The Logic in Computer Science Column

by

Yuri Gurevich

Microsoft Research
One Microsoft Way, Redmond WA 98052, USA
gurevich@microsoft.com

VIEWPOINTS ON "LOGIC ACTIVITIES IN EUROPE", TWENTY YEARS LATER

Luca Aceto
ICE-TCS, School of Computer Science,
Reykjavik University
luca@ru.is

1 Origin of and motivation behind this enterprise

Prompted by a reference to it in a recent CACM editorial by Moshe Vardi [15], I belatedly read the very interesting piece on logic activities in Europe [8] that Yuri Gurevich published in SIGACT News in 1994. While reading Yuri's thought-provoking article, I was struck by the idea that it might be interesting to ask some selected colleagues to contribute opinion pieces to the Bulletin of the EATCS reflecting on the points raised by Yuri in that article twenty years later. Thomas Henzinger, Joost-Pieter Katoen, Wolfgang Thomas and Moshe Vardi accepted my invitation and agreed to share their views with the theoretical-computer-science community at large. You can read their contributions in the "viewpoint pieces" that follow this prefatory note of mine and I trust that you will enjoy reading them as much as I did.

My goal in commissioning those articles was to start a reflection within the theoretical-computer-science community in Europe and in North America, but also more globally, as to how we can improve the research ecosystem in theoretical computer science within our own countries, continents as well as in the world at large. You are all most welcome to provide your opinions and counterpoints as comments or posts on my professional blog[1], where the viewpoint pieces are also published individually, or as contributions to future issues of the Bulletin of the EATCS. All these opinions will be useful input for the Council of the EATCS, so that our association can serve the theoretical-computer-science community in the best possible way.

In order to contribute to this discussion, in the remainder of this note I will offer my own quarter-baked and admittedly disjointed thoughts related to Yuri's

[1] http://processalgebra.blogspot.is/

piece in the hope that others will feel tempted to do a much better job than I.

2 Some personal reflections on Yuri's article

In order to provide some context for some of the thoughts I had while reading Yuri Gurevich's report, let me start by recalling the setting that led to its writing.

In the autumn 1992, Yuri visited a "fair number of West European centers of logic research." In Yuri's words, he tried "to learn more about logic investigations and applications in Europe", where logic is intended as "logic in computer science".

At the start of his report, Yuri notes that, despite the communication between researchers in theoretical computer science in Europe and the US,

> "...it is amazing, however, how different computer science is, especially theoretical computer science, in Europe and the US. American theoretical computer science centers heavily around complexity theory.
>
> ...A much greater proportion of European research goes into programming language theory, semantics, specification languages, proof systems, verification methods, etc."

Yuri uses the term "formal methods" for all of the aforementioned areas of European focus and says:

> "Europeans put much more faith and efforts into foundational investigations of software and hardware technology and into developing formal methods for use in software and hardware."

Even though European theoretical computer scientists still contribute much research related to formal methods, I think that it is fair to say that work belonging to what Yuri would call formal methods has blossomed in the US too, leading to high impact work. Without any pretension of completeness and focusing only on methods for use in the development of software and hardware, at the time of Yuri's Grand Tour of Western European institutions,

- Clarke and Emerson in the US and Queille and Sifakis in France had already sown the seeds of what will eventually be called *model checking* [6, 14], a major approach for computer-aided verification that, since then, has seen substantial developments and industrial applications throughout the world;
- Vardi and Wolper had developed the automata-theoretic approach to model checking linear-time temporal logic [16];

- Alur and Dill had already carried out their seminal work on the model of *timed automata* [2, 3], which has found considerable application over the last twenty years in modelling and analysis of real-time systems and is embodied in software tools for computer-aided verification (with the leading ones being developed in Europe);

- the model of hybrid automata proposed by Alur, Courcoubetis, Henzinger and Ho in [1] was already supported by the HYTECH software tool developed at Cornell by Henzinger and Ho [10].

Indeed, in hindsight, it is rather ironic that the year of publication of Yuri's piece coincides with what I consider to be a watershed event for the application of formal methods in industry in the US, namely the discovery of the Pentium FDIV bug [13]. The Pentium Bug brought model-checking and theorem-proving based verification for hardware systems in the limelight. Today the use of formal methods and verification tools is commonplace at companies such as Amazon [12], Facebook [5], Intel (see the slides by John Harrison at `http://www.cl.cam.ac.uk/~jrh13/slides/oregonsummerschool-26jul12/slides.pdf`), Microsoft [4] and NASA [11]. Based on this evidence, it seems to me that there is belief in the worth (and, in fact, in the need for the use) of formal methods in the US too. The growing recognition of research in formal methods in the US is also witnessed by the rich funding received by the NSF Expedition in Computing projects DeepSpec (`http://deepspec.org/`) and ExCAPE (`https://excape.cis.upenn.edu/`).

