Logic in program analysis and verification

Patrick Cousot

NYU, New York

pcousot@cs.nyu.edu   cs.nyu.edu/~pcousot

Sunday, Nov 15$^{\text{th}}$, 2020

Subject of discussion

- For program specification and verification, logic is a natural choice.
- However, for static analysis, logic is rarely used, even as a user interface.
- We briefly discuss the weaknesses of logic from this static analysis perspective.

Which logic for specification?

Specification

- decidable logics (such as Presburger arithmetic [12]):
  - validity can be mechanically checked
  - incomplete (the invariant of a program that computes the multiplication $*$ using iteration and addition $+$ is not expressible)
- first-order logic:
  - undecidable (user-interaction is needed for proofs)
  - incomplete (no recursion mechanism, transitive closure is not expressible [11])
- higher-order logic:
  - necessary to discuss the relative completeness go Hoare logic
  - necessary to discuss the soundness of static analyzers (e.g. hyperproperties in $\wp(\wp(S))$ where S is the semantic domain)

Which logic for property representation in a static analyzer?

Internal representation of abstract properties

- great advantage: uniform representation by (the abstract syntax) of a formula in the logic
  - many operations have simple implementations (e.g. connectives)
  - exploited in the static analysis of Prolog [10]
- great disadvantage: uniformity
  - no (useful) normal form
  - efficient algorithms require specific representations (e.g. matrices+systems of generators for linear equalities or inequalities [8])
  - algorithmically, syntax-based representation uniformity is not tenable

Abstract domains

Abstract domains

- order-theoretic/algebraic concept of properties (representation + operations)
- hard to translate in logic (e.g. how to express "to be a number between a and b")
- the semantics is formally defined by concretization to sets
- operations (e.g. logical connectives, transformers) are (predictable and efficient) algorithms

- in logic, the failure of theorem provers or SMT solvers may be very hard to explain [9]

Combinations of abstract domains

- the uniformity of representation of properties is lost with abstract domains
- combinations of abstract domains handle non-uniform representations
- communication of shared information between abstract domains

- example: the reduced product [3] for conjunction
- the combination of theories in SMT solvers is a reduced product [5] (the shared information is equalities and disqualifies for Nelson-Oppen [13])

Induction

Proofs by induction

- infering inductive arguments in proofs is the basis for verification and analysis of programs
- asking the users for induction hypotheses makes verification simpler than program analysis [6]
- hardly scale up (invariants are much larger than programs [4])

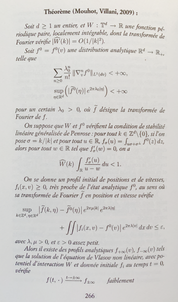
- induction in logic is predefined
- no mechanism in logic to specify how to automate approximate induction or co-induction
- the complexity of an object and its logical denotation may be completely unrelated.

Extrapolation and interpolation

- induction tailored to a level of abstraction [1]
- often based on geometric considerations (e.g. widenings extrapolate in the direction of growth)
- finitary abstract domains are not expressive [2] (e.g. liquid types [14])
- the evolution of the iterates is monitored for induction [7]

Conclusion

- logic reduces the representations of properties and formal reasonings to purely syntactic manipulations (copy/paste :)
- this is great for logicians to reason about proofs ($\neq$ making proofs)
- mathematicians do not use logics to make proofs



Théorème (Mouhot, Villani, 2009) :

Soit $d \geq 1$ *un entier, et* $W : \mathbb{T}^d \to \mathbb{R}$ *une fonction périodique paire, localement intégrable, dont la transformée de Fourier vérifie* $|\widehat{W}(k)| = O(1/|k|^2)$.

Soit $f^0 = f^0(v)$ *une distribution analytique* $\mathbb{R}^d \to \mathbb{R}_+$, *telle que*

$$\sum_{n \geq 0} \frac{\lambda_0^n}{n!} \|\nabla_v^n f^0\|_{L^1(dv)} < +\infty,$$

$$\sup_{\eta \in \mathbb{R}^d} \left(|\widetilde{f}^0(\eta)| \, e^{2\pi\lambda_0|\eta|} \right) < +\infty$$

pour un certain $\lambda_0 > 0$, *où* $\widetilde{f}$ *désigne la transformée de Fourier de* f.

