CVC5 in Lean 4: cvc.lean

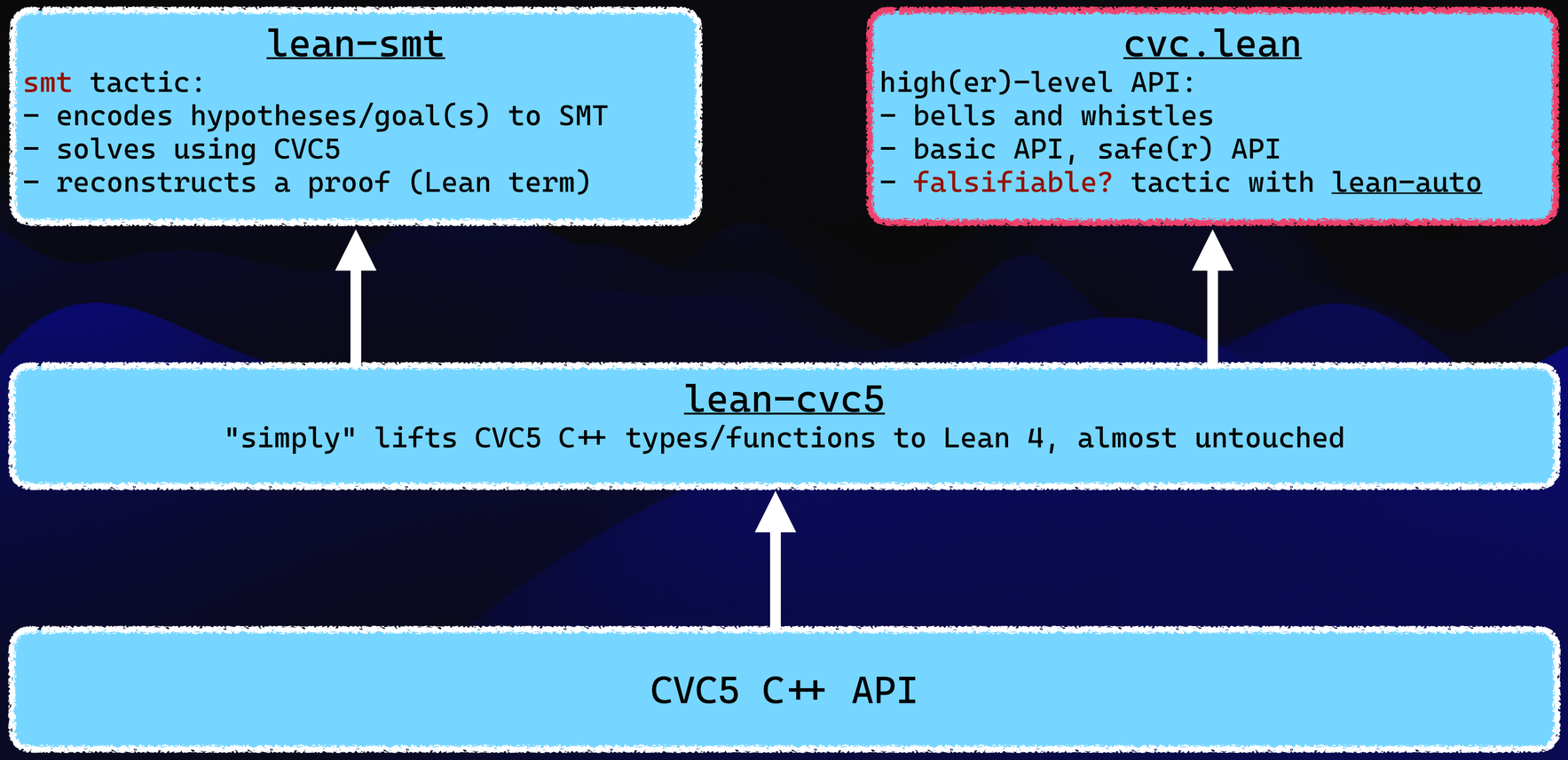
API, Safety, tactics, and integration with lean-auto

repository: https://github.com/anzenlang/cvc.lean

slides available at https://anzenlang.io/blog

Adrien Champion

Overview: Lean libraries



Adrien Champion

Extending lean-cvc5

Currently exposes little of the CVC5 API

We are adding

- ✅ most of the functions corresponding to SMT-LIB commands
- ✅ quantified terms and quantifier elimination
- ✅ interpolants and abducts
- function synthesis
- information retrieval on unknown result

