A tale of two computer scientists

Dale Miller

Inria Saclay &
LIX, École Polytechnique
Palaiseau, France

Logic Mentoring Workshop
17 February 2023

Family

Nadia, Alexis, & Catuscia Palamidessi (CP)

Mathematical Structures in Computer Science

Article contents

Extract

A special issue on structural proof theory, automated reasoning and computation in celebration of Dale Miller's 60th birthday

Published online by Cambridge University Press: **08 October 2019**

David Baelde, Amy Felty, Gopalan Nadathur and Alexis Saurin

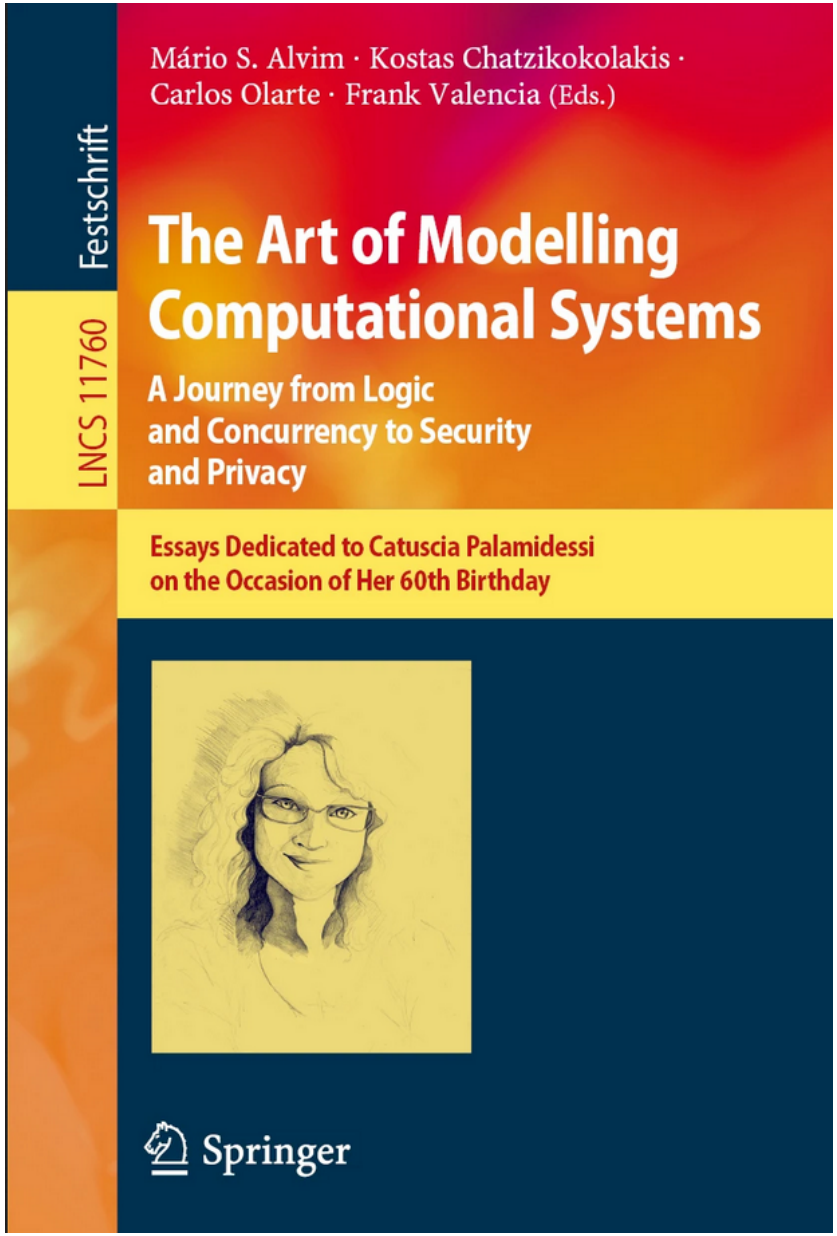
Article | Metrics

Save PDF | Share | Cite | Rights & Permissions

Extract

HTML view is not available for this content. However, as you have access to this content, a full PDF is available via the 'Save PDF' action button.

