
Via Sarfatti 25, 20136 Milano, Italy
L.Trevisan@UniBocconi.it
https://lucatrevisan.github.io

A big difference between academic blogs and other forms of academic communication is the immediacy: the ability to post about results that have just been announced, about events that just took place, or about thoughts that just happened, and to start an interactive discussions. In a complementary way, this column can be an opportunity to look back.

For the first few editions of this column, I will ask writers of some of the most popular theoretical computer science blogs to write guest columns in which they look back at how they started and at some of their favorite posts and interactions.

To get started, Lance Fortnow, the OG of theoretical computer science blogging, will tell us what inspired him to get started, how he set the tone for his blog, what's up with the green background, and he will give us an updated version of one of his favorite posts.

Lance's blog, which is now co-written with Bill Gasarch, is at https://blog.computationalcomplexity.org/

BEATCS no 136

COMPUTATIONAL COMPLEXITY

Lance Fortnow
Illinois Institute of Technology

Thanks to Luca Trevisan for starting this column about theoretical computer science blogs and for asking me to write one of the first columns. Luca said he was interested in knowing what inspired me to start my blog, how I thought about topics (that is, the balance between technical and non-technical content and at what level to address the technical content), why I chose the green background and so on. He also suggested I update a favorite post.

This column thus expands and updates two June 2009 blog posts, The Story of the Blog [3] and A Kolmogorov Complexity Proof of the Lovász Local Lemma [2], the latter about the most mind-blowing STOC talk I ever attended.

1 The Story of the Blog

In 2002 I read a Newsweek article [7] on the then new phenomenon on blogging and I decided to give it a try. I wanted to blog on a specific topic and so I started blogging on the topic I know best. On August 22, 2002 I started "My Computational Complexity Web Log", the first major blog devoted to theoretical computer science. I used the Blogger platform before it was bought out by Google. Back then it had few choices for themes and I chose one with a green background, a color that has become our trademark. I can tell across a large conference room who is looking at the blog.

The blog gained in readership after a sardonic post [4] entitled "Finding the order of the multiplicative group (mod n)... zuh?" in Jason Kottke's popular blog.

> Weblogs are usually pretty easy for readers to get into. Lance Fortnow's Computational Complexity Web Log is probably the most difficult-to-read weblog I've ever come across. But that's OK because if you're into computational complexity, it's just the thing.

All of a sudden I had thousands of readers interested in computational complexity. In that first year I wrote mostly technical posts. I did a set of posts entitled Foundations of Complexity giving an introduction to the field and a series Complexity Class of the Week where I would review results and questions about a specific

complexity class. Later on I wrote a monthly series of my Favorite Theorems in the field.

The technical posts take considerable effort to write and proofread, and seemed less interesting to the majority of my readers who didn't come from the theoretical computer science community. I started writing more opinion and academic-oriented posts. The blog became a meeting place for the theoretical computer science community where we would have some discussions, sometimes quite heated, over the issues of concern to our community. Most of the young people I would meet at conferences knew me more for the blog than for my research. For a while a Google search on my name led to the blog before it led to me.

Changes happen. I shortened the name of the blog. In March of 2007 after a post on turtles [1], I felt I was just going through the motions and decided to retire from blogging. Bill Gasarch took over the blog and kept it going. But I had too much I wanted to say and rejoined the blog in January 2008. Since then Bill and I have co-written this blog and have together experimented with podcasting, vidcasting and typecasting, where we simply transcribe a discussion between us. We often hosted guest bloggers, some of whom, like Scott Aaronson, would go on to write popular blogs on their own. There was a time this blog had new posts every working day but as I took on more administrative roles, we typically do about one to two posts a week. I have supplemented the blog with a Twitter account for short comments and announcements.

Now we have many excellent blogs in theoretical computer science ranging from very technical to very amusing. We strive to be the blog of record, the weblog people turn to to learn about the issues and happenings in the community. We try to cover the major results in complexity, remember those we've tragically lost and the centenaries of the founders of our field.

Weblogs have become the places that brings our ever growing academic community together in a way our conferences no longer can. As this blog approaches 20 years, 3000 posts, 25000 comments and 10 million page views, I'm glad Computational Complexity is able to play its part in that effort.

2 A Kolmogorov Complexity Proof of the Lovász Local Lemma

When I was a graduate student at MIT in the late 1980s, Joel Spencer traveled up from New York to teach a course on the Probabilistic Method, one of the most useful courses I had for future research. The probabilistic method is a general approach to show that certain combinatorial objects, such as Ramsey Graphs, exists by showing that they occur with some positive probability over some distribution

on graphs. In most cases you can find such graphs by just trying random examples.

In that class I first learned about the Lovász Local Lemma, that if you had a set of binary random variables with high enough probability and enough independence, then there was a positive probability they all were true. However the proof did not lead to a randomized constructive algorithm.

In 90's I attended a talk on an algorithmic proof of the Lovász Local Lemma. I had difficulty following the talk and couldn't even understand how the parameters related to the lemma. I tried asking the speaker afterwards but was dismissed with a "You don't understand."

As a best paper and best student paper winner at STOC 2009, I felt I should attend Robin Moser's talk on his paper "A Constructive Proof of the Lovász Local Lemma" [5] but I didn't have high hopes. How wrong I was. Not only did Moser present an amazing result but also an incredibly inventive simple proof that he came up with while preparing the talk.

Smartly Moser focused on an application of the Lovász Local Lemma to satisfiability instead of the full lemma during the talk. He had a beautiful short can't-believe-that-works construction with a simple information-theoretic argument, which I converted in my head to Kolmogorov complexity, because I think better computationally. As I watched enthralled, I said to Eric Allender sitting next to me "Are we really seeing a Kolmogorov proof the the Lovász Local Lemma?" Indeed we were.

After the talk I quickly wrote up the Kolmogorov proof which I posted early the next morning.

Theorem 1. *Suppose we have a k-CNF formula ϕ with n variables and m clauses and each clause shares a variable with at most r other clauses. Then there is a constant d such that if $r < 2^{k-d-1}$ then ϕ is satisfiable. Moreover we can find that assignment in time polynomial in m and n.*

The full algorithm consists of two short routines Solve and Fix below.

Algorithm 1 Solve(ϕ)

```
Pick a random assignment of φ
while There is an unsatisfiable clause C do
  Fix(C)
end while
```

Assume Fix(C) always terminates. Every clause that was satisfied before we called Fix(C) will still remain satisfied and C will also now be satisfied. So Solve makes at most m calls to Fix.

Algorithm 2 Fix(C)

Replace the variables of C with new random values
while While there is an unsatisfied clause D that shares a variable with C **do**
Fix(D)
end while

We need to show all the Fix(C) terminate. Suppose the algorithm makes s Fix calls including all the recursive ones. We will show s is bounded and thus the algorithm terminates.

Fix a Kolmogorov random string x of length $n + sk$ (random relative to ϕ, k, s, r, m and n) and assume the algorithm uses the first n bits as the initial assignment and k bits each to replace the variables in each Fix call.

If we know which clause is being fixed, we know the clause is violated so we know all the bits of this clause and thus we learn k bits of x. We then replace those bits with another part of x.

So we can describe x by the list of clauses we fix plus the remaining n bits of the final assignment. We can describe the C such that Fix(C) is called by Solve by $m \log m$ bits and the remaining fixed clauses by $\log r + O(1)$ bits because either it is one of r clauses that intersects the previous clause or we indicate the end of a recursive call (keeping track of the recursion stack).

Since x was random we have

$$m \log m + s(\log r + O(1)) + n \geq n + sk$$

and thus

$$s(k - \log r - d) \leq m \log m$$

for some constant d.

By assumption we have $r \leq 2^{k-d-1}$ or equivalently $k - \log r - d \geq 1$ and thus s must be bounded by $m \log m$.

We choose the x randomly which with high probability will be Kolmogorovly random and the algorithm will run in polynomial time. QED

In follow-up work with Gábor Tardos [6], Moser builds on these techniques to give a constructive version of the full Lovász Local Lemma with the original parameters which yields $r < 2^k/e$ for satisfiability. The Moser-Tardos paper received the Gödel Prize in 2020.

References

[1] Lance Fortnow. Turles. *Computational Complexity Weblog*, March 2007. https://blog.computationalcomplexity.org/2007/03/turtles.html.

[2] Lance Fortnow. A Kolmogorov complexity proof of the Lovász Local Lemma. *Computational Complexity Weblog*, June 2009. https://blog.computationalcomplexity.org/2009/06/kolmogorov-complexity-proof-of-lov.html.

[3] Lance Fortnow. The story of the blog. *Computational Complexity Weblog*, June 2009. https://blog.computationalcomplexity.org/2009/06/story-of-blog.html.

[4] Jason Kottke. Finding the order of the multiplicative group (mod n)...zuh? *kottke.org*, September 2002. https://kottke.org/02/09/order-multiplicative-group.

[5] Robin A. Moser. A constructive proof of the Lovász Local Lemma. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, page 343–350, New York, NY, USA, 2009. Association for Computing Machinery.

[6] Robin A. Moser and Gábor Tardos. A constructive proof of the general Lovász local lemma. *J. ACM*, 57(2), February 2010.

[7] Newsweek Staff. A blog about writing about blogs. *Newsweek*, August 2002. https://www.newsweek.com/blog-about-writing-about-blogs-144269.

THE DISTRIBUTED COMPUTING COLUMN

Seth Gilbert
National University of Singapore
seth.gilbert@comp.nus.edu.sg

In this issue of the distributed computing column, Michel Raynal and Gadi Taubenfeld revisit a classical question: what can we accomplish in shared memory systems that are anonymous and symmetric? The article is a nice introductory presentation of symmetry and anonymity. It differentiates symmetry and anonymity in the context of processors, and looks at both process and memory anonymity. The article uses two fundamental problems to illustrate: mutual exclusion and consensus. In each case, it illustrates what can be accomplished, giving a good illustration of both what is feasible and the limitations in symmetric and anonymous systems. This new distributed computing column is a great way to start the new year, looking back at a classic problem!

BEATCS no 136

Symmetry and Anonymity in Shared Memory Concurrent Systems

Michel Raynal[⋆], Gadi Taubenfeld[◇]

[⋆]Univ Rennes IRISA, Inria, CNRS, 35042 Rennes, France
[◇]The Interdisciplinary Center, Herzliya, Israel

1 Introduction

More than forty years ago, Dana Angluin asked the following question at one of the first symposia on the theory of computing (STOC 1980) [2]:

> "How much does each processor in a network of processors need to know about its own identity, the identities of other processors, and the underlying connection network in order for the network to be able to carry out useful functions?"

If addressing computability issues was mainly perceived as a theoretical question in 1980, due to the fantastic boom in the development of concurrent and distributed systems (whether the communication is through shared memory or message-passing), today this question is becoming more and more important. In such a context, this technical article visits three important anonymity-related notions. The first two concern restrictions on the identities of the processes (namely, symmetry and anonymity). The third one is relatively new: it concerns the notion of an anonymous shared memory [32].

2 Process Symmetry

The notion of *symmetry* is pervasive in many domains, from philosophy and arts (mainly architecture and painting) to scientific areas such as physics, chemistry, and mathematics [12]. In informatics, a form of process symmetry (related to fairness) was introduced by Edsger W. Dijkstra in his famous one-page article that presented the mutual exclusion problem and a solution to it [10]. More precisely, when he defined the problem, Dijkstra wrote, "The solution must be symmetrical

between the n computers; as a result we are not allowed to introduce a static priority." More than twenty years later (1989), the notion of *process symmetry* was explicitly defined and investigated by Eugene Styer and Gary L. Peterson in an article presented at the ACM conference on principles of distributed computing (PODC) [30].

Definition Let us consider a system of processes, each with its own unique identity. In such a context, an algorithm is *symmetric* if the only way to distinguish processes is by comparing their identifiers (names). This means that the set of the process identifiers defines a specific data type such that the identifiers can be written, read, and compared, but there is no way to "look inside" an identifier, which means that other operations cannot manipulate process identifiers. Thus, identifiers cannot be used to index the entries of a shared array.

Two types of symmetric algorithms can be defined according to how much information can be derived from comparing two identifiers [30, 31]. The family of *symmetric algorithms with arbitrary comparisons* allows the three operations =, >, and < to be applied on process identifiers. Thus, symmetry with arbitrary comparisons allows to totally order the processes according to their identifiers. The family of *symmetric algorithms with only equality* is more restrictive. It allows only the operation = to be applied to process identifiers (in this case, there is no notion of order on process identifiers).

Process symmetry is important for several reasons. One is related to the fact that, while the size of the possible process name-space can be huge (e.g., 2^{32}), the number of processes n is usually relatively much smaller (e.g., $n = 100$). Such a very large name-space does not allow process identifiers to be used as an index to access shared registers. There are two ways to address this issue. One consists in using a process renaming algorithm which allows the size of the name-space to be reduced to n in failure-free systems (this requires an algorithm that needs $2\lceil \log n \rceil + 1$ atomic read/write registers [30]), and to $2n - 1$ in asynchronous systems where processes may crash (see [4, 9]). Another option consists in designing symmetric algorithms. It is important to observe that symmetry with equality means "egalitarian" because a process cannot use its identifier to obtain specific rights. In this sense, process symmetry with equality is the "last" step before process anonymity.

The RW communication model This model is the most basic communication model, namely the only way for processes to communicate is by reading and writing atomic shared registers [16, 19, 27, 31] (the shared registers are the cells of a distributed Turing machine which allows processes to cooperate). A register that

can be written and read by any process is a multi-writer multi-reader (MWMR) register. If a register can be written by a single (predefined) process and read by all, it is a single-writer multi-reader (SWMR) register.

As already indicated, due to the definition of process symmetry, process identifiers cannot be used as pointers to index shared registers. Consequently, process symmetry requires that the shared read/write registers be MWMR registers.

Illustration: Symmetric deadlock-free mutex in RW systems To illustrate process symmetry, let us consider Algorithm 1. This algorithm, due to Styer and Peterson [30], is a process symmetry with equality mutual exclusion algorithm for n processes, where communication is through MWMR atomic registers (RW communication model).

```
operation acquire() is   % invoked by process p
(1)   repeat
(2)     wait(TURN = ⊥); TURN ← p;
(3)     repeat
(4)       for each k ∈ {1, ..., n − 1} do
(5)           if (LOCK[k] = ⊥) then LOCK[k] ← p end if end for;
(6)       locked_p ← ∧_{1≤k≤n−1}(LOCK[k] = p)
(7)     until TURN ≠ p ∨ locked_p end repeat;
(8)     if (TURN = p)
(9)       then return()
(10)      else for each k ∈ {1, ..., n − 1} do
(11)           if (LOCK[k] = p) then LOCK[k] ← ⊥ end if end for
(12)    end if
(13)  end repeat.

operation release() is   % invoked by process p
(14)  TURN ← ⊥;
(15)  for each k ∈ {1, ..., n − 1} do
(16)    if (LOCK[k] = p) then LOCK[k] ← ⊥ end if end for;
(17)  return().
```

Algorithm 1: Symmetric with equality mutex in RW systems [30]

The processes have distinct identities p, q, etc., which can be compared only for equality; $\perp$ is a default process identity (different from the processes' identities). The processes communicate through atomic read/write registers. *TURN* is an atomic MWMR register initialized to $\perp$. Then, it may contain the identity of a

process competing for the critical section. $LOCK[1..n-1]$ is an array of atomic MWMR registers, initialized to $\perp$. The entries of $LOCK$ can be seen as locks that a process needs to capture (by writing its identity in each of them) to enter the critical section. Finally, each process p has a local variable denoted $locked_p$, whose initial value is irrelevant.

To enter the critical section, process p invokes acquire(), which consists of a repeat loop (lines 1-13) that p will exit when it executes the return() statement (line 9). Process p first waits until $TURN = \perp$; when this occurs, it writes its identity in $TURN$ (line 2). Then, it enters an internal repeat loop (lines 3-7). Inside this loop, process p strives to deposit its identity in as many free locks as possible (lines 4-5). When this is done, process p computes if –from its asynchronous and local point of view– it has captured all the locks (assignment to $locked_p$ at line 6). If $TURN \neq p$ or $locked_p =$ `true`, process p exits the internal loop. If $TURN = p$ (in this case $TURN$ has not been modified since p wrote its identity in it), p enters the critical section (lines 8-9). In the other case, before re-entering the main repeat loop, p resets to their initial values all the locks it has previously acquired (lines 10-11).

To exit the critical section, process p invokes the operation release(), which resets to their initial values the shared register $TURN$ and all the locks containing its identity (lines 14-16).

It is easy to see that this algorithm uses exactly n atomic read/write registers and is memoryless (a new invocation of acquire() by a process does not use information on its previous invocations). Moreover, the size of each shared register is bounded by $\log(n+1)$. Proofs of this algorithm can be found in [30, 31].

Remark It is interesting to notice that, while mutual exclusion cannot be solved in the RW communication model when the processes have no identifiers (i.e., are anonymous), it can be solved with process symmetry, which, as already mentioned, is the last step before process anonymity.

3 Process Anonymity

Definition For privacy reasons, some applications must hide the identities of the processes involved. On another side, some applications (e.g., sensor networks) are made up of tiny computing entities with no identifier. This defines the process anonymous computing model, which is characterized by the fact that there is no way for a computing entity (process) to be distinguished from another computing entity. In such a model, not only do the processes have no identities, but they have the same code and the same initialization of their local variables (otherwise, some processes could be distinguished from the others). As for process symmetry, the

Symmetric with equality mutual exclusion on top of shared read/write registers was addressed and solved for the first time in 1989 [30]. This article proves the following lower bounds results and presents associated optimal algorithms.

- n shared read/write registers are necessary and sufficient for deadlock-free symmetric mutual exclusion for n processes.
- $(2n - 1)$ shared read/write registers are necessary and sufficient for memoryless starvation-free symmetric mutual exclusion. "Starvation-free" that that any process that tries to enter the critical section eventually enters it. "Memoryless" means that a process that tries to enter the critical section does not use any information about its previous attempts to enter the critical section.

A symmetric with equality leader election algorithm, in which all the processes are required to participate, was also presented in [30]. This algorithm requires three shared read/write registers, which was conjectured to be necessary. It has recently been shown that a single shared read/write register is sufficient [14].

Sidebar 1: Symmetric mutual exclusion and election in RW systems

notion of SWMR is meaningless for process anonymity: any process may apply any operation to any register.

Process-anonymous failure-free shared memory systems have been studied in [5] where (assuming each process knows the number of processes n) a characterization is presented of problems solvable despite process-anonymity. Relations between the broadcast communication abstraction and reliable process-anonymous shared memory systems have been studied in [3]. Anonymous failure-prone shared-memory systems have been studied in [15], where an answer is presented to the question "What can be deterministically implemented in the process-anonymous crash-prone model?" (deterministically means here that randomized algorithms cannot be used).

Illustration: Obstruction-free binary consensus in asynchronous RW systems To illustrate process-anonymity, let us consider the binary consensus problem in an asynchronous read/write system in which any number of processes may crash. Consensus is one of the most important problems of fault-tolerant distributed computing. Similar to mutual exclusion which is at the core of centralized systems, consensus is at the core of many crash-prone distributed computing problems [24].

Process-anonymous systems have been studied since 1980 in the context of message-passing systems in [2], where several impossibility results are established (e.g., the impossibility to deterministically elect a leader). Characterizations of problems that can be solved in reliable asynchronous message-passing systems despite process anonymity, can be found in [6, 34]. Failure detectors suited to crash-prone asynchronous process-anonymous systems have been introduced and investigated in [7, 8].

Sidebar 2: Process anonymity in message-passing systems

In this problem, each process proposes a value (operation propose()), and must decide on a value. Binary means that only the values 0 and 1 can be proposed. The operation propose() returns the value decided by the invoking process. The following properties define consensus:

- Validity: If a process decides on a value, this value was proposed by a process.
- Agreement: no two processes decide on different values.
- Termination (Wait-freedom): The invocation of propose() by a process that does not crash terminates.

One of the most important results of distributed computing is the impossibility to design a deterministic consensus algorithm satisfying the wait-freedom liveness property [16, 20] in the presence of asynchrony and process crashes (be the communication medium message-passing [11], or RW registers [20]). This impossibility result, established in the context of non-anonymous processes, extends trivially to process-anonymous systems. One way to circumvent this impossibility result is to weaken the termination property as follows (this property, called *obstruction-freedom*, was introduced in [17]).

- Termination (Obstruction-freedom): If process p invokes propose() and all other processes that have pending propose() operations pause during a long enough period, then p terminates its operation.

The notion of "long enough" captures the fact that process p is the only process that continues its execution until it returns from its invocation of propose().

Algorithm 2 described below is due to Rachid Guerraoui and Eric Ruppert [15]. It is a process-anonymous binary consensus in which the processes communicate through MWMR registers, namely a two-dimensional array $SM[0..1, 1..]$ whose second dimension is unbounded. Each register $SM[x, y]$ is initialized to the default value down, and it can then take the value up. $SM[x, y]$ can be seen as a flag

raised forever by a process when some condition is satisfied. Process p locally manages a current estimate of the decision value $est_p \in \{0, 1\}$, the opposite value denoted $opposite_p$, and an iteration number k_p.

```
operation propose(v) is    % invoked by process p
(1) est_p ← v; k_p ← 0;
(2) repeat
(3)   k_p ← k_p + 1; opposite_p ← 1 − est_p;
(4)   if (SM[opposite_p, k_p] = down)
(5)     then SM[est_p, k_p] ← up;
(6)       if (k_p > 1) ∧ (SM[opposite_p, k_p − 1] = down) then return(est_p) end if
(7)     else est_p ← opposite_p
(8)   end if
(9) end repeat.
```

Algorithm 2: Obstruction-free binary consensus in RW systems [15]

This algorithm can be seen as running a competition between two teams of processes, the team of the processes that champion 0, and the team of the processes that champion 1. Process p first progresses to its next iteration (line 3). Iteration numbers k can be seen as defining a sequence of rounds executed asynchronously by the processes. Hence, the state of the flags $SM[0, k]$ and $SM[1, k]$ (which are up or down) describes the state of the competition at round k. When process p enters round k, there are two cases.

- If the flag associated with this round s (k_p) and the other value ($opposite_p$) is up (i.e., the predicate of line 4 is not satisfied), p changes its mind passing from the group of processes that champion est_p to the group of processes that champion $opposite_p$ (line 7). It then proceeds to the next round.
- If the flag associated with this round and the other value is down (the predicate of line 4 is then satisfied), maybe est_p can be decided. To this end, p indicates first that est_p is competing to be the decided value by raising the round k_p flag $SM[est_p, k_p]$ (line 5). The decision involves the two last rounds, namely $(k_p - 1)$ and k_p, attained by p (hence, the sub-predicate $k_p > 1$ at line 6). If p sees both the flags measuring the progress of $opposite_p$ equal to down at round $(k_p - 1)$ and round k_p (predicate $SM[opposite_p, k_p]$ at line 4, and predicate $SM[opposite_p, k_p - 1]$ at line 6), $opposite_p$ is defeated, and p consequently decides est_p.

  To show this is correct, let us consider the smallest round k during which a process decides. Moreover, let p_i be a process that decides during this round, v the value it decides, and τ the time at which p reads $SM[1 - v, k_p -$

1] before deciding (line 6 of round k). As p decides, at time τ we have $SM[1-v, k_p - 1] = \texttt{down}$. This means that, before time τ, no process changed its mind from v to $1-v$ at line 6. The rest of the proof consists in showing that no process p_j started round k before time τ with $est_j = 1 - v$. A proof of this algorithm ensures the consensus is given in [15].

4 Memory Anonymity

Definition Control (processes) and data (memory) are the two pillars of computing. So, while anonymity can be applied to processes, what is the meaning of "memory anonymity"? It means that different processes can have different names for the same register. Let the shared memory be made up of $m \geq 1$ atomic registers $AM[1..m]$. While in a non-anonymous memory $AM[x]$ denotes the same register for all the processes, in an anonymous memory $AM[x]$ can denote some register for process p and a different register for another process q. So, there is an addressing disagreement on the names used by the processes to access the registers. More precisely, an anonymous memory $AM[1..m]$ is such that:

- For each process p an adversary defined a permutation $f_p()$ over the set $\{1, 2, \cdots, m\}$, such that when p uses the address $AM[x]$, it actually accesses $AM[f_p(x)]$,
- No process knows the permutations, and
- All the registers are initialized to the same default value denoted $\perp$.

The notion of anonymous shared memory has been recently introduced in [32]. The work in [32] was inspired by Michael O. Rabin's paper on solving the choice coordination problem [23].

An example of anonymous memory is presented in Table 1. To make apparent the fact that $AM[x]$ can have a different meaning for different processes, we write $AM_p[x]$ when process p invokes $AM[x]$.

identifiers for an external observer	identifiers for process p	identifiers for process q
$AM[1]$	$AM_p[2]$	$AM_q[3]$
$AM[2]$	$AM_p[3]$	$AM_q[1]$
$AM[3]$	$AM_p[1]$	$AM_q[2]$
permutation	$f_p() : [2, 3, 1]$	$f_q() : [3, 1, 2]$

Table 1: An illustration of an anonymous memory model

Motivating anonymous shared memory Anonymous shared memory have two main motivations. The first is related to the basics of computing, namely, computability and complexity lower/upper bounds. Increasing our knowledge of what can (or cannot) be done in the context of both anonymous processes and anonymous memories, and providing associated necessary and sufficient conditions, helps us determine the weakest system assumptions under which the fundamental election problem can be solved.

The second motivation is application-oriented. In [25, 26], it is shown how the process of genome-wide epigenetic modifications, which allows cells to utilize the DNA, can be modeled as an *anonymous* shared memory system where, in addition to the shared memory, also the processes (that is, proteins modifiers) are anonymous. Epigenetic refers in part to post-translational modifications of the histone proteins on which the DNA is wrapped. Such modifications play an important role in the regulation of gene expression.

The authors model histone modifiers (which are a specific type of proteins) as two different types of writer processors and two different types of eraser processors that communicate by accessing an *anonymous* shared memory array which corresponds to a stretch of DNA, and for such a setting formally define the epigenetic consensus problem.

Thus, anonymous shared memories are useful in biologically inspired distributed systems [21, 22], and mastering fundamental distributed computing problems in such an adversarial context could reveal to be important from an application point of view. The similarities and differences between distributed computations in biological and computational (shared-memory and message-passing) systems are explored in [21, 22].

The RMW communication model In addition to the basic RW communication model used previously, we also consider a second communication model denoted RMW (Read-Modify-Write). This model is the RW communication model enriched with the operation Compare&Swap(), which is an atomic conditional write. More precisely, when process p invokes Compare&Swap($AM_p[x]$, *old*, *new*), where *old* and *new* are two values, it atomically assigns the value *new* to $AM_p[x]$ and returns `true` if $AM_p[x] = old$. Otherwise, $AM_p[x]$ is not modified, and the value `false` is returned.

Necessary and sufficient conditions for mutual exclusion and election in symmetric processes and anonymous memory systems Considering a failure-free system where processes are not anonymous, but the memory is anonymous, the following necessary and sufficient conditions relate the number n of processes and the size m of the anonymous memory to solve two classical problems that are

mutual exclusion and election. These conditions capture the minimal information about the pair $\langle n, m\rangle$ needed to break the symmetry that allows these problems to be solved despite memory anonymity. Let $M(n)$ be the set of the positive integers which are relatively prime with the integers 2, ..., n, i.e., $M(n) = \{m : \forall\, \ell \in \{2, ..., n\} : \mathsf{gcd}(\ell, m) = 1\}$, and let $M'(n) = M(n) \setminus \{1\}$.

- Mutual exclusion can be solved by a symmetric algorithm in a system made up of n (non-anonymous) processes communicating through an anonymous memory of size m accessed by RMW (resp. RW) operations if and only if $m \in M(n)$ (resp. $m \in M'(n)$), $m \in M(n)$ (resp. $m \in M'(n)$). The upper bound was established in [1] and the lower bound in [32].
- Leader election can be solved by a symmetric algorithm, in which all the processes are required to participate, in the systems that consist of of n (non-anonymous) processes communicating through an anonymous memory of size m accessed by RMW or RW operations if and only if $\mathsf{gcd}(m, n) = 1$ [14].

Let us observe that while the RMW operations allow mutual exclusion to be solved in more cases than the RW operations alone, this is no longer true for leader election. Let us also observe that while the conditions $m \in M(n)$ and $m \in M'(n)$ are at the heart of their proofs, they do not appear explicitly in the algorithms.

Illustration: Symmetric processes anonymous memory deadlock-free mutex in RMW systems Algorithm 3 (from [1]) is a symmetric with equality anonymous memory deadlock-free mutual exclusion algorithm. It is based on RMW communication operations and assumes $m \in M'(n)$. The registers of the memory are initialized to $\perp$, and p denotes the identifier of the process that invokes the acquire() or release() operation. Let a register be *free* if it contains $\perp$. We say that a register $AM_p[x]$ is *owned* by process p if $AM_p[x] = p$.

When it invokes acquire(), process p enters a repeat loop in which it first tries to own as many registers as possible by writing its identity in as many free registers as possible (line 2). Then p a reads all the registers (line 3) and computes how many registers –from its local and asynchronous point of view– contains the identity that appears the most frequently ($most\_present_p$, line 4) and how many registers it owns ($owned_p$, line 5). If $owned_p < most\_present_p$, process p momentarily withdraws from the competition, resetting to $\perp$ the registers it owns (line 7), until it sees all registers equal to $\perp$ (lines 8-10). If $owned_p \geq most\_present_p$, continues competing until it owns a majority of registers (line 12). When this occurs, p enters the critical section. When it invokes release() process p resets all the registers it owns to their initial value (line 13. A proof of this algorithm can be found in [1].

operation acquire() **is**
(1) **repeat**
(2) **for each** $x \in \{1, ..., m\}$ **do** Compare&Swap($AM_p[x], \perp, p$) **end for**;
(3) **for each** $x \in \{1, ..., m\}$ **do** $view_p[x] \leftarrow AM_p[x]$ **end for**;
(4) $most\_present_p \leftarrow$
maximum number of times the same non-$\perp$ value appears in $view_p$;
(5) $owned_p \leftarrow (|\{x \in \{1, \ldots, m\} : view_p[x] = p\}|)$;
(6) **if** $owned_p < most\_present_p$ **then**
(7) **for each** $x \in \{1, ..., m\}$ **do**
if $(view_p[x] = p)$ **then** $AM_p[x] \leftarrow \perp$ **end if end for**;
(8) **repeat**
(9) **for each** $x \in \{1, ..., m\}$ **do** $view_p[x] \leftarrow AM_p[x]$ **end for**
(10) **until** $\forall\ x \in \{1, \ldots, m\} : view_p[x] = \perp$ **end repeat**
(11) **end if**
(12) **until** $owned_p > m/2$ **end repeat**.

operation release() **is**
(13) **for each** $x \in \{1, ..., m\}$ **do** Compare&Swap($AM_p[x], p, \perp$) **end for**.

