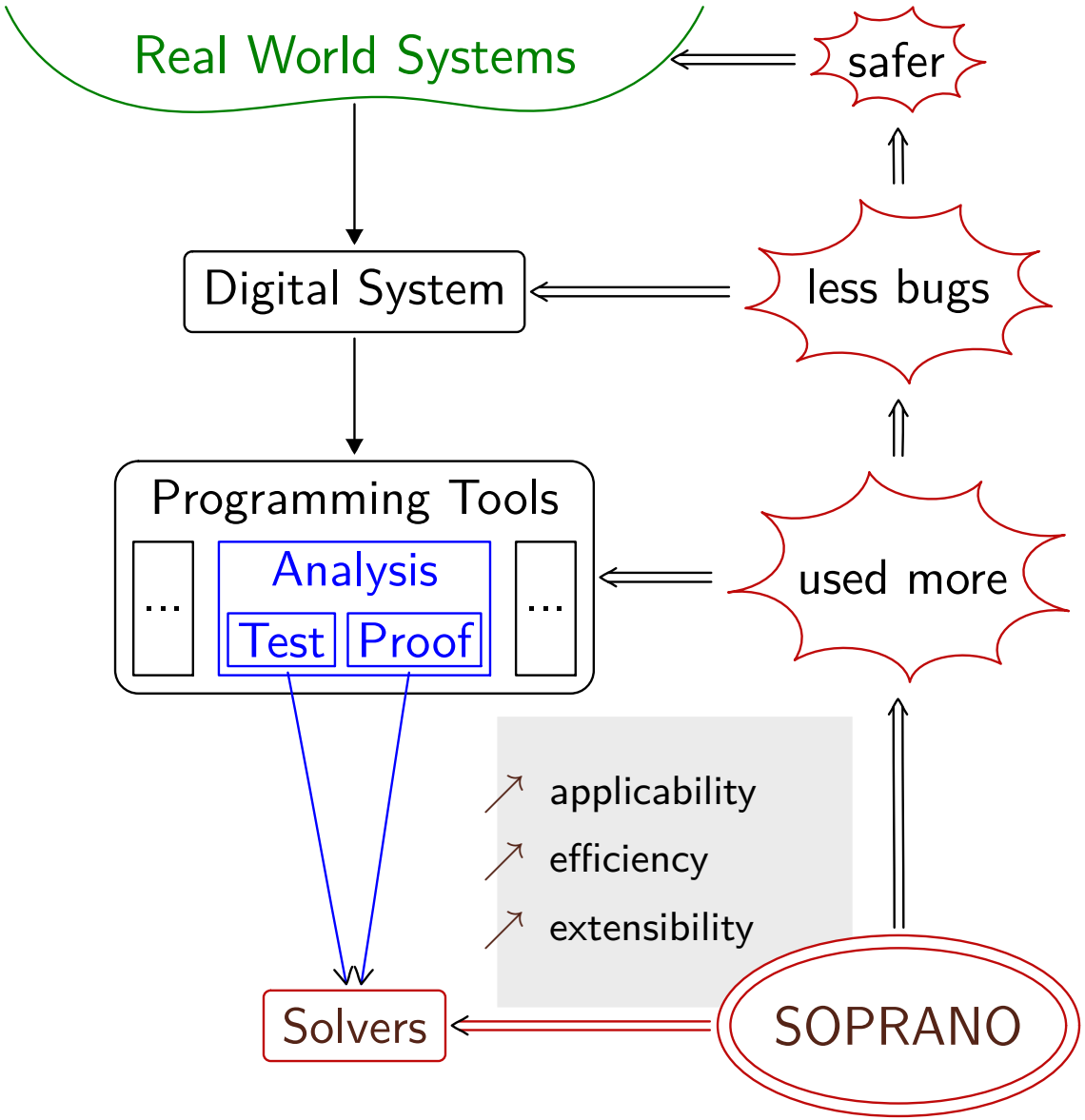
ANR-Project SOPRANO

Revue mi-parcours

4 Octobre

Context


Real World Systems
safer
Digital System
less bugs
Programming Tools
...
Analysis
Test
Proof
...
used more
applicability
efficiency
extensibility
Solvers
SOPRANO

Challenge for SPARK2014

Currently uses SMT solvers:

- ✔ Good handling of arithmetic (integers, bitvectors, reals)
- ✔ Good handling of axioms
- ✔ Reasonably fast
- ✔ Agreed semantics between all provers (SMTLIB)

Challenge for SPARK2014

Currently uses SMT solvers:

- ✔ Good handling of arithmetic (integers, bitvectors, reals)
- ✔ Good handling of axioms
- ✔ Reasonably fast
- ✔ Agreed semantics between all provers (SMTLIB)

- ✘ Many properties involving x/y, $x \times y$, x^y, $x \bmod y$, $x \operatorname{rem} y$
- ✘ Most properties involving floating-point values
- ✘ Properties involving conversions between types (integers $\leftrightarrow$ bitvectors, integers $\leftrightarrow$ reals, integers $\leftrightarrow$ floats)

Interesting and Simple Real Examples

(AdaCore)

```
1  procedure User_Rule_7 (X, Y, Z, A : Float;
                          Res         : out Boolean)
3  begin
      pragma Assume (Z ≥ 0.0);
5     pragma Assume (X ≥ Y);
      pragma Assume (Y ≥ Z);
7     pragma Assume (X > Z);
      pragma Assume (A ≥ 1.0);
9     Res := (X − Y) / (X − Z) ≤ A;
      pragma Assert (Res);       −− valid
11 end User_Rule_7;
```

✔ 25/28 examples are now solved!

Domain Specific Approach of CP

$$X_i \in [1;10] \implies X_0 \oplus X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus X_5 \oplus X_6 \oplus X_7 \in [8;80]$$

Z3 (SMT): 3s
COLIBRI (CP): 0.1s

Domain Specific Approach of CP

$$X_i \in [1;10] \implies X_0 \oplus X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus X_5 \oplus X_6 \oplus X_7 \in [8;80]$$

Z3 (SMT): 3s
COLIBRI (CP): 0.1s

$$X_i \in [1;10] \implies X_0 \otimes X_1 \otimes X_2 \otimes X_3 \otimes X_4 \otimes X_5 \otimes X_6 \otimes X_7 \in [1;10^8]$$

Z3 (SMT): 31min
COLIBRI (CP): 0.1s

The Differences between SMT and CP

SMT:

- ✔ Clear design and separation
- ✔ Fast decision procedures with learning
- ✘ Model generation requires additional work
- ✘ Hard and inefficient to mix theories

The Differences between SMT and CP

SMT:

- ✔ Clear design and separation
- ✔ Fast decision procedures with learning
- ✘ Model generation requires additional work
- ✘ Hard and inefficient to mix theories

CP:

- ✔ Trivial to generate model
- ✔ Easy and efficient to mix constraints
- ✔ Many propagators: powerful reduction
- ✘ Many propagators: difficult learning
- ✘ Few separations: hard to maintain