On suppose que W *et* f^0 *vérifient la condition de stabilité linéaire généralisée de Penrose : pour tout* $k \in \mathbb{Z}^d \setminus \{0\}$, *si l'on pose* $\sigma = k/|k|$ *et pour tout* $u \in \mathbb{R}$, $f_\sigma(u) = \int_{u\sigma + \sigma^\perp} f^0(z)\, dz$, *alors pour tout* $w \in \mathbb{R}$ *tel que* $f'_\sigma(w) = 0$, *on a*

$$\widehat{W}(k) \int_{\mathbb{R}} \frac{f'_\sigma(u)}{u - w} \, du < 1.$$

On se donne un profil initial de positions et de vitesses, $f_i(x, v) \geq 0$, *très proche de l'état analytique* f^0, *au sens où sa transformée de Fourier* $\widetilde{f}$ *en position et vitesse vérifie*

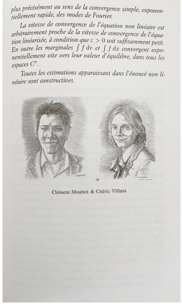
$$\sup_{k \in \mathbb{Z}^d, \eta \in \mathbb{R}^d} \left| \widetilde{f}(k, \eta) - \widetilde{f}^0(\eta) \right| e^{2\pi\mu|k|} \, e^{2\pi\lambda|\eta|} + \iint \left| f_i(x, v) - f^0(v) \right| e^{2\pi\lambda|v|} \, dx \, dv \leq \varepsilon,$$

avec $\lambda, \mu > 0$, *et* $\varepsilon > 0$ *assez petit.*

Alors il existe des profils analytiques $f_{+\infty}(v)$, $f_{-\infty}(v)$ *tels que la solution de l'équation de Vlasov non linéaire, avec potentiel d'interaction* W *et donnée initiale* f_i *au temps* $t = 0$, *vérifie*

$$f(t, \cdot) \xrightarrow{t \to \pm\infty} f_{\pm\infty} \qquad \textit{faiblement}$$

266



plus précisément au sens de la convergence simple, exponentiellement rapide, des modes de Fourier.

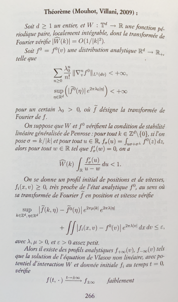
La vitesse de convergence de l'équation non linéaire est arbitrairement proche de la vitesse de convergence de l'équation linéarisée, à condition que $\varepsilon > 0$ *soit suffisamment petit. En outre les marginales* $\int f \, dv$ *et* $\int f \, dx$ *convergent exponentiellement vite vers leur valeur d'équilibre, dans tous les espaces* C^r.

Toutes les estimations apparaissant dans l'énoncé non linéaire sont constructives.

Clément Mouhot & Cédric Villani

Conclusion

- logic reduces the representations of properties and formal reasonings to purely syntactic manipulations (copy/paste :)
- this is great for logicians to reason about proofs ($\neq$ making proofs)
- mathematicians do not use logics to make proofs



Théorème (Mouhot, Villani, 2009) :

Soit $d \geq 1$ *un entier, et* $W : \mathbb{T}^d \to \mathbb{R}$ *une fonction périodique paire, localement intégrable, dont la transformée de Fourier vérifie* $|\widehat{W}(k)| = O(1/|k|^2)$.

Soit $f^0 = f^0(v)$ *une distribution analytique* $\mathbb{R}^d \to \mathbb{R}_+$, *telle que*

$$\sum_{n\geq 0} \frac{\lambda_0^n}{n!} \|\nabla_v^n f^0\|_{L^1(dv)} < +\infty,$$

$$\sup_{\eta\in\mathbb{R}^d} \left(|\widetilde{f}^0(\eta)|\, e^{2\pi\lambda_0|\eta|}\right) < +\infty$$

pour un certain $\lambda_0 > 0$, *où* $\widetilde{f}$ *désigne la transformée de Fourier de* f.

On suppose que W *et* f^0 *vérifient la condition de stabilité linéaire généralisée de Penrose : pour tout* $k \in \mathbb{Z}^d \setminus \{0\}$, *si l'on pose* $\sigma = k/|k|$ *et pour tout* $u \in \mathbb{R}$, $f_\sigma(u) = \int_{u\sigma+\sigma^\perp} f^0(z)\, dz$, *alors pour tout* $w \in \mathbb{R}$ *tel que* $f'_\sigma(w) = 0$, *on a*

$$\widehat{W}(k) \int_{\mathbb{R}} \frac{f'_\sigma(u)}{u - w}\, du < 1.$$

On se donne un profil initial de positions et de vitesses, $f_i(x, v) \geq 0$, *très proche de l'état analytique* f^0, *au sens où sa transformée de Fourier* $\widetilde{f}$ *en position et vitesse vérifie*

