Global optimization in Coq

Present and future work in ValidSDP

Érik Martin-Dorel[1] Pierre Roux[2]

[1]IRIT, Université Paul Sabatier, Toulouse, France

[2]ONERA, Toulouse, France

Jeudi 7 Juin 2018

Journées FastRelax, Inria Sophia Antipolis

Overview

ValidSDP:

- A reflexive tactic (`validsdp`) for polynomial positivity using numerical solvers and floating-point computations
  [CPP'2017, https://hal.archives-ouvertes.fr/hal-01510979]

⇝ semi-decision procedure `validsdp` presented at FastRelax last year:
[http://fastrelax.gforge.inria.fr/files/2017/martin-dorel.pdf]

Overview

ValidSDP:

- A reflexive tactic (`validsdp`) for polynomial positivity using numerical solvers and floating-point computations
  [CPP'2017, https://hal.archives-ouvertes.fr/hal-01510979]

⇝ semi-decision procedure `validsdp` presented at FastRelax last year:
[http://fastrelax.gforge.inria.fr/files/2017/martin-dorel.pdf]

This talk:

1. A new Coq tactic (`validsdp_intro`) for global optimization in Coq that accepts a large set of options
2. Quick overview of the implementation
3. Discussion on possible future work

Agenda

Goals accepted by validsdp

Goals of the form

$$
\begin{array}{l}
\mathtt{x_1, x_2, \ldots : R} \\
\mathtt{==========} \\
\mathtt{g_1\ x_1\ x_2\ \ldots} \lesseqgtr \mathtt{h_1\ x_1\ x_2\ \ldots} \rightarrow \\
\mathtt{g_2\ x_1\ x_2\ \ldots} \lesseqgtr \mathtt{h_2\ x_1\ x_2\ \ldots} \rightarrow \\
\mathtt{\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots} \rightarrow \\
\mathtt{p\ x_1\ x_2\ \ldots} \lesseqgtr \mathtt{q\ x_1\ x_2\ \ldots}
\end{array}
$$

Now, the polynomials $\mathtt{g}_i$, $\mathtt{h}_i$, $\mathtt{p}$, $\mathtt{q}$ need not be unfolded (as we added some support of polynomial composition in Ltac reification and Gallina).

Overview of validsdp_intro I

- Automatically prove lower/upper bounds of polynomial expressions
- Under polynomials constraints, or overy the whole real line if desired
- Introduce the resulting inequalities as a new hyp in the goal.

Overview of validsdp_intro II

Syntax:

- `validsdp_intro e [using (hyp1, ...) | using *] [with (param1, ...)] as (?| H | (Hl, Hu)).`
- `validsdp_intro e lower [using (hyp1, ...) | using *] [with (param1, ...)] as (?| Hl)`
- `validsdp_intro e upper [using (hyp1, ...) | using *] [with (param1, ...)] as (?| Hu)`

where e is a term of type R representing a multivariate polynomial expression with rational constants and real-valued variables.

Overview of validsdp_intro III

- `using (hyp1, ...)` $\rightarrow$ select the hypotheses from the context to be considered by the solver $\rightsquigarrow$ input domain of the considered optimization problem.
  These hypotheses should be multivariate polynomial inequalities with rational constants and real-valued variables
- `using *` $\rightarrow$ heuristic to select relevant inequalities from the context
- if "`using ...`" is omitted, the polynomial expression e is bounded over the whole vector space

Overview of validsdp_intro IV

- `with s_sdpa` $\rightarrow$ use the SDPA solver (as in this talk)
- `with s_csdp` $\rightarrow$ use the CSDP solver
- `with s_mosek` $\rightarrow$ use the Mosek solver
- `with s_verbose` n $\rightarrow$ set the verbosity level (default: 0)

- `as Hl` or `as (Hl, Hu)` $\rightarrow$ name of the inequalities added to the context.

Example

```
Lemma test :
  ∀x y: R, let p := fun x y ↦ 2/3 * x ^ 2 + y ^ 2 in True.
Proof.
intros x y p.
validsdp_intro (p x y + 1) lower as H.
(* ... *)
Qed.
```

Demo: https://github.com/validsdp/validsdp/blob/4704929/theories/testsuite.v

Agenda

Sum of Squares (SOS) Polynomials

Definition (SOS Polynomial)

A polynomial p is SOS if there are polynomials $q_1, \ldots, q_m$ s.t.

$$p = \sum_i q_i^2.$$

- If p SOS then $p \geq 0$

Sum of Squares (SOS) Polynomials

Definition (SOS Polynomial)

A polynomial p is SOS if there are polynomials $q_1, \ldots, q_m$ s.t.

$$p = \sum_i q_i^2.$$

- If p SOS then $p \geq 0$
- p SOS iff there exist $z := \left[1, x_0, x_1, x_0 x_1, \ldots, x_n^d\right]$ and $Q \succeq 0$ (i.e., for all $x, x^T Q x \geq 0$) s.t.

$$p = z^T Q\, z.$$

$\Rightarrow$ SOS can be encoded as semi-definite programming (SDP).