The genesis of this special issue was in a meeting that took place at Université Paris Diderot on December 15 and 16, 2016. Dale Miller, Professor at École polytechnique, had turned 60 a few days earlier. In a career spanning over three decades and in work conducted in collaboration with several students and colleagues, Dale had had a significant influence in an area that can be described as structural proof theory and its application to computation and reasoning. In recognition of this fact, several of his collaborators thought it appropriate to celebrate the occasion by organizing a symposium on topics broadly connected to his areas of interest and achievements. The meeting was a success in several senses: it was attended by over 35 people, there were 15 technical presentations describing new results, and, quite gratifyingly, we managed to spring the event as a complete surprise to Dale.



Some dates in the past 40 years

1983 DM: PhD in Mathematics, Carnegie Mellon, Proofs in higher-order logic

1983-1997 DM: faculty at the University of Pennsylvania

Some dates in the past 40 years

1983	DM: PhD in Mathematics, Carnegie Mellon, Proofs in higher-order logic
1983-1997	DM: faculty at the University of Pennsylvania
1988	CP: PhD in Informatics, University of Pisa, logic programming.
1988-1997	CP: faculties of the universities of Pisa & Genoa
1988-1991	CP: visiting positions, CWI, Amsterdam

Some dates in the past 40 years

1983	DM: PhD in Mathematics, Carnegie Mellon, Proofs in higher-order logic
1983-1997	DM: faculty at the University of Pennsylvania
1988	CP: PhD in Informatics, University of Pisa, logic programming.
1988-1997	CP: faculties of the universities of Pisa & Genoa
1988-1991	CP: visiting positions, CWI, Amsterdam
1996	Married in Pisa
1996-1997	DM: sabbatical in Genoa

Some dates in the past 40 years

1983	DM: PhD in Mathematics, Carnegie Mellon, Proofs in higher-order logic
1983-1997	DM: faculty at the University of Pennsylvania
1988	CP: PhD in Informatics, University of Pisa, logic programming.
1988-1997	CP: faculties of the universities of Pisa & Genoa
1988-1991	CP: visiting positions, CWI, Amsterdam
1996	Married in Pisa
1996-1997	DM: sabbatical in Genoa
1997-2002	Professors at Penn State University
1999	Birth of daughter (USA)

Some dates in the past 40 years

1983	DM: PhD in Mathematics, Carnegie Mellon, Proofs in higher-order logic
1983-1997	DM: faculty at the University of Pennsylvania
1988	CP: PhD in Informatics, University of Pisa, logic programming.
1988-1997	CP: faculties of the universities of Pisa & Genoa
1988-1991	CP: visiting positions, CWI, Amsterdam
1996	Married in Pisa
1996-1997	DM: sabbatical in Genoa
1997-2002	Professors at Penn State University
1999	Birth of daughter (USA)
2002-	Directors of Research, Inria Saclay, France
2003	Birth of son (France)

Cultural backgrounds

We come from non-academic backgrounds.

CP: grew up in Tuscany in the 1960/70s with a rather negative view of the roles of woman in society.

DM: grew up in central Pennsylvania in the 1960/70s in a very conservative, quiet, static community.

We both dreamed of being more.

Some successes with school and teachers lead us to consider academics. Both of us knew very little about academics.

CP: thought God created professors.

DM: thought they were remarkable and exotic people.

On additive properties of general sequences

P. Erdős, A. Sárközy [1], V.T. Sós [*, 1]

Mathematical Institute of the Hungarian Academy of Sciences, H-1053 Budapest, Reáltanoda u. 13–15, Hungary

On additive properties of general sequences

P. Erdős, A. Sárközy [1], V.T. Sós [*, 1]

Mathematical Institute of the Hungarian Academy of Sciences, H-1053 Budapest, Reáltanoda u. 13–15, Hungary

SET EXISTENCE PROPERTY FOR INTUITIONISTIC THEORIES WITH DEPENDENT CHOICE

Harvey M. FRIEDMAN

Department of Mathematics, Ohio State University, Columbus, OH 43210, USA

Andrej ŠČEDROV

Department of Mathematics, University of Pennsylvania, Philadelphia, PA 19104, USA

On additive properties of general sequences

P. Erdős, A. Sárközy [1], V.T. Sós [*, 1]

Mathematical Institute of the Hungarian Academy of Sciences, H-1053 Budapest, Reáltanoda u. 13–15, Hungary