Algorithm 3: Symmetric mem.-anony. deadlock-free mutex in RMW systems [1]

It is interesting to note that it is not known whether a symmetric memory-anonymous *starvation-free* mutual exclusion exists. This constitutes a challenging research problem.

Illustration: Symmetric processes memory-anonymous consensus As far as consensus is concerned, a process symmetry with equality obstruction-free consensus algorithm for $n \geq 1$ and $m \geq 2n - 1$ anonymous RW registers is presented in [32].

Illustration: Symmetric processes memory-anonymous election A symmetric algorithm electing a leader in a system where the processes communicate through RW registers is described in [14]. As shown in [13], such an algorithm can be used to de-anonymize an anonymous memory. This election algorithm requires that all the processes participate in the algorithm. It assumes that $\mathsf{gcd}(m, n) = 1$, which is shown to be necessary and sufficient for the election of a single leader. If up to d leaders can be elected the condition becomes $\mathsf{gcd}(m, n) \leq d$ (the number of elected leaders is then ℓ such that $1 \leq \ell \leq d$).

5 Full Anonymity

The ultimate question is now: Are there problems that can be solved when both the processes and the memory are anonymous?

An illustration: Consensus in the RW model As shown in [28], it appears that it is possible to solve obstruction-free consensus in a fully anonymous crash-prone system made up of $n = 2$ processes and $m \geq 3$ anonymous registers, as shown by Algorithm 4.

The anonymous memory is made up of $m \geq 3$ MWMR atomic registers $AM[1..m]$. Each anonymous process p manages a local array $view_p[1..m]$ which will contain a local copy of the anonymous memory, a local variable k_k that is an index to address the entries of $view_p[1..m]$, and a local estimate of the decision value est_p.

```
operation propose(v) is    % invoked by anonymous process p
(1)  est_p ← v;
(2)  repeat
(3)     for each k_p ∈ {1, ..., m} do view_p[k_p] ← AM_p[k_p] end for;
(4)     if (∃ w appearing in a majority of entries of view_p[1..m])
                                                  then est_p ← w end if;
(5)     k_p ← arbitrary index j such that view_p[k_p] ≠ est_p if any, otherwise 0;
(6)     if (k_p ≠ 0) then AM_p[k_p] ← est_p end if;
(7)  until view_p[1] = view_p[2] = ··· = view_p[m] = est_p) end repeat;
(8)  return(est_p).
```

Algorithm 4: Fully anony. obst.-free consensus for 2-process RW systems [28]

When process p invokes propose(v), it first deposits v in est_p (line 1), and enters a repeat loop in which it first scans the anonymous registers (line 3). If it sees a majority value w, it adopts w as its new estimate of the decision value (line 4), and writes it in an anonymous register that storing a different value from w (line 5-6). The process p repeats the previous statements until, after scanning the anonymous memory, all its registers contain the same estimate value, which is then decided. A proof of this algorithm is given in [28].

While obstruction-free consensus can be solved for two processes despite full anonymity, crashes and asynchrony, neither an algorithm nor a necessary and sufficient condition are known for the case $n > 2$. This constitutes a challenging research problem.

It is interesting to note that, while it is possible to solve binary consensus for two processes in a fully anonymous crash-prone system using only 3-valued

registers, this is not possible to do so using only 2-valued registers (i.e., bits). It was recently proved in [33] that there is no obstruction-free consensus algorithm for two non-anonymous processes using only anonymous bits. Thus, as shown in [33], it follows that anonymous bits are strictly weaker than anonymous (and hence also non-anonymous) multi-valued registers.

Illustration: Consensus in the RMW model As consensus with the wait-freedom liveness property cannot be solved in a non-anonymous RW system [11, 20], it cannot be solved either in an anonymous asynchronous crash-prone RW system. This impossibility no longer holds in a crash-prone fully anonymous system where the processes communicate with RMW operations, as shown by Algorithm 5 [28].

operation propose(v) **is** % invoked by process p
(1) **for each** $k \in \{1, 2, ..., m\}$ **do** Compare&Swap($AM_p[k], \perp, v$) **end for**;
(2) $max \leftarrow$ max($AM_p[1], ...AM_p[m]$);
(3) return(max).

Algorithm 5: Fully anony. obst.-free consensus in RMW systems [28]

This very simple algorithm assumes that the proposed values can be totally ordered and works for any value of n and m. All the registers of the anonymous memory $AM[1..m]$ are initialized to $\perp$. The algorithm is based on a "first write, then read" access pattern. Each process p strives to write the value v it proposes in any order in all the registers. Then it returns the greatest value it reads from the anonymous memory (max is a local variable).

Assuming at least one process does not crash, there is a finite time after which (whatever the concurrency/failures pattern), each anonymous register contains a non-$\perp$ value. Moreover, a greater value cannot erase a smaller value already written in a register. This guarantees that a single value can be decided.

Mutex and election in a fully anonymous system Recent results concern mutex and election in fully anonymous system. On the negative side, none of these problems can solved in systems where the anonymous processes communicate through RW registers. Differently they can be solved when communication is through RMW registers The reader will consult [29] for mutex and [18] for election. (In the election problem where processes are anonymous it is required that when a processes terminates the algorithm it knows if it or not a leader.).

6 Conclusion

The purpose of this article was to be a simple introductory presentation of the notions of process symmetry, process anonymity, and memory anonymity in asynchronous systems where communication is through shared memory. To this end, two fundamental (and practically relevant) problems encountered in concurrent and distributed systems have been considered [24]: mutual exclusion in failure-free systems and consensus in crash-prone systems. The following tables summarize what can be done in this context. They also clearly express the additional computability power provided by the RMW communication model with respect to the RW communication model.

Mutual exclusion	communication	nec. & suf. condition	reference
process symmetry (with eq.)	RW	$n > 1, m \geq n$	[30]
memory anonymity	RW	$n > 1, m \in M'(n)$	[1] UB, [32] LB
memory anonymity	RMW	$n > 1, m \in M(n)$	[1]
full anonymity	RW	impossible	[29]
full anonymity	RMW	$n > 1, m \in M(n)$	[29]

Table 2: Deadlock-free mutual exclusion (asynchronous failure-free systems)

Table 2 concerns mutual exclusion (LB and UB stand for lower bound and upper bound, respectively). Table 3 concerns consensus. Let us remember that n is the number of processes, m is the size of the memory, $M(n) = \{m : \forall \ell \in \{2, ..., n\} : \mathsf{gcd}(\ell, m) = 1\}$, and $M'(n) = M(n) \setminus \{1\}$. Also, in the tables, when we write "memory anonymity", we mean that the memory is anonymous and the processes are symmetric.

Consensus	comm.	progress property	sufficient condition	reference
process anonymity	RW	obstruction-freedom	$n > 1, m = \infty$	[15]
memory anonymity	RW	obstruction-freedom	$n > 1, m \geq 2n - 1$	[32]
full anonymity	RW	obstruction-freedom	$n = 2, m \geq 3$	[28]
full anonymity	RMW	wait-freedom	$n > 1, m \geq 1$	[28]

Table 3: Consensus (asynchronous crash-prone systems)

References

[1] Aghazadeh Z., Imbs D., Raynal M., Taubenfeld G., and Woelfel Ph., Optimal memory-anonymous symmetric deadlock-free mutual exclusion. *Proc. 38th*

ACM Symposium on Principles of Distributed Computing (PODC'19), ACM press, pp. 157-166 (2019)

[2] Angluin D., Local and global properties in networks of processes. *Proc. 12th Symposium on Theory of Computing (STOC'80)*, ACM Press, pp. 82-93, (1980)

[3] Aspnes J., Fich F.E., and Ruppert E., Relationship between broadcast and shared memory in reliable anonymous distributed systems. *Distributed Computing*, 18(3):209-219 (2006)

[4] Attiya H., Bar-Noy A., Dolev D., Peleg D., and Reischuk R., Renaming in an asynchronous environment. *Journal of the ACM*, 37(3):524-548, 1990.

[5] Attiya H., Gorbach A., and Moran S., Computing in totally anonymous asynchronous shared-memory systems. *Information and Computation*, 173(2):162-183 (2002)

[6] Attiya H., Snir M. and Warmuth M.K., Computing on an anonymous ring. *Journal of the ACM*, 35(4):845-875 (1988)

[7] Bonnet F. and Raynal M., The price of anonymity: optimal consensus despite asynchrony, crash and anonymity. *ACM Transactions on Autonomous and Adaptive Systems*, 6(4), 28 pages (2011)

[8] Bonnet F. and Raynal M., Anonymous asynchronous systems: the case of failure detectors. *Distributed Computing*, 26(3):141-158 (2013)

[9] Castañeda A., Rajsbaum S., and Raynal M., The renaming problem in shared memory systems: an introduction. *Computer Science Review*, 5:229-251 (2011)

[10] Dijkstra E.W., Solution of a problem in concurrent programming control. *Communications of the ACM*, 8(9):569 (1965)

[11] Fischer M.J., Lynch N.A., and Paterson M.S., Impossibility of distributed consensus with one faulty process. *Journal of the ACM*, 32(2):374-382 (1985)

[12] Gardner M., *The ambidextrous universe: mirror asymmetry and time-reversed worlds.* 293 pages, Penguin Books (1982)

[13] Godard E., Imbs D., Raynal M., Taubenfeld G., Leader-based de-anonymization of an anonymous read/write memory. *Theoretical Computer Science*, 836(10):110-123 (2020)

[14] Godard E., Imbs D., Raynal M., Taubenfeld G., From Bezout identity to space-optimal leader election in anonymous memory systems. *Proc. 39th ACM Symposium on Principles of Distributed Computing (PODC'20)*, ACM press, pp. 41-50 (2020)

[15] Guerraoui R. and Ruppert E., Anonymous and fault-tolerant shared-memory computations. *Distributed Computing*, 20:165-177 (2007)

[16] Herlihy M., Wait-free synchronization. *ACM Transactions on Programming Languages and Systems*, 13 (1):124-149 (1991)

[17] Herlihy M.P., Luchangco V., and Moir M., Obstruction-free synchronization: double-ended queues as an example. *Proc. 23th Int'l IEEE Conference on Distributed Computing Systems (ICDCS'03)*, IEEE Press, pp. 522-529 (2003)

[18] Imbs D., Raynal M., and Taubenfeld G., Election in fully anonymous shared memory systems: tight space bounds and algorithms. *Submitted to publication*, LIPIcs, 14 pages (2021)

[19] Lamport L., On interprocess communication, Part I: basic formalism. *Distributed Computing*, 1(2):77-85 (1986)

[20] Loui M.C., and Abu-Amara H.H., Memory requirements for agreement among unreliable asynchronous processes. *Parallel and Distributed Computing: Vol. 4 of Advances in Computing Research,* JAI Press, 4:163-183 (1987)

[21] Navlakha S. and Bar-Joseph Z., Algorithms in nature: the convergence of systems biology and computational thinking. *Molecular systems biology*, 546(7):1–11, 2011.

[22] Navlakha S. and Bar-Joseph Z., Distributed information processing in biological and computational systems. *Communications of the ACM*, 58(1):94-102 (2015)

[23] Rabin M. O. The choice coordination problem. *Acta Informatica*, 17:121–134 (1982)

[24] Rajsbaum S. and Raynal M., Mastering concurrent computing through sequential thinking: A half-century evolution. *Communications of the ACM*, Vol. 63(1):78-87 (2020)

[25] Rashid S., Taubenfeld G., and Bar-Joseph Z., Genome wide epigenetic modifications as a shared memory consensus. *6th Workshop on Biological Distributed Algorithms (BDA'18)*, London (2018)

[26] Rashid S., Taubenfeld G. and Bar-Joseph Z., The epigenetic consensus problem. *28th Int'l Colloquium on Structural Information and Communication Complexity (SIROCCO'21)*, Springer LNCS 12810, pp. 146–163 (2021)

[27] Raynal M., *Concurrent programming: algorithms, principles and foundations*. Springer, 515 pages, ISBN 978-3-642-32026-2 (2013)

[28] Raynal M. and Taubenfeld G., Fully anonymous consensus and set agreement algorithms. *Proc. 8th Int'l Conference on Networked Systems (NETYS'20)*, Springer LNCS 12129, pp. 314-328 (2020)

[29] Raynal M. and Taubenfeld G., Mutual exclusion in fully anonymous shared memory systems. *Information Processing Letters*, Vol. 158, 105938, 7 pages (2020)

[30] Styer E., and Peterson G. L. Tight bounds for shared memory symmetric mutual exclusion problems. In *Proc. 8th ACM Symposium on Principles of Distributed Computing*, ACM Press, pp. 177-191 (1989)

[31] Taubenfeld G., *Synchronization algorithms and concurrent programming*. Pearson Education/Prentice Hall, 423 pages, ISBN 0-131-97259-6 (2006)

[32] Taubenfeld G., Coordination without prior agreement. *Proc. 36th ACM Symposium on Principles of Distributed Computing (PODC'17)*, ACM Press, pp. 325-334 (2017)

[33] Taubenfeld G., Set agreement power is not a precise characterization for oblivious deterministic anonymous objects *Proc. 26th International Colloquium on Structural Information and Communication Complexity (SIROCCO 19)*, LNCS 11639, pp. 293-308 (2019)

[34] Yamashita M. and Kameda T., Computing on anonymous networks: Part I - characterizing the solvable cases. *IEEE Transactions on Parallel Distributed Systems*, 7(1):69-89 (1996)

THE LOGIC IN COMPUTER SCIENCE COLUMN

BY

YURI GUREVICH

Computer Science & Engineering
University of Michigan, Ann Arbor, Michigan, USA
gurevich@umich.edu

BEATCS no 136

Wigner's quasidistribution and Dirac's kets

Andreas Blass
Mathematics
University of Michigan

Yuri Gurevich
Computer Science & Engineering
University of Michigan

Alexander Volberg
Mathematics, Michigan State University
Hausdorff Center for Mathematics, University of Bonn

Abstract

In every state of a quantum particle, Wigner's quasidistribution is the unique quasidistribution on the phase space with the correct marginal distributions for position, momentum, and all their linear combinations.

The only difference between a probabilistic classical world and the equations of the quantum world is that somehow or other it appears as if the probabilities would have to go negative . . . Okay, that's the fundamental problem. I don't know the answer to it, . . . if I try my best to make the equations look as near as possible to what would be imitable by a classical probabilistic computer, I get into trouble.

— Richard Feynman,
Simulating Physics with Computers, 1982 [7, p. 480]

Partially supported by the US Army Research Office under W911NF-20-1-0297 (authors 1 and 2) and the NSF grant DMS 1900286 (author 3).

1 Introduction

The story of negative probabilities starts with the 1932 article [13] by Eugene Wigner. In quantum mechanics, probability distributions of the position and momentum of a particle make physical sense but their joint distribution doesn't. Yet Wigner exhibited such a joint distribution. It had some desired properties. However, some of its values were negative. "But of course," wrote Wigner, "this must not hinder the use of it in calculations."

In 1987, Jacqueline and Pierre Bertrand proposed "a new derivation of Wigner's function based on the property of positivity of its integrals along straight lines in phase space" [1]. In 2014, in this Bulletin [2], we sketched a mathematical proof of a characterization of Wigner's quasidistribution as the unique quasidistribution on the phase space $\mathbb{R}^2$ that yields the correct marginal distributions not only for position and momentum but for all their linear combinations. In 2021, that sketch was developed into a complete proof that the characterization is valid in "nice" states, namely the states given by smooth functions with compact support [3].

In this paper, we prove that the characterization is valid in all states, with no exception. Furthermore, the new proof is simpler, conceptually and technically. In particular, the uniqueness is derived from a purely measure-theoretic observation that we prove in §3. The simplicity of the new proof gave us the idea to present it in this Bulletin. We made an effort to give our readers a comprehensible and maybe even enjoyable introduction to some foundational issues of science.

Quisani[1]**:** What are negative probabilities?

Authors: The axiomatic definition of probabilities readily generalizes to *quasiprobabilities*, or *signed probabilities*, where negative values are allowed [3]. Basically, you drop the requirement that probabilities take values in the real segment $[0, 1]$ and allow arbitrary real values.

Q: But what is the intuition behind negative probabilities? An urn cannot have -3 red balls.

A: We don't know. At this point, we find it more fruitful to think about what quasiprobabilities are good for.

[1] A former student of the second author.

Q: You once compared the generalization of probabilities to quasiprobabilities with the generalization of real numbers to complex ones. Complex numbers became indispensable, e.g., in solving algebraic equations. Are there important theoretical problems that have been solved using negative probabilities?

A: We don't know such problems, but we expect that quasiprobabilities will be used to solve theoretical problems. They are already used in practice.

Q: Yes, you mentioned quantum tomography in our 2014 conversation [2]. Being a software engineer, I realize the paramount value of practical applications. But today I would like you to address basic questions. Some of these basic questions you seemed to dodge during our 2014 conversation. For example, you spoke about marginal distributions not only for the position and momentum but also for all their linear combinations. But what are marginal distributions for linear combinations? You never defined them properly.

A: Addressing basic questions is fine, and we will define those marginal distributions. We will try to explain things the best we can.

Q: Do explain. But please take into account that, in the meantime, I was busy with computer engineering. I didn't have time to study quantum mechanics or measure theory.

A: Understood.

2 Preliminaries

2.1 Measures

We recall some basic definitions of measure theory.

A *measurable space* M is a pair (Ω, Σ) where Ω is a nonempty set and Σ a σ-algebra of subsets of Ω. In other words, Σ is a Boolean algebra closed under countable unions. Members of Σ are *measurable sets* of M.

`Example:` The real line $\mathbb{R}$ with the collection of real Borel sets, which is the least σ-algebra containing every open real interval (a, b). $\lhd$

`Example:` The real plane $\mathbb{R}^2$ with the collection of Borel subsets of $\mathbb{R}^2$, which is the least σ-algebra containing every open rectangle $(a, b) \times (c, d)$. $\lhd$

A *measure* μ on a nonempty set Ω is a function such that

1. the domain of μ is a σ-algebra of subsets of Ω known as *μ-measurable* sets,

2. μ assigns a real number or ∞ to each μ-measurable set, and

3. μ is *countably additive* which means that, for all pairwise disjoint measurable sets s_n, we have
$$\mu\Big(\bigcup_{n=1}^{\infty} s_n\Big) = \sum_{n=1}^{\infty} \mu(s_n).$$
Since the union $\bigcup s_n$ is independent of the order of the sets s_n, so is the sum $\sum \mu(s_n)$. That implies absolute convergence by a well-known theorem of Riemann.

If μ does not take value ∞, then μ is *finite*. If μ has no negative values then it is *nonnegative*. Every measure we consider in this paper is either finite or nonnegative.

`Example:` The Lebesgue measure on Euclidean spaces $\mathbb{R}^k$ and finite-dimensional Hilbert spaces $\mathbb{C}^k$, called *length* in the case of $\mathbb{R}$, called *area* in the cases of $\mathbb{R}^2$ and $\mathbb{C}$, and called *volume* in the case of $\mathbb{R}^3$ and in general. A careful treatment of the Lebesgue measure, with all the necessary proofs, is somewhat involved [12, Chapter 11], but the definition itself is simple, and we give a version of it on the example of the interval $(0,1)$ in $\mathbb{R}$.

An open set O in $(0,1)$ is the disjoint union of its maximal intervals; define the length of O to be the sum of the lengths of its maximal intervals. For any set $s \subseteq (0,1)$, the *outer measure* $\lambda^*(s)$ of s is the infimum of the lengths of the open sets O that cover s. The *inner measure* $\lambda_*(s)$ is defined as $1 - \lambda^*((0,1) - s)$.

If $\lambda^*(s) = \lambda_*(s)$, then s is *Lebesgue measurable* and the outer (and also the inner) measure $\lambda^*(s)$ is called the *Lebesgue measure* $\lambda(s)$.

The Lebesgue measure on $\mathbb{R}$ is defined by applying this construction to intervals $(i, i+1)$ for all integers i and adding the resulting measures if all the pieces are measurable.

Lemma 1. *If μ, ν are finite measures on a measurable space M, then their difference $(\mu - \nu)(s) = \mu(s) - \nu(s)$ is a finite measure on M.*

Proof. If measurable sets $s_1, s_2, \ldots$ are pairwise disjoint, then

$$(\mu - \nu)\bigcup_n s_n = \mu\bigcup_n s_n - \nu\bigcup_n s_n = \sum_n \mu(s_n) - \sum_n \nu(s_n)$$
$$= \sum_n (\mu(s_n) - \nu(s_n)) = \sum_n (\mu - \nu)(s_n).$$

The third equality holds because of absolute convergence. □

A function $f : \Omega_1 \to \Omega_2$ from a measurable space $M_1 = (\Omega_1, \Sigma_1)$ to a measurable space $M_2 = (\Omega_2, \Sigma_2)$ is *measurable for* M_1, M_2 if the f-preimage of every measurable set in M_2 is measurable in M_1. If M_2 is the real line $\mathbb{R}$ or complex line $\mathbb{C}$ endowed with the σ-algebra of Borel sets, then f is a *measurable function on* M_1. ◁

2.2 $L^2(\mathbb{R})$, and test functions

If f is a measurable function on a Euclidean space $\mathbb{R}^k$ and μ is a nonnegative measure on $\mathbb{R}^k$, then

$$\int f d\mu = \int_{\mathbb{R}^k} f d\mu = \int_{\mathbb{R}^k} f(x) d\mu(x)$$

means the Lebesgue integral of f with respect to measure μ. For real-valued f, the integral is defined by approximating f with so-called *simple functions*, i.e. functions $g(x)$ taking only finitely many values v_i, each on a measurable set s_i. The integral $\int g(x) d\mu(x)$ is simply $\sum_i v_i \mu(s_i)$. If the supremum of the integrals of simple functions $g(x) \leqslant f(x)$ for all x coincides with the infimum of the integrals of simple functions $g(x) \geqslant f(x)$ for all x, then their common value is the integral $\int f d\mu$, in which case f is *integrable* with respect to measure μ. See details in Chapter 11 of [12].

For $\mathbb{C}$-valued functions f, just integrate the real and imaginary parts separately. It is easy to check that every bounded continuous function is integrable with respect to any finite measure μ.

If s is a measurable subset of $\mathbb{R}^k$, we let $\chi_s(x)$ be 1 for $x \in s$ and 0 otherwise. Then $\int_s f(x)\, d\mu(x)$ means $\int_{\mathbb{R}^k} \chi_s(x) f(x)\, d\mu(x)$.

`Proviso.` By default, Euclidean spaces $\mathbb{R}^k$ come with the Lebesgue measure. ◁

$L^2(\mathbb{R})$ is the Hilbert space of square integrable functions $\psi : \mathbb{R} \to \mathbb{C}$ with the inner product $\langle \psi \,|\, \varphi \rangle$ given by the (Lebesgue) integral $\int_{\mathbb{R}} \psi^*(x)\varphi(x)\,dx$.

Two $L^2(\mathbb{R})$ functions are considered equivalent if they differ only on a set of measure zero. Strictly speaking, $L^2(\mathbb{R})$ vectors are the equivalence classes. It is more convenient though to work with individual functions modulo the equivalence relation.

The forward Fourier transform $\mathcal{F}$ sends an $L^2(\mathbb{R})$ function $\psi(x)$ to

$$\hat{\psi}(\xi) = \frac{1}{\sqrt{2\pi}} \int \psi(x)\, e^{-i\xi x}\, dx$$

provided that the integral exists. Similarly, the inverse Fourier transform $\mathcal{F}^{-1}$ sends a function $\varphi(\xi)$ to

$$\check{\varphi}(x) = \frac{1}{\sqrt{2\pi}} \int \varphi(\xi) e^{i\xi x}\, d\xi,$$

Mathematically x and ξ are real variables. In applications, the dimension of ξ is the inverse of that of x so that ξx is a pure number. Here and in the rest of the paper, integrals are by default integrals over $\mathbb{R}$.

The forward and inverse Fourier transforms are defined also for functions of several variables. In particular, provided the integrals exist, we have

$$\hat{f}(\xi,\eta) = \frac{1}{2\pi} \iint f(x,y)\, e^{-i(\xi x+\eta y)}\, dx\, dy,$$
$$\check{g}(x,y) = \frac{1}{2\pi} \iint g(\xi,\eta) e^{i(\xi x+\eta y)}\, d\xi\, d\eta.$$

If μ is a finite measure on $\mathbb{R}$, its Fourier transform $\hat{\mu}$ is an $\mathbb{R} \to \mathbb{C}$ function:

$$\hat{\mu}(\zeta) = \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} e^{-ix\zeta} d\mu(\zeta).$$

Similarly, if ν is a finite measure on $\mathbb{R}^2$, its Fourier transform $\hat{\nu}$ is an $\mathbb{R}^2 \to \mathbb{C}$ function:

$$\hat{\nu}(\xi,\eta) = \frac{1}{2\pi} \int_{\mathbb{R}^2} e^{-i(\xi x+\eta y)}\, d\nu(x,y).$$

An $L^2(\mathbb{R})$ function $\psi(x)$ is a *Schwartz function* if it is infinitely differentiable

and if it and its derivatives rapidly approach zero when $x \to \pm\infty$ in the sense that, for all nonnegative integers j, k, we have

$$\lim_{x\to\pm\infty} \left| x^j \frac{d^k\psi(x)}{dx^k} \right| = 0.$$

Schwartz functions are also known as *test functions*.

The Fourier transform of a test function is a test function. By the Plancherel theorem [9, Theorem A.19], the Fourier transform $\mathcal{F}$ is a unitary operator on the test functions. But these functions are dense in $L^2(\mathbb{R})$. By continuity, $\mathcal{F}$ is (or rather extends to) a unitary operator on the whole $L^2(\mathbb{R})$.

2.3 Distributions, and exponential operators

Dirac introduced a "function" $\delta(x)$ which is identically zero for all $x \neq 0$ while $\delta(0)$ is infinite, so infinite that $\int_{\mathbb{R}} \delta(x)dx = 1$. This makes no sense. δ is not a function in the usual sense. But it does make sense in the context of integrals of the form $\int_{\mathbb{R}} f(x)\delta(x)dx$ which should be $f(0)$, at least for well behaved functions f. Laurent Schwartz suggested viewing Dirac's δ and similar "generalized functions" as linear functionals on the space of test functions. Thus, Dirac's δ-function would be thought of as the linear functional on test functions f:

$$f \mapsto \int f(x)\delta(x)dx = f(0). \tag{1}$$

If $0 \neq c \in \mathbb{R}$, then

$$\delta(x) = \frac{1}{|c|}\,\delta\Big(\frac{x}{c}\Big). \tag{2}$$

Indeed, if we use the substitution $y = x/c$ and keep integrating from $-\infty$ to ∞, then we have

$$\int f(x)\frac{1}{|c|}\,\delta\Big(\frac{x}{c}\Big)dx = \int f(cy)\delta(y)dy = f(0).$$

Schwartz developed these ideas into a theory of *distributions*, i.e. continuous linear real-valued functionals on the space of test functions (with the suitable topology). Since then distributions play a major role in the theory of differential equations.

Some divergent integrals, e.g. $\int e^{itx}dt$, can be seen as distributions in that sense. In fact, as distributions,

$$\int e^{itx}dt = 2\pi\delta(x). \tag{3}$$

Indeed,

$$\begin{aligned}\int dx\, f(x) \int e^{itx}dt &= \sqrt{2\pi}\int dt\, \frac{1}{\sqrt{2\pi}}\int f(x)e^{itx}dx \\ &= \sqrt{2\pi}\int \check{f}(t)\, dt \\ &= 2\pi\cdot\frac{1}{\sqrt{2\pi}}\int \check{f}(t)e^{-it0}dt = 2\pi f(0).\end{aligned}$$

The exponential e^A of an operator A on $L^2(\mathbb{R})$ is the operator

$$e^A = \sum_{k=0}^{\infty}\frac{A^k}{k!} = I + A + \frac{1}{2}A^2 + \frac{1}{6}A^3 + \dots \tag{4}$$

provided that series converges.

If $(X\psi)(x) = x\cdot\psi(x)$, then

$$(e^X\psi)(x) = \sum_{k=0}^{\infty}\frac{1}{k!}(X^k\psi)(x) = \psi\cdot\sum_{k=0}^{\infty}\frac{1}{k!}x^k = \psi\cdot e^x.$$

If D is the derivative operator $\frac{d}{dx}$ and c a real number, then $e^{cD}\psi(x) = \psi(x+c)$. Indeed,

$$\begin{aligned}e^{cD}\psi(x) &= \sum_{k=0}^{\infty}\frac{(cD)^k\psi(x)}{k!} = \sum_{k=0}^{\infty}\frac{D^kf(x)}{k!}c^k \\ &= \psi(x) + \frac{\psi'(x)}{1!}c + \frac{\psi''(x)}{2!}c^2 + \frac{\psi'''(x)}{3!}c^3 + \dots\end{aligned}$$

which is the Taylor series of $\psi(x+c)$ around point x; think of c as Δx.

Q: I worry about convergence of the Taylor series.