In my humble opinion, however, what has *not* changed is that, unlike in Europe, the research underlying the tremendous advances in formal methods research is still not seen as contributing to theoretical computer science in the US.

The successful applications of formal methods research over the last twenty years have certainly vindicated this belief aired by Yuri in his report:

> "the tour gave me more confidence that a logician may be of great help to software engineers and even himself/herself may be a successful software engineer."

From a European (and personal) perspective, the year 1994 saw also the birth of the Basic Research in Computer Science centre (BRICS) of the Danish National Research Foundation. The centre was richly funded from 1994 till 2006, hosted an international PhD school and attracted top-class researchers from all over the world to Aarhus and Aalborg in Denmark to work on both Volume A and Volume B topics. I believe that a look at research in theoretical computer science in Denmark today clearly indicates that the spirit of BRICS is still very much alive there, and that researchers located in that country carry out high quality research

in both algorithms and complexity, and formal methods. I chose Denmark as an example simply because I was lucky enough to be part of BRICS for many years, but, to my mind, research in both algorithms and complexity, and formal methods is alive and well in many other European countries.

In his report, Yuri wrote:

> "Too often complexity theorists are not interested at all in semantics and (what we call here) formal methods, and too often experts in formal methods are not interested in and do not know modern complexity theory."

This is something I find rather unfortunate. It is, of course, natural to focus on one's own branch of theoretical computer science. However, by not showing interest in each other's work, our community misses excellent opportunities for cross fertilization and this might sometimes slow down advances in our field. As an example, I encourage you to look at the slides for a recent invited talk by Moshe Vardi that are available at `http://www.cs.rice.edu/~vardi/papers/sr15.pdf`.

However, even a cursory look at the proceedings of conferences such as CONCUR, ICALP (Track B) and LICS indicates that there is much research in present-day Volume B theoretical computer science that has strong connections with algorithms and complexity theory and that ought to be of some interest to Volume A researchers. Moreover, Volume B researchers do benefit from Volume A research, and, as highlighted by Vardi's presentation, the algorithmic study of games is an excellent example of the potential for cross-fertilization between the two areas.

Yuri devoted Section 3 of his report to discuss what is good with the research environment in Europe. As I have tried to highlight above, several of the "good points" of theoretical-computer-science research in Europe mentioned by Yuri are very much shared by US research these days, and key approaches to computer-aided verification based on model checking and theorem proving have been developed both in Europe and the US, sometimes in cooperative efforts. The US hosts a very active and influential group of researchers working on semantics of programming languages both at universities and in industrial research laboratories. The work of these colleagues builds on seminal notions (for example, Structural Operational Semantics) and software tools (such as the Coq proof assistant, `https://coq.inria.fr/`) developed in Europe and often sees a welcome cooperation between researchers across the Atlantic.

I guess that Yuri's statement to the effect that

> "Functional Programming, logic programming, automata and formal language theory are more popular in Europe."

is still true to a large extent. However, it seems to me that the popularity of functional programming has increased in the US and that the resurgence of automata theory is visible in American research as well. (To wit, see the research I mentioned earlier in this note and the ExCAPE project.)

I like to think that the sense of community in logic activities in Europe to which Yuri referred in his report still exists today. (I encourage you to read the contribution by Katoen and Thomas for their thoughts on this matter.) However, the recent creation of ACM SIGLOG has given North American researchers in logic and computation a natural meeting point. Moreover, since I expect that many members of SIGLOG are based in Europe, that association can serve, just like the EATCS, as a meeting point for researchers across the Atlantic and can play an important role in developing joint research activities.

Section 4 of Yuri's report was devoted to a discussion of some negative points of the European work on logic and computation, and of the "European system" in general. Yuri pointed out that the European system is more conservative than the one in the US, that academic schools play an overly important role in it and that the connection between industrial and academic research is not as close as it could, and perhaps should, be. For a discussion of these criticisms, I refer you to Thomas Henzinger's thought-provoking viewpoint piece in this volume.