SET EXISTENCE PROPERTY FOR INTUITIONISTIC THEORIES WITH DEPENDENT CHOICE

Harvey M. FRIEDMAN

Department of Mathematics, Ohio State University, Columbus, OH 43210, USA

Andrej ŠČEDROV

Department of Mathematics, University of Pennsylvania, Philadelphia, PA 19104, USA

Uniform proofs as a foundation for logic programming*

Dale Miller**

Computer and Information Science Department, University of Pennsylvania, Philadelphia, PA 19104, USA

Gopalan Nadathur***

Computer Science Department, Duke University, Durham, NC 27706, USA

Frank Pfenning°

Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213, USA

Andre Scedrov°°

Mathematics Department, University of Pennsylvania, Philadelphia, PA 19104, USA

Research trajectories

DM:

- theorem proving (TPS) & higher-order logic
- logic programming λProlog
- structural proof theory, linear logic
- arithmetic & model checking & theorem proving (Abella)

CP:

- logic programming & parallelism and concurrency in LP
- concurrent constraint programming, process calculus
- expressiveness and embedding & operational semantics
- separation of async/sync π-calculus, probabilistic algorithms
- differential privacy, quantitative information flow
- anonymity, privacy, fairness

Problem Solvers and Theorizers, by Gian-Carlo Rota [1]

Mathematicians can be subdivided into two types: **problem solvers** and **theorizers**. Most mathematicians are a mixture of the two.

To the **problem solver**, the supreme achievement in mathematics is the solution to a problem that had been given up as hopeless.

It matters little that the solution may be clumsy; all that counts is that it should be the first and that the proof be correct.

The mathematical concepts required to state mathematical problems are tacitly assumed to be eternal and immutable.

---

[1] an essay in *Indiscreet Thoughts*, Birkhäuser, 1997 (available online)

Problem Solvers and Theorizers, by Gian-Carlo Rota [1]

Mathematicians can be subdivided into two types: **problem solvers** and **theorizers**. Most mathematicians are a mixture of the two.

To the **theorizer**, the supreme achievement of mathematics is a theory that sheds sudden light on some incomprehensible phenomenon.

Success in mathematics does not lie in solving problems but in their trivialization. The moment of glory comes with the discovery of a new theory that does not solve any of the old problems but renders them irrelevant.

To the **theorizer**, the only mathematics that will survive are the definitions.

---

[1] an essay in *Indiscreet Thoughts*, Birkhäuser, 1997 (available online)

A different spectrum in Computer Science

Theory Results can last forever. However, they might not be important. Even if they are, they might take a long time to be recognized.

Design How best to package theory into an exploitable form. Think to programming languages, theorem provers, etc.

Implementation Provide an effective implementation of designs. Should you build on existing technologies or create new tools? What to do if your only algorithms are exponential?

Applications Can you address the applications you probably originally targeted? Are your solutions of good quality? Can you reason about the resulting artifacts?

An example

- **Theory**
  - Familiarity with Church's HOL
  -
  -
- **Design**
  -
  -
- **Implementation**
  -
  -
  -
  -
- **Applications**
  -
  -
  -

An example

Theory	▶ Familiarity with Church's HOL
	▶
	▶
Design	▶ λProlog: HO programming, typing, modules, etc
	▶
Implementation	▶
	▶
	▶
	▶
Applications	▶
	▶
	▶

Range of years: 1985-2022.
Most efforts were joint with G. Nadathur and other colleagues.

An example

Theory	▶ Familiarity with Church's HOL
	▶
	▶
Design	▶ λProlog: HO programming, typing, modules, etc
	▶
Implementation	▶ Prototype - slow but convincing
	▶
	▶
	▶
Applications	▶
	▶
	▶

Range of years: 1985-2022.
Most efforts were joint with G. Nadathur and other colleagues.

An example

Theory	▶ Familiarity with Church's HOL
	▶ Unification under a mixed prefix
	▶
Design	▶ λProlog: HO programming, typing, modules, etc
	▶
Implementation	▶ Prototype - slow but convincing
	▶
	▶
	▶
Applications	▶
	▶
	▶

Range of years: 1985-2022.
Most efforts were joint with G. Nadathur and other colleagues.

