Yuri Gurevich

Prof. Emeritus
Computer Science and Engineering
University of Michigan, Ann Arbor, MI, USA

gurevich@umich.edu
http://web.eecs.umich.edu/∼gurevich/
https://en.wikipedia.org/wiki/Yuri_Gurevich

Recent active employment

1998-2018 Principal Researcher, Microsoft Research Redmond, WA

1982-1998 Prof. of Computer Science & Engineering, University of Michigan, Ann Arbor, MI

1988-1989 Sabbatical at Stanford University and IBM Almaden Research Center, CA

1995-1996 Sabbatical at Centre National de la Recherche Scientific, Paris, France

Honors

- AAAS Fellow (2020)
  Inaugural Fellow of the EATCS (2014)
  Foreign Member of Academia Europaea (2008)
  ACM Fellow (1997)
  John Simon Guggenheim Memorial Foundation Fellow (1995) Memorial Foundation (1995)
- Dr. Honoris Causa of
  - Ural State University (now Ural Federal University) Russia 2005
  - Limburgs Universtair Centrum (now Hasselt University) Belgium 1998
- Gurevich Symposia:
  - 2020 Festschrift: Springer LNCS 12180
  - 2015 Festschrift: Springer LNCS 9300
  - 2010 Festschrift: Springer LNCS 6300
  - 2000 Festschrift: Springer LNCS 1862
- Aisenstadt Chair, Centre de Recherches Mathématiques, Québec, Canada (2010–2011)
  Distinguished Chair, Pacific Institute for Mathematical Studies, Canada, 2003
  Distinguished Visitor Medal, University of Helsinki, Finland, 1995
- University of Michigan Awards:
  - Research Excellence, College of Engineering, 1997–1998
  - Faculty Recognition, the University, 1995–1996
  - Teaching Excellence, EECS Department, 1993–1994
  - Distinguished Faculty Achievement, the University, 1990–1991

Research highlights (in roughly chronological order)

A comprehensive annotated publication list is found at

- `http://web.eecs.umich.edu/~gurevich/annotated.htm`.

See also `http://dblp.uni-trier.de/pers/hd/g/Gurevich:Yuri`.

The general direction is from pure mathematics to applications: to computer science, software engineering, privacy and security, quantum computing.
References [n] below are to the annotated publication list mentioned above.

`Ordered abelian groups (OAGs).` Solving a problem of Tarski, [3] proved the decidability of the first-order theory of OAGs. But the OAG algebra is rarely first-order. [25] extended the decidability to the monadic second-order theory of OAGs where the set variables range over convex subgroups. The extended OAG theory essentially subsumed the OAG literature while the decidability proof became simpler.

`Classical Decision Problem.` By the time of Church-Turing thesis, many standard fragments of pure predicate logic had been proved to be decidable or to be reduction classes. Article [6] completed this classification. [13] extended it to first-order logic with function symbols and explained why complete classifications are possible in such cases, paving the way to a comprehensive book `http://www.springer.com/gp/book/9783540423249` on the Classical Decision Problem.

`Monadic second-order theories and games.` [26] solved some of Shelah's problems in his 1975 paper in Annals of Math. [40] pioneered the use of games in computer science. [64] summarized much of the decade-long Gurevich-Shelah collaboration. E.g., the monadic theory of ω_2 is independent of ZFC [45] while that of ordinals $< \omega_2$ is decidable [Büchi], and [47] solved Rabin's Uniformization Problem. [79] proved that Peano Arithmetic is not interpretable in (the monadic theory of the) real line, but [37] gave such an interpretation. How is that possible? It turned out that, in [37], Peano Arithmetic was interpreted — in the [79] sense of the word — in the real line of the "next world" [57].

`Finite model theory and database theory.` Here is a sample of our contributions. [60] argued that traditional mathematical logic, developed to confront the infinite, is ill suited to deal with finite structures, like databases; more appropriate logics should be tailored for computational complexity. [70] showed that the extensions of first-order logic with monotone and with inflationary (a.k.a. nonmonotone) inductions have the same expressive power on finite structures. [83] proved that every datalog query that is expressible in first-order logic is bounded. [109] observed that ostensibly finite structures, like databases, have implicit infinite parts; it generalized finite model theory to metafinite model theory. [74] conjectured that no reasonable logic captures polynomial time on unordered structures. The conjecture is widely believed to hold, yet there are no polynomial time properties of unordered structures known to be inexpressible in the logic of [150]. [193] introduced Liberal Datalog equi-expressible with existential fixed-point logic.