API:
Terms and SMT Commands

Adrien Champion

Comparison: Term creation

```
-- `lean-cvc5`
def mkTerm : TermManager → (kind : Kind) → (children : Array Term) → Except Error Term

-- `cvc.lean` basic API: specialized constructors
def mult (lhs rhs : Term) : ManagerM Term
def eq (lhs rhs : Term) : ManagerM Term
def mkIte (cnd thn els : Term) : ManagerM Term

-- `cvc.lean` safe API: specialized constructors over strongly-typed terms
def mult [ArithLike α] (a b : Term α) : ManagerM (Term α)
def eq (lhs rhs : Term α) : ManagerM (Term Bool)
def mkIte (cnd : Term Bool) (thn els : Term α) : ManagerM (Term α)
```

github.com/anzenlang/cvc.lean/CvcTest/Demo/Cvc5Summit/Comp.lean

Adrien Champion

Comparison: SMT

```
-- `lean-cvc5`
def assertFormula : (term : Term) → SolverT m Unit
def getValue : (term : Term) → SolverT m Term
def checkSat : SolverT m Result

-- `cvc.lean` basic API
def assert (formula : Term) : SmtM Unit
def getValue (term : Term) : SmtM Term
def checkSat? : SmtM (Option Bool)

-- `cvc.lean` safe API: typed terms, only allow (un)sat-specific commands when legal
def assert (formula : Term Bool) : SmtM Unit
def getValue {α : Type} [I : ValueOfSafeTerm α] (term : Term α) : Smt.SatM α
def checkSat
  (ifSat : Smt.SatT m α) (ifUnsat : Smt.UnsatT m α) (ifUnknown : Smt.UnknownT m α)
: SmtT m α
```

github.com/anzenlang/cvc.lean/CvcTest/Demo/Cvc5Summit/Comp.lean

Adrien Champion

Comparison

A few examples:

- short and simple comparison of term creation/SMT commands
  CvcTest/Demo/Cvc5Summit/Comp.lean

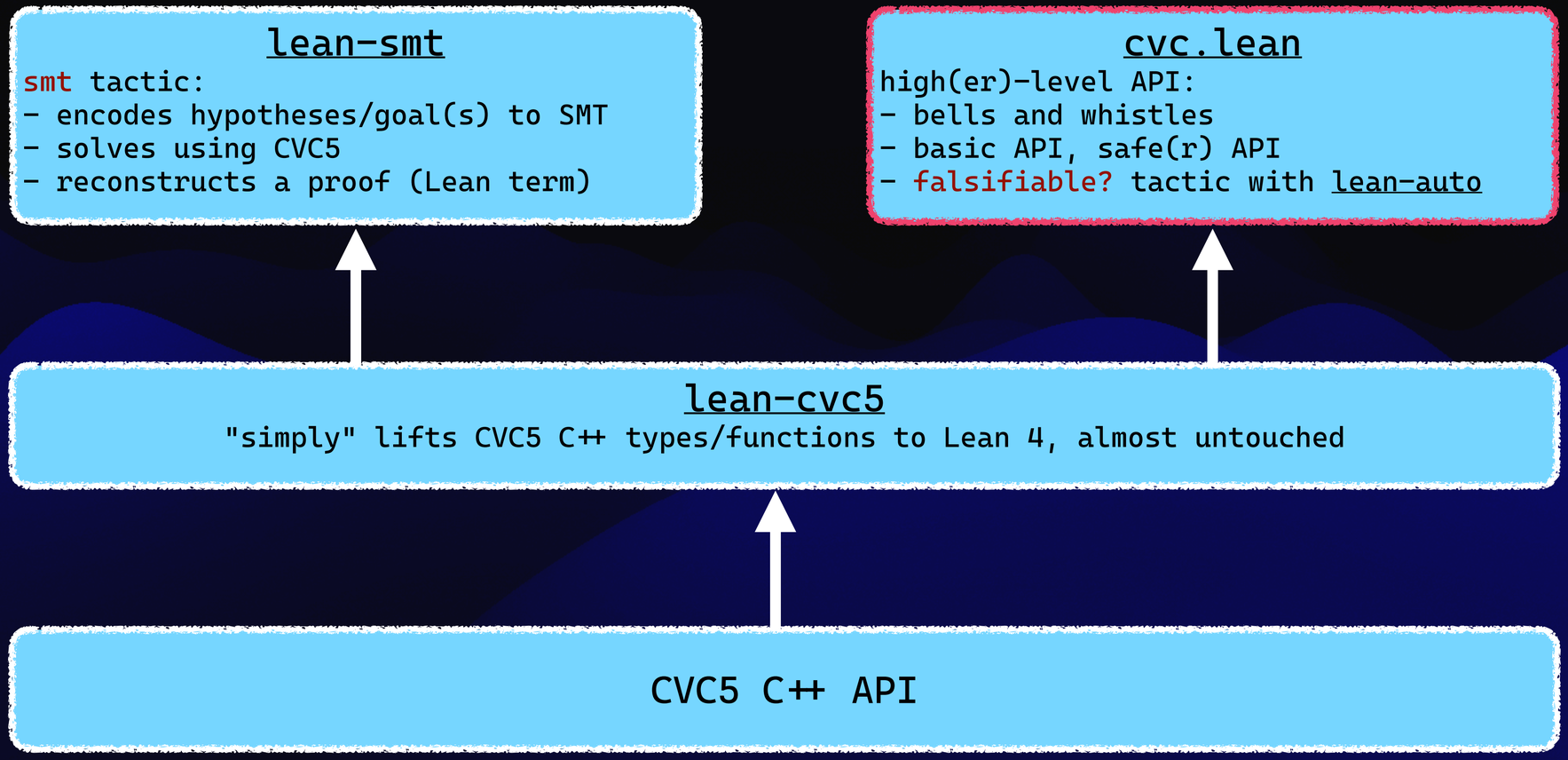
- int-valued function minimization example
  CvcTest/Demo/SimpleMinimizer.lean