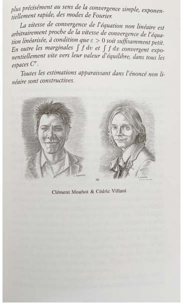
$$\sup_{k\in\mathbb{Z}^d,\, \eta\in\mathbb{R}^d} \left|\widetilde{f}(k,\eta) - \widetilde{f}^0(\eta)\right| e^{2\pi\mu|k|}\, e^{2\pi\lambda|\eta|} + \iint \left|f_i(x,v) - f^0(v)\right| e^{2\pi\lambda|v|}\, dx\, dv \leq \varepsilon,$$

avec $\lambda, \mu > 0$, *et* $\varepsilon > 0$ *assez petit.*

Alors il existe des profils analytiques $f_{+\infty}(v)$, $f_{-\infty}(v)$ *tels que la solution de l'équation de Vlasov non linéaire, avec potentiel d'interaction* W *et donnée initiale* f_i *au temps* $t = 0$, *vérifie*

$$f(t,\cdot) \xrightarrow{t\to\pm\infty} f_{\pm\infty} \qquad \textit{faiblement}$$

266



plus précisément au sens de la convergence simple, exponentiellement rapide, des modes de Fourier.

La vitesse de convergence de l'équation non linéaire est arbitrairement proche de la vitesse de convergence de l'équation linéarisée, à condition que $\varepsilon > 0$ *soit suffisamment petit. En outre les marginales* $\int f\, dv$ *et* $\int f\, dx$ *convergent exponentiellement vite vers leur valeur d'équilibre, dans tous les espaces* C^r.

Toutes les estimations apparaissant dans l'énoncé non linéaire sont constructives.

Clément Mouhot & Cédric Villani

- computer scientists do, maybe that's the problem

The End, Thank you

Bibliography I

[1] Patrick Cousot.
Abstracting induction by extrapolation and interpolation.
In *VMCAI*, volume 8931 of *Lecture Notes in Computer Science*, pages 19–42. Springer, 2015.

[2] Patrick Cousot and Radhia Cousot.
Comparing the Galois connection and widening/narrowing approaches to abstract interpretation.
In *PLILP*, volume 631 of *Lecture Notes in Computer Science*, pages 269–295. Springer.

[3] Patrick Cousot and Radhia Cousot.
Systematic design of program analysis frameworks.
In *POPL*, pages 269–282. ACM Press, 1979.

[4] Patrick Cousot, Radhia Cousot, Jérôme Feret, Laurent Mauborgne, Antoine Miné, and Xavier Rival.
Why does Astrée scale up?
Formal Methods in System Design, 35(3):229–264, 2009.

[5] Patrick Cousot, Radhia Cousot, and Laurent Mauborgne.
Theories, solvers and static analysis by abstract interpretation.
J. ACM, 59(6):31:1–31:56, 2012.

[6] Patrick Cousot, Roberto Giacobazzi, and Francesco Ranzato.
Program analysis is harder than verification: A computability perspective.
In *CAV (2)*, volume 10982 of *Lecture Notes in Computer Science*, pages 75–95. Springer, 2018.

[7] Patrick Cousot, Roberto Giacobazzi, and Francesco Ranzato.
A^2i: abstract^2 interpretation.
Proc. ACM Program. Lang., 3(POPL):42:1–42:31, 2019.

[8] Patrick Cousot and Nicolas Halbwachs.
Automatic discovery of linear restraints among variables of a program.
In *POPL*, pages 84–96. ACM Press, 1978.

Bibliography II

[9] Leonardo Mendonça de Moura and Grant Olney Passmore.
The strategy challenge in SMT solving.
In *Automated Reasoning and Mathematics*, volume 7788 of *Lecture Notes in Computer Science*, pages 15–44. Springer, 2013.

[10] Isabel Garcia-Contreras, José F. Morales, and Manuel V. Hermenegildo.
Incremental analysis of logic programs with assertions and open predicates.
In *LOPSTR*, volume 12042 of *Lecture Notes in Computer Science*, pages 36–56. Springer, 2019.

[11] Erich Grädel.
On transitive closure logic.
In *CSL*, volume 626 of *Lecture Notes in Computer Science*, pages 149–163. Springer, 1991.

[12] Christoph Haase.
A survival guide to Presburger arithmetic.
ACM SIGLOG News, 5(3):67–82, 2018.

[13] Greg Nelson and Derek C. Oppen.
Simplification by cooperating decision procedures.
ACM Trans. Program. Lang. Syst., 1(2):245–257, 1979.

[14] Patrick Maxim Rondon, Ming Kawaguchi, and Ranjit Jhala.
Liquid types.
In *PLDI*, pages 159–169. ACM, 2008.