A: The Taylor series certainly converges on analytic functions, in particular on Gaussian functions

$$\exp\left(-\frac{(x-a)^2}{2b^2}\right).$$

The linear combinations of Gaussian functions are dense in $L^2(\mathbb{R})$, and there is a unique continuous extension of e^{cD} to L^2, namely the shift $f(x) \mapsto f(x+c)$.

3 Pushforward measures, and uniqueness theorem

We recall the definition of pushforward measures and then prove a measure-theoretic uniqueness theorem used in the proof of our main theorem in §9.

Consider measurable spaces $M_1 = (\Omega_1, \Sigma_1)$ and $M_2 = (\Omega_2, \Sigma_2)$. Let μ be a measure on M_1 and let a function $f : \Sigma_1 \to \Sigma_2$ be measurable for M_1, M_2.

Definition 2 (§3.6 in [5])**.** The *pushforward of μ along f*, a.k.a. the *f-pushforward of μ* or the *f-image* of μ, is the measure

$$f_*\mu(e) = \mu\big(f^{-1}(e)\big)$$

on M_2. ◁

It is easy to check that $\nu = f_*\mu$ is indeed a measure on M_2.

Proposition 3. *With notation as above, for every measurable function $g : \Omega_2 \to \mathbb{C}$ on M_2, if $g(f(x))$ is μ-integrable then g is ν-integrable and*

$$\int_{M_2} g(y)d\nu(y) = \int_{M_1} g(f(x))d\mu(x).$$

For real-valued f, Proposition 3 is a modification of Theorem 3.6.1 in book [5] as described in the comments following the proof of theorem in the book. The generalization to $\mathbb{C}$-valued functions is straightforward.

Recall that, by default, real Euclidean spaces $\mathbb{R}^k$ are equipped with the Lebesgue measure. Accordingly, measurable subsets of $\mathbb{R}^k$ are Lebesgue measurable, and integrals are Lebesgue integrals.

Theorem 4 (Uniqueness)**.** *Let μ_1, μ_2 be finite Borel measures on $\mathbb{R}^2$. If $(ax + by)_*\mu_1 = (ax + by)_*\mu_2$ for all a, b not both zero, then $\mu_1 = \mu_2$.*

Proof. Let a, b range over pairs of reals not both zero. By Lemma 1, $\mu = \mu_1 - \mu_2$ is a measure on $\mathbb{R}^2$. It suffices to prove that μ is the zero measure $\mathbb{R}^2$. Let $\nu_{ab} = (ax + by)_*\mu$.

Every ν_{ab} is the zero measure on $\mathbb{R}$. Indeed, if s is a Borel subset s of $\mathbb{R}$, let $S = \{(x,y) : ax + by \in s\}$. We have

$$(ax+by)_*\mu(s) = \mu(S) = \mu_1(S) - \mu_2(S) \\ = (ax+by)_*\mu_1(s) - (ax+by)_*\mu_2(s) = 0.$$

We have.

$$\widehat{\nu}_{ab}(\zeta) = \int_{\mathbb{R}} e^{i\zeta t}\, d\nu_{ab}(t) = \int_{\mathbb{R}^2} e^{i\zeta(ax+by)}\, d\mu(x,y) \\ = \int_{\mathbb{R}^2} e^{i(a\zeta x + b\zeta y)}\, d\mu(x,y),$$

where the second equality uses Proposition 3 with $g(t) = e^{i\zeta t}$; the integrand $e^{i\zeta(ax+by)}$ is a bounded continuous function and therefore is integrable with respect to the (finite) measure μ.

Comparing this with the Fourier transform of μ,

$$\widehat{\mu}(\xi,\eta) = \int_{\mathbb{R}^2} e^{i(\xi x+\eta y)}\, d\mu(x,y),$$

we get

$$\widehat{\mu}(a\zeta, b\zeta) = \widehat{\nu}_{ab}(\zeta).$$

Since every ν_{ab} is the zero measure, every $\widehat{\nu}_{ab}(\zeta) = 0$ for all ζ. It follows that $\widehat{\mu}$ is the function zero.

By Proposition 3.8.6 in book [5], if two Borel measures on $\mathbb{R}^2$ have equal Fourier transforms, then they coincide. Applying this to μ and the zero measure, we conclude that μ is the zero measure. □

The theorem generalizes to higher dimensions, but we restrict our attention to $\mathbb{R}^2$.

4 Marginal distributions

Traditionally, for a (signed) probability distribution of several variables, the marginal (signed) distributions are defined only for single variables or subsets

of the variables. We extend this definition to linear functions of the variables, restricting attention to just two variables.

Let $\mathcal{P}$ be a quasiprobability distribution on real plane $\mathbb{R}^2$ with coordinate axes x and y, and let (a,b) range over pairs of real numbers not both zero.

Definition 5. The $(ax+by)$ *marginal* of $\mathcal{P}$ is the pushforward measure (and in fact quasidistribution) $(ax+by)_*\mathcal{P}$ of $\mathcal{P}$ along the function $z = ax+bp : \mathbb{R}^2 \to \mathbb{R}$. $\lhd$

In particular, $(x)_*\mathcal{P}(s) = \mathcal{P}\big(s \times \mathbb{R}\big)$, and $(y)_*\mathcal{P}(s) = \mathcal{P}\big(\mathbb{R} \times s\big)$, so that $(x)_*\mathcal{P}$ and $(y)_*\mathcal{P}$ are traditional marginals.

Now suppose that $\mathcal{P}$ is given by a density function $f(x,y)$, so that $\mathcal{P}(s) = \iint_s f(x,y)\,dx\,dy$ for all measurable subsets s of $\mathbb{R}^2$. We show that in this case every marginal $(ax+by)_*\mathcal{P}$ is given by a density function which will be denoted $(ax+by)_*f$.

Lemma 6. *For every pair (a,b), the function*

$$g(z) = \begin{cases} \dfrac{1}{|b|}\displaystyle\int f\left(x, \frac{1}{b}(z-ax)\right) dx & \text{if } b \neq 0 \\ \dfrac{1}{|a|}\displaystyle\int f\left(\frac{z}{a}, y\right) dy & \text{otherwise} \end{cases}$$

is the density function $(ax+by)_*f$.

Proof. We consider the case $b \neq 0$; the other case is similar (and a bit simpler). It suffices to prove that $(ax+by)_*\mathcal{P}[u,v] = \int_u^v g(z)\,dz$ on intervals $[u,v]$ with $u \leqslant v$. We have

$$(ax+by)_*\mathcal{P}[u,v] = \iint\limits_{u \leqslant ax+by \leqslant v} f(x,y)\,dx\,dy.$$

Let $z = ax+by$, so that $y = \frac{1}{b}(z-ax)$. Change variables in the integral, from x,y to x,z. The absolute value of the Jacobian determinant of this transformation is $\frac{1}{|b|}$, so we obtain

$$\begin{aligned}(ax+by)_*\mathcal{P}[u,v] &= \iint\limits_{u \leqslant z \leqslant v} \frac{1}{|b|} f\left(x, \frac{1}{b}(z-ax)\right) dx\,dz \\ &= \int_u^v dz \int_{\mathbb{R}} \frac{1}{|b|} f\left(x, \frac{1}{b}(z-ax)\right) dx = \int_u^v g(z)\,dz. \qquad \square\end{aligned}$$

Lemma 7. *$(ax + by)_*\mathcal{P}(u,v) = (acx + bcy)_*\mathcal{P}(cu,cv)$ for every real $c \neq 0$ and every open interval (u,v) of $\mathbb{R}$.*

Proof.

$$\begin{aligned}(ax + by)_*\mathcal{P}(u,v) &= \mathcal{P}\{(x,y) : u < ax + by < v\} \\ &= \mathcal{P}\{(x,y) : cu < acx + bcy < cv\} \\ &= (acx + cby)_*\mathcal{P}(cu,cv).\end{aligned}$$ □

Lemma 8 (Lemma 6.4 in [3])**.** *For all real a, b not both zero and every function $g : \mathbb{R} \to \mathbb{R}$, the following claims are equivalent.*

1. *g is the density function $(ax + bp)_*f$.*
2. *$\widehat{g}(\zeta) = \sqrt{2\pi} \cdot \widehat{f}(a\zeta, b\zeta)$ where $\widehat{f}$ and $\widehat{g}$ are (forward) Fourier transforms of f and g respectively.*

The computation that proves the lemma was essentially done in our proof of the uniqueness theorem above.

5 Position, momentum, and their linear combinations

Consider one particle moving in one dimension. A generalization to more particles in more dimensions is relatively straightforward.

In classical mechanics, the position x and momentum p of the particle determine its current state. The set of all possible classical states is the phase space of the particle. In quantum mechanics, the state space of the particle is the Hilbert space $L^2(\mathbb{R})$.

Using Dirac's bra-ket notation, we write $|\psi\rangle$ for the vector given by function ψ. Unit vectors $|\psi\rangle$ represent states of the particle, and two unit vectors represent the same state if and only if they differ by a scalar factor $e^{i\theta}$.

Q: How come a whole $\mathbb{R} \to \mathbb{C}$ function is needed to represent just one quantum state?

A: Because, in quantum mechanics, a particle is also a wave. If it is in state $|\psi\rangle$, then $|\psi(x)|^2$ is the probability density at x for finding the particle. $\langle\psi\,|\,\psi\rangle$ is the total probability. Accordingly, $\langle\psi\,|\,\psi\rangle$ must be 1, and this is why unit vectors are used to represent states.

In quantum mechanics, observable quantities are represented by Hermitian operators on the state space. In particular, the position observable X and momentum observable P are (represented by) operators

$$(X\psi)(x) = x \cdot \psi(x),$$
$$(P\psi)(x) = -i\hbar\frac{d\psi}{dx}$$

where $\hbar = h/2\pi$ is the reduced Planck constant; h is the (unreduced) Planck constant. We will simplify notation by assuming (by proper choice of units) that $\hbar = 1$.

Q: Functions $X\psi$ and $P\psi$ may fail to be square integrable.

A: Indeed, the operators X, P are undefined in some states.

Q: The formula for P looks mysterious to me. Is momentum also related to the wave character of our particle?

A: Yes, it is. The momentum p corresponds to the wavelength $\lambda = h/p$ (de Broglie relation). So, if a particle had an exact value p of the momentum, its wave function would be (up to a scalar factor)

$$\psi(x) = e^{2\pi ix/\lambda} = e^{ipx/\hbar} = e^{ipx},$$

which is an eigenfunction of P with eigenvalue p. (Yes, this ψ isn't in our Hilbert space. We'll return to this point in §6.) P is designed to be the operator whose eigenvalues are momenta, just as X is the operator whose eigenvalues (corresponding to "eigenfunctions" $\delta(x-q)$) are positions q.

There is a simple mathematical connection between the two operators: $\mathcal{F}P\mathcal{F}^{-1} = X$ where $\mathcal{F}$ is the Fourier transform. It suffices to verify this equality

on test functions.

$$\begin{aligned}\mathcal{F}P(\psi)(\xi) &= \int P\psi(x)e^{-i\xi x}dx = \int -i\frac{d\psi(x)}{dx}e^{-i\xi x}dx \\ &= i\int \psi(x)\frac{de^{-i\xi x}}{dx}dx = \xi\int \psi(x)e^{-i\xi x}dx = \\ &= \xi\cdot\mathcal{F}(\psi)(\xi) = X\mathcal{F}(\psi)(\xi).\end{aligned}$$

The third equality uses integration by parts; the extra terms disappear because $e^{-i\xi x}$ is bounded and the test function ψ approaches zero when the argument goes to $\pm\infty$.

Q: The equality $\mathcal{F}P\mathcal{F}^{-1} = X$ makes me worry about the dimensions. You mentioned earlier that, when x is a length, as here, then the variable ξ of the Fourier transform is a reciprocal length. But here the variable of the Fourier transform seems to be a momentum. How do you reconcile these dimensions?

A: By convention, we're using units where $\hbar = 1$, and the dimension of $\hbar$ is momentum times length, so our convention makes reciprocal length the same as momentum. Thus, our P is dimensionally correct.

6 Dirac's kets

The spectral theory of self-adjoint operators in finite dimensional Hilbert spaces is relatively simple. Suppose that $\mathcal{H}$ is n-dimensional, and consider a self-adjoint operator A on $\mathcal{H}$. Since $\mathcal{H}$ is self-adjoint, all its eigenvalues are real. There exists an orthonormal basis $|1\rangle, \ldots, |n\rangle$ for $\mathcal{H}$ composed of eigenvectors of A. Let $\lambda_1, \ldots, \lambda_n$ be the corresponding eigenvalues.

For simplicity of exposition, we assume that all eigenvalues λ_k are distinct (and thus non-degenerate since the number of them equals the dimension of $\mathcal{H}$). This suffices for our purposes in this paper, and it allows us to label the eigenvectors with the corresponding eigenvalues. We may write $|\lambda_k\rangle$ instead of $|k\rangle$.

It will be convenient, in the infinite dimensional case, to use an alternative characterization of the "distinct eigenvalues" assumption. The operator A is called *cyclic* if there is a vector $|\psi\rangle$ in $\mathcal{H}$ such that the vectors

$$|\psi\rangle,\ A|\psi\rangle,\ A^2|\psi\rangle,\ \ldots,\ A^{n-1}|\psi\rangle$$

span the whole space $\mathcal{H}$; such vector $|\psi\rangle$ is *A-cyclic*.

Claim 9. *A is cyclic if and only if the eigenvalues $\lambda_1, \ldots, \lambda_n$ are all distinct.*

Proof. If the values are distinct, then $\sum_{k=1}^{n} |k\rangle$ is A-cyclic. If, on the other hand, $\lambda_i = \lambda_j$, then for any vector $|\psi\rangle = \sum c_k|k\rangle$, all linear combinations of vectors $A^l|\psi\rangle$ have coefficients of $|i\rangle$ and $|j\rangle$ in the same ratio $c_i : c_j$. Thus these linear combinations fail to span $\mathcal{H}$. □

Think of $\mathcal{H}$ as the state space of a quantum system where unit vectors represent states of the system (and two unit vectors represent the same state if and only if they are collinear). Then every state $|\psi\rangle$ can be written as

$$|\psi\rangle = \sum_j |\lambda_j\rangle\langle\lambda_j \,|\, \psi\rangle \tag{5}$$

where scalars $\langle\lambda_j \,|\, \psi\rangle$ are probability amplitudes of the wave function $|\psi\rangle$, so that the corresponding probabilities are $|\langle\lambda_j \,|\, \psi\rangle|^2$. Accordingly,

$$A|\psi\rangle = \sum_j A|\lambda_j\rangle\langle\lambda_j \,|\, \psi\rangle = \sum_j \lambda_j|\lambda_j\rangle\langle\lambda_j \,|\, \psi\rangle$$

and the expectation of A in a state $|\psi\rangle$ is

$$\langle\psi|A|\psi\rangle = \sum_j \lambda_j\langle\psi \,|\, \lambda_j\rangle\langle\lambda_j \,|\, \psi\rangle = \sum_j \lambda_j \,|\langle\lambda_j \,|\, \psi\rangle|^2 .$$

The infinite dimensional case is much more involved. Let A be a self-adjoint operator on a infinite-dimensional space $\mathcal{H}$. If you measure A, you still receive some real number but it is not necessarily an eigenvalue. It is just an element of the spectrum

$$\sigma(A) = \{\lambda : \text{operator } A - \lambda I \text{ is not invertible}\}$$

of A. Notice that the set of eigenvalues is

$$\{\lambda : \text{operator } A - \lambda I \text{ is not one-to-one}\} .$$

In finite dimensions, any one-to-one linear operator is invertible, but this principle fails in infinite dimensions.

As in the finite dimensional case, we make the simplifying assumption that the spectrum is simple in the sense that A is *cyclic*. That is, there is a vector $|\psi\rangle$, called an *A-cyclic* vector, such that the vectors

$$|\psi\rangle,\ A|\psi\rangle,\ A^2|\psi\rangle,\ \ldots$$

are dense in whole Hilbert space $\mathcal{H}$.

How to generalize the spectral theory of self-adjoint operators in the finite dimensional case to the infinite dimensional case? Paul Dirac came up with an elegant heuristic generalization [6] which works for operators like $aX + bP$ on $L^2(\mathbb{R})$. We sketch Dirac's generalization, restricting attention to the Hilbert space $L^2(\mathbb{R})$ and to cyclic self-adjoint operators A on $L^2(\mathbb{R})$ with $\sigma(A) = \mathbb{R}$.

For each spectrum value $r \in \mathbb{R}$, there is a generalized eigenvector, in short *eigenket*, $|r\rangle$ for A, so that $A|r\rangle = r|r\rangle$. As in the finite dimensional case, we take advantage of the cyclicity assumption to label eigenkets $|r\rangle$ by the corresponding spectrum value r.

Q: What do you mean by eigenvector being generalized?

A: That it does not necessarily belong to $L^2(\mathbb{R})$.

Q: That is confusing. Give me an example.

A: If A is the momentum operator P, then $|r\rangle$ is the function $x \mapsto e^{irx}/\sqrt{2\pi}$. Indeed,

$$P\left(\frac{e^{irx}}{\sqrt{2\pi}}\right) = -i\,\frac{d}{dx}e^{irx} = r\left(\frac{e^{irx}}{\sqrt{2\pi}}\right).$$

The finite-dimensional orthonormality requirement is replaced by *δ-normality*:

$$\langle s\,|\,r\rangle = \delta(r-s).$$

In the case of P, using (3) we have

$$\int \frac{e^{-isx}}{\sqrt{2\pi}} \cdot \frac{e^{irx}}{\sqrt{2\pi}}\,dx = \frac{1}{2\pi}\int e^{i(r-s)x}dx = \delta(r-s).$$

The finite-dimensional expansion (5) with respect to the eigenvectors becomes

the *Dirac basis expansion*:

$$|\psi\rangle = \int |r\rangle\langle r\,|\,\psi\rangle\, dr$$

where $\langle r\,|\,\psi\rangle$ is the density of probability amplitude, so that the corresponding probability density is

$$D_A(r) = |\langle r\,|\,\psi\rangle|^2. \tag{6}$$

The probability distribution on $\mathbb{R}$, given by the probability density function $D_A(r)$, will be denoted $\mathcal{P}_A$.

We have $A|\psi\rangle = \int A|r\rangle\langle r\,|\,\psi\rangle\, dr = \int r|r\rangle\langle r\,|\,\psi\rangle\, dr$, and the expectation of A in state $|\psi\rangle$ is

$$\langle\psi|A|\psi\rangle = \int r\,|\langle r\,|\,\psi\rangle|^2\; dr. \tag{7}$$

In the case $A = P$, the density of probability amplitude is

$$\langle r\,|\,\psi\rangle = \frac{1}{\sqrt{2\pi}}\langle e^{irx}\,|\,\psi(x)\rangle = \frac{1}{\sqrt{2\pi}}\int e^{-irx}\psi(x)\,dx = \widehat{\psi}(r),$$

the probability density is

$$D_P(r) = \left|\widehat{\psi}(r)\right|^2, \tag{8}$$

and $\mathcal{P}_P$ is the corresponding probability distribution.

Q: Is there mathematical justification of Dirac's heuristic generalization?

A: The theory of rigged Hilbert spaces, see [4] for example, mathematically justifies the use of generalized eigenvectors[2]. The operators A are subject to some constraints which are satisfied by the operators X, P and their linear combinations. See §3 in [10] in this connection.

Q: It this "rigged" as in rigged elections? What witty guy came up with this term?

A: This is a translation of a Russian term[3] meaning *equipped* or *rigged* as in "rigging a ship for sailing."

Q: Will you tell me more about rigged Hilbert spaces?

A: Rigged Hilbert spaces deserve a separate column article, but the basic idea

[2] An alternative justification is provided by the spectral theory of operators [9].

[3] оснащенный; see [8].

is that a Hilbert space $\mathcal{H}$ is augmented with two additional spaces to obtain a so-called *Gelfand triple*.

In our case, the triple is

$$\Phi \subset L^2(\mathbb{R}) \subset \Phi^\times$$

where Φ comprises test functions and $\Phi^\times$ comprises distributions. Recall that we touched upon distributions in§2.3.

Q: What kind of distributions are the eigenkets $|r\rangle = e^{irx}$ of P?

A: An antilinear functional

$$\langle f \mid r\rangle = \frac{1}{\sqrt{2\pi}} \int f^*(x) e^{irx} dx$$

on test functions f [11].

Q: I am confused. Earlier, in §2.3, you said that distributions are linear, not antilinear, functionals.

A: Yes, the elements of $\Phi^\times$ are antilinear functionals serving as generalized kets while distributions are linear functionals serving as generalized bras. In our case, it is safe to ignore the distinction and call both of them distributions.

Q: Explain.

A: To see what goes on, consider the finite dimensional case. View elements of $\mathbb{C}^n$ as column vectors, so the inner product $\langle \varphi \mid \psi \rangle$ is given by the matrix product $\varphi^\dagger \cdot \psi$. Thus ψ acts antilinearly on φ whereas φ acts linearly on ψ, but both are column vectors in the same space $\mathbb{C}^n$. The same entities serve as linear and antilinear functionals.

From this point of view, an eigenket of an operator $A : \mathbb{C}^n \to \mathbb{C}^n$ for the eigenvalue λ is a column vector ψ such that $A \cdot \psi = \lambda\psi$. An eigenbra is a column vector φ such that $\varphi^\dagger \cdot A = \lambda\varphi^\dagger$; equivalently, $A^\dagger \cdot \varphi = \lambda^*\varphi$. When A is self-adjoint and its eigenvalues λ are therefore real, the eigenbras and eigenkets coincide.

The situation is similar in infinite dimensions, and the operators whose eigenkets and eigenbras we use are (essentially) self-adjoint. Thus we can safely use the same entities as eigenkets and eigenbras.

7 Linear combinations of position and momentum operators

In this section, we apply Dirac's machinery to linear combinations $aX + bP$ of the operators X and P where a, b are real numbers not both zero.

One can argue that, from physical considerations, the spectrum of every $aX + bP$ is $\mathbb{R}$. This is supported by theory. Every operator $aX + bP$ is self-adjoint[4]. The spectrum of every self-adjoint operator consists of reals [9, Theorem 9.17]. Hence, for every $aX + bP$, the spectrum $\sigma(aX + bP) \subseteq \mathbb{R}$. Furthermore, for every $aX + bP$, we will provide an eigenket $|r\rangle$ of $aX + bP$ for every real r, so that $\sigma(aX + bP) = \mathbb{R}$.

Lemma 10. *Let $Z = aX + bP$ where a, b are real numbers not both zero, let $0 \neq c \in \mathbb{R}$, and let (u, v) be an interval in $\mathbb{R}$. Then, in every state $|\psi\rangle$,*

$$\mathsf{P}[Z \in (u,v)] = \mathsf{P}[cZ \in \{cr : u < r < v\}] = \begin{cases} \mathsf{P}[cZ \in (cu, cv)] & \text{if } c > 0, \\ \mathsf{P}[cZ \in (cv, cu)] & \text{if } c < 0. \end{cases}$$

Q: This seems obvious. The two events are the same and so have the same probability.

A: Z is an observable, and the probabilities are determined by the rules of quantum mechanics.

Q: But we can view Z also as a random variable by repeatedly measuring it in a given state. It is gratifying that both views give the same result, isn't it?

A: Agreed.

Proof. We consider the case $c > 0$; the case $c < 0$ is similar.

As we saw in the previous section, there is a δ-normal system $\langle f(r,x) : r \in \mathbb{R} \rangle$ where $f(r,x)$ is an eigenket of Z for spectrum value r. Then $\left\langle \frac{1}{\sqrt{c}} f(\frac{r}{c}, x) : r \in \mathbb{R} \right\rangle$ is a δ-normal system for cZ. Indeed,

$$(cZ)\left(\frac{1}{\sqrt{c}} f\big(\frac{r}{c}, x\big)\right) = \sqrt{c}\, Z f\big(\frac{r}{c}, x\big) = \sqrt{c}\, \frac{r}{c} f\big(\frac{r}{c}, x\big) = r \cdot \frac{1}{\sqrt{c}} f\big(\frac{r}{c}, x\big),$$

[4] More exactly, $aX + bP$ is essentially self-adjoint [9, Proposition 9.40], and every essentially self-adjoint operator has a unique self-adjoint extension [9, Proposition 9.11].

and, using the δ-normality of $\langle f(r,x) : r \in \mathbb{R}\rangle$ and using (2), we have

$$\Big\langle \frac{1}{\sqrt{c}} f\big(\frac{s}{c}, x\big) \;\Big|\; \frac{1}{\sqrt{c}} f\big(\frac{r}{c}, x\big) \Big\rangle = \frac{1}{c}\delta\Big(\frac{s}{c} - \frac{r}{c}\Big) = \delta(r-s).$$

By (6), in a state $|\psi\rangle$, the probability density functions for Z and cZ are

$$|\langle f(r,x) \,|\, \psi(x)\rangle|^2 \quad \text{and} \quad |\langle f(r/c,x) \,|\, \psi(x)\rangle|^2$$

Using the substitution $r = cs$, we have

$$\begin{aligned}\mathsf{P}[cZ \in (cu, cv)] &= \int_{cu}^{cv} dr \left| \int_{\mathbb{R}} dx \, \frac{1}{\sqrt{c}} f^*(\frac{r}{c}, x)\psi(x) \right|^2 \\ &= \int_u^v ds \left| \int_{\mathbb{R}} dx \, f^*(s,x)\psi(x) \right|^2 = \mathsf{P}[Z \in (u,v)]. \qquad \square\end{aligned}$$

For every real r, the eigenket $|r\rangle$ of X for spectrum value r is the delta function $\delta(x-r)$, which acts on test functions according to

$$f(x) \mapsto \langle f \,|\, r\rangle = \int f^*(x)\delta(x-r)\,dx = f^*(r).$$

$X|r\rangle$ is the distribution that sends a test function $f(x)$ to

$$\begin{aligned}&\int f^*(x)x\delta(x-r)\,dx = \int f^*(x+r)(x+r)\delta(x)\,dx \\ &= r f^*(r) = r\langle f \,|\, r\rangle.\end{aligned}$$

Thus, $X|r\rangle = r|r\rangle$. In particular, $\sigma(X) = \mathbb{R}$. Furthermore, these eigenkets form a δ-normal system:

$$\langle s \,|\, r\rangle = \int \delta(x-s)\delta(x-r)\,dx = \delta(r-s).$$

The density of probability amplitude is $\langle r \,|\, \psi\rangle = \int \delta(x-r)\psi\,dx = \psi(r)$, the probability density function is

$$D_X(r) = |\psi(r)|^2. \tag{9}$$

and $\mathcal{P}_X$ is the corresponding probability distribution.

Next we consider a case $Z = -cX + P$. It is still a special case, but the general case $Z = aX + bP$ easily reduces to that special case; we return to this issue in the next section.

The mapping defined by $U|\psi\rangle = e^{icx^2/2}|\psi\rangle$ is a unitary operator on the test functions:

$$\langle U\psi \,|\, U\varphi\rangle = \langle\psi|U^\dagger U|\varphi\rangle = \langle\psi \,|\, \varphi\rangle.$$

We have $UPU^{-1} = Z$ on test functions [3, §6]. Indeed,

$$\begin{aligned}\big(UPU^{-1}\big)\,\psi &= e^{icx^2/2}\cdot\left(-i\frac{d}{dx}\big(e^{-icx^2/2}\psi\big)\right) = -cx\psi - i\frac{d\psi}{dx}\\ &= -cX\psi + P\psi = Z\psi.\end{aligned}$$

Accordingly, one may expect that U transforms generalized eigenvectors of P into those of Z. This intuition happens to be correct.

For each real r, the distribution $e^{irx+icx^2/2}$ is a generalized eigenvector of Z for r:

$$\begin{aligned}Ze^{irx+icx^2/2} &= (-cX+P)e^{irx+icx^2/2} = -cxe^{irx+icx^2/2} - i\frac{d}{dx}e^{irx+icx^2/2}\\ &= e^{irx+icx^2/2}\big[-cx - i(ir+icx)\big] = re^{irx+icx^2/2}.\end{aligned}$$

It follows that the spectrum of Z is $\mathbb{R}$.

For each real r, let $|r\rangle$ be the eigenket $\frac{1}{\sqrt{2\pi}}e^{irx+icx^2/2}$ of Z for generalized eigenvalue r. These eigenkets form a δ-normal system:

$$\int \frac{e^{-isx-icx^2/2}}{\sqrt{2\pi}}\cdot\frac{e^{irx+icx^2/2}}{\sqrt{2\pi}}\,dx = \frac{1}{2\pi}\int e^{i(r-s)x}dx = \delta(r-s).$$

Accordingly, the density of probability amplitude is

$$\langle r \,|\, \psi\rangle = \frac{1}{\sqrt{2\pi}}\langle e^{irx+icx^2/2} \,|\, \psi(x)\rangle = \frac{1}{\sqrt{2\pi}}\int e^{-irx+icx^2/2}\psi(x)\,dx,$$

the probability density is

$$D_Z(r) = \frac{1}{2\pi}\left|\langle e^{irx+icx^2/2} \,|\, \psi(x)\rangle\right|^2, \tag{10}$$

and the corresponding probability distribution is $\mathcal{P}_Z$.

8 Wigner's quasiprobability distribution

In every $L^2(\mathbb{R})$ state $|\psi\rangle$, Wigner's quasiprobability $\mathcal{P}_W$ is given by probability density function

$$w(x,p) = \frac{1}{2\pi}\int_{\mathbb{R}} \psi^*(x+\frac{\gamma\hbar}{2})\psi(x-\frac{\gamma\hbar}{2})e^{i\gamma p}\,d\gamma. \tag{11}$$

The integral converges in every $L^2(\mathbb{R})$ state $|\psi\rangle$. According to Wigner, the x-marginal and p-marginal of his distribution are the probability distributions for X and P respectively [13].

Q: You don't take advantage of using units where $\hbar = 1$. I guess it doesn't hurt to keep $\hbar$ in this case.