On a more technical note, I think that research on "pure functional programming" over the last twenty years and the advent of multi-core machines have increased the practical importance of the functional programming paradigm. For instance, several financial domain-specific languages have a functional core; see the list at `http://www.dslfin.org/resources.html`. On this point, Philip Wadler wrote to me saying:

> Yuri Gurevich, in Section 4.3 of "Logic Activities in Europe", writes that (in 1994) current functional languages "are not sufficiently efficient yet". These days it is common for functional languages to rank highly in programming language benchmark "shootouts", with languages such as Clojure, Erlang, F#, Haskell, OCaml, Racket, Scala all doing well, and often rivalling the performance of C++. Increased interest in parallel and distributed computing has raised the importance of functional programming and immutable data.
>
> Gurevich also conjectures "Maybe a right mixture of imperative and functional programming is needed, an imperative language with a clean and powerful functional components." It is interesting that he chose not to put it the other way around. Many functional languages have a lambda calculus core with direct support for computational effects. Further, monads in Haskell (and adapted to other languages, including Clojure, F#, and Scala) provide a way of embedding "impure"

computational effects within a "pure" functional language, using the type system to delimit what effects can occur where.

For an accessible discussion of the connections between logic and programming and of the "proofs as programs/propositions as types" paradigm, I encourage you to read a recent Communications of the ACM article by Philip Wadler [17].

The final part of Yuri's report was devoted to a discussion of the use of proof assistants in computer system verification and in mathematics. I do not think anyone could have imagined the amazing developments in the use of proof assistants in mathematical research and in software development, as well as the advances in research on their foundations, over the last twenty years. Moreover, this development provides an excellent example of what can be achieved by collaborative enterprises across national and continental borders.

The use of proof assistants in mathematical research has led to the complete formalization of very long and deep proofs—see, for instance, the work on a formal proof of the Kepler conjecture [9], on the four-colour theorem[2] and on the so-called Odd Order Theorem in group theory [7]. In software development, the CompCert project led by Xavier Leroy has investigated the formal verification of realistic compilers usable for critical embedded software and has developed a formally verified optimizing compiler for a large subset of the C programming language. See `https://en.wikipedia.org/wiki/CompCert`.

On the foundational front, the recent development of Homotopy Type Theory (see `http://homotopytypetheory.org/`) has provided yet another example of what can be achieved via a joint American-European effort. I hope that transnational and trans-continental research efforts such as the Homotopy Type Theory one will help foster research cooperation on a global scale in the future, leading to advances in our field that would not be possible (or would be much slower) otherwise.

With a delay of over twenty years, I thank Yuri Gurevich for taking the time to write the article that serves as an inspiration to the contributions that follow this prefatory piece of mine in the Logic Column of this issue of the Bulletin of the EATCS. I hope that these viewpoints and the ensuing further discussion within our community will help us realize what the strengths and weaknesses of our respective research ecosystems are, and that we can always learn by analyzing them critically and with an open mind, as well as by interacting and studying each other's work across the continental or Volume A/Volume B divides.

Acknowledgments I thank Philip Wadler for sending me his opinions on Yuri Gurevich's comments related to pure functional programming in [8, Section 4.3],

[2] `http://research.microsoft.com/en-US/people/gonthier/4colproof.pdf`

and Thomas Henzinger, Joost-Pieter Katoen, Wolfgang Thomas and Moshe Vardi for contributing viewpoint articles to this issue of the Bulletin of the EATCS. Ignacio Fábregas, Álvaro García-Pérez and Moshe Vardi provided comments on drafts of this note that led to improvements in the presentation and helped me to correct some imprecisions. Any remaining infelicity is solely my responsibility.

References

[1] Rajeev Alur, Costas Courcoubetis, Thomas A. Henzinger, and Pei-Hsin Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. In Robert L. Grossman, Anil Nerode, Anders P. Ravn, and Hans Rischel, editors, *Hybrid Systems*, volume 736 of *Lecture Notes in Computer Science*, pages 209–229. Springer, 1992.

[2] Rajeev Alur and David L. Dill. Automata for modeling real-time systems. In Mike Paterson, editor, *Automata, Languages and Programming, 17th International Colloquium, ICALP90, Warwick University, England, July 16–20, 1990, Proceedings*, volume 443 of *Lecture Notes in Computer Science*, pages 322–335. Springer, 1990.

[3] Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[4] Thomas Ball. Correctness via compilation to logic: A decade of verification at Microsoft Research. In Michael Feldman and S. Tucker Taft, editors, *Proceedings of the 2014 ACM SIGAda annual conference on High integrity language technology, HILT 2014, Portland, Oregon, USA, October 18-21, 2014*, pages 69–70. ACM, 2014.