- induction and pre-image computation on transition system
  CvcTest/Safe/Sys.lean
  CvcTest/Safe/SysDemoSw.lean

Tactics:
Theorem proving with Cvc5

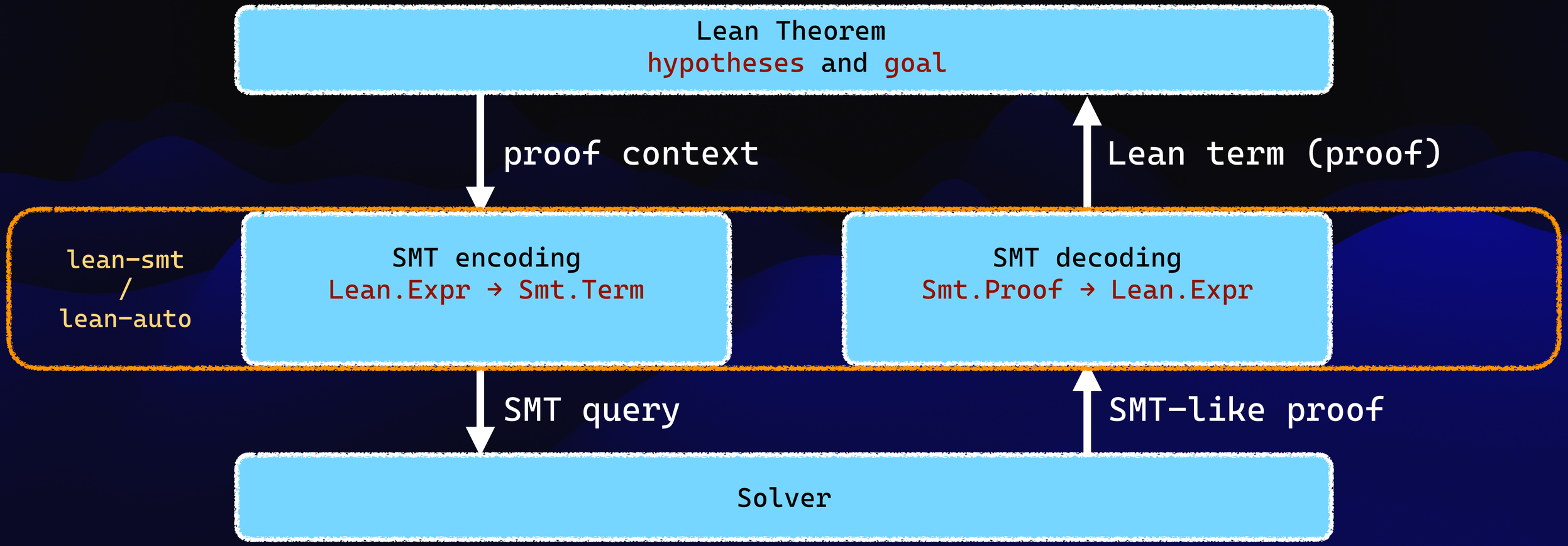
Adrien Champion

Overview: Lean libraries



Adrien Champion

SMT-based theorem proving



Adrien Champion

The falsifiable? tactic

A lean-auto-based toy tactic from cvc.lean:

- on simple arithmetic theorems
  CvcTest/Tactic/Simple.lean

- on more complex fragments:
  CvcTest/Tactic/Fragments.lean

Make sure to also check the smt tactic from lean-smt:

- https://github.com/ufmg-smite/lean-smt

Adrien Champion

Thank you 😺

Links:

- lean-cvc5: https://github.com/abdoo8080/lean-cvc5
- cvc.lean: https://github.com/anzenlang/cvc.lean
- lean-smt: https://github.com/ufmg-smite/lean-smt
- lean-auto: https://github.com/leanprover-community/lean-auto

Adrien Champion