A: Exactly.

Q: Is this obvious that the x-marginal and p-marginal of Wigner's distribution are the probability distributions for X and P ?

A: It is certainly easier to verify than the claim that an arbitrary $(ax+by)$-marginal of Wigner's distribution is the probability distribution for $aX + bY$.

Lemma 11. *In every state* $|\psi\rangle$, $(x)_*\mathcal{P}_W = \mathcal{P}_X$.

Proof. It suffices to prove that, in every state $|\psi\rangle$, the x-marginal x_*w of Wigner's density is the density function $|\psi|^2$. Since two density function coincide if they are proportional, we may neglect constant factors.

By Lemma 6, up to constant factors, the x_*w density function is

$$\begin{aligned}
&x_*w = \int w(x,p)dp = && \text{by Lemma 6}\\
&\int\left[\int \psi^*(x+\frac{\gamma\hbar}{2})\,\psi(x-\frac{\gamma\hbar}{2})\,e^{i\gamma p}\,d\gamma\right]dp = &&\\
&\int\left[\int e^{i\gamma p}dp\right]\psi^*(x+\frac{\gamma\hbar}{2})\,\psi(x-\frac{\gamma\hbar}{2})\,d\gamma = && \text{by (3)}\\
&\int \delta(\gamma)\psi^*(x+\frac{\gamma\hbar}{2})\,\psi(x-\frac{\gamma\hbar}{2})\,d\gamma = && \text{by (1)}\\
&\psi^*(x)\,\psi(x) = |\psi|^2 && \square
\end{aligned}$$

Lemma 12. $w(x,p) = \int \widehat{\psi}(p+\frac{\gamma}{2})^* \, \widehat{\psi}(p-\frac{\gamma}{2}) \, e^{-i\gamma x} \, d\gamma$
up to a constant factor.

Proof. Ignoring constant factors and using the formula

$$\psi(x) = \int \widehat{\psi}(\xi) \, e^{i\xi x} \, d\xi$$

for the inverse Fourier transform, we get

$$\begin{aligned} w(x,p) &= \int \psi(x+\frac{\gamma}{2})^* \, \psi(x-\frac{\gamma}{2}) \, e^{i\gamma p} \, d\gamma \\ &= \iiint \widehat{\psi}(\xi)^* \, e^{-i\xi(x+\frac{\gamma}{2})} \, \widehat{\psi}(\eta) \, e^{i\eta(x-\frac{\gamma}{2})} \, e^{i\gamma p} \, d\xi \, d\eta \, d\gamma \\ &= \iint \left[e^{-i\gamma(\frac{\xi}{2}+\frac{\eta}{2}-p)} d\gamma \right] \widehat{\psi}(\xi)^* \, \widehat{\psi}(\eta) \, e^{-ix(\xi-\eta)} \, d\xi \, d\eta \\ &= \iint \widehat{\psi}(\xi)^* \, \widehat{\psi}(\eta) \, e^{-ix(\xi-\eta)} \, \delta(\frac{\xi}{2}+\frac{\eta}{2}-p) \, d\xi \, d\eta. \end{aligned}$$

The δ function makes it easy to perform the integration with respect to η. Just substitute $2p-\xi$ for η in the remaining factors.

$$w(x,p) = \int \widehat{\psi}(\xi)^* \, \widehat{\psi}(2p-\xi) \, e^{-ix(2\xi-2p)} \, d\xi.$$

Change variables in the integral to $\gamma = 2\xi - 2p$, so ξ becomes $p + \frac{\gamma}{2}$.

$$w(x,p) = \int \widehat{\psi}(p+\frac{\gamma}{2})^* \, \widehat{\psi}(p-\frac{\gamma}{2}) \, e^{-i\gamma x} \, d\gamma. \qquad \square$$

Corollary 13. *The (p)-marginal of Wigner's distribution is the probability distribution $\mathcal{P}_P$ for P.*

The proof is similar to that of Lemma 11, except the formula of Lemma 12 is used.

The generalization to arbitrary $(ax+bp)$-marginals will be proved in §9. A key role in that proof is played by the following lemma.

Lemma 14 (Lemma 6.6 in [3])**.** *For all test functions ψ and real numbers α, β,*

not both zero,

$$\langle\psi|e^{-i(\alpha X+\beta P)}|\psi\rangle = e^{i\alpha\beta\hbar/2}\int \psi^*(y)e^{-i\alpha y}\psi(y-\beta\hbar)\,dy.$$

Actually, Lemma 6.6 in [3] speaks about states $|\psi\rangle$ that are "nice" in the sense that the function ψ is smooth and compactly supported. But the lemma and its proof obviously remain valid for test functions ψ.

9 Characterization theorem

Theorem 15. *In every state $|\psi\rangle$ in $L^2(\mathbb{R})$, Wigner's quasidistribution $\mathcal{P}_W$ is the unique quasidistribution on $\mathbb{R}^2$ such that, for all real numbers a, b not both zero, the marginal $(ax+bp)_*\mathcal{P}_W$ is correct in the sense that it coincides with the probability distribution $\mathcal{P}_Z$ of the observable $Z = aX + bP$.*

Proof. The uniqueness follows from Theorem 4.

We need to prove the equality $(ax+bp)_*\mathcal{P}_W = \mathcal{P}_Z$ for every pair of reals a, b not both zero and in every state $|\psi\rangle$. By virtue of Lemmas 7 and 10, we restrict attention to two cases:

1. $a = 1$ and $b = 0$, so that $Z = X$,

2. $b = 1$, so that $Z = aX + P$.

Case (1) is taken care of by Lemma 11. In the rest of the proof we consider case (2). Even though $b = 1$, we sometimes write $aX + bP$ and $ax + bp$ anyway.

Fix an arbitrary real number a. We prove that, in every state,

$$(ax+bp)_*\mathcal{P}_W = (ax+p)_*\mathcal{P}_W = \mathcal{P}_Z. \tag{12}$$

In every state $|\psi\rangle$, the quasidistribution $\mathcal{P}_W$ is given by the probability density function $w(x,p)$ specified by formula (11). By Lemma 6, the marginal quasidis-

tribution $(ax+p)_*\mathcal{P}_W$ is given by quasiprobability density function

$$\begin{aligned} g(r) &= \int w(x, r-ax)\,dx \\ &= \frac{1}{2\pi}\iint \psi^*(x+\frac{\gamma\hbar}{2})\,\psi(x-\frac{\gamma\hbar}{2})\,e^{i\gamma(r-ax)}\,d\gamma\,dx. \end{aligned}$$

The probability distribution $\mathcal{P}_Z$ is given by the probability density function D_Z of Z specified in (10). To prove (12), it suffices to prove

$$g(r) = D_Z(r). \tag{13}$$

Both sides of (13) are continuous as functions of the state in the $L^2(\mathbb{R})$ metric. By continuity, it suffices to prove that the equality (13) holds in every "nice" state $|\psi\rangle$ provided that the nice states are dense in $L^2(\mathbb{R})$.

Theorem 6.2 in [3] does just that. In that theorem, a state $|\psi\rangle$ is nice if ψ is smooth and compactly supported. While Theorem 6.2 addresses both, the uniqueness and the correctness aspects, the emphasis in its proof is on uniqueness, and the correctness proof may be a bit confusing. We explain it here. Notice that two density functions coincide if they are proportional. Accordingly, we ignore constant factors in equations below.

Since (ignoring the factor $1/2\pi$)

$$w(x,p) = \int \psi^*(x+\frac{\gamma\hbar}{2})\,\psi(x-\frac{\gamma\hbar}{2})\,e^{i\gamma p}\,d\gamma,$$

its Fourier transform is

$$\widehat{w}(\alpha,\beta) = \iiint dx\,dp\,d\gamma\,\psi^*(x+\frac{\gamma\hbar}{2})\,\psi(x-\frac{\gamma\hbar}{2})\,e^{i\gamma p}\,e^{-i\alpha x}\,e^{-i\beta p}.$$

Here p occurs only in two of the exponential factors, so the integration over p produces

$$\int dp\,e^{i(\gamma-\beta)p} = \delta(\gamma-\beta).$$

The delta function now makes the integral over γ trivial; just substitute β for γ in the integrand. Thus,

$$\widehat{w}(\alpha,\beta) = \int dx\,\psi^*(x+\frac{\beta\hbar}{2})\,\psi(x-\frac{\beta\hbar}{2})\,e^{-i\alpha x}.$$

Introducing a new integration variable $y = x + \frac{\beta\hbar}{2}$, we get

$$\widehat{w}(\alpha,\beta) = \int dy\, \psi^*(y)\, \psi(y - \beta\hbar)\, e^{-i\alpha y}\, e^{i\alpha\beta\hbar/2}.$$

By Lemma 14,

$$\widehat{w}(\alpha,\beta) = \langle\psi|e^{-i(\alpha X+\beta P)}|\psi\rangle.$$

In particular, if Z is defined as $aX + bP$ and if we substitute $a\zeta$ and $b\zeta$ for α and β, we get

$$\widehat{w}(a\zeta, b\zeta) = \langle\psi|e^{-i\zeta Z}|\psi\rangle.$$

By Lemma 8, $\langle\psi|e^{-i\zeta Z}|\psi\rangle$ is the Fourier transform of the marginal $g(z) = (ax + bp)_* w(x,p)$. But this same $\langle\psi|e^{-i\zeta Z}|\psi\rangle$ is also, up to a constant factor, the Fourier transform of the density function D_Z of Z in the state ψ. Indeed, by (4),

$$e^{-i\zeta Z} = \sum_k \frac{1}{k!}(-i\zeta Z)^k,$$

and we restrict attention to test functions ψ, so that convergence is no problem. We have

$$\langle\psi|e^{-i\zeta Z}|\psi\rangle = \left\langle \psi \,\middle|\, \sum_k \frac{1}{k!}(-i\zeta Z)^k \,\middle|\, \psi \right\rangle = \sum_k \frac{(-i\zeta)^k}{k!}\langle\psi|Z^k|\psi\rangle$$

Now we use Dirac's machinery. Let $|r\rangle$ be the eigenket $\frac{1}{\sqrt{2\pi}}e^{irx+icx^2/2}$ of Z for spectrum value r. We have $Z^2|r\rangle = Z(r|r\rangle) = r^2|r\rangle$ and similarly for other powers of Z. By the preceding computation, (7), and (10), we have

$$\begin{aligned}\langle\psi|e^{-i\zeta Z}|\psi\rangle &= \sum_k \frac{(-i\zeta)^k}{k!}\int r^k |\langle r\,|\,\psi\rangle|^2\, dr = \int \sum_k \frac{(-i\zeta)^k}{k!} r^k |\langle r\,|\,\psi\rangle|^2\, dr \\ &= \int e^{-i\zeta r}|\langle r\,|\,\psi\rangle|^2\, dr = \int e^{-i\zeta r} D_Z(r)\, dr,\end{aligned}$$

as required. □

Acknowledgment

We thank Rafael de la Madrid for useful comments.

References

[1] Jacqueline Bertrand and Pierre Bertrand, “A tomographic approach to Wigner’s function," Foundations of Physics 17:4 (1987) 397–405

[2] Andreas Blass and Yuri Gurevich, “Negative probabilities," Bulletin of EATCS, February 2014

[3] Andreas Blass and Yuri Gurevich, “Negative probabilities: What they are for," Journal of Physics A 54 (2021), article 315303, `https://doi.org/10.1088/1751-8121/abef4d`

[4] Arno Böhm, “The rigged Hilbert space and quantum mechanics," Springer 1978

[5] Vladimir I. Bogachev, “Measure theory,” Vol. 1, Springer 2007

[6] Paul Dirac, “The principles of quantum mechanics," 4th edition, Oxford University Press 1958

[7] Richard P. Feynman, “Simulating physics with computers," International Journal of Theoretical Physics, Vol. 21, Nos. 6/7 (1982) 467–488

[8] Israel M. Gelfand and Naum Y. Vilenkin, “Generalized functions," Vol. 4, Academic Press 1964; the Russian original published in 1961 by Физматлит

[9] Brian C. Hall, “Quantum theory for mathematicians," Springer 2013

[10] Rafael de la Madrid, “Quantum mechanics in rigged Hilbert space language," PhD thesis, Universidad de Valladolid, 2001

[11] Rafael de la Madrid, “The role of the rigged Hilbert space in quantum mechanics," European Journal of Physics 26 (2005) 287–312, doi:10.1088/0143-0807/26/2/008

[12] Walter Rudin, “Principles of mathematical analysis," McGraw-Hill, 3rd edition, 1976

[13] Eugene P. Wigner, “On the quantum correction for thermodynamic equilibrium," Physical Review 40 (1932) 749–759

The Computational Complexity Column

by

Michal Koucký

Computer Science Institute, Charles University
Malostranské nám. 25, 118 00 Praha 1, Czech Republic
koucky@iuuk.mff.cuni.cz
https://iuuk.mff.cuni.cz/~koucky/

BEATCS no 136

META-COMPUTATIONAL AVERAGE-CASE COMPLEXITY: A NEW PARADIGM TOWARD EXCLUDING HEURISTICA

Shuichi Hirahara
National Institute of Informatics, Tokyo, Japan
s_hirahara@nii.ac.jp

Abstract

One of the central open questions in computational complexity theory is to connect worst-case hardness of NP to average-case hardness of NP. This question is well known as whether Heuristica is excluded from Impagliazzo's five worlds. It is known that standard proof techniques that relate worst-case and average-case complexities are incapable of excluding Heuristica; thus, in order to make progress, we need to develop new proof techniques.

Recently, new worst-case to average-case connections that cannot be proved by previous proof techniques are established based on *meta-complexity*. Meta-complexity refers to the computational complexity of problems that themselves ask complexity. In this article, we present the emerging paradigm of "meta-computational average-case complexity," i.e., a new approach of analyzing average-case complexity via meta-complexity of time-bounded Kolmogorov complexity.

1 Introduction

Traditionally, the complexity of a computational problem is measured in terms of worst-case complexity; that is, the performance of an algorithm is measured with respect to the worst input. However, it is often criticized that the worst-case complexity measure is too pessimistic: the worst input might not be encountered in practice; thus, worst-case complexity may not be relevant to "real-life" complexity. Moreover, some NP-complete problems, such as the Hamiltonian path problem, can be solved efficiently with high probability when the input is chosen randomly from natural distributions [GS87].

A better approach to measuring "real-life" complexity would be to use *average-case complexity*. It is natural to assume that a real-life instance is generated by some efficiently computable procedure. This motivates us to study the complexity of DistNP, i.e., the average complexity of NP over instances generated by some randomized polynomial-time algorithms.

The theory of average-case complexity has two primary motivations. As mentioned above, one is to understand the practical performance of algorithms. The other is that average-case hardness of NP is the foundation for the security of cryptosystems: the existence of an average-case hard problem in NP is necessary for most cryptographic primitives to be secure [IL89]. The relationship among $\mathsf{P} \neq \mathsf{NP}$, average-case hardness of NP, and cryptography was clearly presented in the influential work of Impagliazzo [Imp95]. We review Impagliazzo's five worlds in Section 2 and explain the importance of studying average-case complexity of NP.

Arguably, one of the most important open questions in the theory of average-case complexity is to exclude Heuristica from Impagliazzo's five worlds, i.e., to prove average-case hardness of NP from worst-case hardness of NP. This question can be, for example, formally stated as follows.

Open Question 1. Does $\mathsf{P} \neq \mathsf{NP}$ imply $\mathsf{DistNP} \not\subseteq \mathsf{AvgP}$?

Here, the statement $\mathsf{DistNP} \not\subseteq \mathsf{AvgP}$ means that there exists a pair of a problem L and a polynomial-time samplable distribution $\mathcal{D}$ with respect to which there is no average-polynomial-time algorithm that solves L. We review the basics of average-case complexity in Section 3, following the excellent survey of Bogdanov and Trevisan [BT06a].

There are at least three reasons why it is difficult to resolve Open Question 1. Standard proof techniques that connect worst-case complexity to average-case complexity, such as black-box reductions [FF93; BT06b], hardness amplification procedures [Vio05b; Vio05a], and relativizing proof techniques [Imp11; HN21], are shown to be incapable of resolving Open Question 1. (We will explain the details of the limits of black-box reductions in Section 5.) To make progress on the central problem, we need to develop new types of proof techniques that are not subject to any of these technical barriers.

Recently, a new type of proof techniques that are not subject to some of the barriers are developed. [Hir18] presented the first worst-case-to-average-case connection that goes beyond the limits of black-box reductions. Building on this result, [Hir21a] showed the following worst-case-to-average-case connections.

Theorem 2 ([Hir21a])**.**

1. *If* $\mathsf{UP} \not\subseteq \mathsf{DTIME}(2^{O(n/\log n)})$*, then* $\mathsf{DistNP} \not\subseteq \mathsf{AvgP}$.
2. *If* $\mathsf{PH} \not\subseteq \mathsf{DTIME}(2^{O(n/\log n)})$*, then* $\mathsf{DistPH} \not\subseteq \mathsf{AvgP}$.
3. *If* $\mathsf{NP} \not\subseteq \mathsf{DTIME}\left(2^{O(n/\log n)}\right)$*, then* $\mathsf{DistNP} \not\subseteq \mathsf{Avg}_{\mathsf{P}}\mathsf{P}$. Here, $\mathsf{Avg}_{\mathsf{P}}\mathsf{P}$ stands for P-*computable average-polynomial-time*, which interpolates P and AvgP; see Section 3 for a definition.

The first item of the worst-case-to-average-case connections of Theorem 2 cannot be proved by neither black-box reductions [FF93; BT06b] nor hardness amplification procedures [Vio05a; Vio05b]; any proof of Theorem 2 must simultaneously overcome the two barriers, i.e., the limits of black-box reductions and the "impossibility" of hardness amplification procedures. This is the reason why it took 35 years to find a proof of Theorem 2 since Levin [Lev86] laid the foundation of average-case complexity.

The main goal of this article is to present a proof of the first and second items of Theorem 2 as well as technical tools developed in the proof. Our presentation follows recent alternative proofs of [Hir21a], which are independently obtained by [Hir21b; GK22]. Along the way, we put together several results that are scattered in several papers.

The reader may wonder where the time complexity $2^{O(n/\log n)}$ comes from. It comes from the fact that a polynomial p is a "$(1/\epsilon \log n)$-exponential function"[1] for a small constant $\epsilon > 0$ in the sense that the $\epsilon \log n$-iterated composition $p^{(\epsilon \log n)}(n)$ of p is at most $2^{n/\log n}$. How this property is used in the proof will be explained in Section 8. In fact, the time complexity is shown to be tight for relativizing proof techniques: Building on the work of Impagliazzo [Imp11],

[1] The name is an analogue of a half-exponential function, which is a function f such that $f(f(n)) \leq 2^n$.

Hirahara and Nanashima [HN21] showed that there is an oracle A such that $\mathsf{DistPH}^A \subseteq \mathsf{AvgP}^A$ and $\mathsf{UP}^A \cap \mathsf{coUP}^A \not\subseteq \mathsf{DTIME}(2^{o(n/\log n)})^A$, which indicates that the time complexity $2^{O(n/\log n)}$ achieved in Theorem 2 cannot be improved to $2^{o(n/\log n)}$ using relativizing proof techniques. We note, however, that the first and third items of Theorem 2 is not necessarily relativizing; whether they can be relativized or not remains open,[2] whereas the second item of Theorem 2 does relativize.

Given the quantitatively tight barrier, it is natural to wonder if one can achieve the time complexity $2^{o(n/\log n)}$. Chen, Hirahara, and Vafa [CHV22] achieved this in fine-grained complexity settings. For example, they showed that nondeterministic linear time $\mathsf{NTIME}(n)$ cannot be solved in quasi-linear time on average if $\mathsf{UP} \not\subseteq \mathsf{DTIME}\left(2^{O(\sqrt{n\log n})}\right)$. The conclusion is weaker than Theorem 2, but the worst-case hardness assumption is also significantly weakened. A high-level idea of [CHV22] is that a quasi-linear function $p(n) = \widetilde{O}(n)$ is a "$\sqrt{\log n/n}$-exponential function" in the sense that $p^{(\sqrt{n/\log n})}(n) \leq 2^{O(\sqrt{n\log n})}$.

The proof is based on meta-complexity of time-bounded Kolmogorov complexity. *Meta-complexity* refers to the computational complexity of problems that themselves ask complexity. One representative example of meta-computational problems is MINKT [Ko91], which is the problem of computing the t-time-bounded Kolmogorov complexity $\mathrm{K}^t(x)$ of x, given $(x, 1^t)$ as input. Throughout this article, time-bounded Kolmogorov complexity and the (meta-)complexity of MINKT play a central role. We review these notions in Section 4. At a very high level, the reason why meta-complexity is useful is that worst-case meta-complexity exactly characterizes the average-case complexity of the polynomial hierarchy PH: [Hir20a] showed that $\mathsf{DistPH} \subseteq \mathsf{AvgP}$ if and only if $\mathrm{GapMINKT}^{\mathsf{PH}} \in \mathsf{P}$, where $\mathrm{GapMINKT}^{\mathsf{PH}}$ is the problem of approximating PH-oracle time-bounded Kolmogorov complexity in the worst case. Meta-complexity enables us to analyze average-case complexity from a view point of worst-case complexity. Analyzing worst-case complexity is often easier than analyzing average-case complexity. Indeed, by going through the statements on worst-case meta-complexity (see the right half of Fig. 1), new statements on average-case complexity of PH are proved in [Hir20a; Hir21a]: One-sided-error heuristics that succeed with small probability, errorless heuristics that succeed with high probability, average-polynomial-time algorithms (AvgP), and P-computable average-polynomial-time algorithms ($\mathsf{Avg_PP}$) are all equivalent for DistPH.

The aforementioned work [Hir18] showed the equivalence between the average-case complexity of MINKT and the worst-case complexity of an approximate version of MINKT. We present this result in Section 5 and explain why meta-complexity enables overcoming the limits of black-box reductions. Along the way, we introduce the notion of *k-wise direct product generator* [Hir20c], which is the main technical tool extensively used throughout this article.

For those who are familiar with randomness extractor, it may be useful to contrast the recent development of meta-complexity with the development of the theory of randomness extractor. The influential work of Trevisan [Tre01] presented a generic construction of an extractor from a pseudorandom generator construction. The k-wise direct product generator is also a (very simple) pseudorandom generator construction. Trevisan [Tre01] exploited the property of a pseudorandom generator construction *information-theoretically* to construct an extractor, which is an information-theoretic object. In contrast, the recent development of meta-complexity is given by exploiting the property of a pseudorandom generator construction *computationally*. In a bit more detail, we apply the reconstruction property of a pseudorandom generator construction

[2] The only non-relativizing part is Theorem 3, which relies on the PCP theorem.

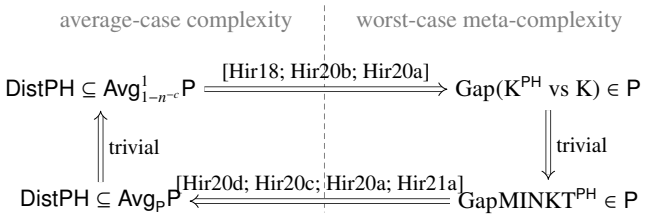


Figure 1: The paradigm of analyzing average-case complexity by worst-case meta-complexity. The figure shows that the following are equivalent for DistPH: $\mathsf{Avg}^1_{1-n^{-c}}\mathsf{P}$, i.e., one-sided-error heuristics that succeed with probability n^{-c}; AvgP, i.e., errorless heuristic schemes, or equivalently, average-polynomial-time algorithms; $\mathsf{Avg}_\mathsf{P}\mathsf{P}$, i.e., P-computable average-polynomial-time algorithms. This equivalence is given as a corollary of the characterization of the average-case complexities of PH by worst-case meta-complexity of GapMINKT$^{\mathsf{PH}}$.

(Lemma 12) to *efficient* algorithms, whereas Trevisan [Tre01] applied it to *inefficient* algorithms.

Two fundamental theorems of Kolmogorov complexity play a key role in the proof of Theorem 2: Language compression and symmetry of information. These theorems are known to be true unconditionally for *resource-unbounded* Kolmogorov complexity [ZL70]. We present *time-bounded* analogues of language compression and symmetry of information in Sections 6 and 7, respectively, under the assumption that DistNP ⊆ AvgP. The proofs of these results are "meta-computational" in the sense that we use an efficient hypothetical algorithm that computes time-bounded Kolmogorov complexity; this means that these proofs must be significantly different from the resource-unbounded cases, as resource-unbounded Kolmogorov complexity cannot be computed in finite steps.

The outline of the proof of Theorem 2 is depicted in Fig. 2. In Section 8, we introduce the notion of universal heuristic scheme. We construct universal heuristic schemes for PH in Section 9. As alluded in the figure, the final result does not refer to meta-complexity at all, whereas the proof goes through statements on worst-case meta-complexity. Indeed, the only known proofs are via meta-complexity!

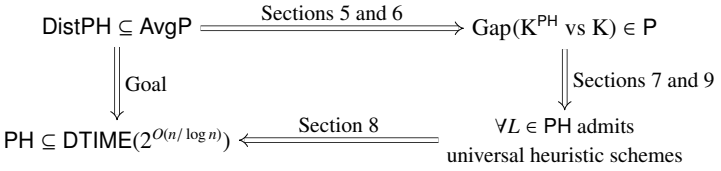


Figure 2: A proof strategy for the second item of Theorem 2.

Notation $[n]$ denotes $\{1,\dots,n\}$. $|x|$ denotes the length of a string $x \in \{0,1\}^*$. For a function $p\colon \mathbb{N} \to \mathbb{N}$ and $i \in \mathbb{N}$, the function $p^{(i)}\colon \mathbb{N} \to \mathbb{N}$ is recursively defined as follows: $p^{(0)}(n) := n$ and $p^{(i+1)}(n) := p^{(i)}(p(n))$ for every $n \in \mathbb{N}$.

2 Impagliazzo's Five Worlds

What is the importance of the P versus NP question? One cannot underestimate significant impacts that a proof of P = NP would have. An efficient algorithm for NP would have great impacts on mathematics in general. If P = NP, for any mathematical theorem φ, one can find a proof π for φ in time $\mathsf{poly}(|\pi|, |\varphi|)$. This means that mathematicians do not need to work hard to find proofs. In a world where P = NP, the only task of mathematicians is to come up with the statements of interesting theorems; then, they can use the efficient automated theorem proving algorithm to find proofs of the theorems. Another consequence of P = NP is that the security of any public-key cryptosystem can be broken. This does not just mean that the privacy of communications is compromised. All the wealth invested to Bitcoin can be stolen by the person who finds a proof of P = NP, as Bitcoin relies on the security of a public-key cryptosystem.[3]

However, most researchers conjecture that P ≠ NP, so it is likely that the answer to the P versus NP question is negative. Given that most researchers believe P ≠ NP, it is more intriguing to investigate what follows from P ≠ NP. In principle, we can expect that a secure public-key cryptosystem can be constructed; however, currently, there is a large gap between P ≠ NP and the possibility of cryptography, which was clearly explained by Impagliazzo [Imp95].

Impagliazzo [Imp95] proposed five possible worlds which are consistent with the current knowledge of complexity theory and named them as follows: Algorithmica, Heuristica, Pessiland, Minicrypt, and Cryptomania. Algorithmica is a world in which NP is easy in the worst case, e.g., P = NP. Heuristica is a world in which there exists an efficient heuristic for NP, i.e., NP is easy on average, but NP is hard in the worst case. For example, one canonical definition of Heuristica is a world in which P ≠ NP and DistNP ⊆ AvgP.[4] Pessiland is a world in which NP is hard on average and there is no one-way function; this is the worst of all the possible worlds. The existence of one-way functions is often considered as a minimal assumption under which complexity-theory-based cryptography is possible [IL89]. In Pessiland, cryptography is not possible and NP cannot be solved efficiently on average. Minicrypt is a world in which one-way functions exist but a public-key cryptosystem does not exist. Finally, Cryptomania is a world in which a public-key cryptosystem exists. A popular conjecture is that our world is Cryptomania. The security of the RSA cryptosystem, which is widely used in practice, is not yet broken despite significant efforts by many researchers; thus, it is reasonable to conjecture that the RSA cryptosystem is secure (at least against classical computers [Sho99]) and, in particular, our world is Cryptomania.

What is known about Impagliazzo's five worlds? It is easy to see the following four implications: $\exists$ a public-key cryptosystem $\Longrightarrow$ $\exists$ a one-way function $\Longrightarrow$ DistNP $\not\subseteq$ AvgP $\Longrightarrow$ NP ≠ P $\Longrightarrow$ True. The converses of these implications correspond to the open problems of excluding Minicrypt, Pessiland, Heuristica, and Algorithmica, respectively. Excluding these worlds is equivalent to showing that our world is Cryptomania, i.e., a secure public-key cryptosystem exists; excluding any one of them is one of the most important open questions in complexity theory and cryptography. Currently, no world is excluded from Impagliazzo's five worlds. The difficulty is that there are many technical barriers suggesting that standard proof techniques would not work. For example, to prove P ≠ NP (i.e., to exclude Algorithmica from

[3] It is worth mentioning that Bitcoin also relies on a cryptographic primitive called *Proof of Work* [DN92], which is based on average-case hardness of NP; see, e.g., [BRSV17; BRSV18; HS21].

[4] There are many variants of Heuristica, depending on what types of algorithms we consider. For example, a world in which NP $\not\subseteq$ BPP and DistNP ⊆ AvgBPP can be considered as a variant of Heuristica.