[5] Cristiano Calcagno, Dino Distefano, Jérémy Dubreil, Dominik Gabi, Pieter Hooimeijer, Martino Luca, Peter W. O'Hearn, Irene Papakonstantinou, Jim Purbrick, and Dulma Rodriguez. Moving fast with software verification. In Klaus Havelund, Gerard J. Holzmann, and Rajeev Joshi, editors, *NASA Formal Methods - 7th International Symposium, NFM 2015, Pasadena, CA, USA, April 27–29, 2015, Proceedings*, volume 9058 of *Lecture Notes in Computer Science*, pages 3–11. Springer, 2015.

[6] E.M. Clarke and E.A. Emerson. Design and synthesis of synchronization skeletons using branching-time temporal logic. In D. Kozen, editor, *Proceedings of the Workshop on Logic of Programs,* Yorktown Heights, volume 131 of *Lecture Notes in Computer Science*, pages 52–71. Springer, 1981.

[7] Georges Gonthier, Andrea Asperti, Jeremy Avigad, Yves Bertot, Cyril Cohen, François Garillot, Stéphane Le Roux, Assia Mahboubi, Russell O'Connor, Sidi Ould Biha, Ioana Pasca, Laurence Rideau, Alexey Solovyev, Enrico Tassi, and Laurent Théry. A machine-checked proof of the odd order theorem. In Sandrine Blazy, Christine Paulin-Mohring, and David Pichardie, editors, *Interactive Theorem Proving - 4th International Conference, ITP 2013, Rennes, France, July 22-26, 2013.*

Proceedings, volume 7998 of *Lecture Notes in Computer Science*, pages 163–179. Springer, 2013.

[8] Yuri Gurevich. Logic activities in europe. *SIGACT News*, 25(2):11–24, 1994.

[9] Thomas C. Hales, Mark Adams, Gertrud Bauer, Dat Tat Dang, John Harrison, Truong Le Hoang, Cezary Kaliszyk, Victor Magron, Sean McLaughlin, Thang Tat Nguyen, Truong Quang Nguyen, Tobias Nipkow, Steven Obua, Joseph Pleso, Jason Rute, Alexey Solovyev, An Hoai Thi Ta, Trung Nam Tran, Diep Thi Trieu, Josef Urban, Ky Khac Vu, and Roland Zumkeller. A formal proof of the kepler conjecture. *CoRR*, abs/1501.02155, 2015.

[10] Thomas A. Henzinger and Pei-Hsin Ho. HYTECH: the cornell hybrid technology tool. In Panos J. Antsaklis, Wolf Kohn, Anil Nerode, and Shankar Sastry, editors, *Hybrid Systems II*, volume 999 of *Lecture Notes in Computer Science*, pages 265–293. Springer, 1994.

[11] Gerard J. Holzmann. Mars code. *Communications of the ACM*, 57(2):64–73, 2014.

[12] Chris Newcombe, Tim Rath, Fan Zhang, Bogdan Munteanu, Marc Brooker, and Michael Deardeuff. How Amazon web services uses formal methods. *Communications of the ACM*, 58(4):66–73, 2015.

[13] Vaughan R. Pratt. Anatomy of the Pentium bug. In Peter D. Mosses, Mogens Nielsen, and Michael I. Schwartzbach, editors, *TAPSOFT'95: Theory and Practice of Software Development, 6th International Joint Conference CAAP/FASE, Aarhus, Denmark, May 22-26, 1995, Proceedings*, volume 915 of *Lecture Notes in Computer Science*, pages 97–107. Springer, 1995.

[14] J. P. Queille and J. Sifakis. Specification and verification of concurrent systems in CESAR. In *International symposium on programming (Turin, 1982)*, volume 137 of *Lecture Notes in Computer Science*, pages 337–351. Springer, 1982.

[15] Moshe Y. Vardi. Why doesn't ACM have a SIG for theoretical computer science? *Communications of the ACM*, 58(8):5, 2015.

[16] Moshe Y. Vardi and Pierre Wolper. Reasoning about infinite computations. *Information and Computation*, 115(1):1–37, 1994.

[17] Philip Wadler. Propositions as types. *Communications of the ACM*, 58(12):75–84, 2015.

Computer Science in Europe

Thomas A. Henzinger
IST Austria
tah@ist.ac.at

It saddens me but it would be difficult to refute a claim that, in the past two decades, Europe has been falling further behind the United States in the dynamism of the information technology industry, the popularity of the computer science major, and the impact of frontier research in computing. The vast majority of Turing awards still goes to researchers who work in the United States. It is particularly disconcerting that the main strengths of European computer science appear largely unchanged from 1994: on the academic side, Europe's research leaders are still concentrated disproportionately in formal methods, and on the industrial side, Europe's technology leaders are still found primarily in the "old" economy, exemplified by the automotive industry.