An example

Theory	▶ Familiarity with Church's HOL
	▶ Unification under a mixed prefix
	▶
Design	▶ λProlog: HO programming, typing, modules, etc
	▶
Implementation	▶ Prototype - slow but convincing
	▶
	▶
	▶
Applications	▶ Specification of operational semantics
	▶ Tactics/tacticals in theorem proving
	▶

Range of years: 1985-2022.
Most efforts were joint with G. Nadathur and other colleagues.

An example

Theory	▶ Familiarity with Church's HOL
	▶ Unification under a mixed prefix
	▶
Design	▶ λProlog: HO programming, typing, modules, etc
	▶
Implementation	▶ Prototype - slow but convincing
	▶ Teyjus: explicit subst, abstract machine
	▶
	▶
Applications	▶ Specification of operational semantics
	▶ Tactics/tacticals in theorem proving
	▶

Range of years: 1985-2022.
Most efforts were joint with G. Nadathur and other colleagues.

An example

Theory	▶ Familiarity with Church's HOL
	▶ Unification under a mixed prefix
	▶ Pattern unification
Design	▶ λProlog: HO programming, typing, modules, etc
	▶
Implementation	▶ Prototype - slow but convincing
	▶ Teyjus: explicit subst, abstract machine
	▶
	▶
Applications	▶ Specification of operational semantics
	▶ Tactics/tacticals in theorem proving
	▶

Range of years: 1985-2022.
Most efforts were joint with G. Nadathur and other colleagues.

An example

Theory	▶ Familiarity with Church's HOL
	▶ Unification under a mixed prefix
	▶ Pattern unification
Design	▶ λProlog: HO programming, typing, modules, etc
	▶
Implementation	▶ Prototype - slow but convincing
	▶ Teyjus: explicit subst, abstract machine
	▶ Teyjus V2: only pattern unification
	▶
Applications	▶ Specification of operational semantics
	▶ Tactics/tacticals in theorem proving
	▶

Range of years: 1985-2022.
Most efforts were joint with G. Nadathur and other colleagues.

An example

Theory	▶ Familiarity with Church's HOL
	▶ Unification under a mixed prefix
	▶ Pattern unification
Design	▶ λProlog: HO programming, typing, modules, etc
	▶
Implementation	▶ Prototype - slow but convincing
	▶ Teyjus: explicit subst, abstract machine
	▶ Teyjus V2: only pattern unification
	▶ ELPI
Applications	▶ Specification of operational semantics
	▶ Tactics/tacticals in theorem proving
	▶

Range of years: 1985-2022.
Most efforts were joint with G. Nadathur and other colleagues.
Development of ELPI by E. Tassi & C. Sacerdoti Coen.

An example

Theory	▶ Familiarity with Church's HOL ▶ Unification under a mixed prefix ▶ Pattern unification
Design	▶ λProlog: HO programming, typing, modules, etc ▶ Coq-ELPI plugin
Implementation	▶ Prototype - slow but convincing ▶ Teyjus: explicit subst, abstract machine ▶ Teyjus V2: only pattern unification ▶ ELPI
Applications	▶ Specification of operational semantics ▶ Tactics/tacticals in theorem proving ▶

Range of years: 1985-2022.
Most efforts were joint with G. Nadathur and other colleagues.
Development of ELPI by E. Tassi & C. Sacerdoti Coen.

An example

Theory	▶ Familiarity with Church's HOL ▶ Unification under a mixed prefix ▶ Pattern unification
Design	▶ λProlog: HO programming, typing, modules, etc ▶ Coq-ELPI plugin
Implementation	▶ Prototype - slow but convincing ▶ Teyjus: explicit subst, abstract machine ▶ Teyjus V2: only pattern unification ▶ ELPI
Applications	▶ Specification of operational semantics ▶ Tactics/tacticals in theorem proving ▶ Hierarchy builder via Coq-ELPI

Range of years: 1985-2022.
Most efforts were joint with G. Nadathur and other colleagues.
Development of ELPI by E. Tassi & C. Sacerdoti Coen.

Research and teaching balance

We both see ourselves as researchers first.
During half of our careers, we had the typical teaching loads of American and Italian professors.
As a junior professor, I found teaching useful for me.

- It helped me learn CS culture.
- When research yielded no results for months, I could put a day's extra work into teaching, and that produced immediate satisfaction.