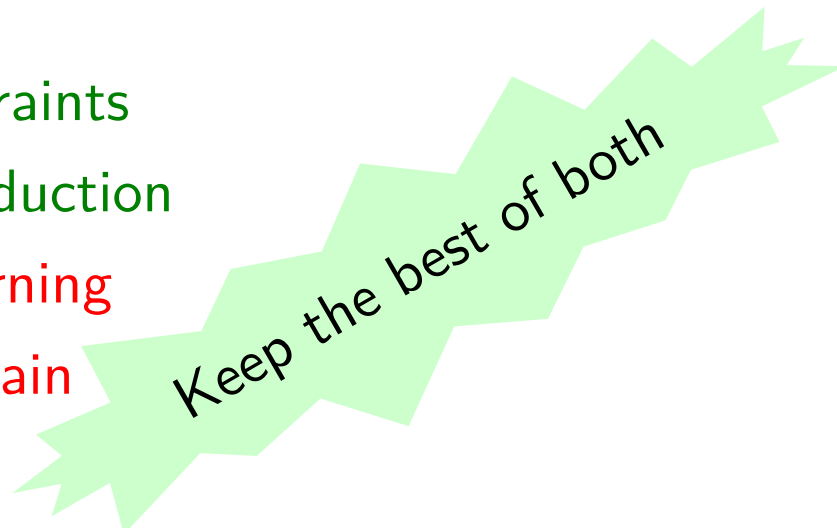
The Differences between SMT and CP

SMT:

- ✔ Clear design and separation
- ✔ Fast decision procedures with learning
- ✘ Model generation requires additional work
- ✘ Hard and inefficient to mix theories

CP:

- ✔ Trivial to generate model
- ✔ Easy and efficient to mix constraints
- ✔ Many propagators: powerful reduction
- ✘ Many propagators: difficult learning
- ✘ Few separations: hard to maintain



Objectives

- **Obj1:** design a **new collaboration framework for solvers**
- **Obj2:** design **new decision procedures**
- **Obj3:** a **new open-source platform**
- **Obj4:** ensure **industrial-adequacy**

Objectives

- **Obj1:** design a **new collaboration framework for solvers**
  - ✔ First class domain and propagation engine
- **Obj2:** design **new decision procedures**
- **Obj3:** a **new open-source platform**
- **Obj4:** ensure **industrial-adequacy**

Objectives

- **Obj1:** design a **new collaboration framework for solvers**
  - ✔ First class domain and propagation engine
  - ➠ First class conflict and learning engine
- **Obj2:** design **new decision procedures**
- **Obj3:** a **new open-source platform**
- **Obj4:** ensure **industrial-adequacy**

Objectives

- **Obj1:** design a **new collaboration framework for solvers**
  - ✔ First class domain and propagation engine
  - ➠ First class conflict and learning engine
- **Obj2:** design **new decision procedures**
  - ✔ Floating-point theory
- **Obj3:** a **new open-source platform**
- **Obj4:** ensure **industrial-adequacy**

Objectives

- **Obj1:** design a **new collaboration framework for solvers**
  - ✔ First class domain and propagation engine
  - ➠ First class conflict and learning engine
- **Obj2:** design **new decision procedures**
  - ✔ Floating-point theory
  - ✔ Bitvector/integer theory
- **Obj3:** a **new open-source platform**
- **Obj4:** ensure **industrial-adequacy**

Objectives

- **Obj1:** design a **new collaboration framework for solvers**
  - ✔ First class domain and propagation engine
  - ➠ First class conflict and learning engine
- **Obj2:** design **new decision procedures**
  - ✔ Floating-point theory
  - ✔ Bitvector/integer theory
  - ✘ Nonlinear, ...
- **Obj3:** a **new open-source platform**
- **Obj4:** ensure **industrial-adequacy**

Objectives

- **Obj1:** design a **new collaboration framework for solvers**
  - ✔ First class domain and propagation engine
  - ➠ First class conflict and learning engine
- **Obj2:** design **new decision procedures**
  - ✔ Floating-point theory
  - ✔ Bitvector/integer theory
  - ✘ Nonlinear, ...
- **Obj3:** a **new open-source platform**
  - ✔ Framework implemented
- **Obj4:** ensure **industrial-adequacy**

Objectives

- **Obj1:** design a **new collaboration framework for solvers**
  - ✔ First class domain and propagation engine
  - ➠ First class conflict and learning engine
- **Obj2:** design **new decision procedures**
  - ✔ Floating-point theory
  - ✔ Bitvector/integer theory
  - ✘ Nonlinear, ...
- **Obj3:** a **new open-source platform**
  - ✔ Framework implemented
  - ➠ Usual theories
- **Obj4:** ensure **industrial-adequacy**

Objectives

- **Obj1:** design a **new collaboration framework for solvers**
  - ✔ First class domain and propagation engine
  - ➠ First class conflict and learning engine
- **Obj2:** design **new decision procedures**
  - ✔ Floating-point theory
  - ✔ Bitvector/integer theory
  - ✘ Nonlinear, ...
- **Obj3:** a **new open-source platform**
  - ✔ Framework implemented
  - ➠ Usual theories
  - ✘ New theories
- **Obj4:** ensure **industrial-adequacy**

Objectives

- **Obj1:** design a **new collaboration framework for solvers**
  - ✔ First class domain and propagation engine
  - ➠ First class conflict and learning engine
- **Obj2:** design **new decision procedures**
  - ✔ Floating-point theory
  - ✔ Bitvector/integer theory
  - ✘ Nonlinear, ...
- **Obj3:** a **new open-source platform**
  - ✔ Framework implemented
  - ➠ Usual theories
  - ✘ New theories
- **Obj4:** ensure **industrial-adequacy**
  - ✔ Integrated in existing solvers

Objectives

- **Obj1:** design a **new collaboration framework for solvers**
  - ✔ First class domain and propagation engine
  - ➠ First class conflict and learning engine
- **Obj2:** design **new decision procedures**
  - ✔ Floating-point theory
  - ✔ Bitvector/integer theory
  - ✘ Nonlinear, ...
- **Obj3:** a **new open-source platform**
  - ✔ Framework implemented
  - ➠ Usual theories
  - ✘ New theories
- **Obj4:** ensure **industrial-adequacy**
  - ✔ Integrated in existing solvers
  - ✔ Great results on existing benchmarks