To close the gap, Europe desperately needs new organizational structures in academia, a greater entrepreneurial spirit of society, an improved image for computer science as a career choice, especially among women, the mandatory acquisition of computational thinking and coding skills in secondary education, and more emphasis on principles of systems building which are critical to industry in university curricula of computer science. Israel offers a role model for closing the gap with the United States with regard to the first three points —academic structures, entrepreneurial culture, and the public image of computer science— and has been a leader in computer science education.

There are a few encouraging signs of European computer science changing. The European systems community has begun to organize itself through efforts such as the Eurosys conference and some countries are trying to remedy their deficiencies in systems research. Germany, for example, founded the Max Planck Institute for Software Systems. Several European countries and institutions have started to copy key aspects of the American career model, such as tenure tracks that give faculty early independence and doctoral programs that give students a broad graduate education. Student mobility and structured doctoral education are strongly supported by the Marie Curie program of the European Union and by the funding agencies of some countries, to counteract the wide-spread habit of researchers advancing in the same lab from undergraduate to faculty level. There have been some remarkable institutional changes. EPFL has demonstrated that

changes in the organization and recruiting can lead to dramatic improvements in the scientific reputation and attractiveness of an institution. Even entirely new institutions have been founded, such as IST Austria, which naturally find it easier to implement new structures such as a tenure track and an institutional doctoral school.

The most significant development can be found, perhaps surprisingly, on the European level. I am referring to the creation of the European Research Council, which supports frontier research based purely on scientific criteria. This program has no counterpart in the United States, but if it manages to remain scientifically independent and well-funded, I am confident that its impact will change the game. These are big if's, of course, and the ERC is constantly being threatened by national interests and sectorial lobbies that favor traditional programs which distribute the available funds to more different countries, sectors, and groups. Given that politicians love to pride themselves with the founding of "strategic" consortia, centers, and flagships, and industry likes to get every possible cut of public money, the initial success of the ERC has been all the more remarkable. Let's work together so that it will trump the less effective funding formats and lift the strength of computer science in Europe.

Comments of 2016 on Yuri Gurevich's Paper "Logic Activities in Europe" of 1994

Joost-Pieter Katoen and Wolfgang Thomas
RWTH Aachen University
katoen@cs.rwth-aachen.de, thomas@cs.rwth-aachen.de

Prologue

In his noteworthy paper on "Logic Activities in Europe" of 1994, Yuri Gurevich focussed on logic in relation to computer science. Following comments on "the incredible breadth of European foundational research" (in his Section 3.4), he addressed concrete programs and frameworks of research in rather short Sections 3.5–3.7 (with less than a page altogether), titled "ESPRIT", "Conference centers", and "Disparate remarks". Here he discussed the efforts of the European Community to "overcome the fragmentation of the European research by countries", he points to the research centers of Dagstuhl, Luminy, and Udine, and he mentions "strong, unifying, and active organizations" (EATCS, EACSL, and FoLLI). All this conveys a sense of appreciation of the European situation and European efforts.

In this note we try an update of these sections of Yuri's paper twenty years later, following an invitation of Luca Aceto, president of the EATCS. It is perhaps surprising that Yuri's main observations match quite well what can be said also today. In our comments we summarize the development since 1994, and we add two sections, on the landscape of conferences, workshops, and research schools, and on academic publishing in our domain.

European research funding

Starting with the ESPRIT program (European Strategic Program for Research and Development in Information Technology) that was launched in 1983, several frameworks opened possibilities to obtain funding (up to Framework Programs FP8 that started in 2014, also known as Horizon 2020). From ESPRIT in 1995 to Horizon 2020 there is a great leap both in scope and amount of funding. There

were and are complaints about the bureaucracy involved in EU projects, but it is fair to say that the European funding contributed a lot to create a landscape of research which as a whole is now more European than the sum of the national research efforts. An important aspect is the increased exchange of researchers by the funding of PhD student and postdoc positions that were and are built into European research projects as well as facilitated by the Marie Curie program. This led to an integration of European research far beyond the level in 1994, both regarding integration across countries and across disciplines.

It will be interesting to see how the significantly increased focus of H2020 on applications and industrial exploitation will affect this situation. Whereas up to FP7 participation of industrial partners was mostly restricted to case study providers, H2020 requires a much more prominent role of industry in the projects. In addition, the exploitation of project results (in terms of spin-offs and industrial adoption of project results) has become a major evaluation criterion. There is not much room for theory-oriented research. As a result, the only more research-oriented sub-programs such as FET-OPEN had acceptance rates of below 5% in 2015. When this trend continues, we fear that the importance of the Framework Programs for fundamental science may be at risk.