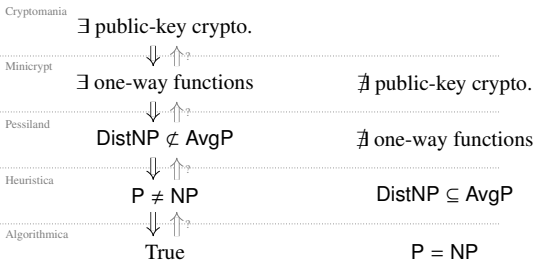


Figure 3: Impagliazzo's five worlds. "⇒" indicates known implications. "$\stackrel{?}{\Rightarrow}$" indicates open questions, each of which corresponds to excluding Algorithmica, Heuristica, Pessiland, and Minicrypt.

the possible worlds), one needs to develop a proof technique that simultaneously overcomes the relativization barrier [BGS75], the algebrization barrier [AW09], the natural proof barrier [RR97], and the locality barrier [CHOPRS20]. Similarly, to exclude Heuristica, one needs to develop a proof technique that simultaneously overcomes the limits of black-box reductions [FF93; BT06b; AGGM06; BB15], the "impossibility" of hardness amplification procedures [Vio05b; Vio05a], and the relativization barrier [Wat12; Imp11; HN21]. In this article, we present a proof technique that is not subject to the first two barriers but is still subject to the relativization barrier; overcoming the relativization barrier is left open.

3 Average-Case Complexity

The theory of average-case complexity studies a *distributional problem* $(L, \mathcal{D})$, which is a pair of a language $L \subseteq \{0,1\}^*$ and a family $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ of distributions over instances of "size" n. For example, we define the uniform distribution $\mathcal{U}$ to be the family $\{\mathcal{U}_n\}_{n\in\mathbb{N}}$ such that $\mathcal{U}_n$ is the uniform distribution over $\{0,1\}^n$. In this specific example, all the strings in the support of $\mathcal{U}_n$ are binary strings of length n; we call such a family of distributions *length-preserving*; however, we emphasize that there is no restriction on $\mathcal{D}_n$ in general, except that $\mathcal{D}_n$ is a distribution over $\{0,1\}^*$.[5]

The seminal work of Levin [Lev86] introduced a robust notion of average-case easiness. An algorithm A is said to be an *average-polynomial-time algorithm for* $(L, \mathcal{D})$ if $A(x, 1^n) = L(x)$ for every $x \in \{0,1\}^*$ and there exists some constant $\epsilon > 0$ such that for all large $n \in \mathbb{N}$, it holds that $\mathbb{E}_{x\sim\mathcal{D}_n}[t_A(x,1^n)^\epsilon] \leq n^{O(1)}$, where $t_A(x,1^n)$ is an upper bound on the running time of an algorithm A on input $(x,1^n)$.[6] The class of distributional problems for which there exist average-polynomial-time algorithms is denoted by AvgP. When there exists an average-case-polynomial upper bound $t_A\colon \{0,1\}^* \to \mathbb{N}$ on the running time of A such that t_A is computable in polynomial time (in the worst case), we say that A is P-*computable average-polynomial-time*; the class of distributional problems solvable by P-computable average-polynomial-time algorithms is

[5] One restriction that automatically follows from $\mathcal{D} \in \mathrm{PSAMP}$ is that the length of any string in the support of $\mathcal{D}_n$ is at most $n^{O(1)}$.

[6] We identify a language $L \subseteq \{0,1\}^*$ with its characteristic function $L\colon \{0,1\}^* \to \{0,1\}$.

denoted by $\mathsf{Avg}_{\mathsf{P}}\mathsf{P}$ [Hir21a].

One may wonder why the constant ϵ in the definition of average-polynomial-time is not fixed to 1. If $\epsilon = 1$, the definition just says that the expected running time $\mathbb{E}_{x\sim\mathcal{D}_n}[t_A(x, 1^n)]$ of an algorithm is bounded by a polynomial in the size parameter n, which appears to be a natural definition at first glance. A short answer to this question is that having a constant exponent $\epsilon > 0$ in the definition makes average-case hardness results stronger: $(L, \mathcal{D}) \notin \mathsf{AvgP}$ implies that $(L, \mathcal{D})$ cannot be solved by average-polynomial-time algorithms in the naïve definition.

Another reason is that AvgP is equivalent to another (perhaps more) intuitive notion: *errorless heuristic scheme* [Imp95; BT06a]. A polynomial-time algorithm A is said to be an *errorless heuristic scheme* if for every $(n, \delta^{-1}) \in \mathbb{N}$,

$$\Pr_{x\sim\mathcal{D}_n}\left[A(x, 1^n, 1^{\delta^{-1}}) \neq L(x)\right] \leq \delta$$

and $A(x, 1^n, 1^{\delta^{-1}}) \in \{L(x), \perp\}$ for every $x \in \mathrm{supp}(\mathcal{D}_n)$, where $\perp$ indicates a failure of A, i..e., the algorithm does not know the answer; that is, A is allowed to *fail* (i.e., output $\perp$) but is not allowed to *err* (i.e., output the wrong answer $1 - L(x)$). For a function $\delta\colon \mathbb{N} \to [0, 1]$, the class of distributional problems that can be solved by errorless heuristics with failure probability $\delta(n)$ on instances of size n is denoted by $\mathsf{Avg}_{\delta}\mathsf{P}$.

What distributions should we consider? A family of distributions $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ is said to be *polynomial-time samplable* if there exists a randomized polynomial-time algorithm M such that the distribution of $M(1^n)$ is identical to $\mathcal{D}_n$ for every $n \in \mathbb{N}$. The class of polynomial-time samplable distributions is denoted by PSamp. We define DistNP to be the class of distributional problems $(L, \mathcal{D})$ such that $L \in \mathsf{NP}$ and $\mathcal{D} \in$ PSamp.

It will be useful to consider a family $\mathcal{D} = \{\mathcal{D}_{a,b}\}_{a,b\in\mathbb{N}}$ of distributions indexed by a pair $(a, b) \in \mathbb{N}^2$ of integers. Such a family can be converted into a family $\mathcal{D}' = \{\mathcal{D}'_n\}_{n\in\mathbb{N}}$ indexed by a single integer n by defining $\mathcal{D}'_{\langle a,b\rangle} := \mathcal{D}_{a,b}$ for every $(a, b) \in \mathbb{N}^2$, where $\langle\text{-},\text{-}\rangle\colon \mathbb{N}\times\mathbb{N} \to \mathbb{N}$ is a bijection defined as, for example, $\langle a, b\rangle := \sum_{i=0}^{a+b} i + a$.

Buhrman, Fortnow, and Pavan [BFP05] showed an important theorem that will be useful throughout this article. They showed pr-BPP = pr-P in Heuristica, i.e., randomized polynomial-time algorithms can be derandomized in polynomial time. The underlying tool of derandomization is the notion of *pseudorandom generator*. A function $G = \left\{G_n : \{0, 1\}^{s(n)} \to \{0, 1\}^n\right\}_{n\in\mathbb{N}}$ is said to be a *pseudorandom generator* secure against linear-sized circuits if for all large $n \in \mathbb{N}$, for every circuit D of size n,

$$\left|\Pr_{z\sim\{0,1\}^{s(n)}}[D(G_n(z)) = 1] - \Pr_{w\sim\{0,1\}^n}[D(w) = 1]\right| \leq \frac{1}{n}.$$

Note that the existence of a pseudorandom generator with seed length $s(n)$ enables derandomizing randomized polynomial-time algorithms in time $2^{O(s(n)+\log n)}$; more background on derandomization can be found in [Vad12].

Theorem 3 (Buhrman, Fortnow, and Pavan [BFP05])**.** *If* $\mathsf{DistNP} \subseteq \mathsf{AvgP}$*, then there exists a pseudorandom generator*

$$G = \left\{G_n : \{0, 1\}^{O(\log n)} \to \{0, 1\}^n\right\}_{n\in\mathbb{N}}$$

computable in time $n^{O(1)}$ *and secure against linear-sized circuits.*

Proof Sketch. Theorem 3 is based on the following four results.

1. DistNP $\subseteq$ AvgP implies NE = E [BCGL92].
2. DistNP $\subseteq$ AvgP implies pr-MA = pr-NP [KS04].
3. If NE = E and pr-MA = pr-NP, then $\mathsf{E} \not\subset \mathsf{i.o.SIZE}(2^{\epsilon n})$ for some constant $\epsilon > 0$ [BFP05]. (Proof Sketch: If not, we obtain $\mathsf{E} \subseteq \mathsf{i.o.MATIME}(2^{o(n)}) = \mathsf{i.o.NTIME}(2^{o(n)})$ using a PCP theorem. However, this contradicts the time hierarchy and NE = E; see [Hir20a] for the details.)
4. If $\mathsf{E} \not\subset \mathsf{i.o.SIZE}(2^{\epsilon n})$ for some constant $\epsilon > 0$, then there exists a polynomial-time-computable pseudorandom generator secure against linear-sized circuits [IW97].

□

4 Meta-Complexity of Time-Bounded Kolmogorov Complexity

4.1 Kolmogorov Complexity

The Kolmogorov complexity $\mathrm{K}(x)$ of a string $x \in \{0,1\}^*$ is defined to be the minimum length of a program that prints x. For example, $\mathrm{K}(1^n) = \log n + O(1)$ because the string 1^n can be described by a program "`print '1' n times`", whose description length is $\log n + O(1)$. Note that an integer $n \in \mathbb{N}$ can be represented as a binary string of length $\lceil \log n \rceil$ and the length of the other part of the program is constant.

The t-time-bounded Kolmogorov complexity $\mathrm{K}^t(x)$ of x is defined to be the minimum length of a program that prints x in time t. We also consider the conditional Kolmogorov complexity $\mathrm{K}^t(x \mid y)$ of x given y, which is the minimum length of a program that prints x given y as input in time t. More generally, we consider oracle time-bounded Kolmogorov complexity:

Definition 4 (Time-bounded Kolmogorov complexity)**.** *For strings $x, y \in \{0,1\}^*$, a time bound $t \in \mathbb{N} \cup \{\infty\}$, and an oracle A, the A-*oracle *t-*time-bounded Kolmogorov complexity *of x given y is defined as*

$$\mathrm{K}^{t,A}(x \mid y) := \min\left\{|M| \;\middle|\; M^A \text{ outputs } x \text{ in time } t\right\}.$$

Here, $|M|$ denotes the length of a binary encoding of an oracle Turing machine M.[7] *We omit the superscript A if $A = \emptyset$, the superscript t if $t = \infty$, and "$\mid y$" if y is the empty string.*

The Kolmogorov complexity of an n-bit string $x \in \{0,1\}^n$ satisfies $0 \le \mathrm{K}(x) \le n + O(1)$ because any string x can be described by a program "`print` x". The upper bound is almost tight:

Fact 5. *For any integer $s \ge 1$, the number of strings $x \in \{0,1\}^*$ such that $\mathrm{K}(x) < s$ is less than 2^s. In particular, $\Pr_{x \sim \{0,1\}^n}[\mathrm{K}(x) \ge s] \ge 1 - 2^{s-n}$.*

[7] More formally, we fix an efficient universal Turing machine U such that for every oracle Turing machine M, there exists a string $d_M \in \{0,1\}^*$ such that $U^A(d_M, x) = M^A(x)$ for every input $x \in \{0,1\}^*$ and every oracle $A \subseteq \{0,1\}^*$. Then, we define $|M|$ to be the length $|d_M|$ of d_M.

Proof. For every string x such that $\mathrm{K}(x) < s$, there is a program M_x of length less than s that prints x. The map $x \mapsto M_x$ is injective. The number of programs of length less than s is at most $\sum_{i=0}^{s-1} 2^i < 2^s$, so is the number of strings x such that $\mathrm{K}(x) < s$. □

More background on Kolmogorov complexity can be found in the book of Li and Vitányi [LV19].

4.2 Meta-Complexity

Here, we review the notion of meta-complexity that is relevant to Theorem 2. For broader background on meta-complexity and its recent development, see the survey of Allender [All21].

We first introduce the representative meta-computational problem: MINKT [Ko91] is defined to be the language

$$\mathrm{MINKT} := \{(x, 1^t, 1^s) \mid \mathrm{K}^t(x) \leq s\}.$$

This is a problem in NP because given a program M as certificate one can check whether $|M| \leq s$ and M prints x in time t. Its NP-completeness is a long-standing open question [Ko91]. Note that MINKT is computationally equivalent to the problem of computing the t-time-bounded Kolmogorov complexity $\mathrm{K}^t(x)$ of x on input $(x, 1^t)$; in other words, MINKT asks to compute the time-bounded Kolmogorov complexity of x. The complexity of MINKT is referred to *meta-complexity*, as MINKT itself asks for the complexity of printing an input x.

Although it is unknown whether MINKT can be solved *exactly* in Heuristica, we show in the next section that an *approximate* version of MINKT, denoted by GapMINKT, can be solved in Heuristica. To introduce the problem formally, we recall the framework of promise problems [Gol06]. A *promise problem* Π is a pair $(\Pi_{\mathrm{Yes}}, \Pi_{\mathrm{No}})$ of languages. An algorithm A is said to *solve* a promise problem $\Pi = (\Pi_{\mathrm{Yes}}, \Pi_{\mathrm{No}})$ if A accepts every $x \in \Pi_{\mathrm{Yes}}$ and rejects every $x \in \Pi_{\mathrm{No}}$. Note that if there exists an algorithm that solves a promise problem $\Pi = (\Pi_{\mathrm{Yes}}, \Pi_{\mathrm{No}})$, it must be disjoint, i.e., $\Pi_{\mathrm{Yes}} \cap \Pi_{\mathrm{No}} = \emptyset$. It is common to require a promise problem to be disjoint in the definition; however, we also consider "non-disjoint" promise problems [Hir20b], i.e., promise problems that are non-disjoint under plausible assumptions, which will be useful in Section 6.

Definition 6 ([Ko91; Hir20b])**.** *For a polynomial $\tau\colon \mathbb{N} \to \mathbb{N}$ and an oracle $A \subseteq \{0,1\}^*$, let*

$$\Pi^A_{\mathrm{Yes}} := \left\{(x, 1^t, 1^s) \mid \mathrm{K}^{t,A}(x) \leq s\right\},$$
$$\Pi^A_{\mathrm{No}} := \left\{(x, 1^t, 1^s) \mid \mathrm{K}^{\tau(|x|,t),A}(x) > s + \log \tau(|x|, t)\right\}.$$

Then, we define

- $\mathrm{Gap}_\tau\mathrm{MINKT}^A := (\Pi^A_{\mathrm{Yes}}, \Pi^A_{\mathrm{No}})$ *and*
- $\mathrm{Gap}_\tau(\mathrm{K}^A \text{ vs } \mathrm{K}) := (\Pi^A_{\mathrm{Yes}}, \Pi^\emptyset_{\mathrm{No}})$.

We say that $\mathrm{GapMINKT}^A \in \mathsf{P}$ *if there exists some polynomial τ such that* $\mathrm{Gap}_\tau\mathrm{MINKT}^A \in \mathsf{P}$. *For a complexity class $\mathbb{C}$, we say that* $\mathrm{GapMINKT}^{\mathbb{C}} \in \mathsf{P}$ *if* $\mathrm{GapMINKT}^A \in \mathsf{P}$ *for every $A \in \mathbb{C}$. We omit the superscript A if $A = \emptyset$.*

For example, $\mathrm{GapMINKT}^{\mathrm{SAT}}$ is the problem of approximating SAT-oracle time-bounded Kolmogorov complexity. The problem $\mathrm{Gap}(\mathrm{K}^{\mathrm{SAT}} \text{ vs } \mathrm{K})$ is computationally harder than $\mathrm{GapMINKT}^{\mathrm{SAT}}$ and is an example of "non-disjoint" promise problems: It is non-disjoint if $\mathsf{E}^{\mathsf{NP}} \neq \mathsf{E}$ [Hir20b].

Nevertheless, we show in Section 6 that Gap($\mathrm{K}^{\mathrm{SAT}}$ vs K) can be solved in polynomial time if $\mathsf{DistPH} \subseteq \mathsf{AvgP}$.

One common misunderstanding about GapMINKT$^{\mathrm{SAT}}$ is that it should be NP-hard as SAT is NP-complete. In fact, proving NP-hardness of GapMINKT$^{\mathrm{SAT}}$ would significantly improve Theorem 2: $\mathsf{P} \neq \mathsf{NP}$ implies $\mathsf{DistPH} \not\subseteq \mathsf{AvgP}$ if GapMINKT$^{\mathrm{SAT}}$ is NP-hard. Thus, it is an important open problem to prove the NP-hardness of GapMINKT$^{\mathrm{SAT}}$, despite the intuition that approximating SAT-oracle time-bounded Kolmogorov complexity seems much harder than computing SAT.

The problem GapMINKT can be equivalently defined as the problem of computing an approximate value of $\mathrm{K}^t(x)$ up to some additive error.

Fact 7 ([Hir21a])**.** *The following are equivalent.*

1. GapMINKT $\in \mathsf{P}$.
2. *There exist a polynomial-time algorithm* $\widetilde{\mathrm{K}}$ *and a polynomial* p *such that*
$$\mathrm{K}^{p(t)}(x) - \log p(t) \leq \widetilde{\mathrm{K}}(x, 1^t) \leq \mathrm{K}^t(x)$$
for every string $x \in \{0,1\}^*$ *and every integer* $t \geq |x|$.

Proof Sketch. Given a polynomial-time algorithm M that solves GapMINKT, the algorithm $\widetilde{\mathrm{K}}$ can be defined as follows:
$$\widetilde{\mathrm{K}}(x, 1^t) := \min\{s \in \mathbb{N} \mid M(x, 1^t, 1^s) = 1\}.$$
□

Fact 7 shows that GapMINKT is equivalent to the problem of computing $\mathrm{K}^t(x)$ up to an additive error of
$$\mathrm{K}^t(x) - (\mathrm{K}^{p(t)}(x) - \log p(t)) = \mathrm{cd}^{t,p(t)}(x) + \log p(t),$$
where $\mathrm{cd}^{t,s}(x) := \mathrm{K}^t(x) - \mathrm{K}^s(x)$ is called (s,t)-*time-bounded computational depth*. This notion plays a key role in Section 8.

Definition 8 (Time-Bounded Computational Depth; [AFMV06; Hir21a])**.** *For time bounds* $s \in \mathbb{N}$ *and* $t \in \mathbb{N} \cup \{\infty\}$ *and a string* $x \in \{0,1\}^*$*, the* (s,t)-time-bounded computational depth *of* x *is defined as*
$$\mathrm{cd}^{s,t}(x) := \mathrm{K}^s(x) - \mathrm{K}^t(x).$$
We omit the superscript t *if* $t = \infty$.

The notion of (time-unbounded) computational depth $\mathrm{cd}^t(\text{-})$ was introduced by Antunes, Fortnow, Melkebeek, and Vinodchandran [AFMV06]. One fundamental property of computational depth is that it is small for most strings. For example, [AFMV06] showed that $\mathrm{cd}^{\mathsf{poly}(n)}(x) \leq k$ holds with probability at least $1 - 2^{-k+O(\log n)}$ over a random input x drawn from a *polynomial-time-computable distribution* $\mathcal{D}$. Here, a polynomial-time-computable distribution is a distribution whose cumulative function can be computed in polynomial time. Under a plausible assumption, Antunes and Fortnow [AF09] generalized it to polynomial-time-*samplable* distributions; [Hir21a] proved the same result in Heuristica.

Theorem 9 ([AF09; Hir21a])**.** *Assume either* $\mathsf{E} \not\subseteq \mathsf{i.o.DSPACE}(2^{\epsilon n})$ *for some constant* $\epsilon > 0$ *or* $\mathsf{DistNP} \subseteq \mathsf{AvgP}$*. Then, for every* $\mathcal{D} \in \mathrm{PSAMP}$*, there exists a polynomial* t *such that for every* $n \in \mathbb{N}$*,*
$$\Pr_{x \sim \mathcal{D}_n}\left[\mathrm{cd}^{t(n)}(x) > k\right] \leq 2^{-k+\log t(n)}.$$

5 Non-Black-Box Worst-Case-to-Average-Case Reduction

Feigenbaum and Fortnow [FF93] and Bogdanov and Trevisan [BT06b] presented fundamental limits of proof techniques called *(black-box) reductions*. A natural approach to showing a worst-case-to-average-case connection would be to construct a reduction. In order to show the implication from the worst-case hardness of a language L to the average-case hardness of a distributional problem $(L', \mathcal{D})$, it suffices to design a reduction R that takes an arbitrary oracle O that solves $(L', \mathcal{D})$ on average and decides L in the worst case. Typically, the correctness of a reduction can be shown regardless of the complexity of an oracle O; i.e., R^O decides L for *every* oracle $O \subseteq \{0,1\}^*$. Such a reduction R is called *black-box*. Bogdanov and Trevisan [BT06b] showed the following limits of black-box reductions.

Theorem 10 ([BT06b])**.** *Let L be any language outside* $\mathsf{NP/poly} \cap \mathsf{coNP/poly}$. *Then, there is no randomized polynomial-time nonadaptive reduction from L to* DistNP.

To exclude Heuristica, it suffices to show that some NP-complete problem is reducible to DistNP. NP-complete problems are believed to be outside $\mathsf{NP/poly} \cap \mathsf{coNP/poly}$, as otherwise PH collapses [Yap83]. Theorem 10 indicates that, unless PH collapses, NP-complete problems cannot be reducible to DistNP via a randomized polynomial-time nonadaptive reduction.

How can we overcome this limits? One approach is to consider *adaptive* reductions. There are adaptive reductions in the literature (e.g., [HILL99; MR07]) and there exists an artificial language that admits an adaptive random-self-reduction but not nonadaptive one [FFLS94]. However, it is unknown if there exists an adaptive reduction for a natural language that goes beyond the limits $\mathsf{NP/poly} \cap \mathsf{coNP/poly}$ of black-box reductions. The other approach is to use *non-black-box* reductions. Here, we say that a reduction R is non-black-box if the reduction exploits an efficiency of an oracle, i.e., R^O may fail to decide L correctly if O cannot be decided in polynomial time.[8]

We now demonstrate that GapMINKT admits a worst-case-to-average-case connection. The connection is proved by a non-black-box reduction that exploits the efficiency of an oracle, i.e., an efficient hypothetical algorithm that solves DistNP.

Theorem 11 ([Hir18; Hir20b])**.** *If* $\mathsf{DistNP} \subseteq \mathsf{AvgP}$*, then* $\mathrm{GapMINKT} \in \mathsf{P}$.

To prove this theorem, we introduce a *k-wise direct product generator* [Hir20c], which turned out to be a fundamental tool for analyzing Kolmogorov complexity. A k-wise direct product generator $\mathrm{DP}_k \colon \{0,1\}^n \times \{0,1\}^{nk} \to \{0,1\}^{nk+k}$ is defined as follows:

$$\mathrm{DP}_k(x; z) := (z^1, \ldots, z^k, \langle z^1, x\rangle, \ldots, \langle z^k, x\rangle)$$

for every $x \in \{0,1\}^n$ and every $z = (z^1, \ldots, z^k) \in (\{0,1\}^n)^k$, where $\langle x, y\rangle$ denotes the inner product of x and $y \in \{0,1\}^n$ over GF(2), i.e., $\langle x, y\rangle := (\sum_{i=1}^n x_i y_i) \mod 2$. The k-wise direct product generator $\mathrm{DP}_k(x; \text{-})$ is a pseudorandom generator secure against an algorithm D if $\mathrm{K}(x \mid D) > k + O(\log |x|)$. More formally, we have the following property:

Lemma 12 (Deterministic Reconstruction for DP_k; see [Hir21a])**.** *Assume that* $\mathsf{DistNP} \subseteq \mathsf{AvgP}$. *Then, there exists a polynomial p such that, for every $n \in \mathbb{N}$, $x \in \{0,1\}^n$, parameters $k, \epsilon^{-1}, s \in \mathbb{N}$, and for every randomized circuit D of size s such that*

$$\left| \Pr_{z,r}[D(\mathrm{DP}_k(x; z); r) = 1] - \Pr_{w,r}[D(w; r) = 1] \right| \geq \epsilon,$$

[8] In fact, more importantly, O should not depend on the input x, which is used in the proof of Theorem 11.

where $z \sim \{0,1\}^{nk}$, $w \sim \{0,1\}^{nk+k}$, *and* $r \sim \{0,1\}^{s}$, *it holds that*

$$\mathrm{K}^{p(ns/\epsilon)}(x \mid D) \leq k + \log p(ns/\epsilon).$$

This lemma can be proved by standard proof techniques of pseudorandomness [Vad12, Chapter 7] together with Theorem 3. We briefly present a proof sketch.

Proof Sketch. Using Yao's next-bit predictor, given some prefix of $\mathrm{DP}_k(x; z)$, the next bit can be predicted. Note that the first nk bits of $\mathrm{DP}_k(x; z)$ are completely uniform and cannot be predicted. Thus, there exists some index $i \in [k]$ such that $\langle z^i, x\rangle$ can be predicted with probability at least $\frac{1}{2} + \frac{\epsilon}{k}$, given $z^1, \ldots, z^k, \langle z^1, x\rangle, \ldots, \langle z^{i-1}, x\rangle$ as input. Note that $\langle z^i, x\rangle$ is the z^i-th bit of the Hadamard code of x. Using the list-decoding algorithm of Goldreich and Levin [GL89], one can decode x from the next-bit predictor. We have described a randomized algorithm M that prints x given $z^1, \ldots, z^k, \langle z^1, x\rangle, \ldots, \langle z^{i-1}, x\rangle$ as input. However, the random bits $z^1, \ldots, z^k$ as well as the internal randomness of M can be generated from the pseudorandom generator G of Theorem 3; i.e., there exists a short seed $\sigma \in \{0,1\}^{O(\log n)}$ from which $z^1, \ldots, z^k$ can be generated. Now we are ready to describe a program M' that prints x: Given a seed σ and an "advice string" $\alpha := (\langle z^1, x\rangle, \ldots, \langle z^{i-1}, x\rangle) \in \{0,1\}^{i-1}$, the program M' computes $(z^1, \ldots, z^k) := G(\sigma)$ and outputs the output of M on input $z^1, \ldots, z^k, \langle z^1, x\rangle, \ldots, \langle z^{i-1}, x\rangle$. The length of the program M' is approximately at most $|\sigma| + |\alpha| \leq k + O(\log n)$. □

In the literature of pseudorandom generator constructions and randomness extractors [TV07; TUZ07; Uma09], the length of the advice string α is referred to as the *advice complexity* of the pseudorandom generator construction $\mathrm{DP}_k(x; \text{-})$. Trevisan and Vadhan [TV07] showed that the advice complexity of any pseudorandom generator construction that extends its seed by k bits must be at least $k - 3$; thus, the simple pseudorandom generator construction $\mathrm{DP}_k(x; \text{-})$ achieves the optimal advice complexity up to an additive logarithmic term.[9]

Given Lemma 12, the proof of Theorem 11 is fairly simple.

Proof of Theorem 11. Consider a family $\mathcal{D} := \{\mathcal{D}_{n,t}\}_{n,t\in\mathbb{N}}$ of distributions over instances of MINKT such that $\mathcal{D}_{n,t}$ is the distribution defined by the following sampling procedure: Pick $x \sim \{0,1\}^n$ uniformly at random and output $(x, 1^t, 1^{n-2})$.

Since $(\mathrm{MINKT}, \mathcal{D}) \in \mathsf{DistNP} \subseteq \mathsf{AvgP}$, there exists an errorless polynomial-time heuristic M that solves $(\mathrm{MINKT}, \mathcal{D})$ with failure probability at most $\frac{1}{4}$. We now present a randomized algorithm R that solves GapMINKT: The algorithm R takes $(x, 1^t, 1^s)$ as input and accepts if and only if $M(\mathrm{DP}_k(x; z), 1^{t'}) \in \{1, \perp\}$ for a random $z \sim \{0,1\}^{nk}$, where $n := |x|$, k and $t' = \mathsf{poly}(t, n)$ are parameters chosen later.

To see the correctness of R, consider any YES instance $(x, 1^t, 1^s)$ of GapMINKT; i.e., $\mathrm{K}^t(x) \leq s$. Since the string $\mathrm{DP}_k(x; z)$ can be described from the seed z and a program that prints x in time t, we have

$$\mathrm{K}^{t'}(\mathrm{DP}_k(x; z)) \leq |z| + \mathrm{K}^t(x) + O(\log n) \leq nk + s + O(\log n) \leq nk + k - 2$$

[9] In the original paper [Hir18], the pseudorandom generator of Theorem 3 is not used; in this case, to obtain a small approximation error in Theorem 11, one needs to use a pseudorandom generator construction that has small *randomness complexity* in addition to small advice complexity. Whether the approximation error of [Hir18] can be improved or not under the assumption that MINKT is easy on average (under which the existence of a pseudorandom generator is unknown) remains open.

for a sufficiently large t', where we choose a large $k = s + O(\log n)$ so that the last inequality holds. This ensures that $(\mathrm{DP}_k(x; z), 1^{t'}, 1^{nk+k-2})$ is a YES instance of MINKT for every z. By the property of an errorless heuristic scheme, M outputs either 1 or $\perp$ on input $(\mathrm{DP}_k(x; z), 1^{t'}, 1^{s'})$; hence, R accepts with probability 1 over a choice of $z \sim \{0,1\}^{nk}$.