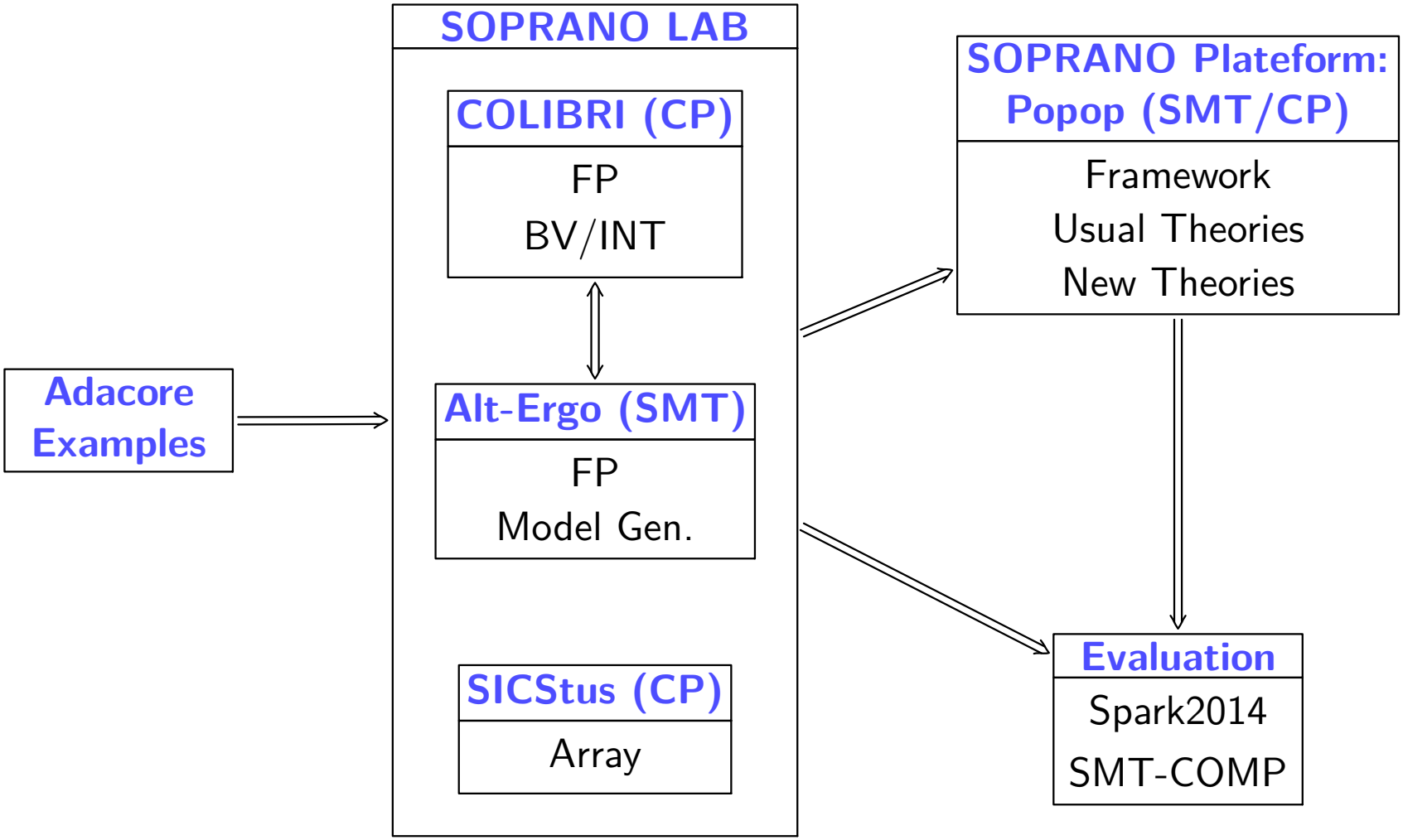
Objectives

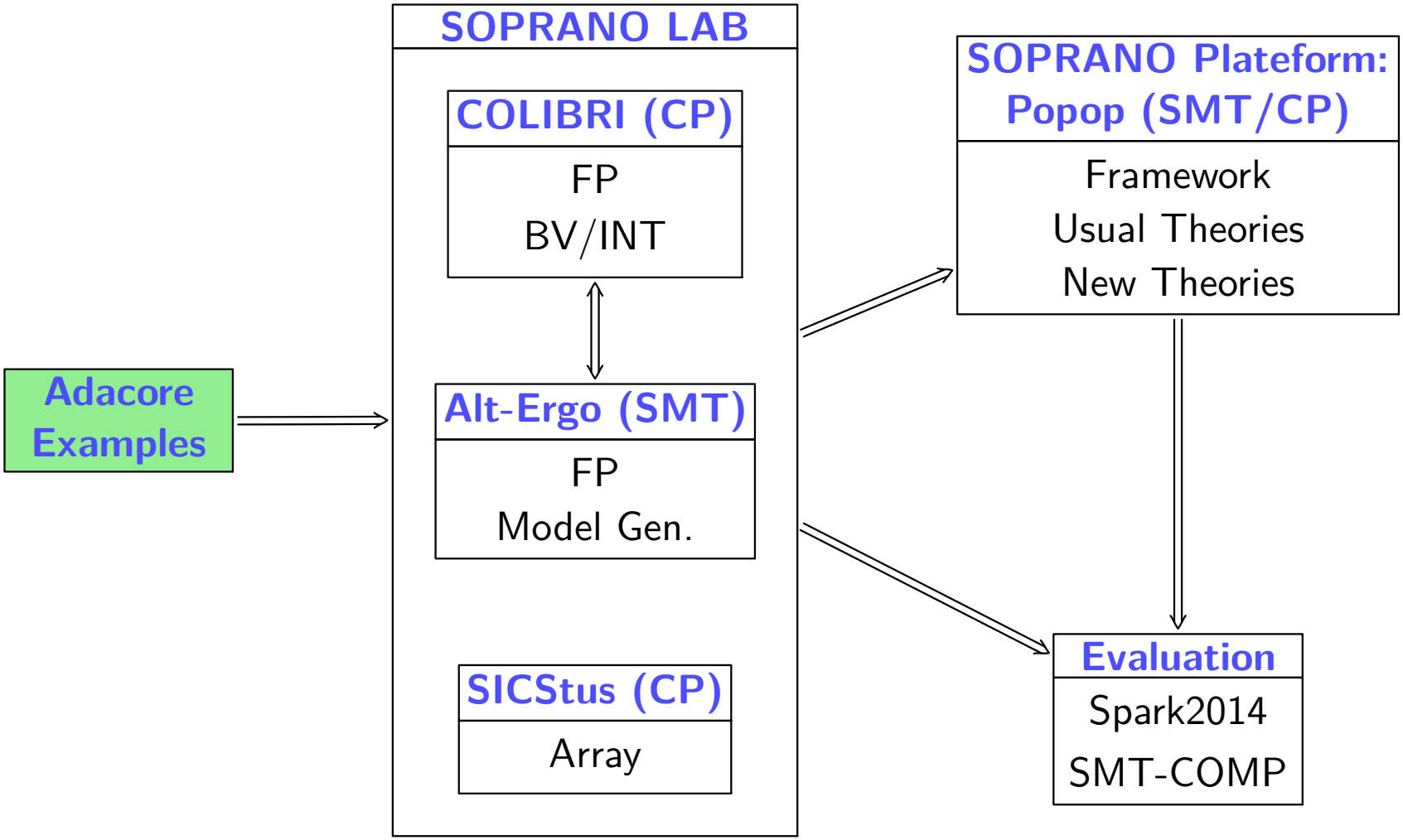
- **Obj1:** design a **new collaboration framework for solvers**
  - ✔ First class domain and propagation engine
  - ➠ First class conflict and learning engine
- **Obj2:** design **new decision procedures**
  - ✔ Floating-point theory
  - ✔ Bitvector/integer theory
  - ✘ Nonlinear, ...
- **Obj3:** a **new open-source platform**
  - ✔ Framework implemented
  - ➠ Usual theories
  - ✘ New theories
- **Obj4:** ensure **industrial-adequacy**
  - ✔ Integrated in existing solvers
  - ✔ Great results on existing benchmarks
  - ✔ Currently connected and tested in SPARK2014