The EU-frameworks were complemented by the European Science Foundation (ESF) which offered funding through national funding agencies. With some regret we realize that many programs of ESF terminated. The administrative overhead was smaller (as perceived from the scientists), and it seems that some competition between different funding lines is healthy.

A remarkable new development started in 2007 with the ERC (European Research Council) grants. They are complementary to the classical EU funding schemes in the sense that merits of individuals are acknowledged and their projects supported. The ERC grants are respected today as top awards, matching well the highest awards of the EU countries. By the end of 2014 more than 600 ERC projects were finalized.

European workshops and conferences, and research schools

In 1994, the main conferences in Europe in theoretical computer science (ICALP, STACS, MFCS, FCT) were already well established, and the division into "Western" (ICALP and STACS) and "Eastern" (MFCS and FCT) conferences had become largely obsolete. With a definite focus on logic there were, for example, CSL (founded 1987) and CONCUR (founded in 1990). In 1999, ICALP was structured into Track A (for algorithms and complexity) and Track B (for formal

methods and semantics); later a Track C was added to deal with varying subjects of special interest. As with all such divisions, problems arose in the "grey zones"; for example, automata were first put into Track A and later in Track B. On a more international level, one should mention LICS (Logic in Computer Science, which is considered the leading conference in logic related to computer science) and CAV (Computer-Aided Verification); both of them are held in Europe around every third year.

A major breakthrough in creating a European venue of "logic" (in a broad sense, and with a focus on computer science and programming in particular) was the establishment of ETAPS (European Joint Conference on Theory and Practice of Software) in 1998. It started as a multiconference bundling the five conferences FoSSaCS (Foundations of Software Science and Computation Structures), ESOP (European Symposium on Programming), FASE (Fundamental Aspects of Software Engineering), TACAS (Tools and Algorithms for the Construction and Analysis of Systems, that originally started as the European version of CAV), and CC (Compiler Construction); in 2011 a sixth conference POST (Principles of Security and Trust) was added. CC left ETAPS in 2016. ETAPS serves now as the main meeting point in the area of formal methods in Europe; usually about a dozen of workshops is associated to an ETAPS edition, and the conference receives 600–700 submissions annually.

Besides this, many more conferences and workshops were created that helped to define a European culture of research, probably with a density and vivid exchange between researchers that is not seen anywhere else in the world. Any list given at this point will be incomplete; let us just mention FM, RTA and TLCA (now joined to FSCD), CALCO, ICDT as some examples. For an illustration of the rich list of workshops with high quality, it may suffice to refer to the proceedings published with EPTCS (Electronic Proceedings in Theoretical Computer Science, discussed in more detail below), where many items can be associated to "logic" and many are held in conjunction with European conferences or at European locations.

A last aspect should not be forgotten: In various frameworks the work of young researchers is supported by research schools (often called spring school, summer school, etc.). They are most important to get young people together, to have a lively exchange between young and "established" researchers (the docents), and to foster cooperation in Europe and world-wide. The FoLLI schools (of the Federation of Logic, Language, and Information) and the Marktoberdorf summer schools (in particular the "blue series" devoted to logic) are master examples. The significance of these events is acknowledged by EATCS; since 2014 there is an annual EATCS summer school for young researchers.

Research centers

Since its foundation in 1990, Schloss Dagstuhl (Saarland, Germany) has developed into a leading address for small meetings in top-level research; every week one or two seminars are held in various domains of computer science. Much progress in logic, semantics, verification, and formal methods was greatly supported (sometimes even made possible) by these seminars. From time to time "perspectives workshops" are held on long-term challenges, interdisciplinary cooperation, etc. There does not seem to be any other place in the world matching this profile. The formal status of Dagstuhl was strengthened (beyond a locally funded institute) by inclusion in the "Leibniz-Gemeinschaft", in which institutions of nation-wide significance are funded by the German federal ministry of research. In Italy, Bertinoro was established as a similar center (however covering many other subjects besides computer science). It is interesting to note that these centers of (at least) European relevance are just funded with national money. The successful Dagstuhl concept has given rise to similar initiatives both within Europe (the already mentioned Bertinoro and the Lorentz Center in the Netherlands) as well as outside Europe such as NII Shonan (in Japan) and Mysore (in India) whose scope as Bertinoro goes beyond computer science.