Conversely, we claim that R rejects any No instance $(x, 1^t, 1^s)$ of $\mathrm{Gap}_\tau\mathrm{MINKT}$ with high probability over the internal randomness of R (i.e., the random choice $z \sim \{0,1\}^{nk}$), where τ is a polynomial chosen later. Intuitively, this is because an algorithm M cannot distinguish the output distribution $\mathrm{DP}_k(x; \text{-})$ of the k-wise direct product generator from the uniform distribution if $\mathrm{K}^{\mathsf{poly}(t)}(x \mid M) > k + O(\log n) = s + O(\log n)$. More formally, we prove the contrapositive: Assume that R accepts $(x, 1^t, 1^s)$ with probability at least $\frac{7}{8}$ over a choice of z; that is,

$$\Pr_z\left[M(\mathrm{DP}_k(x; z), 1^{t'}) \in \{1, \perp\}\right] \geq \frac{7}{8}.$$

We claim that $(x, 1^t, 1^s)$ is *not* a No instance of $\mathrm{Gap}_\tau\mathrm{MINKT}$ for some polynomial τ. If we pick $w \sim \{0,1\}^{nk+k}$, then by Fact 5, with probability at least $\frac{1}{2}$, it holds that $\mathrm{K}(w) > |w| - 2$, in which case $(w, 1^{t'}, 1^{|w|-2})$ is a No instance of MINKT; hence,

$$\Pr_w\left[M(w, 1^{t'}) \in \{1, \perp\}\right] \leq \Pr_w\left[M(w, 1^{t'}) = \perp\right] + \Pr_w\left[(w, 1^{t'}, 1^{|w|-2}) \in \mathrm{MINKT}\right] \leq \frac{1}{4} + \frac{1}{2} = \frac{3}{4}.$$

Let D be a circuit such that $D(w) := 1$ iff $M(w, 1^{t'}) \in \{1, \perp\}$. Then, it follows from the two inequalities above that

$$\Pr_z\left[D(\mathrm{DP}_k(x; z)) = 1\right] - \Pr_w\left[D(w) = 1\right] \geq \frac{7}{8} - \frac{3}{4} = \frac{1}{8}.$$

By Lemma 12, we obtain

$$\mathrm{K}^{p(t,n)}(x \mid D) \leq k + \log p(t, n)$$

for some polynomial p. Since the circuit D can be described using $O(\log t')$ bits, we conclude that

$$\mathrm{K}^{\tau(t,n)}(x) \leq s + \log \tau(t, n) \tag{1}$$

for some polynomial τ, which means that $(x, 1^t, 1^s)$ is not a No instance of $\mathrm{Gap}_\tau\mathrm{MINKT}$.

We conclude that R is a one-sided-error randomized polynomial-time algorithm for $\mathrm{Gap}_\tau\mathrm{MINKT}$; that is, $\mathrm{Gap}_\tau\mathrm{MINKT} \in \text{pr-}\mathsf{coRP} \subseteq \text{pr-}\mathsf{BPP} = \text{pr-}\mathsf{P}$, where the last equality follows from Theorem 3. □

Why is the reduction R constructed in the proof of Theorem 11 non-black-box? The key point is that Eq. (1) may not hold when there is no polynomial-time algorithm M that decides MINKT. Specifically, the circuit D comes from the hypothetical efficient algorithm M that solves MINKT. It would be possible to show that $\mathrm{K}^{\tau(t,n),\mathrm{MINKT}}(x) \leq s + \log \tau(t, n)$ unconditionally; however, removing the MINKT oracle seems to be impossible in general. In fact, under plausible conjectures, the reduction R must be non-black-box. For example, the conjecture by Rudich [Rud97] implies that $\mathrm{GapMINKT} \notin \mathsf{coNP/poly}$; thus, if R were non-black-box, then we would get a contradiction to the limits of black-box reductions (Theorem 10). Other evidences that Theorem 11 is inherently non-black-box can be found in [Hir18; HW20].

It is instructive to emphasize that meta-complexity plays a key role in constructing the non-black-box reduction. The problem GapMINKT is a meta-computational problem that asks

the complexity of generating a given string x in a given time limit t. It is this meta-computational property of GapMINKT that enables us to exploit the efficiency of a hypothetical algorithm M in the proof of Theorem 11.

We mention that the proof techniques of Theorem 11 actually show the equivalence between the worst-case hardness of GapMINKT and the existence of a polynomial-time-computable hitting set generator. A family $H = \left\{H_n : \{0,1\}^{s(n)} \to \{0,1\}^n\right\}_{n\in\mathbb{N}}$ is said to be a *hitting set generator* secure against i.o.P if for every polynomial-time algorithm M, there exist infinitely many integers $n \in \mathbb{N}$ such that $\Pr_{x\sim\{0,1\}^n}[M(x) = 1] \geq \frac{1}{2}$ implies $M(H_n(z)) = 1$ for some seed $z \in \{0,1\}^{s(n)}$.

Proposition 13 ([Hir20a; Hir21b])**.** *Under the assumption that* $\mathsf{E} \not\subseteq \text{i.o.}\mathsf{SIZE}(2^{\epsilon n})$ *for some constant* $\epsilon > 0$*, the following are equivalent.*

1. GapMINKT $\in \mathsf{P}$.
2. *There exists a constant c such that no polynomial-time-computable hitting set generator*

$$H = \left\{H_n : \{0,1\}^{n-c\log n} \to \{0,1\}^n\right\}_{n\in\mathbb{N}}$$

   is secure against i.o.P.
3. *There exists a constant c such that* $(\text{MINKT}, \mathcal{U}^*) \in \mathsf{Avg}_{\frac{1}{4}}\mathsf{P}$*, where* $\mathcal{U}^* = \left\{\mathcal{U}^*_{n,t}\right\}_{n,t\in\mathbb{N}}$ *is the family of distributions $\mathcal{U}^*_{n,t}$ that sample $(x, 1^t, 1^{n-c\log n})$ for $x \sim \{0,1\}^n$.*
4. *For every length-preserving distribution $\mathcal{D} \in \mathrm{PSAMP}$, there exists a polynomial t_0 such that for every $t \geq t_0$, there exists an errorless heuristic scheme that computes $\mathrm{K}^{t(|x|)}(x)$ over a random instance x chosen from $\mathcal{D}$.*

Although we include a proof sketch for completeness, the reader may skip the proof.

Proof Sketch. Item 2 $\Rightarrow$ 1: The idea is to use a "universal" hitting set generator $H = \{H_n\}$. Let H_n be the function that takes as input a program M of length less than $n - c\log n$ (which can be encoded as a binary string of length $n - c\log n$) and outputs the output of M if M halts in time n^2. Clearly, H_n is computable in $\mathsf{poly}(n)$. The image of H_n contains the set $X_n := \left\{x \in \{0,1\}^n \;\middle|\; \mathrm{K}^{n^2}(x) < n - c\log n\right\}$. Since H is not secure against i.o.P, there exists a polynomial-time algorithm M that rejects every $x \in X_n$ and accepts at least a half of all the n-bit strings. Then, GapMINKT can be solved by the following algorithm R: Given $(x, 1^t, 1^s)$ as input, R picks $z \sim \{0,1\}^{nk}$ and $w \sim \{0,1\}^t$ randomly and outputs $1 - M(\mathrm{DP}_k(x; z) \cdot w)$, where $n := |x|$ and $k := s + O(\log nt)$. Here, w is used for padding the input of M so that the time bound $|\mathrm{DP}_k(x; z) \cdot w|^2$ is sufficiently larger than t. The correctness can be proved in a way similar to Theorem 11. See [Hir20a, Theorem 8.7] for the details.

Item 1 $\Rightarrow$ 3: Let M be a polynomial-time algorithm that solves Gap_τMINKT for some polynomial τ. Then, an errorless heuristic M' for $(\text{MINKT}, \mathcal{U}^*)$ can be defined as follows: $M'(x, 1^t, 1^s) := \perp$ if $M(x, 1^t, 1^s) = 1$; otherwise, $M'(x, 1^t, 1^s) := 0$, where s is fixed to $n - c\log n$. To see the correctness of M', observe that M' can err only on YES instances of MINKT If $(x, 1^t, 1^s) \in \text{MINKT}$, then we have $M(x, 1^t, 1^s) = 1$, which implies that M' outputs $\perp$; thus, M' does not err. The probability that M' outputs $\perp$ is bounded by the probability that $M(x, 1^t, 1^s) = 1$, which happens only if $(x, 1^t, 1^s)$ is not a No instance of Gap_τMINKT. It follows from Fact 5

that $\mathrm{K}(x) \geq n - 2 > s + \log \tau(n,t) = n - c\log n + \log \tau(n,t)$ with probability at least $\frac{3}{4}$ for a large constant c; hence, the probability that M' fails is at most $\frac{1}{4}$.

Item 3 $\Rightarrow$ 2: Let c be a constant chosen later. Let H be an arbitrary family

$$H = \left\{H_n : \{0,1\}^{n-c\log n} \to \{0,1\}^n\right\}_{n\in\mathbb{N}},$$

which is computable in time $\ll t(n)$. Given an errorless heuristic M for $(\mathrm{MINKT}, \mathcal{U}^*)$, we define M' so that $M'(x) := 0$ if $M(x, 1^{t(n)}, 1^{n-c'\log n}) \in \{1, \perp\}$ and $M'(x) := 1$ otherwise, for every $x \in \{0,1\}^n$. We show that M' "avoids" H:

1. The algorithm M' accepts at least a half of $\{0,1\}^n$ because at least a $(1 - o(1))$-fraction of $\{0,1\}^n$ is a No instance of MINKT and the failure probability of M is at most $\frac{1}{4}$.

2. We claim that M' rejects every string x in the image of H_n: Note that $x \in \{0,1\}^n$ can be described by an integer $n \in \mathbb{N}$ and a seed $z \in \{0,1\}^{n-c\log n}$ such that $H_n(z) = x$, which implies that $\mathrm{K}^{t(n)}(x) \leq n - c\log n + O(\log n) \leq n - c'\log n$ for a sufficiently large c; this means that $(x, 1^{t(n)}, 1^{n-c'\log n}) \in \mathrm{MINKT}$, which is rejected by M'.

Item 1 $\Leftrightarrow$ 4: This is proved in [Hir21b] using SoI of Section 7. We omit the proof. □

We conclude this section by describing applications to derandomization. Given that a hitting set generator is used to derandomize randomized algorithms [GVW11], it is natural to wonder if one can characterize *complexity-theoretic* hitting set generators using the proof techniques of Proposition 13. Here, a hitting set generator or a pseudorandom generator $G = \left\{G : \{0,1\}^{s(n)} \to \{0,1\}^n\right\}_{n\in\mathbb{N}}$ with seed length $s(n)$ is said to be *complexity-theoretic* if it can be computed in time $2^{O(s(n)+\log n)}$. This notion is more appropriate for derandomization [NW94] and is weaker than *cryptographic* generators. For example, the pseudorandom generator of Theorem 3 is complexity-theoretic. In [Hir20b], the existence of complexity-theoretic hitting set generators is characterized by the worst-case hardness of approximating Levin's Kt complexity [Lev84], which is an exponential-time variant of time-bounded Kolmogorov complexity. We refer the reader to [Hir20b] for details.

6 Algorithmic Language Compression

The language compression theorem [Sip83; BFL01; BLM05] for resource-unbounded Kolmogorov complexity refers to the following simple and fundamental fact:

Fact 14 (Language Compression)**.** *Let L be a decidable language. Then, for every $n \in \mathbb{N}$ and every $x \in \{0,1\}^n \cap L$,*

$$\mathrm{K}(x) \leq \log|L \cap \{0,1\}^n| + O(\log n).$$

Proof Sketch. Any string $x \in L \cap \{0,1\}^n$ can be described by the index of x in the enumeration of $L \cap \{0,1\}^n$ and an integer $n \in \mathbb{N}$. □

Sipser [Sip83] considered a time-bounded analogue of the language compression theorem and showed that $\mathrm{K}^{\mathsf{poly}(n),\mathrm{SAT}}(x \mid r) \leq |L \cap \{0,1\}^n| + O(\log n)$ for $x \in L \in \mathsf{P}$, where r is a random string. Here, we show that the SAT oracle and the random string r can be removed if $\mathsf{DistNP} \subseteq \mathsf{AvgP}$. At a very high level, the idea is that the SAT oracle can be replaced with

an errorless heuristic for DistNP. We also remove the random string using Theorem 3. It will be useful to state the language compression theorem for a family of languages indexed by an integer t, which we call an *ensemble of languages*:

Definition 15 (Ensemble of Languages)**.** *For every language $L \subseteq \{0,1\}^*$ and every $t \in \mathbb{N}$, let L_t denote $\{x \in \{0,1\}^* \mid (x, 1^t) \in L\}$. We say that a language $L \subseteq \{0,1\}^*$ is an* ensemble of languages *if there exists a polynomial p_L such that $|x| \leq p_L(t)$ for any $t \in \mathbb{N}$ and any $x \in L_t$. We identify an ensemble $L \subseteq \{0,1\}^*$ of languages with a family $\{L_t\}_{t\in\mathbb{N}}$.*

The language compression theorem can be stated as follows.

Theorem 16 (Time-Bounded Language Compression [Hir21a])**.** *Let A be an oracle and $L = \{L_t\}_{t\in\mathbb{N}} \in \mathsf{NP}^A$ be an ensemble of languages. If $\mathsf{DistNP}^A \subseteq \mathsf{AvgP}$, then there exists a polynomial p such that*

$$\mathrm{K}^{p(t)}(x) \leq \log|L_t| + \log p(t)$$

for every $(x, 1^t) \in L$.

We present an "algorithmic" proof of the language compression theorem, which generalizes the non-black-box reduction of Theorem 11. Instead of just showing Theorem 16, we construct a polynomial-time algorithm that accepts $x \in L_t$ and rejects any string x such that $\mathrm{K}^{\mathsf{poly}(t)}(x) > \log|L_t| + O(\log t)$. Formally:

Theorem 17 (Algorithmic Language Compression [Hir21a])**.** *Let A be an oracle and $L = \{L_t\}_{t\in\mathbb{N}} \in \mathsf{NP}^A$ be an ensemble of languages. If $\mathsf{DistNP}^A \subseteq \mathsf{AvgP}$, then there exists a polynomial p such that a promise problem $\Pi = (\Pi_{\mathrm{Yes}}, \Pi_{\mathrm{No}})$ defined as*

$$\Pi_{\mathrm{Yes}} := \left\{(x, 1^t, 1^k) \;\middle|\; x \in L_t\right\},$$
$$\Pi_{\mathrm{No}} := \left\{(x, 1^t, 1^k) \;\middle|\; \mathrm{K}^{p(t)}(x) > k + \log p(t), k \geq \log|L_t| + 1\right\}$$

is in pr-P.

The language compression theorem follows from the disjointness of Π_{Yes} and Π_{No}.

Proof of Theorem 16 from Theorem 17. The existence of an algorithm that separates Π_{Yes} from Π_{No} implies that $\Pi_{\mathrm{Yes}} \cap \Pi_{\mathrm{No}} = \emptyset$. Consider any string $x \in L_t$. Let $k := \log|L_t| + 1$. Since $(x, 1^t, 1^k) \in \Pi_{\mathrm{Yes}}$, we obtain $(x, 1^t, 1^k) \notin \Pi_{\mathrm{No}}$, which implies that $\mathrm{K}^{p(t)}(x) \leq k + \log p(t) = \log|L_t| + 1 + \log p(t)$. □

Now we present the algorithmic proof of the language compression theorem. The proof is quite similar to Theorem 11: For an errorless heuristic B for some distributional problem in DistNP^A, we consider a randomized non-black-box reduction B' that outputs 1 on input x iff $B(\mathrm{DP}_k(x; z)) \in \{1, \perp\}$ for a random choice of z.

Proof of Theorem 17. Let $L' = \left\{(\mathrm{DP}_k(x; z), 1^n, 1^t, 1^k) \;\middle|\; x \in L_t \cap \{0,1\}^n\right\}$. We claim that $L' \in \mathsf{NP}^A$. Note that $(w, 1^n, 1^t, 1^k) \in L'$ if and only if there exist $x \in \{0,1\}^n$, a certificate y for $x \in L_t$, and $z \in \{0,1\}^{n\cdot k}$ such that $w = \mathrm{DP}_k(x; z)$, which can be verified in polynomial time; thus, $L' \in \mathsf{NP}^A$. Consider a distribution $\mathcal{D} = \{\mathcal{D}_{n,t,k}\}_{n,t,k\in\mathbb{N}}$ such that $\mathcal{D}_{n,t,k}$ picks $w \sim \{0,1\}^{nk+k}$ and outputs $(w, 1^n, 1^t, 1^k)$. Since $(L', \mathcal{D}) \in \mathsf{DistNP}^A \subseteq \mathsf{AvgP}$, there exists an errorless heuristic B that solves L' with failure probability at most $\frac{1}{4}$.

We first claim that the number of YES instances in L' is relatively small. For each n, t, and $k \in \mathbb{N}$, we have

$$\begin{aligned}
&\Pr_{w\sim\{0,1\}^{nk+k}}\left[(w, 1^n, 1^t, 1^k) \in L'\right] \\
&= \Pr_{w\sim\{0,1\}^{nk+k}}\left[\exists x \in L_t \cap \{0,1\}^n, \exists z \in \{0,1\}^{nk}, w = \mathrm{DP}_k(x; z)\right] \\
&\leq |L_t \cap \{0,1\}^n| \cdot 2^{nk} \cdot 2^{-nk-k} \leq |L_t| \cdot 2^{-k}. \qquad (2)
\end{aligned}$$

We present a randomized algorithm B' that solves the promise problem Π defined in Theorem 17. On input $(x, 1^t, 1^k)$, the algorithm B' lets $n := |x|$ and picks $z \sim \{0,1\}^{nk}$ randomly, and accepts if and only if $B(\mathrm{DP}_k(x; z), 1^n, 1^t, 1^k) \in \{1, \perp\}$. We claim the correctness of B' below. Let p_L be the polynomial of Definition 15.

Claim 18. *For all large $t \in \mathbb{N}$ and every $n \leq p_L(t)$, the following hold for every $x \in \{0,1\}^n$.*

1. *If $(x, 1^t, 1^k) \in \Pi_{\mathrm{YES}}$, then $\Pr_z[B'(x, 1^t, 1^k; z) = 1] = 1$.*
2. *If $(x, 1^t, 1^k) \in \Pi_{\mathrm{No}}$, then $\Pr_z[B'(x, 1^t, 1^k; z) = 1] < \frac{7}{8}$.*

Proof. Assume that $x \in L_t$. By the definition of L', we have $(\mathrm{DP}_k(x; z), 1^n, 1^t, 1^k) \in L'$ for every z. Since B is an errorless heuristic, we obtain $B(\mathrm{DP}_k(x; z), 1^n, 1^t, 1^k) \in \{1, \perp\}$, which implies that $B'(x, 1^t, 1^k; z) = 1$. This completes the proof of Item 1.

To prove Item 2 by way of contradiction, we assume that $\Pr_z[B'(x, 1^t, 1^k; z) = 1] \geq \frac{7}{8}$; that is,

$$\Pr_z\left[B(\mathrm{DP}_k(x; z), 1^n, 1^t, 1^k) \in \{1, \perp\}\right] \geq \frac{7}{8}. \qquad (3)$$

On the other hand,

$$\begin{aligned}
\Pr_w\left[B(w, 1^n, 1^t, 1^k) \in \{1, \perp\}\right] &\leq \Pr_w\left[L'(w, 1^n, 1^t, 1^k) = 1\right] + \Pr_w\left[B(w, 1^n, 1^t, 1^k) = \perp\right] \\
&\leq |L_t| \cdot 2^{-k} + \frac{1}{4}, \qquad (4)
\end{aligned}$$

where the last inequality uses Eq. (2). Since $(x, 1^t, 1^k) \in \Pi_{\mathrm{No}}$, we have $|L_t| \cdot 2^{-k} \leq \frac{1}{2}$. By Eqs. (3) and (4), we obtain

$$\Pr_z\left[B(\mathrm{DP}_k(x; z), 1^n, 1^t, 1^k) \in \{1, \perp\}\right] - \Pr_w\left[B(w, 1^n, 1^t, 1^k) \in \{1, \perp\}\right] \geq \frac{7}{8} - \left(\frac{1}{2} + \frac{1}{4}\right) = \frac{1}{8}.$$

By Lemma 12, we obtain $\mathrm{K}^{p(t)}(x) \leq k + \log p(t)$ for some polynomial p. This is a contradiction to the assumption that $(x, 1^t, 1^k) \in \Pi_{\mathrm{No}}$. ⋄

It follows from Claim 18 that $\Pi \in$ pr-coRP. Using Theorem 3, we conclude that $\Pi \in$ pr-BPP = pr-P. □

The algorithmic language compression theorem generalizes the non-black-box reduction of Theorem 11. By algorithmically compressing an ensemble of languages $L_{t,s} := \{x \in \{0,1\}^* \mid \mathrm{K}^t(x) \leq s\}$, it can be shown that GapMINKT $\in$ P. More generally:

Corollary 19 ([Hir20b])**.** *Let A be an oracle. If* $\mathrm{DistNP}^A \subseteq \mathrm{AvgP}$*, then* $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K}) \in \mathrm{P}$.

Proof. Consider an ensemble $L = \{L_{\langle n,t,s\rangle}\}_{n,t,s\in\mathbb{N}}$ of languages defined as[10]

$$L_{\langle n,t,s\rangle} := \left\{x \in \{0,1\}^n \;\middle|\; \mathrm{K}^{t,A}(x) \leq s\right\}.$$

Observe that $|L_{\langle n,t,s\rangle}| \leq 2^{s+1}$ by Fact 5. It is easy to observe that $L \in \mathsf{NP}^A$. Applying Theorem 17 to L, we obtain a polynomial-time algorithm that solves the promise problem $(\Pi_{\text{YES}}, \Pi_{\text{No}})$ such that

$$\begin{aligned}\Pi_{\text{YES}} &:= \left\{(x, 1^{\langle n,t,s\rangle}, 1^k) \;\middle|\; \mathrm{K}^{t,A}(x) \leq s\right\},\\ \Pi_{\text{No}} &:= \left\{(x, 1^{\langle n,t,s\rangle}, 1^k) \;\middle|\; \mathrm{K}^{p(n,t,s)}(x) > k + \log p(n,t,s), k \geq \log\left|L_{\langle n,t,s\rangle}\right| + 1\right\}\end{aligned}$$

for some polynomial p. $\mathrm{Gap}_\tau(\mathrm{K}^A \text{ vs } \mathrm{K})$ is reducible to this promise problem via the reduction that maps $(x, 1^t, 1^s)$ to $(x, 1^{\langle |x|,t,s\rangle}, 1^{s+2})$ for some polynomial τ. □

The following lemma shows that any string that can be efficiently compressed with some PH oracle can also be compressed without the oracle if $\mathsf{DistPH} \subseteq \mathsf{AvgP}$.

Lemma 20 ([Hir20b])**.** *Let A be an oracle. If* $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K}) \in \mathsf{P}$*, then there exists a polynomial p such that, for every $x \in \{0,1\}^*$ and every $t \geq |x|$,*

$$\mathrm{K}^{p(t)}(x) \leq \mathrm{K}^{t,A}(x) + \log p(t).$$

Proof. Let $(\Pi_{\text{YES}}, \Pi_{\text{No}}) := \mathrm{Gap}_\tau(\mathrm{K}^A \text{ vs } \mathrm{K})$. The hypothesis implies that $\Pi_{\text{YES}} \cap \Pi_{\text{No}} = \emptyset$. Since $(x, 1^t, 1^s) \in \Pi_{\text{YES}}$ for $s := \mathrm{K}^{t,A}(x)$, we obtain $(x, 1^t, 1^s) \notin \Pi_{\text{No}}$, which implies that $\mathrm{K}^{\tau(|x|,t)}(x) \leq s + \log \tau(|x|,t) = \mathrm{K}^{t,A}(x) + \log \tau(|x|,t)$. □

7 Symmetry of Information

Yet another fundamental theorem of Kolmogorov complexity is *symmetry of information*, which was established by Kolmogorov and Levin [ZL70]. It states that for every $x \in \{0,1\}^*$ and $y \in \{0,1\}^*$,

$$\mathrm{K}(x \mid y) + \mathrm{K}(y) \lesssim \mathrm{K}(x,y) \lesssim \mathrm{K}(y \mid x) + \mathrm{K}(x),$$

where "$\lesssim$" indicates that the inequality holds up to an additive logarithmic term. Note that the second inequality is trivial because the pair of strings (x,y) can be computed by combining a program of size $\mathrm{K}(x)$ that prints x with a program of size $\mathrm{K}(y \mid x)$ that prints y given x as input. The highly non-trivial part of symmetry of information is the first inequality. Here, we consider a time-bounded analogue of symmetry of information.

Definition 21. Symmetry of information for time-bounded Kolmogorov complexity (SoI) *refers to the following hypothesis: There exists a polynomial p such that for any strings $x \in \{0,1\}^*$ and $y \in \{0,1\}^*$, for every $t \geq |x| + |y|$,*

$$\mathrm{K}^{p(t)}(x \mid y) + \mathrm{K}^{p(t)}(y) - \log p(t) \leq \mathrm{K}^t(x,y). \tag{SoI}$$

[10] We include n in the parameter so that L is an ensemble of languages.

Longpré and Watanabe [LW95] showed that $\mathsf{P} = \mathsf{NP}$ implies SoI, and that SoI implies the non-existence of one-way functions. In terms of Impagliazzo's five world, SoI holds in Algorithmica, while SoI does not hold in Minicrypt. It has been a long-standing open question to determine whether SoI holds in Heuristica or Pessiland. Recently, building on [Hir21a], this open question was independently resolved by [Hir21b; GK22]: SoI holds in Heuristica.

Theorem 22 ([Hir21b; GK22])**.** *If* $\mathsf{DistNP} \subseteq \mathsf{AvgP}$*, then SoI holds.*

Proof. Let $\widetilde{\mathrm{K}}$ be the polynomial-time algorithm of Fact 7, which satisfies the property that there exists a polynomial p such that for every $x \in \{0,1\}^*$ and every $t \geq |x|$,

$$\mathrm{K}^{p(t)}(x) - \log p(t) \leq \widetilde{\mathrm{K}}(x; 1^t) \leq \mathrm{K}^t(x) \tag{5}$$

Fix strings $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$ and an integer $t \geq n + m$. The proof of SoI is given by analyzing the following three values for $z \sim \{0,1\}^{nk}$, $w \sim \{0,1\}^{nk+k}$, $z' \sim \{0,1\}^{mk}$, and $w' \sim \{0,1\}^{mk+k}$:

$$\begin{array}{lccc} \widetilde{\mathrm{K}}(& \mathrm{DP}_k(x;z), & \mathrm{DP}_\ell(y;z') & ;1^{t'}), \\ \widetilde{\mathrm{K}}(& w, & \mathrm{DP}_\ell(y;z') & ;1^{t'}), \\ \widetilde{\mathrm{K}}(& w, & w' & ;1^{t'}), \end{array}$$

where $t' = t^{O(1)}$, $k \approx \mathrm{K}^t(x,y) - \ell$, and $\ell \approx \mathrm{K}^{\mathsf{poly}(t)}(y)$ are parameters chosen later.

First, observe that Fact 5 implies that $\mathrm{K}(w,w') \geq |w| + |w'| - 2$ with probability at least $\frac{3}{4}$. Let $\theta := |w| + |w'| - 2 - \log p(t')$; using Eq. (5), we obtain

$$\Pr_{w,w'}\left[\widetilde{\mathrm{K}}(w,w';1^{t'}) \geq \theta\right] \geq \frac{3}{4}. \tag{6}$$

Next, we set the parameter ℓ to be $\mathrm{K}^{p'(t)}(y) - \log p'(t) - 1$, where p' is some large polynomial. Consider a randomized circuit D that takes w' as input as well as random bits w and outputs 1 if and only if $\widetilde{\mathrm{K}}(w,w';1^{t'}) \geq \theta$. By the contrapositive of Lemma 12, $\mathrm{DP}_\ell(y;\text{-})$ is a pseudorandom generator secure against D; i.e., D cannot distinguish $\mathrm{DP}_\ell(y;z')$ and w' in the sense that

$$\left|\Pr_{w,z'}\left[\widetilde{\mathrm{K}}(w,\mathrm{DP}_\ell(y;z');1^{t'}) \geq \theta\right] - \Pr_{w,w'}\left[\widetilde{\mathrm{K}}(w,w';1^{t'}) \geq \theta\right]\right| < \frac{1}{4},$$

which, together with Eq. (6), implies that

$$\Pr_{w,z'}\left[\widetilde{\mathrm{K}}(w,\mathrm{DP}_\ell(y;z');1^{t'}) \geq \theta\right] \geq \frac{1}{2}.$$

Finally, we compare $\widetilde{\mathrm{K}}(w,\mathrm{DP}_\ell(y;z');1^{t'})$ with $\widetilde{\mathrm{K}}(\mathrm{DP}_k(x;z),\mathrm{DP}_\ell(y;z');1^{t'})$. On one hand, since $|w| = |z| + k$ and $|w'| = |z'| + \ell$, we have

$$\Pr_{w,z'}\left[\widetilde{\mathrm{K}}(w,\mathrm{DP}_\ell(y;z');1^{t'}) \geq |z| + |z'| + k + \ell - 2 - \log p(t')\right] \geq \frac{1}{2}. \tag{7}$$

On the other hand, observe that for some $t' := \mathsf{poly}(t)$,

$$\widetilde{\mathrm{K}}(\mathrm{DP}_k(x;z),\mathrm{DP}_\ell(y;z');1^{t'}) \leq \mathrm{K}^{t'}(\mathrm{DP}_k(x;z),\mathrm{DP}_\ell(y;z')) \leq \mathrm{K}^t(x,y) + |z| + |z'| + O(\log n)$$

holds because the strings $\mathrm{DP}_k(x;z)$ and $\mathrm{DP}_\ell(y;z')$ can be computed from k, ℓ, z, z', and a program of size $\mathrm{K}^t(x,y)$ that outputs (x,y) in time t. We now set $k := \mathrm{K}^t(x,y) - \ell + O(\log t)$ so that

$$\Pr_{z,z'}\left[\widetilde{\mathrm{K}}(\mathrm{DP}_k(x;z), \mathrm{DP}_\ell(y;z'); 1^{t'}) < |z| + |z'| + k + \ell - 2 - \log p(t')\right] = 1. \tag{8}$$

Let D_y be a randomized circuit that takes an input w and random bits z' and outputs 1 if and only if $\widetilde{\mathrm{K}}(w, \mathrm{DP}_\ell(y;z'); 1^{t'}) < |z| + |z'| + k + \ell - 2 - \log p(t')$. It follows from Eqs. (7) and (8) that

$$\Pr_{z,z'}\left[D_y(\mathrm{DP}_k(x;z);z') = 1\right] - \Pr_{w,z'}\left[D_y(w;z') = 1\right] \geq 1 - \frac{1}{2} = \frac{1}{2}.$$

Using Lemma 12, we obtain that

$$\mathrm{K}^{\mathsf{poly}(t)}(x \mid D_y) \leq k + O(\log t) = \mathrm{K}^t(x,y) - \mathrm{K}^{p'(t)}(y) + O(\log t).$$

It follows that for some large polynomial q,

$$\begin{aligned}\mathrm{K}^{q(t)}(x \mid y) \leq \mathrm{K}^t(x \mid D_y) + \mathrm{K}^t(D_y \mid y) + O(1) &\leq \mathrm{K}^t(x,y) - \mathrm{K}^{p'(t)}(y) + O(\log t)\\ &\leq \mathrm{K}^t(x,y) - \mathrm{K}^{q(t)}(y) + \log q(t)\end{aligned}$$

as desired. □

We mention that SoI holds if $\mathrm{Gap}(\mathrm{K}^{\mathrm{SAT}}$ vs K) is easy because $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ follows from $\mathrm{Gap}(\mathrm{K}^{\mathrm{SAT}}$ vs K) $\in \mathsf{P}$ [Hir20a].