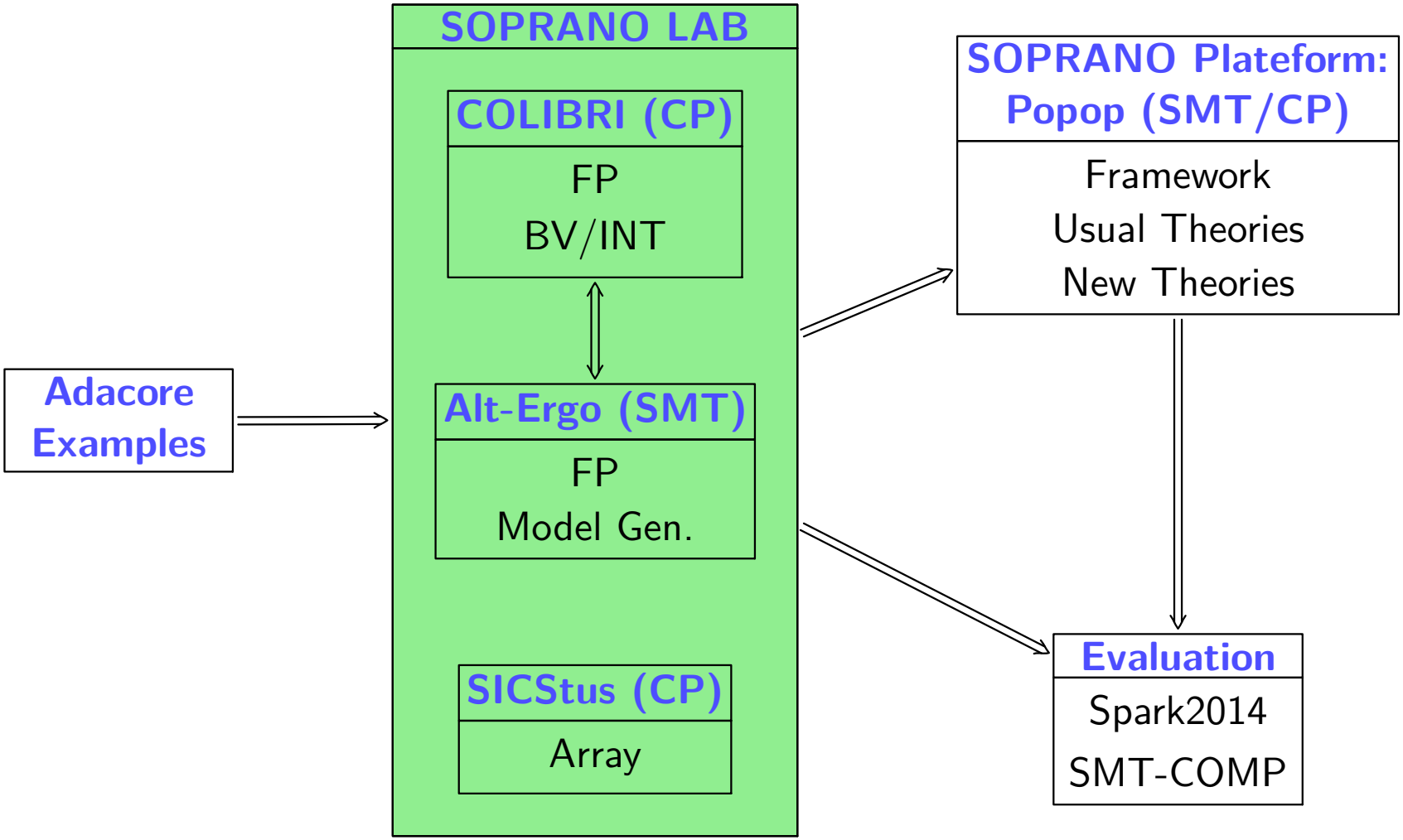
Objectives

- **Obj1:** design a **new collaboration framework for solvers**
  - ✔ First class domain and propagation engine
  - ➠ First class conflict and learning engine
- **Obj2:** design **new decision procedures**
  - ✔ Floating-point theory
  - ✔ Bitvector/integer theory
  - ✘ Nonlinear, ...
- **Obj3:** a **new open-source platform**
  - ✔ Framework implemented
  - ➠ Usual theories
  - ✘ New theories
- **Obj4:** ensure **industrial-adequacy**
  - ✔ Integrated in existing solvers
  - ✔ Great results on existing benchmarks
  - ✔ Currently connected and tested in SPARK2014

➠ Not yet published


SOPRANO LAB
COLIBRI (CP)
FP
BV/INT
Alt-Ergo (SMT)
FP
Model Gen.
SICStus (CP)
Array
Adacore Examples
SOPRANO Plateform: Popop (SMT/CP)
Framework
Usual Theories
New Theories
Evaluation
Spark2014
SMT-COMP


SOPRANO LAB
COLIBRI (CP)
FP
BV/INT
Alt-Ergo (SMT)
FP
Model Gen.
SICStus (CP)
Array
Adacore Examples
SOPRANO Plateform: Popop (SMT/CP)
Framework
Usual Theories
New Theories
Evaluation
Spark2014
SMT-COMP


SOPRANO LAB
COLIBRI (CP)
FP
BV/INT
Alt-Ergo (SMT)
FP
Model Gen.
SICStus (CP)
Array
Adacore
Examples
SOPRANO Plateform:
Popop (SMT/CP)
Framework
Usual Theories
New Theories
Evaluation
Spark2014
SMT-COMP

Floating-Point Arithmetic: Huge Improvements

(CEA, Inria, UPSud, OcamlPro)

$$A \leq \frac{X \ominus Y}{X \ominus Z} \leq B \quad \text{with } \ldots$$

$$\sqrt{X^2 \ominus Y^2} \leq X \quad \text{with } \ldots$$

$$\frac{X}{\sqrt{X^2 \oplus Y^2}} \leq 1 \quad \text{with } \ldots$$

Floating-Point Arithmetic: COLIBRI

(CEA, UPSud)

- Precise domain propagation
- Distance graph on floating-point numbers

Floating-Point Arithmetic: COLIBRI

(CEA, UPSud)

- Precise domain propagation
- Distance graph on floating-point numbers
- Monotonic functions:
  $\mathrm{o}(f(x)) < \mathrm{o}(y) \implies \mathrm{o}(x) \leq \mathrm{o}(f^{-1}(\mathrm{o}(y)))$

Floating-Point Arithmetic: COLIBRI

(CEA, UPSud)

- Precise domain propagation
- Distance graph on floating-point numbers
- Monotonic functions:
  $o(f(x)) < o(y) \implies o(x) \leq o(f^{-1}(o(y)))$
- Instantiated for many functions

Floating-Point Arithmetic: COLIBRI

(CEA, UPSud)

- Precise domain propagation
- Distance graph on floating-point numbers
- Monotonic functions:
  $o(f(x)) < o(y) \implies o(x) \leq o(f^{-1}(o(y)))$
- Instantiated for many functions
- Linearization of constraints for simplex

Floating-Point Arithmetic: Alt-Ergo

(OCamlPro, UPSud)

Floating-point rounding operator on rational constants

```
1 axiom rounding_operator_1 :
    forall x : real .
3   forall i , j : real .
    forall md : fpa_rounding_mode.
5   forall p,m : int
     [round(m,p,md,x), x in [ i , j ]].
7      i ≤ x ≤ j →
       round(m,p,md,i) ≤ round(m,p,md,x) ≤ round(m,p,md,j)
9
```

(Kailiang Ji, post-doc SOPRANO)

Floating-Point Arithmetic: Recap

Progression of COLIBRI and Alt-Ergo on AdaCore benchmarks:

	Before SOPRANO	Current Results
COLIBRI	18 / 28	25 / 28
Alt-Ergo	2 / 28	19 / 28

Bitvector and Integer Arithmetic

(CEA)

- High-level view of bitvectors
- New propagations for integers $\leftrightarrow$ bitvectors
- Integration in Popop (SOPRANO platform)
- Presentation at *CP meets Verification 2016 Workshop*

