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 trans-national 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.