Dependencies

Coq 8.6 or 8.7 with the following libraries:

- MathComp
- Paramcoq
- CoqEAL
- SsrMultinomials
- Flocq
- Coquelicot
- Coq.Interval

OCaml library:

- OSDP

Outline of the formalization

Outline of the formalization

1. Effective multivariate polynomials (now integrated in CoqEAL)
   - CoqEAL [Cano, Cohen, Dénès, Mörtberg, Rouhling, Siles]
   - $\rightsquigarrow$ uses SSReflect and MathComp [Gonthier et al.]
   - proof: SsrMultinomials [Strub]
   - implem.: FMapAVL from Coq stdlib
   - coefficients: $\mathbb{Q}$ as `bigQ` from Coq stdlib

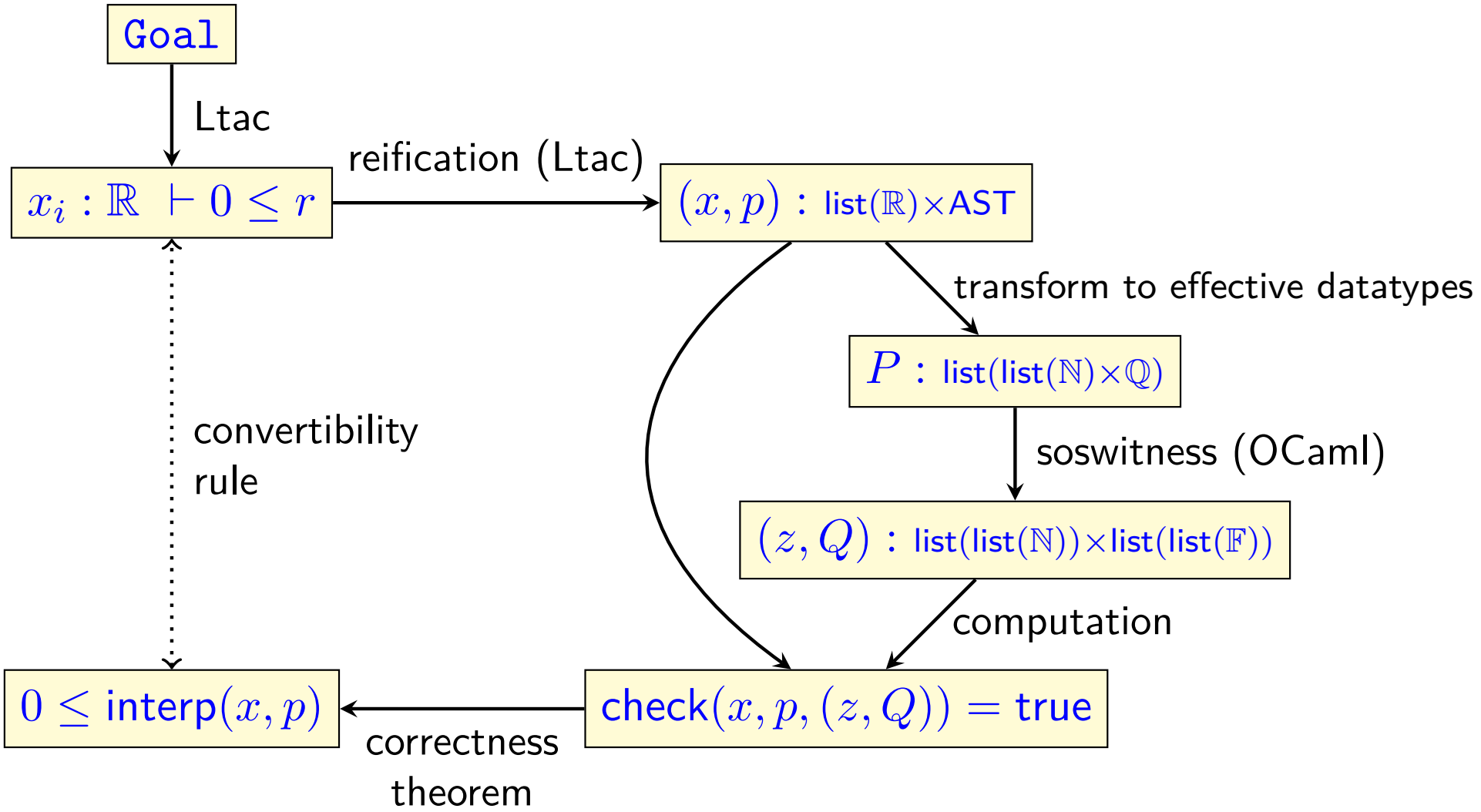
Outline of the formalization

1. Effective multivariate polynomials (now integrated in CoqEAL)
   - CoqEAL [Cano, Cohen, Dénès, Mörtberg, Rouhling, Siles]
   - $\rightsquigarrow$ uses SSReflect and MathComp [Gonthier et al.]
   - proof: SsrMultinomials [Strub]
   - implem.: FMapAVL from Coq stdlib
   - coefficients: $\mathbb{Q}$ as `bigQ` from Coq stdlib
2. Effective check for positive definite matrices
   - CoqEAL
   - proof: previous work
   - implem.: lists of lists, CoqEAL
   - coefficients: floating-point numbers from CoqInterval [Melquiond]

Outline of the formalization

1. Effective multivariate polynomials (now integrated in CoqEAL)
   - CoqEAL [Cano, Cohen, Dénès, Mörtberg, Rouhling, Siles]
   - $\rightsquigarrow$ uses SSReflect and MathComp [Gonthier et al.]
   - proof: SsrMultinomials [Strub]
   - implem.: FMapAVL from Coq stdlib
   - coefficients: $\mathbb{Q}$ as `bigQ` from Coq stdlib
2. Effective check for positive definite matrices
   - CoqEAL
   - proof: previous work
   - implem.: lists of lists, CoqEAL
   - coefficients: floating-point numbers from CoqInterval [Melquiond]
3. Reflexive tactic
   - OCaml code as a wrapper for SDP solvers
   - Ltac code using Continuation-Passing Style

The validsdp tactic – the big picture



Agenda

Conclusion

- Context: formally verified global optimization in Coq for multivariate polynomials
- Coq reflexive tactics accepting a large set of options
  - Input: polynomial expressions with real variables and rational constants
  - Use off-the-shelf SDP solvers as untrusted oracles