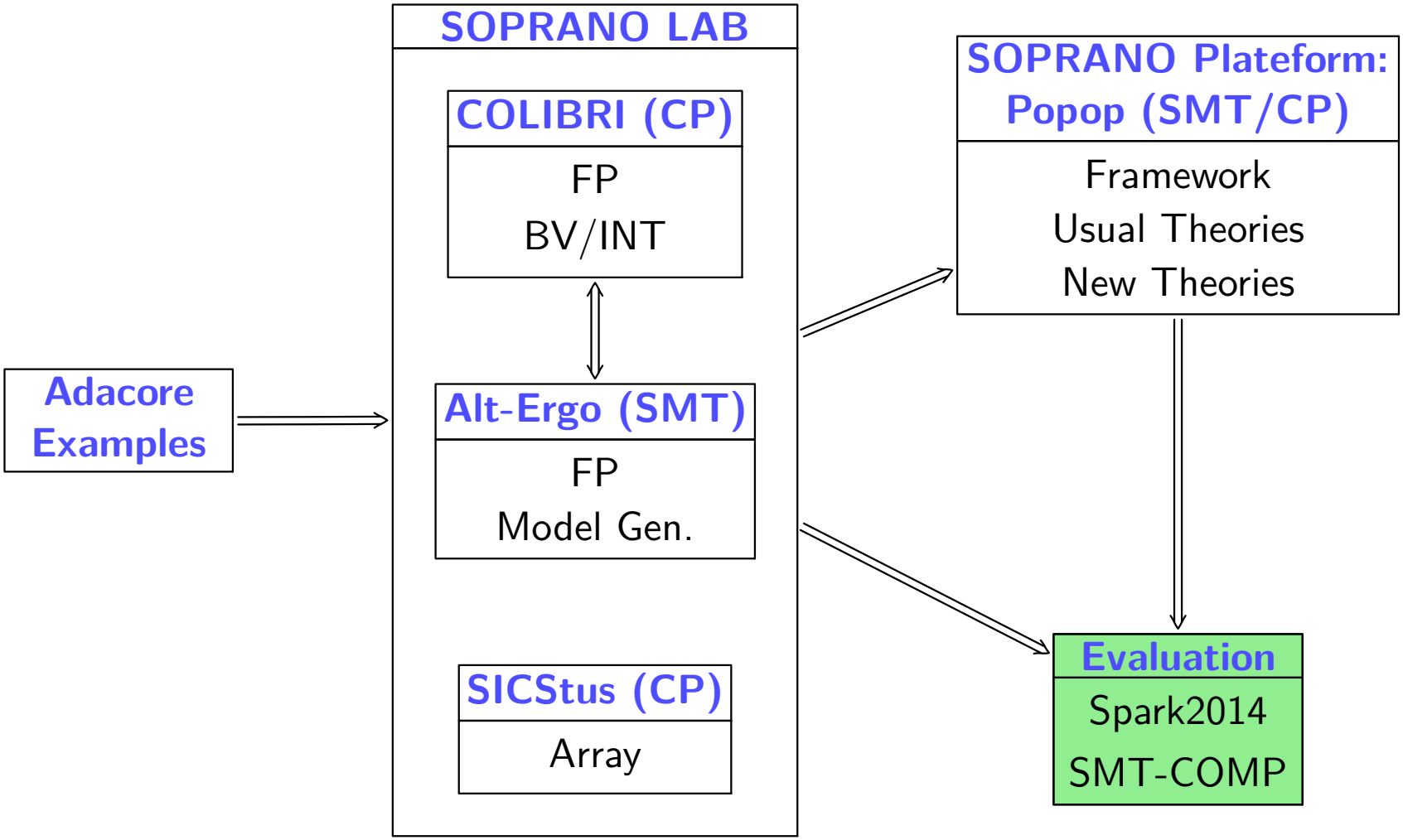
(Zakaria Chihani, post-doc CEA)

Array Theory with CP

(Inria, CEA)

- Large arrays badly handled by SMT solvers
- New domain for arrays
- Propagation and learning
- Integration in SICStus clpfd CP solver

(Quentin Plazar, PhD SOPRANO)


SOPRANO LAB
COLIBRI (CP)
FP
BV/INT
Alt-Ergo (SMT)
FP
Model Gen.
SICStus (CP)
Array
Adacore Examples
SOPRANO Plateform: Popop (SMT/CP)
Framework
Usual Theories
New Theories
Evaluation
Spark2014
SMT-COMP

Evaluations

(AdaCore, CEA, UPSud, OCamlPro)

- AdaCore examples

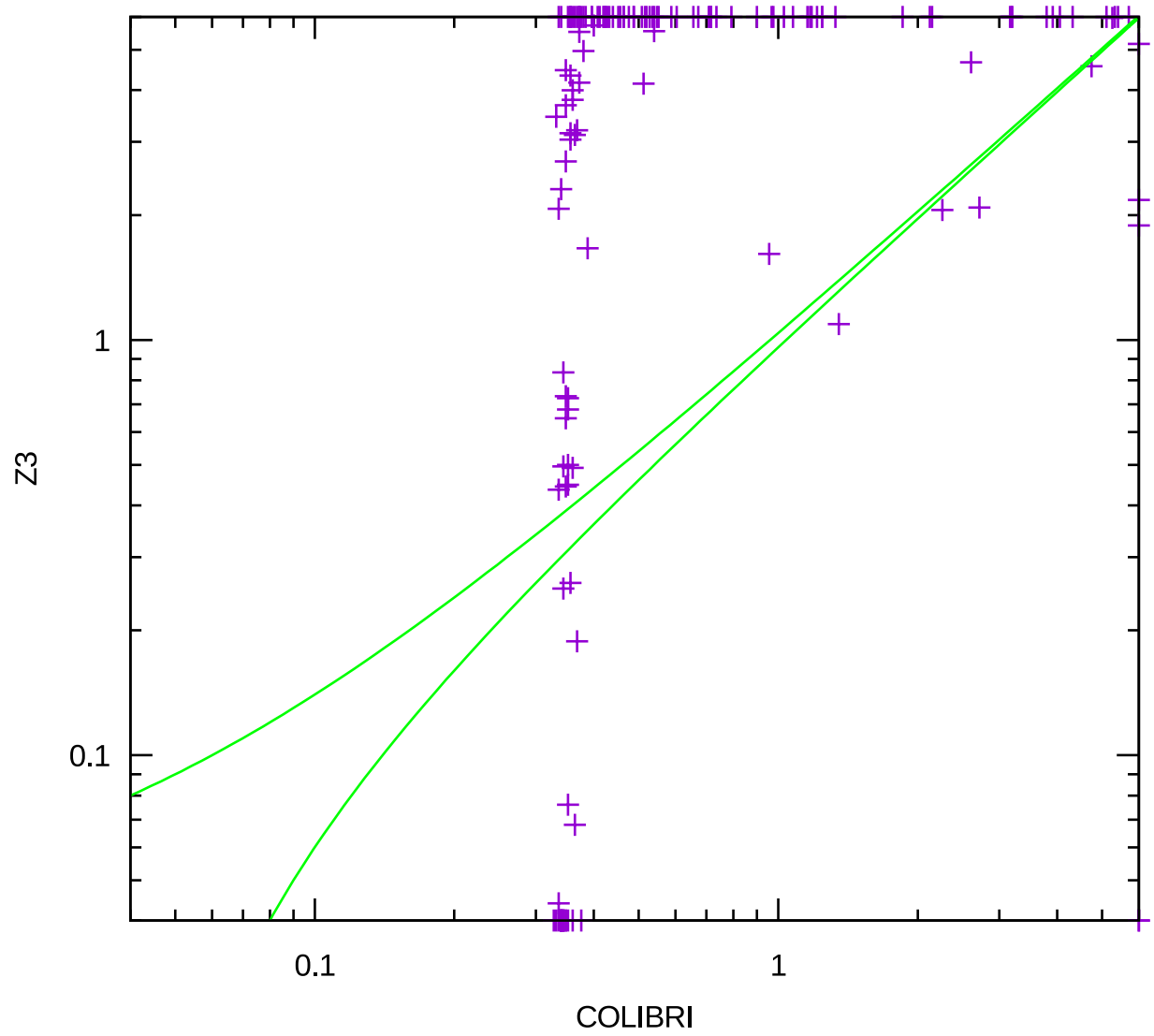
Evaluations

(AdaCore, CEA, UPSud, OCamlPro)