Corollary 23. *If* $\mathrm{Gap}(\mathrm{K}^{\mathrm{SAT}}$ vs K) $\in \mathsf{P}$*, then SoI holds.*

We conclude this section by raising an open question.

Open Question 24. Does SoI hold in Pessiland? That is, does the non-existence of one-way functions imply SoI?

8 Universal Heuristic Scheme

Antunes and Fortnow [AF09] showed that the running time of a heuristic scheme that works with respect to every polynomial-time-samplable distribution can be characterized by using the notion of computational depth: Under a plausible assumption, $\{L\} \times \mathrm{PSAMP} \subseteq \mathsf{AvgP}$ if and only if there exists an algorithm S that decides L on input x in time $2^{O(\mathrm{cd}^{\mathsf{poly}(|x|)}(x)+\log|x|)}$, i.e., S runs in exponential time in the *time-unbounded* computational depth $\mathrm{cd}^{\mathsf{poly}(|x|),\infty}(x)$. Here, we study a faster algorithm that runs in exponential time in the *time-bounded* computational depth, which we call a universal heuristic scheme.

A universal heuristic scheme for L is an algorithm that takes an additional parameter t and decides L on input x in time $2^{O(\mathrm{cd}^{t,p(t)}(x)+\log t)}$. More formally:

Definition 25 (Universal Heuristic Scheme)**.** *An algorithm S is said to be a* universal heuristic scheme *for a language L if there exists a polynomial p such that for every x and every $t \geq p(|x|)$, the algorithm S outputs $L(x)$ on input (x,t) in time $2^{O(\mathrm{cd}^{t,p(t)}(x))} \cdot t^{O(1)}$.*

It is not hard to see that the existence of a universal heuristic scheme implies an algorithm that runs in time $2^{O(n/\log n)}$ on inputs of length n.

Theorem 26 ([Hir21a])**.** *If there exists a universal heuristic scheme S for a language L, then $L \in \mathsf{DTIME}\left(2^{O(n/\log n)}\right)$.*

Proof. Let p be the polynomial in Definition 25. We present an algorithm A that solves L on inputs of length n. Let x be an input of length n. Let I be a parameter chosen later, and let $k := 2n/I$. Define $t_i := p^{(i)}(n)$ for each $i \in [I]$. The algorithm A simulates the universal heuristic scheme S on inputs $(x, 1^{t_1}), (x, 1^{t_2}), \ldots, (x, 1^{t_I})$ in parallel. If one of the simulations halts with output being $b \in \{0, 1\}$, then A outputs b and halts. In other words, $A(x)$ is defined to be $S(x, 1^{t_i})$, where $i \in [I]$ is the index such that the running time of $S(x, 1^{t_i})$ is the smallest among $i \in [I]$.

We claim that A solves L in time $2^{O(n/\log n)}$. The correctness of A follows from the definition that S outputs the correct answer $L(x)$ when it halts. To bound the running time of A, consider the following telescoping sum:

$$\sum_{i=1}^{I} \mathrm{cd}^{t_i,t_{i+1}}(x) = \mathrm{cd}^{t_1,t_{I+1}}(x) \leq n + O(1),$$

where the last inequality follows because $\mathrm{K}^{t_1}(x) \leq n + O(1)$ and $\mathrm{K}^{t_{I+1}}(x) \geq 0$. By taking the minimum term of the left-hand side, we obtain

$$I \cdot \min\{\mathrm{cd}^{t_i,t_{i+1}}(x) \;\big|\; i \in [I]\} \leq n + O(1),$$

from which it follows that there exists $i \in [I]$ such that $\mathrm{cd}^{t_i,t_{i+1}}(x) \leq n/I + O(1) \leq k$. The running time of S on input $(x, 1^{t_i})$ is at most $2^{O\left(\mathrm{cd}^{t_i,t_{i+1}}(x)+\log t_i\right)}$. Let $c > 1$ be a constant such that $p(n) \leq n^c$ for all large n; then, we have $t_i = p^{(i)}(n) \leq n^{c^i}$ for all large n and every i. In particular, if we choose $I = \epsilon \log n$ for some small constant $\epsilon > 0$, we obtain $t_i \leq 2^{c^I \log n} \leq 2^{\sqrt{n}}$. We conclude that the running time of S on input $(x, 1^{t_i})$ is at most $2^{O\left(\mathrm{cd}^{t_i,t_{i+1}}(x)+\log t_i\right)} \leq 2^{O(n/\log n)}$. Thus, A also runs in time at most $2^{O(n/\log n)}$. □

Remark 27. Instead of the worst-case algorithm A, it is possible to construct an efficient heuristic algorithm using Theorem 9. Specifically, one can construct an algorithm A' such that given parameters $I \in \mathbb{N}$ and $\delta^{-1} \in \mathbb{N}$, with probability at least $1 - \delta$ over an instance x drawn from a distribution $\mathcal{D} \in \mathrm{PSAMP}$, the algorithm A' decides L on input x in time $2^{O\left((\log 1/\delta)/I + c^I \log n\right)}$ for some constant c.

In light of Theorem 26, in order to prove Theorem 2, it suffices to construct a universal heuristic scheme for each language in PH. It is easier to construct a weak variant of universal heuristic schemes, which we introduce below.

Definition 28 ([Hir21a])**.** *A* weak universal heuristic scheme *for a language L is a polynomial-time algorithm S such that, for some polynomial p, for any $n \in \mathbb{N}$, any $t \geq p(n)$, and any $x \in \{0, 1\}^n$, if $\mathrm{cd}^{t,p(t)}(x) \leq k$, then $S(x, 1^t, 1^{2^k}) = L(x)$.*

We also introduce a strong variant that can check an input x is an easy instance or not in polynomial time.

Definition 29 ([Hir21b])**.** *A* strong universal heuristic scheme *for a language L is a pair (S, C) of polynomial-time algorithms such that, for some polynomial p, for any $n \in \mathbb{N}$, any $t \geq p(n)$, and any $x \in \{0, 1\}^n$,*

1. *if $\mathrm{cd}^{t,p(t)}(x) \leq k$, then $C(x, 1^t, 1^k) = 1$, and*

2. *if* $C(x, 1^t, 1^k) = 1$*, then* $S(x, 1^t, 1^{2^k}) = L(x)$.

S and C are referred to as a solver *and a* checker*, respectively.*

These different notions of universal heuristic schemes are in fact equivalent in Heuristica.

Theorem 30 ([Hir21a; Hir21b])**.** *Assume that* GapMINKT $\in$ P*. Then, the following are equivalent for any language L:*

1. *There exists a strong universal heuristic scheme for L.*
2. *There exists a universal heuristic scheme for L.*
3. *There exists a weak universal heuristic scheme for L.*

Moreover, under the stronger assumption that DistNP $\subseteq$ AvgP*, the following statement is also equivalent.*

4. $\{L\} \times \mathrm{PSAMP} \subseteq \mathsf{Avg_P P}$.

Proof. It is easy to show Item 1 $\Rightarrow$ Item 2 $\Rightarrow$ Item 3. Let (S, C) be a strong universal heuristic scheme. We define a universal heuristic scheme S' as follows: Given $(x, 1^t)$ as input, S' finds the minimum $k \in \mathbb{N}$ such that $C(x, 1^t, 1^k) = 1$ and outputs $S(x, 1^t, 1^{2^k})$. Note that $k \leq \mathrm{cd}^{t,p(t)}(x)$ by the property of the string universal heuristic scheme (S, C); thus, the running time of S' is bounded by $2^{O(k+\log t)} \leq 2^{O(\mathrm{cd}^{t,p(t)}(x)+\log t)}$. Next, we show that any universal heuristic scheme S' can be converted into a weak universal heuristic scheme S''. We define S'' as follows: Given $(x, 1^t, 1^{2^k})$ as input, S'' simulates S' on input $(x, 1^t)$ up to $2^{O(k+\log t)}$ steps and outputs the output of S' if S' halts; if S' does not halt, the output is defined to be, e.g., 0. Then, S'' is a polynomial-time algorithm. The correctness of S'' follows from the definition that S' halts in time $2^{O(\mathrm{cd}^{t,p(t)}(x)+\log t)} \leq 2^{O(k+\log t)}$ if $\mathrm{cd}^{t,p(t)}(x) \leq k$.

We now show Item 3 $\Rightarrow$ Item 1. Given a weak universal heuristic scheme S for L, we need to construct a checker C. The idea of constructing C is to estimate the time-bounded computational depth of an input by using the polynomial-time algorithm for GapMINKT whose existence is guaranteed by Theorem 11. Let $\widetilde{\mathrm{K}}$ be the polynomial-time algorithm of Fact 7 such that for every $x \in \{0,1\}^*$ and every $t \geq |x|$,[11]

$$\mathrm{K}^{p(t)}(x) - \log p(t) \leq \widetilde{\mathrm{K}}(x, 1^t) \leq \mathrm{K}^t(x).$$

Observe that

$$\mathrm{cd}^{p(t),p^{(2)}(t)}(x) - \log p(t) \leq \widetilde{\mathrm{K}}(x, 1^t) - \widetilde{\mathrm{K}}(x, 1^{p^{(2)}(t)}) \leq \mathrm{cd}^{t,p^{(3)}(t)}(x) + \log p^{(3)}(t). \tag{9}$$

We define a checker C as follows: $C(x, 1^t, 1^k) = 1$ if and only if $\widetilde{\mathrm{K}}(x, 1^t) - \widetilde{\mathrm{K}}(x, 1^{p^{(2)}(t)}) \leq k + \log p^{(3)}(t)$. We define a solver S' so that $S'(x, 1^t, 1^{2^k}) := S'(x, 1^{p(t)}, 1^{2^{k'}})$, where $k' := k + \log p(t) + \log p^{(3)}(t)$.

Below, we claim that (S', C) is a strong universal heuristic scheme by showing that it satisfies the two properties of Definition 29.

[11] We may assume without loss of generality that the polynomial p in Fact 7 is the same polynomial with Definition 28.

1. If $\mathrm{cd}^{t,p^{(3)}(t)}(x) \leq k$, then by the upper bound of Eq. (9), we have $C(x, 1^t, 1^k) = 1$.
2. If $C(x, 1^t, 1^k) = 1$, then by the definition of C and by the lower bound of Eq. (9), we obtain $\mathrm{cd}^{p(t),p^{(2)}(t)}(x) \leq k + \log p(t) + \log p^{(3)}(t) = k'$. It follows from the property of the weak heuristic scheme S that $S'(x, 1^t, 1^{2^k}) = S(x, 1^{p(t)}, 1^{2^{k'}}) = L(x)$.

The implication from Item 1 to 4 can be proved using Theorem 9. The converse can be proved by considering the "time-bounded universal distribution". We omit the detailed proof of the equivalence between Item 1 and Item 4, which can be found in [Hir21a]. □

9 Constructing Universal Heuristic Schemes

We now use SoI to construct a weak universal heuristic scheme for every language in PH. For simplicity, we first construct a weak universal heuristic for every language in NP.

Theorem 31. *If* $\mathrm{Gap}(\mathrm{K}^{\mathrm{SAT}}\text{ vs }\mathrm{K}) \in \mathsf{P}$*, then for every language* $L \in \mathsf{NP}$*, there exists a weak universal heuristic scheme for* L*.*

Proof. Let V be a polynomial-time verifier for $L \in \mathsf{NP}$. For every $x \in L$, let y_x denote the lexicographically first certificate y_x such that $V(x, y_x) = 1$. The following claim is the key to the construction of a weak universal heuristic scheme.

Claim 32. *There exists a polynomial* q *such that for every* $x \in L$ *and every* $t \geq |x|$*,*

$$\mathrm{K}^{q(t)}(y_x \mid x) \leq \mathrm{cd}^{t,q(t)}(x) + \log q(t).$$

Proof. Note that SoI holds because of Corollary 23. Using SoI, we obtain

$$\begin{aligned}
&\mathrm{K}^{p^{(3)}(t)}(y_x \mid x) \\
&\leq \mathrm{K}^{p^{(2)}(t)}(y_x, x) - \mathrm{K}^{p^{(3)}(t)}(x) + \log p^{(3)}(t) && \text{(by SoI)} \\
&\leq \mathrm{K}^{p(t),\mathrm{SAT}}(y_x, x) + \log p^{(2)}(t) - \mathrm{K}^{p^{(3)}(t)}(x) + \log p^{(3)}(t) && \text{(by Lemma 20)} \\
&\leq \mathrm{K}^{t}(x) - \mathrm{K}^{p^{(3)}(t)}(x) + O(\log p^{(3)}(t)) \\
&= \mathrm{cd}^{t,p^{(3)}(t)}(x) + O(\log p^{(3)}(t)),
\end{aligned}$$

where the last inequality holds because y_x can be computed from x in polynomial time given oracle access to SAT. The claim follows by letting $q(t) := p^{(3)}(t)^{O(1)}$. ◇

We now present a weak universal heuristic scheme S for L: The algorithm S takes $(x, 1^t, 1^{2^k})$ as input and computes the set Y of strings $y \in \{0,1\}^*$ such that there exists a program of length at most $k + \log q(t)$ that takes x as input and outputs y in time $q(t)$. In other words, we define

$$Y := \left\{ y \in \{0,1\}^* \;\middle|\; \mathrm{K}^{q(t)}(y \mid x) \leq k + \log q(t). \right\}$$

Note that $|Y| \leq 2^{k+\log q(t)+1}$ and Y can be computed in time $\mathsf{poly}(|x|, t, 2^k)$ by enumerating all the programs of length at most $k + \log q(t)$. The algorithm S outputs 1 if and only if there exists a string $y \in Y$ such that $V(x, y) = 1$. Clearly, S is a polynomial-time algorithm. We prove the correctness of S: It is evident that S does not err on any input $x \notin L$. Consider an input $x \in L$ such that $\mathrm{cd}^{t,q(t)}(x) \leq k$. Then, by Claim 32 we have $\mathrm{K}^{q(t)}(y_x) \leq k + \log q(t)$, which implies $y_x \in Y$ and thus S accepts. □

This enables us to complete a proof of a special case of Theorem 2.

Corollary 33. *If* $\mathsf{DistPH} \subseteq \mathsf{AvgP}$*, then* $\mathsf{NP} \subseteq \mathsf{DTIME}(2^{O(n/\log n)})$.

Proof. See Fig. 4. □

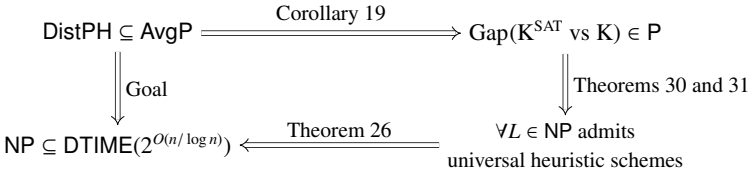


Figure 4: The proof of Corollary 33.

To extend Theorem 31 to all the levels of PH, we use an inductive argument that constructs weak universal heuristic schemes for the k-th level Σ_k^{p} of PH from weak universal heuristic schemes for $\Sigma_{k-1}^{\mathsf{p}}$.

Theorem 34. *Let* $k \in \mathbb{N}$*. If* $\mathsf{Dist}\Sigma_{k+1}^{\mathsf{p}} \subseteq \mathsf{AvgP}$*, then for every language* $L \in \Sigma_k^{\mathsf{p}}$*, there exists a weak universal heuristic scheme for* L*.*

Proof. We prove this by induction on $k \in \mathbb{N}$. The base case ($k = 0$) is trivial because every language $L \in \Sigma_0^{\mathsf{p}} = \mathsf{P}$ admits a weak universal heuristic scheme. Let $k \geq 1$. Let V be a language in Π_{k-1}^{p} such that $x \in L$ if and only if $V(x, y) = 1$ for some $y \in \{0,1\}^{\mathsf{poly}(|x|)}$. For every $x \in L$, let y_x be the lexicographically first string y such that $V(x, y) = 1$. Using the same proof idea of Claim 32, it is easy to prove the following.

Claim 35. *There exists a polynomial* q *such that for every* $x \in L$ *and every* $t \geq |x|$*,*

$$\mathrm{K}^{q(t)}(y_x \mid x) \leq \mathrm{cd}^{t,q(t)}(x) + \log q(t).$$

By the induction hypothesis, there exists a weak universal heuristic scheme S for $V \in \Pi_{k-1}^{\mathsf{p}}$. Let p be the polynomial in Definition 28. Using S, we now present a weak universal heuristic scheme S' for L: The algorithm S' takes $(x, 1^t, 1^{2^k})$ as input and computes the set

$$Y := \left\{ y \in \{0,1\}^* \;\middle|\; \mathrm{K}^{q(t)}(y \mid x) \leq k + \log q(t). \right\}$$

Note that $|Y| \leq 2^{k+\log q(t)+1}$ and Y can be computed in time $\mathsf{poly}(|x|, t, 2^k)$ by an exhaustive search. The algorithm S' outputs 1 if and only if there exists a string $y \in Y$ such that $S((x, y), 1^{t'}, 1^{2^{k'}}) = 1$, where $t' = t^{O(1)}$ and $k' = O(k + \log t)$ are parameters chosen later. Clearly, S' is a polynomial-time algorithm.

We claim the correctness of S'. Assume that $\mathrm{cd}^{t,2p(t')}(x) \leq k$. We claim that for some parameter $t' = q(t)^{O(1)}$ and for every $y \in Y$, the $(t', p(t'))$-time-bounded computational depth of (x, y) is at most k', which will imply that the output of the weak universal heuristic scheme S is correct on input (x, y). For every $y \in Y$, we have

$$\begin{aligned}
\mathrm{cd}^{t',p(t')}(x, y) &\leq \mathrm{K}^{q(t)}(x) + \mathrm{K}^{q(t)}(y \mid x) - \mathrm{K}^{p(t')}(x, y) + O(1) \\
&\leq \mathrm{cd}^{q(t),2p(t')}(x) + k + \log q(t) + O(1) \\
&\leq 2k + \log q(t) + O(1) =: k',
\end{aligned}$$

where the first inequality follows from the definition of time-bounded computational depth, the second inequality follows from the fact that $\mathrm{K}^{2p(t')}(x) \leq \mathrm{K}^{p(t')}(x, y) + O(1)$ and $y \in Y$, and the third inequality follows from the assumption that $\mathrm{cd}^{q(t),2p(t')}(x) \leq \mathrm{cd}^{t,q'(t)}(x) \leq k$. By the correctness of the weak universal heuristic scheme S, we obtain $S((x, y), 1^{t'}, 1^{2^{k'}}) = V(x, y)$. If $x \in L$, Claim 35 implies that $y_x \in Y$; thus, we have $S((x, y_x), 1^{t'}, 1^{2^{k'}}) = V(x, y_x) = 1$, which implies that S' outputs 1. If $x \notin L$, then $V(x, y) = 0$ for every string y; thus, we obtain $S((x, y), 1^{t'}, 1^{2^{k'}}) = V(x, y) = 0$, which implies that S' outputs 0. □

This completes the proof of the second item of Theorem 2, as shown in Fig. 5.

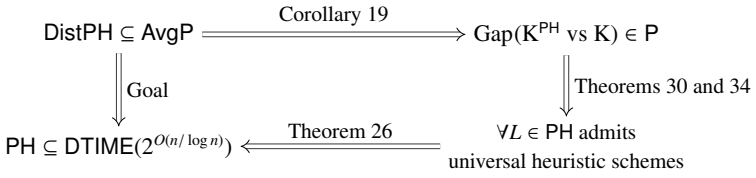


Figure 5: The proof of the second item of Theorem 2.

Finally, we construct universal heuristic schemes for UP.

Theorem 36. *If* DistNP ⊆ AvgP*, then for every language* $L \in \mathsf{UP}$*, there exists a weak universal heuristic scheme for* L*.*

Proof. Let V be a UP-type verifier for L; that is, for every $x \in L$, there exists a unique certificate y_x such that $V(x, y_x) = 1$. Consider an ensemble $L = \{L_{\langle n,t,s\rangle}\}_{n,t,s\in\mathbb{N}}$ of languages defined as

$$L_{\langle n,t,s\rangle} := \{(x, y) \mid x \in \{0, 1\}^n, V(x, y) = 1, \mathrm{K}^t(x) \leq s\}.$$

It is easy to observe that $L \in \mathsf{NP}$. Using that $L \in \mathsf{UP}$, we can observe that $|L_{\langle n,t,s\rangle}| \leq 2^{s+1}$ because the number of strings x such that $\mathrm{K}^t(x) \leq s$ is at most 2^{s+1} by Fact 5 and for each x there is at most one certificate y such that $V(x, y) = 1$. Applying Theorem 16 to L, there exists a polynomial p such that for every $t \geq n$, for every $(x, y_x) \in L_{\langle n,t,s\rangle}$ such that $s := \mathrm{K}^t(x)$, it holds that

$$\mathrm{K}^{p(t)}(x, y_x) \leq s + 1 + \log p(t) = \mathrm{K}^t(x) + \log 2p(t).$$

Since SoI follows from Theorem 22, we have

$$\mathrm{K}^{p^{(2)}(t)}(y_x \mid x) \leq \mathrm{K}^{p(t)}(x, y_x) - \mathrm{K}^{p^{(2)}(t)}(x) + \log p^{(2)}(t).$$

Combining these two inequalities, we obtain

$$\mathrm{K}^{p^{(2)}(t)}(y_x \mid x) \leq \mathrm{cd}^{t,p^{(2)}(t)}(x) + O(\log t). \tag{10}$$

We now present a weak universal heuristic scheme S for L. Given an input $(x, 1^t, 1^{2^k})$ such that $\mathrm{cd}^{t,p^{(2)}(t)}(x) \leq k$, the algorithm S computes the set

$$Y := \left\{y \in \{0, 1\}^* \;\middle|\; \mathrm{K}^{p^{(2)}(t)}(y \mid x) \leq k + O(\log t)\right\}$$

and accepts if and only if there exists $y \in Y$ such that $V(x, y) = 1$. The correctness of S follows from Eq. (10) because it implies $y_x \in Y$ for every $x \in L$. □

This enables us to complete the proof of Theorem 2.

Proof of the first item of Theorem 2. Assume $\mathsf{DistNP} \subseteq \mathsf{AvgP}$. By Theorem 36, every language $L \in \mathsf{UP}$ admits a *weak* universal heuristic scheme, which can be converted into a universal heuristic scheme by Theorem 30, from which we obtain $L \in \mathsf{DTIME}(2^{O(n/\log n)})$ by Theorem 26. □

In the original paper [Hir21a], a more general result than Theorem 36 was proved: $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ implies that $\mathsf{NP}_{\mathsf{sv}}$ admits universal heuristic schemes. Here, $\mathsf{NP}_{\mathsf{sv}}$ stands for *size-verifiable* NP and is the class of languages $L \in \mathsf{NP}$ such that some AM protocols can verify that the number of certificates for L is approximately small. The notion of size-verifiability was introduced by Akavia, Goldreich, Goldwasser, and Moshkovitz [AGGM06]. It is easy to observe that $\mathsf{UP} \subseteq \mathsf{FewP} \subseteq \mathsf{NP}_{\mathsf{sv}} \subseteq \mathsf{NP}$ and that $\mathsf{NP}_{\mathsf{sv}} = \mathsf{NP}$ if $\mathsf{NP} \subseteq \mathsf{coAM}$; however, the complexity of $\mathsf{NP}_{\mathsf{sv}}$ is not well understood. It is an interesting open problem to extend UP of Theorem 36 to NP.

Open Question 37. Does $\mathsf{NP} \not\subseteq \mathsf{DTIME}(2^{O(n/\log n)})$ imply $\mathsf{DistNP} \not\subseteq \mathsf{AvgP}$? In particular, is it possible to construct universal heuristic schemes for NP under the assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$?

10 Future Research Directions

We conclude this article by presenting three research directions which we believe are most promising and exciting.

The first research direction is to develop non-relativizing proof techniques. The only non-relativizing part of the proofs in this article is Theorem 3, which constructs a complexity-theoretic pseudorandom generator in Heuristica. In fact, there is a relativizing proof showing the existence of a pseudorandom generator from the stronger assumption that $\mathsf{DistP}^{\mathsf{NP}} \subseteq \mathsf{AvgP}$ [HN21]. Given that there is a quantitatively tight relativization barrier [HN21], we would need a non-relativizing proof technique to obtain better worst-case-to-average-case connections for NP or PH. A recent line of work [AHMPS08; HOS18; Ila20a; ILO20; Ila20b; ACMTV21; LP21; Hir21b] developed apparently non-relativizing proof techniques: Although Ko [Ko91] showed that GapMINKT cannot be shown to be NP-hard using a relativizing proof technique, NP-hardness of computing sublinear-time-bounded conditional Kolmogorov complexity is proved in [ACMTV21; LP21; Hir21b]. Note that NP-hardness of GapMINKT excludes Heuristica by Theorem 11; even NP-hardness of $\mathrm{GapMINKT}^{\mathsf{PH}}$ would significantly improve Theorem 2. Trying to prove NP-hardness of meta-computational problems would lead us to new non-relativizing proof techniques. A state-of-the-art result along this research line is due to Ilango [Ila20b; Ila21], who showed NP-hardness of variants of the Minimum Circuit Size Problem [KC00], which is another representative meta-computational problem.

The second research direction is to use meta-complexity to exclude Pessiland. Impagliazzo and Levin [IL90, Proposition 1] presented a proof sketch of the characterization of the existence of a one-way function: there exists no one-way function if and only if the randomized t-time-bounded Kolmogorov complexity of x can be approximated with high probability over a random input x drawn from any unknown t-time-samplable distribution. Their results can be seen as an approach toward excluding Pessiland: If the randomized t-time-bounded Kolmogorov complexity is NP-hard under t'-time randomized reductions for $t' \ll t$, then Pessiland does

not exist, i.e, (error-prone) average-case hardness of NP implies the existence of a one-way function. More recently, Liu and Pass [LP20] proved the equivalence between the non-existence of a one-way function and the existence of a randomized polynomial-time *error-prone* heuristic that computes $\mathrm{K}^t(x)$ with high probability over a random input x chosen from the uniform distribution. The main gap between these results and the results presented in this article is the difference between *error-prone* average-case complexity (denoted by HeurP in [BT06a]) and *errorless* average-case complexity (AvgP), which was recently investigated in [HS22a]. An intermediate statement is SoI, which is sandwiched between errorless heuristics for MINKT and error-prone heuristics for MINKT [Hir21b]. Nanashima [Nan21] showed that the existence of a one-way function follows from $\mathsf{NP} \not\subseteq \mathsf{BPP}$ (i.e., both Heuristica and Pessiland can be excluded) if there exists a randomized nonadaptive reduction from NP to the adversary of auxiliary-input hitting set generators. Note that the existence of a hitting set generator is equivalent to the hardness of GapMINKT, at least under non-black-box reductions (Proposition 13). Given these results, we conjecture that meta-complexity would play a central role in excluding Pessiland, as well as Heuristica.

The last research direction is to determine average-case complexity of natural distributional problems in DistNP. We started off this article by explaining that the motivation of studying average-case complexity is to understand the "real-life" complexity of (natural) distributional problems. The original motivation of Levin [Lev86], who laid the foundation of average-case complexity, was also the same and showed that the Tiling problem is DistNP-complete. However, a relatively few distributional problems in DistNP are shown to be DistNP-complete, compared to the highly successful theory of NP-completeness [Coo71; Lev73; Kar72]. The difficulty is that an average-case reduction disturbs the distribution of distributional problems, which makes it difficult to prove DistNP-completeness of distributional problems $(L, \mathcal{D})$ for natural distributions $\mathcal{D}$. We envision that meta-complexity could come to the rescue: Meta-complexity enables us to analyze *average-case* complexity through the lens of *worst-case* complexity, for which a reduction is easier to construct. [Hir20a; HS22b] showed that $(\mathrm{MINKT}^{\mathsf{PH}}, \mathcal{U})$ is DistPH-complete, which may be a step toward determining average-case complexity of natural distributional problems in DistPH or DistNP.

References

[ACMTV21] Eric Allender, Mahdi Cheraghchi, Dimitrios Myrisiotis, Harsha Tirumala, and Ilya Volkovich. "One-Way Functions and a Conditional Variant of MKTP". In: *Proceedings of the Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*. 2021, 7:1–7:19. DOI: 10.4230/LIPIcs.FSTTCS.2021.7.

[AF09] Luis Filipe Coelho Antunes and Lance Fortnow. "Worst-Case Running Times for Average-Case Algorithms". In: *Proceedings of the Conference on Computational Complexity (CCC)*. 2009, pp. 298–303. DOI: 10.1109/CCC.2009.12.

[AFMV06] Luis Antunes, Lance Fortnow, Dieter van Melkebeek, and N. V. Vinodchandran. "Computational depth: Concept and applications". In: *Theor. Comput. Sci.* 354.3 (2006), pp. 391–404. DOI: 10.1016/j.tcs.2005.11.033.