  - Numerical approach with formal floating-point arithmetic and Cholesky decomposition

Discussion: global optimization with elementary functions?

- On the one hand: ValidSDP (proving bounds on multivariate polynomials)
- On the other hand: Coq-Interval (proving bounds on univariate expressions with elementary functions)
- Timeliness of merging both? Realistic benchmarks examples?

Concluding remarks

- Possible extension of ValidSDP: add support of elementary functions
- Do systematic benchmarks with `validsdp_intro`
- Ltac implementation is hard to debug (untyped functional language, with CPS style)
- Currently, reification is done fully in Ltac but it would be interesting to switch to Template-Coq to increase performance
- Main bottleneck:
  floating-point arithmetic is emulated in Coq (1000x overhead)

→ Pierre and I will soon supervise an ENS student project to add native floats in Coq

- We currently use `vm_compute` but plan to provide a tactic option to use `native_compute` instead

Références

P. Roux, "Formal proofs of rounding error bounds - with application to an automatic positive definiteness check," *J. Autom. Reasoning*, vol. 57, no. 2, pp. 135–156, 2016.

Références

P. Roux, "Formal proofs of rounding error bounds - with application to an automatic positive definiteness check," *J. Autom. Reasoning*, vol. 57, no. 2, pp. 135–156, 2016.

P. Roux, Y. Voronin, and S. Sankaranarayanan, "Validating numerical semidefinite programming solvers for polynomial invariants," in *SAS 2016, Edinburgh, UK, Proceedings*, pp. 424–446, 2016.

Références


P. Roux, "Formal proofs of rounding error bounds - with application to an automatic positive definiteness check," *J. Autom. Reasoning*, vol. 57, no. 2, pp. 135–156, 2016.


P. Roux, Y. Voronin, and S. Sankaranarayanan, "Validating numerical semidefinite programming solvers for polynomial invariants," in *SAS 2016, Edinburgh, UK, Proceedings*, pp. 424–446, 2016.


E. Martin-Dorel and P. Roux, "A Reflexive Tactic for Polynomial Positivity using Numerical Solvers and Floating-Point Computations," in *CPP 2017, Paris*, pp. 90–99, ACM, janvier 2017.

Thank you!

Questions

https://github.com/validsdp/validsdp