- AdaCore examples
- Prepare participation at SMT-COMP

COLIBRI on SMT-COMP FP category

(CEA)



Evaluations

(AdaCore, CEA, UPSud, OCamlPro)

- AdaCore examples
- Prepare participation at SMT-COMP
- Participation at SMT-COMP 2017

Evaluations

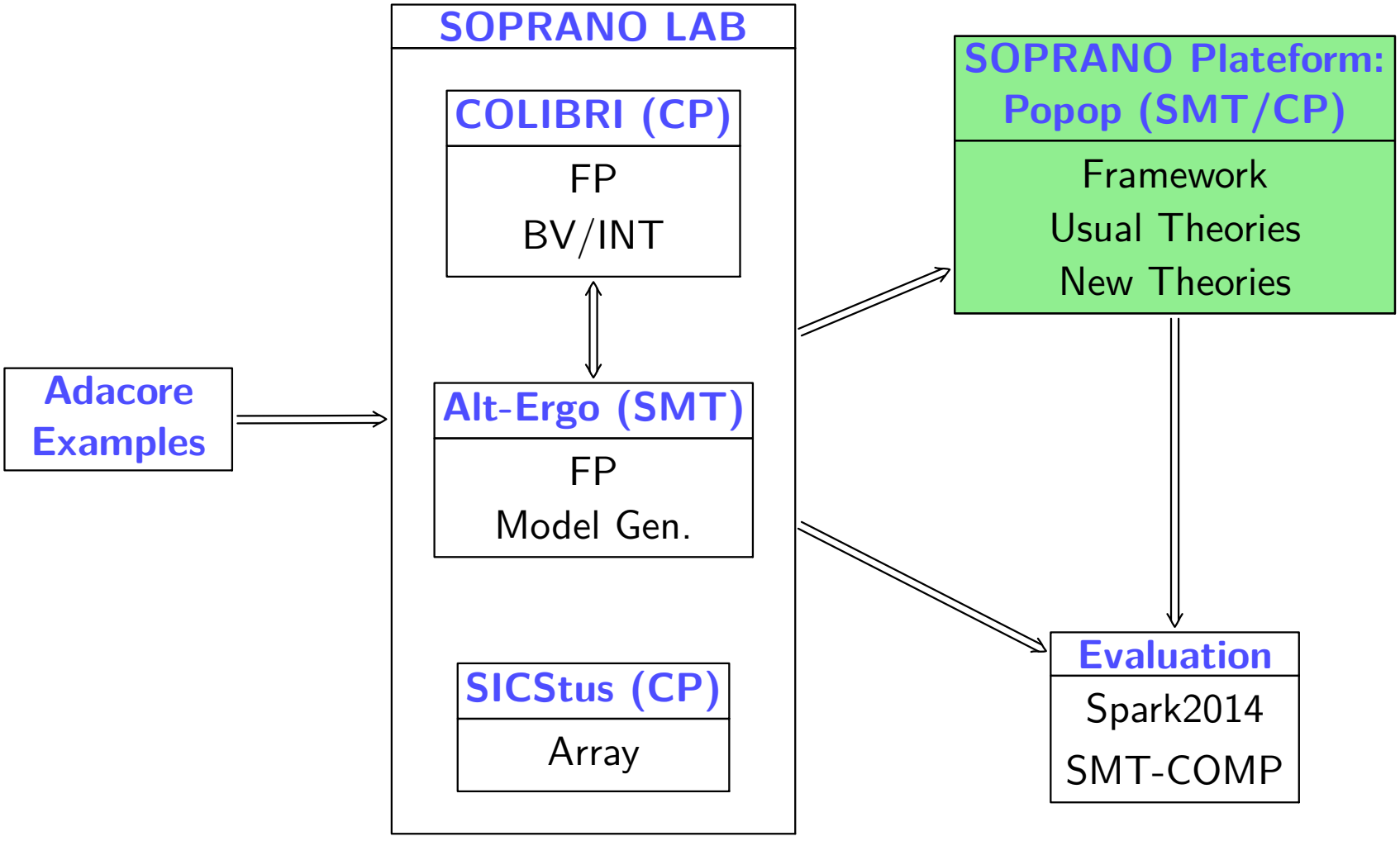
(AdaCore, CEA, UPSud, OCamlPro)

- AdaCore examples
- Prepare participation at SMT-COMP
- Participation at SMT-COMP 2017
- Connection from SPARK to COLIBRI and Alt-Ergo

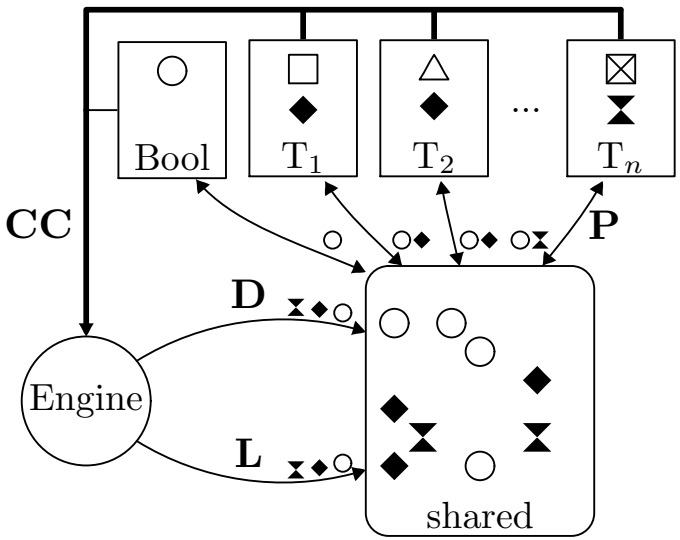
Evaluations

(AdaCore, CEA, UPSud, OCamlPro)

- AdaCore examples
- Prepare participation at SMT-COMP
- Participation at SMT-COMP 2017
- Connection from SPARK to COLIBRI and Alt-Ergo
- Soon, evaluation directly on SPARK programs


SOPRANO LAB
COLIBRI (CP)
FP
BV/INT
Alt-Ergo (SMT)
FP
Model Gen.
SICStus (CP)
Array
Adacore Examples
SOPRANO Plateform: Popop (SMT/CP)
Framework
Usual Theories
New Theories
Evaluation
Spark2014
SMT-COMP

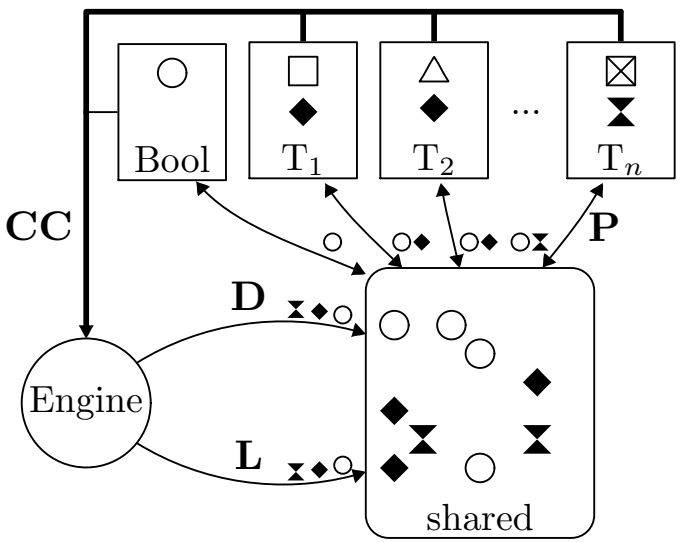
Popop Framework (Implemented)