Developments in academic publishing

The ever increasing prices for published research and the commercialization of publishing (in stock-market companies, such as Elsevier, and enterprises held by private equity firms, such as Springer) led to activities in the scientific community to start alternative models of publication, aiming at open access at small fees. Back in 1994, this development was visible only in its nucleus. The open access platform arXiv had been founded in 1991 and from 1999 was hosted at Cornell University, giving it a firm perspective. In France, the open access archive HAL (Hyper Articles en Ligne) has been launched in 2011.

The community of logic in computer science was active and successful in opening non-commercial high-level journals in their domain. The first was ACM Transactions on Computational Logic, founded in 2000 with Krzysztof Apt as first editor-in-chief, the second was Logical Methods in Computer Science, founded in 2005 with Jiri Adamek as editor-in-chief, who also cared for the enormous work of hosting the journal (as an overlay of arXiv) at the Technical University of Braunschweig. These efforts (jointly by many people and institutions, not just in Europe) significantly changed the infrastructure conditions in which our research takes place. It is worth to mention that both journals have rapidly obtained a very good status among the traditional journals in logic.

On the level of conference proceedings, the "classical venue" back in 1994 was Springer LNCS. Reacting to price increases that prohibited many libraries to order these proceedings, alternative publication venues were founded. First we mention LIPIcs (Leibniz International Proceedings in Informatics) for high-quality conferences, founded in 2008 as a joint project of STACS, FSTTCS (Foundations of Software Technology and Theoretical Computer Science, an India-based top-level conference), and Dagstuhl. Secondly there is EPTCS (Electronic Proceedings in Theoretical Computer Science) for high-level workshops, started in the same year by Rob van Glabbeek. The conferences ICALP and ETAPS stayed with LNCS (in the newly created subseries ARCoSS, whose editors-in-chief were simultaneously president of EATCS and chair of the ETAPS steering committee, respectively), but various logic-related conferences (among them CSL and CONCUR) joined LIPIcs. In 2015 ICALP followed. In the present rapid development the perspectives are quite open, but already the fact that there is now competition between commercial and non-commercial publication models is an improvement of the situation, regarding the main purpose of publication, namely the (affordable) exchange of scientific ideas and results.

Associations

As mentioned by Yuri Gurevich in 1994, EATCS was then active and strong as a scientific association in theoretical computer science. Today one can say that this European institution in fact serves as the representation world-wide in this domain. In the (sub-) domain of logic, EACSL (European Association of Computer Science Logic) has established itself as another strong and visible reference point. In the field of formal systems development, FME (Formal Methods Europe) is rather active and amongst others responsible for the FM symposium. Many awards and prizes have been established to acknowledge excellent contributions; as examples we mention the Presburger Award (by EATCS, for young researchers) and the Ackermann Award (by EACSL, for dissertations in the area of logic in computer science), both named after logicians. On a more advanced level, SIGLOG, mentioned below, launched the Alonzo Church Award for Outstanding Contributions to Logic and Computation in cooperation with EATCS, EACSL, and the Kurt Gödel Society.

Summing up, logic in computer science continues to have a very strong representation in Europe. It looks stronger than in the U.S. where algorithms and complexity theory still seem to enjoy higher estimation. The foundation of ACM SIGLOG (Special Interest Group on Logic and Computation, established in 2014) may change this; here we have a venue of logic in computer science which is visible and appreciated world-wide. It should also be useful in connecting research

in Europe and the U.S. with that of many Asian countries in which the last twenty years have seen a considerable increase of research activities in logic related to computer science.

On the Two Sides of the Atlantic in Logic and Computation

Moshe Y. Vardi
Rice University
vardi@cs.rice.edu

In his 1977 EWD Note 611, "On the fact that the Atlantic Ocean has two sides,"[1] Edsger Dijkstra noted the different attitudes towards computing research in Northern America and Western Europe. Yuri Gurevich noted the same phenomenon in his 1992 report, "Logic Activities in Europe."[2] In a 2015 Communications of the ACM editorial I revisited this issue and asked "Why Doesn't ACM Have a SIG for Theoretical Computer Science?" [3] The key issue raised in that editorial was the split between Volume-A-type and Volume-B-type research in Theoretical Computer Science (TCS), referring to the 1990 Handbook of Theoretical Computer Science, with Jan van Leeuwen as editor. The handbook consisted of Volume A, focusing on algorithms and complexity, and Volume B, focusing on formal models and semantics. In other words, Volume A is the theory of algorithms, while Volume B is the theory of systems (hardware and software). North American TCS tends to be quite heavily focused on Volume A, while European TCS tends to encompass both Volume A and Volume B. The ACM Special Interest Group on Algorithms and Computation Theory (SIGACT) is, de facto, a special-interest group for Volume-A TCS.