As senior researchers in France, we advise students on Ph.D.s and elect to do some small amount of teaching.

I find other ways to deal with months without research results.

- hack on prototypes
- work on writing a monograph or textbook

Research and administrative balance

- Ph. D. advising
- Conference & journal reviewing, program committee member
- Program committee chair, general chair
- Journal editors, editor-in-chief, special interest groups, etc
- Team leader, department head

Research and administrative balance

- Ph. D. advising
- Conference & journal reviewing, program committee member
- Program committee chair, general chair
- Journal editors, editor-in-chief, special interest groups, etc
- Team leader, department head
- Director of laboratory, of Graduate School, etc.
- Dean, provost, university president,
- Directory of CNRS/Inria, advisor to the President, etc.

Research and administrative balance

- Ph. D. advising
- Conference & journal reviewing, program committee member
- Program committee chair, general chair
- Journal editors, editor-in-chief, special interest groups, etc
- Team leader, department head
- Director of laboratory, of Graduate School, etc.
- Dean, provost, university president,
- Directory of CNRS/Inria, advisor to the President, etc.

The academic world is a striking environment, shaped largely by peer reviewing, volunteering for these jobs, etc. Governments generally apply only indirect influence based on funding.

Moving to a teaching and/or administration emphasis is a valuable trajectory for researchers to consider.

Some perspectives that have been useful for me

- I envisioned my Ph.D. dissertation as my first and worst piece of work. I would move upwards from there. This thought helped me write that first document.

Some perspectives that have been useful for me

- I envisioned my Ph.D. dissertation as my first and worst piece of work. I would move upwards from there. This thought helped me write that first document.
- Will my work last?

Some perspectives that have been useful for me

- I envisioned my Ph.D. dissertation as my first and worst piece of work. I would move upwards from there. This thought helped me write that first document.
- Will my work last? Hard to tell, of course, but:
  - Test your results in a richer setting:
    - move from $\mathcal{R}$ to $\mathcal{R}^n$; e.g., Euler-Poincaré formula $V - E + F = 2$ vs $f_0 - f_1 + f_2 - \cdots + (-1)^d f_d = 1$.
    - from first-order to higher-order quantification; etc.

Some perspectives that have been useful for me

- I envisioned my Ph.D. dissertation as my first and worst piece of work. I would move upwards from there. This thought helped me write that first document.
- Will my work last? Hard to tell, of course, but:
  - Test your results in a richer setting:
    - move from $\mathcal{R}$ to $\mathcal{R}^n$; e.g., Euler-Poincaré formula $V - E + F = 2$ vs $f_0 - f_1 + f_2 - \cdots + (-1)^d f_d = 1$.
    - from first-order to higher-order quantification; etc.
  - Move along the theory-design-implementation-application spectrum as much as possible.

Some perspectives that have been useful for me

- I envisioned my Ph.D. dissertation as my first and worst piece of work. I would move upwards from there. This thought helped me write that first document.
- Will my work last? Hard to tell, of course, but:
  - Test your results in a richer setting:
    - move from $\mathcal{R}$ to $\mathcal{R}^n$; e.g., Euler-Poincaré formula $V - E + F = 2$ vs $f_0 - f_1 + f_2 - \cdots + (-1)^d f_d = 1$.
    - from first-order to higher-order quantification; etc.
  - Move along the theory-design-implementation-application spectrum as much as possible.
  - People keep returning to logic and formalizations. This is a topics that will last. I see myself as trying to contribute to that tradition.

Some perspectives that have been useful for me

- I envisioned my Ph.D. dissertation as my first and worst piece of work. I would move upwards from there. This thought helped me write that first document.
- Will my work last? Hard to tell, of course, but:
  - Test your results in a richer setting:
    - move from $\mathcal{R}$ to $\mathcal{R}^n$; e.g., Euler-Poincaré formula $V - E + F = 2$ vs $f_0 - f_1 + f_2 - \cdots + (-1)^d f_d = 1$.
    - from first-order to higher-order quantification; etc.
  - Move along the theory-design-implementation-application spectrum as much as possible.
  - People keep returning to logic and formalizations. This is a topics that will last. I see myself as trying to contribute to that tradition.
- Read *Proofs and Refutations* by Imre Lakatos (1976).

Questions?

Art by Nadia Miller