Engine:

- equivalence class
- semantic values
- pluggable domains
- event loop
- pluggable learning

Popop Framework (Implemented)



Engine:

- equivalence class
- semantic values
- pluggable domains
- event loop
- pluggable learning

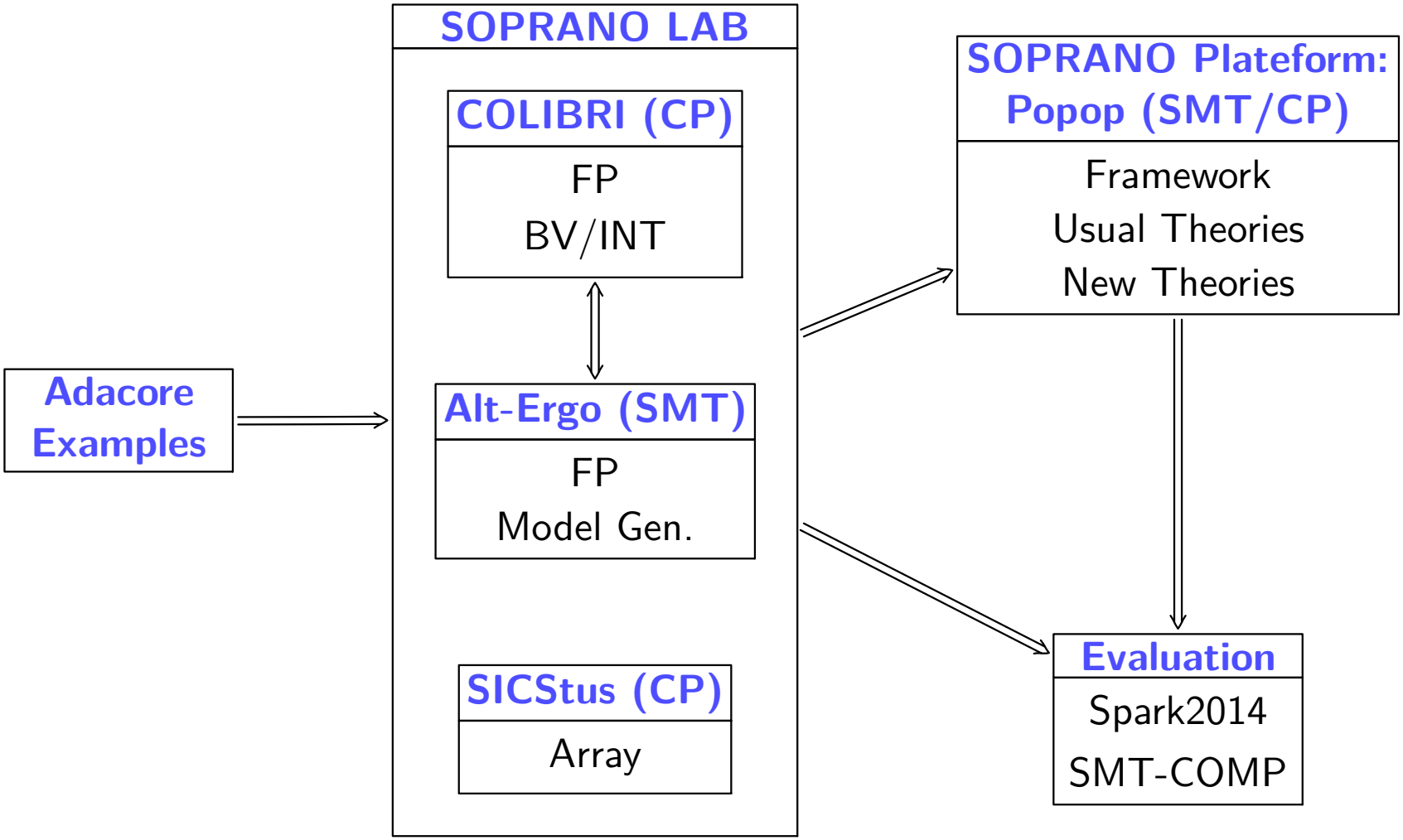
Theories:

- Bool: friendly with other theories
- equalities/disequalities, `ite`
- uninterpreted function
- linear rational arithmetic
- bitvectors (partial)

Popop Framework: Roadmap

Roadmap:

- implementation of learning for bitvector operators
- structures, algebraic datatypes, arrays
- floating-point arithmetic
- non-linear rational arithmetic
- integer linear arithmetic and bitvectors on arithmetic operators


SOPRANO LAB
COLIBRI (CP)
FP
BV/INT
Alt-Ergo (SMT)
FP
Model Gen.
SICStus (CP)
Array
Adacore Examples
SOPRANO Plateform: Popop (SMT/CP)
Framework
Usual Theories
New Theories
Evaluation
Spark2014
SMT-COMP

Current releases (CEA, OcamlPro)

OCI open-source:

- Benchmarking framework
- Continuous integration (used daily for Frama-C, maybe Coq)

Ocplib-Simplex open-source:

- Simplex implementation
- Parametric on constants and polynomial

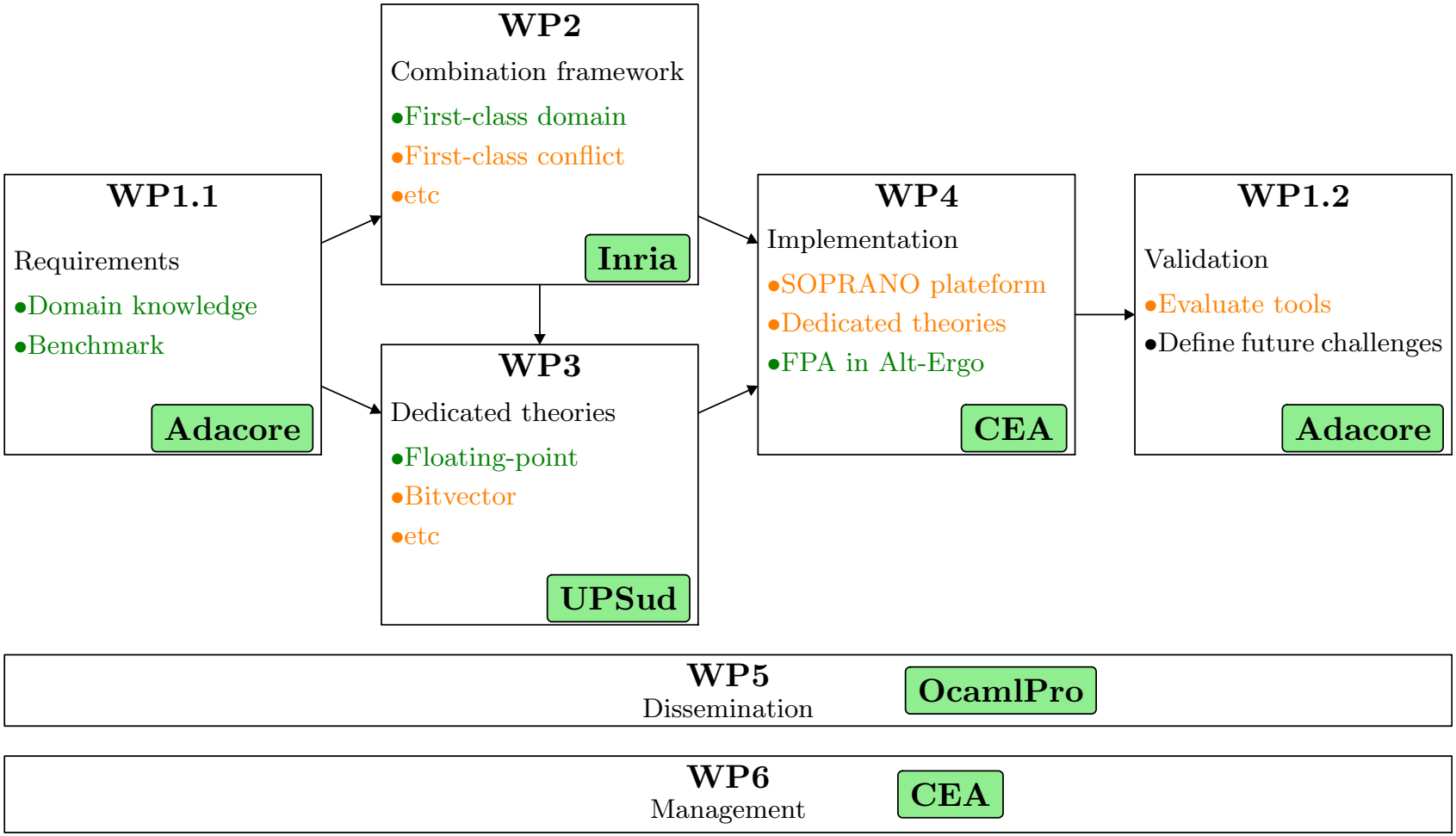
Alt-Ergo: new open-source release with model generation