`Average case complexity.` Many NP-hard problems with natural probability distributions on instances are easy on average. For example, the Hamiltonian Path problem on random graphs with a fixed edge probability is linear time on average [71], and the membership problem for the modular group is polynomial time on average [175]. In 1986, Leonid Levin gave an appropriate definition of average-case completeness. Unfortunately average-case reductions are much harder to construct; Levin exhibited just one average-case complete problem. [76] developed the theory of average-case completeness and exhibited a number of average-case complete problems, including a

version of halting problem. [85] questioned whether $P \stackrel{?}{=} NP$ is the right question. It is conceivable that the negative answer would provide little evidence for the intrinsic difficulty of NP problems. The paper argued in favor of the average-case version of $P \stackrel{?}{=} NP$. [88] presented the first algebraic average-case complete problem, a little modification of the membership problem for the modular group. [96] developed the theory of randomizing reductions of search problems.

`What is an algorithm?` The question isn't what function, if any, an algorithm computes, but what it is at its natural abstraction level. [103] defined abstract state machines (ASMs) in order to simulate arbitrary algorithms step-for-step on their natural abstraction levels. [141] axiomatized sequential algorithms and proved that sequential algorithms are behaviorally identical to sequential ASMs. In subsequent papers, these results were extended to parallel [157] and to interactive [176, 182] algorithms.

`Software engineering.` Abstract state machines (ASMs) mentioned above have been used by the ASM community to specify software and hardware architectures, compilers, databases, programming languages, various distributed systems, etc. At Microsoft, my group on Foundations of Software Engineering built an ASM-based tool, called Spec Explorer, which helped Microsoft when the European Union demanded in the early 2000s that Microsoft produce high-level executable specifications of various communication protocols. [190] provided foundations for a new differential compression method used now in all major Microsoft products.

`Access control.` Logic is a natural foundation for authorization languages. It allows one to write high-level policy rules in a human-readable form. The challenge is to have your logic expressive and feasible at the same time. [191] introduced Distributed-Knowledge Authorization Language (DKAL), based on existential fixed-point logic [73] and more expressive that the languages in the literature and yet feasible. Subsequently we worked on developing DKAL itself [197,200,203,212,216] and on the logical foundations of DKAL [193,194,204,208]. The surprising feasibility of DKAL led to the discovery of primal logic which is decidable in linear (or, in some variants, expected linear) time [198,205,211,215,221].

`Other security issues, and privacy` This part is better reflected in patents. Some of them are based on or inspired by the ideas in [206]. But there are some publications as well. [202] describes a novel approach, implemented at Microsoft, to the Security Assessment Sharing Problem. [218] focuses on explicating subtle implicit assumptions endemic to the software developments kits (SDKs) of Facebook, Google, Microsoft, etc. [222] introduces the notion of inverse privacy. An item of your personal information is private if you have it but nobody else does; it is inversely private if somebody has it but you do not. It turns out that inverse privacy is ubiquitous. We analyze the provenance of inverse privacy and argue that technology and appropriate public policy can reduce inverse privacy to a minimum.

`Quantum computing.` [217,223] develop efficient algorithms for compiling single-qubit unitary gates into circuits over the V basis, an alternative universal basis to the standard {H,T} basis. No-go theorems prove that, under reasonable assumptions, classical hidden-variable theories cannot reproduce the predictions of quantum mechanics. Recent expectation no-go theorems prove that hidden-variable theories cannot predict the expectations of observables; [236] establishes the strongest expectation-focused no-go theorem to date. Somewhat surprisingly, the only rigorous models of non-abelian anyons are based on category theory. [225] explains the use of category theory in this context. A question arises whether category theory is really needed for the purpose. The answer isn't black and white. Some aspects of category theory are needed [225] but the framework can be greatly simplified [work in progress].

Patents

Most of the patents are found at
`https://patents.google.com/?inventor=yuri+gurevich`.

Logic in Computer Science column

My continuous column on "Logic in Computer Science" in the Bulletin of the European Association for Theoretical Computer Science is running for 30 years, from 1988, three times a year. Writing for the general computer-science audience is challenging. Often I find great experts willing to do that; otherwise I am writing an article myself or with my long-term collaborator Andreas Blass.