[AGGM06] Adi Akavia, Oded Goldreich, Shafi Goldwasser, and Dana Moshkovitz. "On basing one-way functions on NP-hardness". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 2006, pp. 701–710. DOI: 10.1145/1132516.1132614.

[AHMPS08] Eric Allender, Lisa Hellerstein, Paul McCabe, Toniann Pitassi, and Michael E. Saks. "Minimizing Disjunctive Normal Form Formulas and AC^0 Circuits Given a Truth Table". In: *SIAM J. Comput.* 38.1 (2008), pp. 63–84. DOI: 10.1137/060664537.

[All21] Eric Allender. "Vaughan Jones, Kolmogorov Complexity, and the new complexity landscape around circuit minimization". In: *New Zealand journal of mathematics* 52 (2021), pp. 585–604.

[AW09] Scott Aaronson and Avi Wigderson. "Algebrization: A New Barrier in Complexity Theory". In: *TOCT* 1.1 (2009), 2:1–2:54. DOI: 10.1145/1490270.1490272.

[BB15] Andrej Bogdanov and Christina Brzuska. "On Basing Size-Verifiable One-Way Functions on NP-Hardness". In: *Proceedings of the Theory of Cryptography Conference (TCC)*. 2015, pp. 1–6. DOI: 10.1007/978-3-662-46494-6_1.

[BCGL92] Shai Ben-David, Benny Chor, Oded Goldreich, and Michael Luby. "On the Theory of Average Case Complexity". In: *J. Comput. Syst. Sci.* 44.2 (1992), pp. 193–219. DOI: 10.1016/0022-0000(92)90019-F.

[BFL01] Harry Buhrman, Lance Fortnow, and Sophie Laplante. "Resource-Bounded Kolmogorov Complexity Revisited". In: *SIAM J. Comput.* 31.3 (2001), pp. 887–905. DOI: 10.1137/S009753979834388X.

[BFP05] Harry Buhrman, Lance Fortnow, and Aduri Pavan. "Some Results on Derandomization". In: *Theory Comput. Syst.* 38.2 (2005), pp. 211–227. DOI: 10.1007/s00224-004-1194-y.

[BGS75] Theodore P. Baker, John Gill, and Robert Solovay. "Relativizations of the P =? NP Question". In: *SIAM J. Comput.* 4.4 (1975), pp. 431–442. DOI: 10.1137/0204037.

[BLM05] Harry Buhrman, Troy Lee, and Dieter van Melkebeek. "Language compression and pseudorandom generators". In: *Computational Complexity* 14.3 (2005), pp. 228–255. DOI: 10.1007/s00037-005-0199-5.

[BRSV17] Marshall Ball, Alon Rosen, Manuel Sabin, and Prashant Nalini Vasudevan. "Average-case fine-grained hardness". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 2017, pp. 483–496. DOI: 10.1145/3055399.3055466.

[BRSV18] Marshall Ball, Alon Rosen, Manuel Sabin, and Prashant Nalini Vasudevan. "Proofs of Work From Worst-Case Assumptions". In: *Proceedings of the International Cryptology Conference (CRYPTO)*. 2018, pp. 789–819. DOI: 10.1007/978-3-319-96884-1_26.

[BT06a] Andrej Bogdanov and Luca Trevisan. "Average-Case Complexity". In: *Foundations and Trends in Theoretical Computer Science* 2.1 (2006). DOI: 10.1561/0400000004.

[BT06b] Andrej Bogdanov and Luca Trevisan. "On Worst-Case to Average-Case Reductions for NP Problems". In: *SIAM J. Comput.* 36.4 (2006), pp. 1119–1159. DOI: 10.1137/S0097539705446974.

[CHOPRS20] Lijie Chen, Shuichi Hirahara, Igor Carboni Oliveira, Ján Pich, Ninad Rajgopal, and Rahul Santhanam. "Beyond Natural Proofs: Hardness Magnification and Locality". In: *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*. 2020, 70:1–70:48. DOI: 10.4230/LIPIcs.ITCS.2020.70.

[CHV22] Lijie Chen, Shuichi Hirahara, and Neekon Vafa. "Average-case Hardness of NP and PH from Worst-case Fine-grained Assumptions". In: *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*. 2022, 67:1–67:17.

[Coo71] Stephen A. Cook. "The Complexity of Theorem-Proving Procedures". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 1971, pp. 151–158. DOI: 10.1145/800157.805047.

[DN92] Cynthia Dwork and Moni Naor. "Pricing via Processing or Combatting Junk Mail". In: *Proceedings of the International Cryptology Conference (CRYPTO)*. 1992, pp. 139–147. DOI: 10.1007/3-540-48071-4_10.

[FF93] Joan Feigenbaum and Lance Fortnow. "Random-Self-Reducibility of Complete Sets". In: *SIAM J. Comput.* 22.5 (1993), pp. 994–1005. DOI: 10.1137/0222061.

[FFLS94] Joan Feigenbaum, Lance Fortnow, Carsten Lund, and Daniel A. Spielman. "The Power of Adaptiveness and Additional Queries in Random-Self-Reductions". In: *Comput. Complex.* 4 (1994), pp. 158–174. DOI: 10.1007/BF01202287.

[GK22] Halley Goldberg and Valentine Kabanets. "A Simpler Proof of the Worst-Case to Average-Case Reduction for Polynomial Hierarchy via Symmetry of Information". In: *Electronic Colloquium on Computational Complexity (ECCC)* 007 (2022).

[GL89] Oded Goldreich and Leonid A. Levin. "A Hard-Core Predicate for all One-Way Functions". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 1989, pp. 25–32. DOI: 10.1145/73007.73010.

[Gol06] Oded Goldreich. "On Promise Problems: A Survey". In: *Theoretical Computer Science, Essays in Memory of Shimon Even*. 2006, pp. 254–290. DOI: 10.1007/11685654_12.

[GS87] Yuri Gurevich and Saharon Shelah. "Expected Computation Time for Hamiltonian Path Problem". In: *SIAM J. Comput.* 16.3 (1987), pp. 486–502. DOI: 10.1137/0216034.

[GVW11] Oded Goldreich, Salil P. Vadhan, and Avi Wigderson. "Simplified Derandomization of BPP Using a Hitting Set Generator". In: *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*. Springer, 2011, pp. 59–67. DOI: 10.1007/978-3-642-22670-0_8.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. "A Pseudorandom Generator from any One-way Function". In: *SIAM J. Comput.* 28.4 (1999), pp. 1364–1396. DOI: 10.1137/S0097539793244708.

[Hir18] Shuichi Hirahara. "Non-black-box Worst-case to Average-case Reductions within NP". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 2018, pp. 247–258.

[Hir20a] Shuichi Hirahara. "Characterizing Average-Case Complexity of PH by Worst-Case Meta-Complexity". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 2020, pp. 50–60.

[Hir20b] Shuichi Hirahara. "Non-Disjoint Promise Problems from Meta-Computational View of Pseudorandom Generator Constructions". In: *Proceedings of the Computational Complexity Conference (CCC)*. 2020, 20:1–20:47. DOI: 10.4230/LIPIcs.CCC.2020.20.

[Hir20c] Shuichi Hirahara. "Unexpected hardness results for Kolmogorov complexity under uniform reductions". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 2020, pp. 1038–1051. DOI: 10.1145/3357713.3384251.

[Hir20d] Shuichi Hirahara. "Unexpected Power of Random Strings". In: *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*. 2020, 41:1–41:13. DOI: 10.4230/LIPIcs.ITCS.2020.41.

[Hir21a] Shuichi Hirahara. "Average-case hardness of NP from exponential worst-case hardness assumptions". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 2021, pp. 292–302. DOI: 10.1145/3406325.3451065.

[Hir21b] Shuichi Hirahara. "Symmetry of Information in Heuristica". manuscript. 2021.

[HN21] Shuichi Hirahara and Mikito Nanashima. "On Worst-Case Learning in Relativized Heuristica". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 2021.

[HOS18] Shuichi Hirahara, Igor Carboni Oliveira, and Rahul Santhanam. "NP-hardness of Minimum Circuit Size Problem for OR-AND-MOD Circuits". In: *Proceedings of the Computational Complexity Conference (CCC)*. 2018, 5:1–5:31. DOI: 10.4230/LIPIcs.CCC.2018.5.

[HS21] Shuichi Hirahara and Nobutaka Shimizu. "Nearly Optimal Average-Case Complexity of Counting Bicliques Under SETH". In: *Proceedings of the Symposium on Discrete Algorithms (SODA)*. 2021, pp. 2346–2365. DOI: 10.1137/1.9781611976465.140.

[HS22a] Shuichi Hirahara and Rahul Santhanam. "Errorless versus Error-prone Average-case Complexity". In: *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*. 2022, 38:1–38:23.

[HS22b] Shuichi Hirahara and Rahul Santhanam. "Excluding PH Pessiland". In: *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*. 2022, 78:1–78:25.

[HW20] Shuichi Hirahara and Osamu Watanabe. "On Nonadaptive Security Reductions of Hitting Set Generators". In: *Proceedings of the Approximation, Randomization, and Combinatorial Optimization (APPROX/RANDOM)*. 2020, 15:1–15:14. DOI: 10.4230/LIPIcs.APPROX/RANDOM.2020.15.

[IL89] Russell Impagliazzo and Michael Luby. "One-way Functions are Essential for Complexity Based Cryptography (Extended Abstract)". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 1989, pp. 230–235. DOI: 10.1109/SFCS.1989.63483.

[IL90] Russell Impagliazzo and Leonid A. Levin. "No Better Ways to Generate Hard NP Instances than Picking Uniformly at Random". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 1990, pp. 812–821. DOI: 10.1109/FSCS.1990.89604.

[Ila20a] Rahul Ilango. "Approaching MCSP from Above and Below: Hardness for a Conditional Variant and $\mathsf{AC}^0[p]$". In: *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*. 2020, 34:1–34:26. DOI: 10.4230/LIPIcs.ITCS.2020.34.

[Ila20b] Rahul Ilango. "Constant Depth Formula and Partial Function Versions of MCSP are Hard". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 2020, pp. 424–433.

[Ila21] Rahul Ilango. "The Minimum Formula Size Problem is (ETH) Hard". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 2021.

[ILO20] Rahul Ilango, Bruno Loff, and Igor Carboni Oliveira. "NP-Hardness of Circuit Minimization for Multi-Output Functions". In: *Proceedings of the Computational Complexity Conference (CCC)*. 2020, 22:1–22:36. DOI: 10.4230/LIPIcs.CCC.2020.22.

[Imp11] Russell Impagliazzo. "Relativized Separations of Worst-Case and Average-Case Complexities for NP". In: *Proceedings of the Conference on Computational Complexity (CCC)*. 2011, pp. 104–114. DOI: 10.1109/CCC.2011.34.

[Imp95] Russell Impagliazzo. "A Personal View of Average-Case Complexity". In: *Proceedings of the Structure in Complexity Theory Conference*. 1995, pp. 134–147. DOI: 10.1109/SCT.1995.514853.

[IW97] Russell Impagliazzo and Avi Wigderson. "*P* = *BPP* if *E* Requires Exponential Circuits: Derandomizing the XOR Lemma". In: *Proceedings of the Symposium on the Theory of Computing (STOC)*. 1997, pp. 220–229. DOI: 10.1145/258533.258590.

[Kar72] Richard M. Karp. "Reducibility Among Combinatorial Problems". In: *Proceedings of a symposium on the Complexity of Computer Computations*. 1972, pp. 85–103.

[KC00] Valentine Kabanets and Jin-yi Cai. "Circuit minimization problem". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 2000, pp. 73–79. DOI: 10.1145/335305.335314.

[Ko91] Ker-I Ko. "On the Complexity of Learning Minimum Time-Bounded Turing Machines". In: *SIAM J. Comput.* 20.5 (1991), pp. 962–986. DOI: 10.1137/0220059.

[KS04] Johannes Köbler and Rainer Schuler. "Average-case intractability vs. worst-case intractability". In: *Inf. Comput.* 190.1 (2004), pp. 1–17. DOI: 10.1016/j.ic.2003.05.002.

[Lev73] Leonid Anatolevich Levin. "Universal sequential search problems". In: *Problemy Peredachi Informatsii* 9.3 (1973), pp. 115–116.

[Lev84] Leonid A. Levin. "Randomness Conservation Inequalities; Information and Independence in Mathematical Theories". In: *Information and Control* 61.1 (1984), pp. 15–37. DOI: 10.1016/S0019-9958(84)80060-1.

[Lev86] Leonid A. Levin. "Average Case Complete Problems". In: *SIAM J. Comput.* 15.1 (1986), pp. 285–286. DOI: 10.1137/0215020.

[LP20] Yanyi Liu and Rafael Pass. "On One-way Functions and Kolmogorov Complexity". In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 2020, pp. 1243–1254.

[LP21] Yanyi Liu and Rafael Pass. "On One-way Functions from NP-Complete Problems". In: *Electron. Colloquium Comput. Complex.* 28 (2021), p. 59.

[LV19] Ming Li and Paul M. B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications, 4th Edition*. Texts in Computer Science. Springer, 2019. ISBN: 978-3-030-11297-4. DOI: 10.1007/978-3-030-11298-1.

[LW95] Luc Longpré and Osamu Watanabe. "On Symmetry of Information and Polynomial Time Invertibility". In: *Inf. Comput.* 121.1 (1995), pp. 14–22. DOI: 10.1006/inco.1995.1120.

[MR07] Daniele Micciancio and Oded Regev. "Worst-Case to Average-Case Reductions Based on Gaussian Measures". In: *SIAM J. Comput.* 37.1 (2007), pp. 267–302. DOI: 10.1137/S0097539705447360.

[Nan21] Mikito Nanashima. "On Basing Auxiliary-Input Cryptography on NP-Hardness via Nonadaptive Black-Box Reductions". In: *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*. 2021, 29:1–29:15. DOI: 10.4230/LIPIcs.ITCS.2021.29.

[NW94] Noam Nisan and Avi Wigderson. "Hardness vs Randomness". In: *J. Comput. Syst. Sci.* 49.2 (1994), pp. 149–167. DOI: 10.1016/S0022-0000(05)80043-1.

[RR97] Alexander A. Razborov and Steven Rudich. "Natural Proofs". In: *J. Comput. Syst. Sci.* 55.1 (1997), pp. 24–35. DOI: 10.1006/jcss.1997.1494.

[Rud97] Steven Rudich. "Super-bits, Demi-bits, and NP/qpoly-natural Proofs". In: *Proceedings of the Randomization and Approximation Techniques in Computer Science (RANDOM/APPROX)*. 1997, pp. 85–93. DOI: 10.1007/3-540-63248-4_8.

[Sho99] Peter W. Shor. "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer". In: *SIAM Review* 41.2 (1999), pp. 303–332. DOI: 10.1137/S0036144598347011.

[Sip83] Michael Sipser. "A Complexity Theoretic Approach to Randomness". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 1983, pp. 330–335. DOI: 10.1145/800061.808762.

[Tre01] Luca Trevisan. "Extractors and pseudorandom generators". In: *J. ACM* 48.4 (2001), pp. 860–879. DOI: 10.1145/502090.502099.

[TUZ07] Amnon Ta-Shma, Christopher Umans, and David Zuckerman. "Lossless Condensers, Unbalanced Expanders, And Extractors". In: *Combinatorica* 27.2 (2007), pp. 213–240. DOI: 10.1007/s00493-007-0053-2.

[TV07] Luca Trevisan and Salil P. Vadhan. "Pseudorandomness and Average-Case Complexity Via Uniform Reductions". In: *Computational Complexity* 16.4 (2007), pp. 331–364. DOI: 10.1007/s00037-007-0233-x.

[Uma09] Christopher Umans. "Reconstructive Dispersers and Hitting Set Generators". In: *Algorithmica* 55.1 (2009), pp. 134–156. DOI: 10.1007/s00453-008-9266-z.

[Vad12] Salil P. Vadhan. "Pseudorandomness". In: *Foundations and Trends in Theoretical Computer Science* 7.1-3 (2012), pp. 1–336. DOI: 10.1561/0400000010.

[Vio05a] Emanuele Viola. "On Constructing Parallel Pseudorandom Generators from One-Way Functions". In: *Proceedings of the Conference on Computational Complexity (CCC)*. 2005, pp. 183–197. DOI: 10.1109/CCC.2005.16.

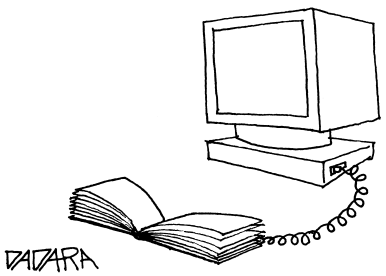
[Vio05b] Emanuele Viola. "The complexity of constructing pseudorandom generators from hard functions". In: *Computational Complexity* 13.3-4 (2005), pp. 147–188. DOI: 10.1007/s00037-004-0187-1.

[Wat12] Thomas Watson. "Relativized Worlds without Worst-Case to Average-Case Reductions for NP". In: *TOCT* 4.3 (2012), 8:1–8:30. DOI: 10.1145/2355580.2355583.

[Yap83] Chee-Keng Yap. "Some Consequences of Non-Uniform Conditions on Uniform Classes". In: *Theor. Comput. Sci.* 26 (1983), pp. 287–300. DOI: 10.1016/0304-3975(83)90020-8.

[ZL70] AK Zvonkin and LA Levin. "The complexity of finite objects and the algorithmic concepts of randomness and information". In: *UMN (Russian Math. Surveys)* 25.6 (1970), pp. 83–124.

Book Introduction by the Authors



Book Introduction by the Authors

Invited by

Christian Choffrut

Christian.Choffrut@liafa.univ-paris-diderot.fr
Laboratoire IRIF
Université Paris Diderot

BEATCS no 136

PROFINITE SEMIGROUPS AND SYMBOLIC DYNAMIC

Jorge Almeida,
Alfredo Costa,
Revekka Kyriakoglou,
Dominique Perrin
Lecture Notes in Mathematics 2274.
(Christian Choffrut, IRIF Université Paris Diderot)

Most probably, if the reader of this bulletin has ever heard the term "profinite" this is via Eilenberg's theorem on pseudo-varieties of finite monoids. Actually, Birkhoff's theory of varieties had to be reworked in order to suit finite structures. This is precisely what Reitermann did in 1982 by introducing a new type of "equations" or "identities" using operations that no longer belong to the structure, typically $x^\omega = x^{\omega+1}$ which can be understood as saying that the finite monoids of the variety have only trivial subgroups. Technically, the idea is based on the notion of free profinite monoid $\widehat{A^*}$ generated by A (and its elements the pseudowords), defined in two equivalent ways: "metrically" as completion of the free monoid for a specific metric (two words are close if they can be distinguished by small finite monoids) and "algebraically" as projective limits of finite monoids in the same spirit that p-adic numbers are the projective limit of the rings $/p^n$. The free monoid A^* is identified with the set of pseudowords having finite (profinite!) length. This publication collects the very last results in the area since the "Finite Semigroups and Universal Algebras" of Jorge Almeida in 1995.

Theoretical computer scientists are familiar with the notion of right infinite, left infinite, two way infinite words and more generally linear structures labelled by finite alphabets. The authors claim that pseudowords are a generalization and that nonfinite pseudowords "start with a right infinite word, end with a left infinite word and have something in the middle" which make them completely distinct from the usual right-, left- or two-way infinite words and invite us to be cautious with the possible misleading interpretation of the symbol ω. This formalism may seem abstract but let me mention a known and old result showing the relevance to the traditional theory of finite automata: there exists an isomorphism between the algebra of clopen subsets of the free profinite monoid and the Boolean algebra of recognizable languages (those recognized by finite automata).

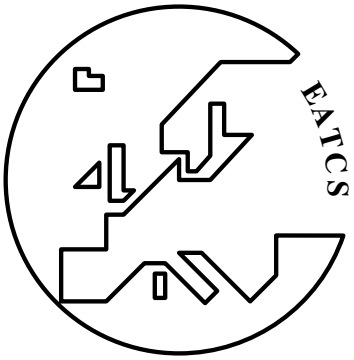
The book starts with an alluring prelude arousing the curiosity of the reader.

In particular and as an illustration of the concepts developed in further chapters, it shows how to use p-adic arithmetic to elegantly prove Skolem-Mahler theorem concerning the zeros of the series associated with rational fractions having rational coefficients. Half of the rest recalls basic definitions and concepts: Hausdorff topological spaces, inverse limits, free groups, finite automata, semigroups, Green relations for semigroups, shift spaces, bifix codes ... The more advanced part investigates the closure in $\widehat{A^*}$ of the uniformly recurrent subsets appearing in minimal shifts and their so-called return words. Most of these shifts are defined by the finite blocks appearing in the fixed points of substitutions of the free monoid, the best known examples of which are the Fibonacci and Thue-Morse words and more generally the DOL-sequences introduced by Lindenmayer. Languages, i.e., subsets of finite words, can be profitably studied by algebraic structures such as congruences and semigroups. This allowed M.P. Schützenberger to characterize the star free regular languages as those languages having only trivial subgroups in the syntactic monoid. In a somewhat similar but sophistigated manner, uniformly recurrent subsets can be associated with so-called Schützenberger groups in $\widehat{A^*}$. The main result, due to the first two authors, shows that a combinatorial property on finite graphs associated with the finite blocks of the shift is a sufficient condition for this group to be a free profinite group. The last chapter has a slightly different focus and works with bifix codes that are factors of some uniformly recurrent subsets. It shows in particular that under some assumptions these codes are finite bases of a free group.

The book is self-contained. Almost all the results are proved in the text otherwise they are referred to some among numerous solved exercices at the end of each chapter. By always choosing the right arguments the proofs are remarkably elegant and appear natural in spite of their technical difficulty.

BEATCS no 136

European Association for Theoretical Computer Science



E A T C S

EATCS

HISTORY AND ORGANIZATION

EATCS is an international organization founded in 1972. Its aim is to facilitate the exchange of ideas and results among theoretical computer scientists as well as to stimulate cooperation between the theoretical and the practical community in computer science.

Its activities are coordinated by the Council of EATCS, which elects a President, Vice Presidents, and a Treasurer. Policy guidelines are determined by the Council and the General Assembly of EATCS. This assembly is scheduled to take place during the annual **I**nternational **C**olloquium on **A**utomata, **L**anguages and **P**rogramming (ICALP), the conference of EATCS.

MAJOR ACTIVITIES OF EATCS

- Organization of ICALP;
- Publication of the "Bulletin of the EATCS;"
- Award of research and academic career prizes, including the EATCS Award, the Gödel Prize (with SIGACT), the Presburger Award, the EATCS Distinguished Dissertation Award, the Nerode Prize (joint with IPEC) and best papers awards at several top conferences;
- Active involvement in publications generally within theoretical computer science.

Other activities of EATCS include the sponsorship or the cooperation in the organization of various more specialized meetings in theoretical computer science. Among such meetings are: CIAC (Conference of Algorithms and Complexity), CiE (Conference of Computer Science Models of Computation in Context), DISC (International Symposium on Distributed Computing), DLT (International Conference on Developments in Language Theory), ESA (European Symposium on Algorithms), ETAPS (The European Joint Conferences on Theory and Practice of Software), LICS (Logic in Computer Science), MFCS (Mathematical Foundations of Computer Science), WADS (Algorithms and Data Structures Symposium), WoLLIC (Workshop on Logic, Language, Information and Computation), WORDS (International Conference on Words).

Benefits offered by EATCS include:

- Subscription to the "Bulletin of the EATCS;"
- Access to the Springer Reading Room;
- Reduced registration fees at various conferences;
- Reciprocity agreements with other organizations;
- 25% discount when purchasing ICALP proceedings;
- 25% discount in purchasing books from "EATCS Monographs" and "EATCS Texts;"
- Discount (about 70%) per individual annual subscription to "Theoretical Computer Science;"
- Discount (about 70%) per individual annual subscription to "Fundamenta Informaticae."

Benefits offered by EATCS to Young Researchers also include:

- Database for Phd/MSc thesis
- Job search/announcements at Young Researchers area

(1) THE ICALP CONFERENCE

ICALP is an international conference covering all aspects of theoretical computer science and now customarily taking place during the second or third week of July. Typical topics discussed during recent ICALP conferences are: computability, automata theory, formal language theory, analysis of algorithms, computational complexity, mathematical aspects of programming language definition, logic and semantics of programming languages, foundations of logic programming, theorem proving, software specification, computational geometry, data types and data structures, theory of data bases and knowledge based systems, data security, cryptography, VLSI structures, parallel and distributed computing, models of concurrency and robotics.

SITES OF ICALP MEETINGS:

- Paris, France 1972
- Saarbrücken, Germany 1974
- Edinburgh, UK 1976
- Turku, Finland 1977
- Udine, Italy 1978
- Graz, Austria 1979
- Noordwijkerhout, The Netherlands 1980
- Haifa, Israel 1981
- Aarhus, Denmark 1982
- Barcelona, Spain 1983
- Antwerp, Belgium 1984
- Nafplion, Greece 1985
- Rennes, France 1986
- Karlsruhe, Germany 1987
- Tampere, Finland 1988
- Stresa, Italy 1989
- Warwick, UK 1990
- Madrid, Spain 1991
- Wien, Austria 1992
- Lund, Sweden 1993
- Jerusalem, Israel 1994
- Szeged, Hungary 1995
- Paderborn, Germany 1996
- Bologne, Italy 1997
- Aalborg, Denmark 1998
- Prague, Czech Republic 1999
- Genève, Switzerland 2000
- Heraklion, Greece 2001
- Malaga, Spain 2002
- Eindhoven, The Netherlands 2003
- Turku, Finland 2004
- Lisabon, Portugal 2005
- Venezia, Italy 2006
- Wrocław, Poland 2007
- Reykjavik, Iceland 2008
- Rhodes, Greece 2009
- Bordeaux, France 2010
- Zürich, Switzerland 2011
- Warwick, UK 2012
- Riga, Latvia 2013
- Copenhagen, Denmark 2014
- Kyoto, Japan 2015
- Rome, Italy 2016
- Warsaw, Poland 2017
- Prague, Czech Republic 2018
- Patras, Greece 2019
- Saarbrücken, Germany (virtual conference) 2020
- Glasgow, UK (virtual conference) 2021

(2) THE BULLETIN OF THE EATCS

Three issues of the Bulletin are published annually, in February, June and October respectively. The Bulletin is a medium for *rapid* publication and wide distribution of material such as:

- EATCS matters;
- Technical contributions;
- Columns;
- Surveys and tutorials;
- Reports on conferences;
- Information about the current ICALP;
- Reports on computer science departments and institutes;
- Open problems and solutions;
- Abstracts of Ph.D. theses;
- Entertainments and pictures related to computer science.

Contributions to any of the above areas are solicited, in electronic form only according to for-

mats, deadlines and submissions procedures illustrated at `http://www.eatcs.org/bulletin`. Questions and proposals can be addressed to the Editor by email at `bulletin@eatcs.org`.

(3) OTHER PUBLICATIONS

EATCS has played a major role in establishing what today are some of the most prestigious publication within theoretical computer science.

These include the *EATCS Texts* and the *EATCS Monographs* published by Springer-Verlag and launched during ICALP in 1984. The Springer series include *monographs* covering all areas of theoretical computer science, and aimed at the research community and graduate students, as well as *texts* intended mostly for the graduate level, where an undergraduate background in computer science is typically assumed.

Updated information about the series can be obtained from the publisher.

The editors of the EATCS Monographs and Texts are now M. Henzinger (Vienna), J. Hromkovič (Zürich), M. Nielsen (Aarhus), G. Rozenberg (Leiden), A. Salomaa (Turku). Potential authors should contact one of the editors.

EATCS members can purchase books from the series with 25% discount. Order should be sent to:

Prof.Dr. G. Rozenberg, LIACS, University of Leiden,
P.O. Box 9512, 2300 RA Leiden, The Netherlands

who acknowledges EATCS membership and forwards the order to Springer-Verlag.

The journal *Theoretical Computer Science*, founded in 1975 on the initiative of EATCS, is published by Elsevier Science Publishers. Its contents are mathematical and abstract in spirit, but it derives its motivation from practical and everyday computation. Its aim is to understand the nature of computation and, as a consequence of this understanding, provide more efficient methodologies.

The Editor-in-Chief of the journal currently are D. Sannella (Edinburgh), L. Kari and P.G. Spirakis (Patras).

ADDITIONAL EATCS INFORMATION

For further information please visit `http://www.eatcs.org`, or contact the President of EATCS:

Prof. Artur Czumaj,
Email: `president@eatcs.org`

EATCS MEMBERSHIP

DUES

The dues are € 40 for a period of one year (two years for students / Young Researchers). Young Researchers, after paying, have to contact `secretary@eatcs.org`, in order to get additional years. A new membership starts upon registration of the payment. Memberships can always be prolonged for one or more years.

In order to encourage double registration, we are offering a discount for SIGACT members, who can join EATCS for € 35 per year. We also offer a five-euro discount on the EATCS membership fee to those who register both to the EATCS and to one of its chapters. Additional € 35 fee is required for ensuring the *air mail* delivery of the EATCS Bulletin outside Europe.

HOW TO JOIN EATCS

You are strongly encouraged to join (or prolong your membership) directly from the EATCS website `www.eatcs.org`, where you will find an online registration form and the possibility of secure online payment. Alternatively, contact the Secretary Office of EATCS:

Mrs. Efi Chita,
Computer Technology Institute & Press (CTI)
1 N. Kazantzaki Str., University of Patras campus,
26504, Rio, Greece
Email: `secretary@eatcs.org`,
Tel: +30 2610 960333, Fax: +30 2610 960490

If you are an EATCS member and you wish to prolong your membership or renew the subscription you have to use the Renew Subscription form. The dues can be paid via paypal and all major credit cards are accepted.

For adittional information please contact the Secretary of EATCS:

Prof. Emanuela Merelli
via Madonna delle Carceri, 9
Computer Science Build. 1st floor
University of Camerino,
Camerino 62032, Italy
Email: `secretary@eatcs.org`,
Tel: +39 0737402567