COLIBRI Freeware

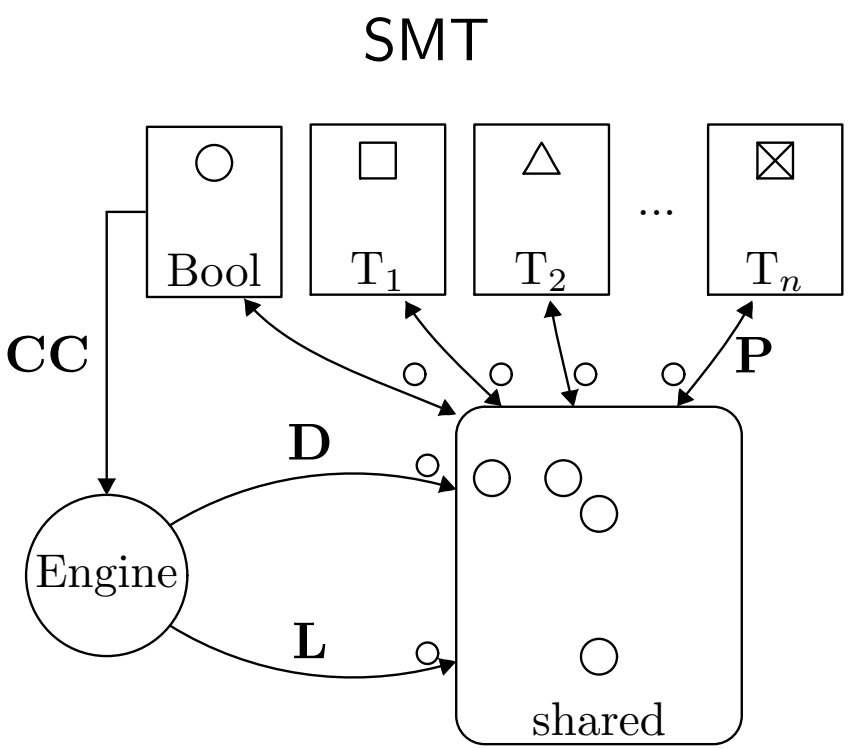
Dissemination

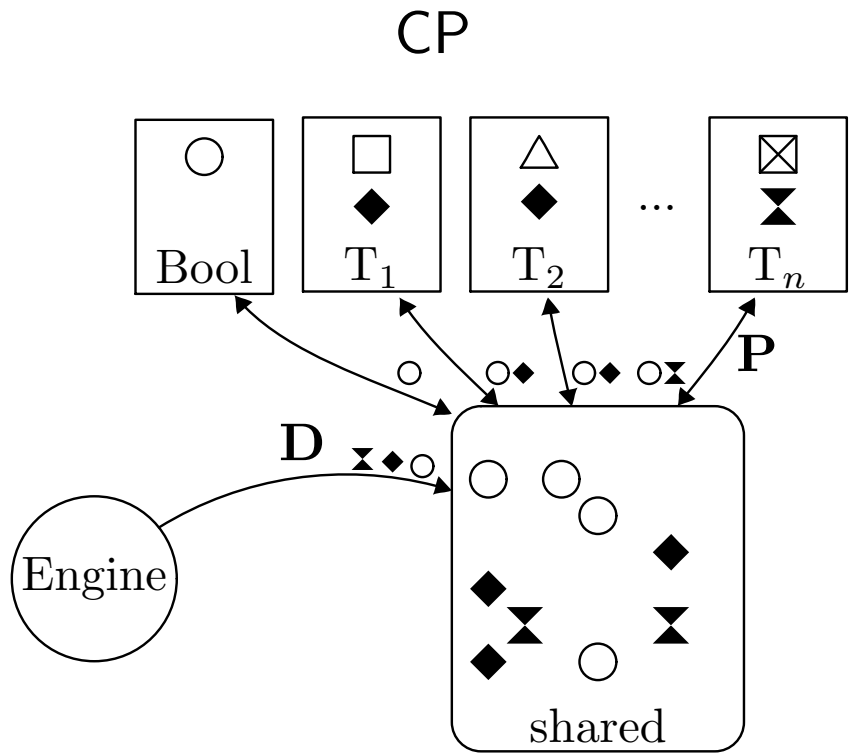
- Chaired *CP meets Verification 2016 Workshop*
- Publications in preparation:
  - Semantic triggers and applications in Alt-Ergo
  - Real behavior of floating point (techniques used in COLIBRI)
  - Quadruple domain representation tackling bitvector theory
  - CP meets SMT: MCSAT generalization

Workflow


WP2
Combination framework
•First-class domain
•First-class conflict
•etc
Inria
WP1.1
Requirements
•Domain knowledge
•Benchmark
Adacore
WP3
Dedicated theories
•Floating-point
•Bitvector
•etc
UPSud
WP4
Implementation
•SOPRANO plateform
•Dedicated theories
•FPA in Alt-Ergo
CEA
WP1.2
Validation
•Evaluate tools
•Define future challenges
Adacore
WP5
Dissemination
OcamlPro
WP6
Management
CEA

The Problems





Floating-Point Arithmetic: Monotonic functions

(CEA, UPSud)

Theorem

Let $D, E \subset \mathcal{R}$, $f : D \mapsto E$ *and* $f^{-1} : E \mapsto D$ *such that*

- $\forall x : D,\ f^{-1}(f(x)) = x$
- f *increasing*

We have

- $\forall x \in D,\ o(y) \in E,\ o(f(x)) < o(y) \implies o(x) \leq o(f^{-1}(o(y)))$
- $\forall x \in D,\ y \in E,\ o(f(x)) < o(f(y)) \implies x < y$

Instantiated for many functions in COLIBRI's DBM