I pointed out in my editorial that this division did not exist prior to the 1980s. In fact, the tables of contents of the proceedings of two North American premier TCS conferences–IEEE Symposium on Foundations of Computer Science (FOCS) and ACM Symposium on Theory of Computing (STOC)–from the 1970s reveal a surprisingly (from today's perspective) high level of Volume-B content. In the 1980s, the level of TCS activities in North America grew beyond the capacity of two annual single-track three-day conferences, which led to the launching of what was known then as "satellite conferences." Shedding the "satellite" topics allowed FOCS and STOC to specialize and develop a narrower focus on TCS. But this narrower focus in turn has influenced what is considered TCS in North

[1] http://www.cs.utexas.edu/~EWD/ewd06xx/EWD611.PDF
[2] http://research.microsoft.com/en-us/um/people/gurevich/Opera/105.pdf
[3] http://doi.acm.org/10.1145/2791388

America. In contrast, the European Association for Theoretical Computer Science (EATCS) expanded the scope of its flagship conference, the International Colloquium on Automata, Languages, and Programming (ICALP), by reorganizing the conference along several tracks. In 2015, ICALP consisted of three tracks: Track A: Algorithms, Complexity and Games; Track B: Logic, Semantics, Automata and Theory of Programming; and Track C: Foundations of Networked Computation: Models, Algorithms and Information Management. The reorganization along tracks allowed EATCS to broaden its scope, rather than narrow it like SIGACT.

But the reality is that if one zooms into Volume-B research, one finds again the Volume-A/Volume-B dichotomy, also reflected in the range of topics of the Symposium on Logic in Computer Science (LICS), the flagship conference of the ACM Special Interest Group on Logic and Computation (SIGLOG). Subvolume A of Volume-B research is concerned with connections between logic, algorithms, and computational complexity. Descriptive-Complexity Theory, for example, aims at bridging computational complexity and logic by studying the expressive power needed to describe problems in given complexity classes. A celebrated result in this area is Fagin's Theorem, which relates NP to Existential Second-Order Logic. Model Checking, as another example, studies the evaluation of logical formalisms, including various temporal logics, over finitely represented structures. Automata theory often provides tools to bridge between logic and algorithms. The Büchi-Elgot-Trakhtenbrot Theorem, for example, provides automata-theoretic tools for solving the satisfiability problem for Monadic Second-Order Logic on finite words.

Sub-volume B of Volume-B research, in contrast, is concerned with semantical and methodological foundations for programming and programming languages. Domain theory, for example, studies special kinds of partially ordered sets called domains. Domain theory is used to specify denotational semantics, especially for functional programming languages. Category theory, as another example, formalizes mathematical structure and its concepts in terms of a collection of objects and of arrows (also called morphisms). Category theory provides powerful modeling idioms and has deep connections to types in programming languages. Concurrency theory studies formalisms for modeling and analyzing concurrent systems. Proof Theory is of major interest in Sub-volume B of Volume B research, and a distinguished result is the Curry-Howard Correspondence, which provides a direct relationship between types in computer programs and formal proofs in certain logics.

The split between Sub-volumes A and B within Volume-B research can perhaps be traced to the standard division of mathematical logic into several branches: computability theory, model theory, proof theory, and set theory. (See the 1989 Handbook of Mathematical Logic, with John Barwise as editor.) While set theory

has no clear computer-science counterpart, Sub-volume A of Volume-B research can be traced to computability theory and model theory, while Sub-volume B of Volume-B research can be traced to proof theory. Indeed, a scientific discipline, as it grows and matures, inevitably grows branches, which gradually grow apart from each other. As scientists are forced to go deeper, it becomes gradually impossible for them to keep track of developments in more than a very small number of branches. In fact different branches develop their own specialized languages, impeding communication between branches.

It is often at the interfaces between branches, however, that the most exciting developments occur. Consider Artificial Intelligence, for example. Since the establishment of the field in the late 1950s, logic has played a key role as the fundamental formalism for describing reasoning. Ultimately, however, logical tools were not fully adequate to capture the common-sense reasoning that characterizes human reasoning. In the 21st Century, probabilistic and statistical approaches have become dominant, for example, in machine learning. Synthesizing the logical and probabilistic approaches is a new frontier, where I expect to see many exciting developments in the next few years.

Finally, while 25 years ago computing-research took place mostly in North America and Western Europe, computing research has since globalized. The Atlantic Ocean is no longer as dominant as it used to be. I look forward to the day when we will write about "The Two Sides of the Pacific/Indian Ocean in Logic and Computation."