Formal methods at MPRI

There is a bug

Just close the window and start again

But not possible with critical systems: transportation, energy, medicine, finance, blockchain...

Our industrial partners: Airbus, NASA, RATP, CEA...

Why bugs?

Engineers are able to build mills, cathedrals, bridges... with no (few) bugs
We are not

Programs are more complex than mills
A mill: 10^2 different parts, a program: 10^8 lines of code

But also Programming languages (e.g. in machine language) do not help us to write correct programs
Programs: do, do, do... what for?

An old question

First "high level" programming languages (Fortran, Lisp, Cobol...): easier and safer to use

Not only bugs

but also malicious attacks: systems malfunction because someone wanted to

Safety and security

Safety is the basis of security

Express a property your program must verify

Be sure it is true for your program
E.g.

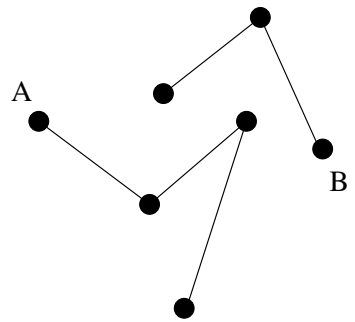
```
r = a; while (r > b) do r = r - b;
```

Specification:

$$\forall a \forall b \exists q \exists r \ (a = b \times q + r \wedge r < b)$$

What does "being true" means?

Logic has given answers since Aristotle (one connection between logic and informatics)
Diversity of answers, diversity of formal methods



"A and B are not connected" is true
Because you can prove it
Because you can check it

Proving and checking

Proving: more general: What does it mean to check Fermat's little theorem?

Checking: more efficient: An algorithm can do it

Both are useful but not for the same problems

Another reason for diversity

What should we do with C, Python, Java...?
(1) Forget them: our job is to invent better programming languages, e.g. including rich type systems that express properties of programs
(2) Wake up: these programs are there and you are not going to start the computer age again

Related: when should we make sure the program meets its specification?
(2) After it is written: the programs are there and...
(1) Before: the specification should guide development

Both are useful but on a different time scale

Yet another reason for diversity

Can we reason about programs with the same logic used for circles and triangles?

Yes: it is the logic (logic is universal)

No: programs are very specific objects Euclid, Aristotle and Frege did not have idea of (in particular they care about facts, we care about transformations)

A (attempt of) consensus: the (universal) logic has an history anyway

Who should make sure the programs meet their specifications?

The / another programmer (code auditing very expensive)
A machine

Proofs, models, programs are to large to be processed (without mistake) by a human
Humans: buggy proofs for buggy programs

Using computers to check proofs (proofs of programs, but also general mathematical proofs)
You build the proof the computer checks: use also the computer to help you: automated theorem proving

Formal methods explained to your grandmother

Four concepts in informatics:
Machine, algorithm, language, data

Formal method: the part of informatics focused on language:
$\lambda o \gamma o \varsigma$ (another connection between logic and informatics)

Formal methods at MPRI

Programs

2.2 Models of programming languages: domains, categories, games (Michele Pagani's talk)
2.4 Functional programming and type systems
2.23.1 Programming of synchronous systems
~~2.35.1 Constraint programming~~
2.37.1 Semantics, languages and algorithms for multicore programming

Proofs

2.1 Linear logic and logical paradigms of computation (Michele Pagani's talk)
2.5.1 ~~Automated deduction~~
2.7.1 Foundations of proof systems
2.7.2 Proof assistants
2.36.1 Proofs of programs

Model checking

2.8.1 Non-sequential theory of distributed systems
2.8.2 Foundations of real time and hybrid systems
2.9.1 Well-Quasi-Orders for Algorithms
2.9.2 Algorithmic verification of programs
2.16 Finite automata modelling

See Laurent Fribourg's talk

Security

~~2.3.2 Foundations of privacy~~
2.30 Cryptographic protocols: computational and symbolic proofs

Static analysis

2.6 Abstract interpretation: application to verification and static analysis