

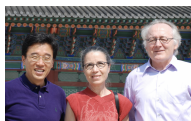
Program Synthesis via Bi-directional Reduced-product Abstract Interpretation

Kwangkeun Yi

(co-work with Yongho Yoon and Woosuk Lee)

Seoul National University, Korea

01/20/2024 @ N40AI at POPL'24, London



Our Story Upon Abstract Interpretation

- ▶ program analysis

- ▶ program synthesis

Our Story Upon Abstract Interpretation

- ▶ program **analysis**
workbench: a more-sound & global analysis for full C
 - ▶ **commercialization**: SparrowFasoo.com (~ 2010)
 - ▶ safety+security checkers for over 25 programming languages
 - ▶ less-sound, lots of compromises

- ▶ program **synthesis**

Our Story Upon Abstract Interpretation

- ▶ program **analysis**
workbench: a more-sound & global analysis for full C
 - ▶ **commercialization**: SparrowFasoo.com (~ 2010)
 - ▶ safety+security checkers for over 25 programming languages
 - ▶ less-sound, lots of compromises
 - ▶ **scalability**: sparse analysis framework (TOPLAS'14, PLDI'12)
 - ▶ **precision**: selective x-sensitivity (TOPLAS'18, TOPLAS'16, PLDI'14)
 - ▶ github.com/ropas/sparrow
- ▶ program **synthesis**

Our Story Upon Abstract Interpretation

- ▶ program **analysis**
workbench: a more-sound & global analysis for full C
 - ▶ **commercialization**: SparrowFasoo.com (~ 2010)
 - ▶ safety+security checkers for over 25 programming languages
 - ▶ less-sound, lots of compromises
 - ▶ **scalability**: sparse analysis framework (TOPLAS'14, PLDI'12)
 - ▶ **precision**: selective x-sensitivity (TOPLAS'18, TOPLAS'16, PLDI'14)
 - ▶ github.com/ropas/sparrow
- ▶ program **synthesis**
 - ▶ superb in performance (PLDI'23, PLDI'20)
 - ▶ by bi-directional reduced-product AI with finite refinements

Analysis Commercialization: Our Story Upon Abstract Interpretation



Guarantees Quality & Security in Software
Provide products and services across SDLC to increase software value.



Sparrow SAST

Intelligent static application security testing solution

Find and fix security vulnerabilities at the speed of DevOps



Sparrow's static analysis solution supports over 25 major programming languages and frameworks.


Java, JSP, C/C++, C#, Python, Swift, Rust, Go, TypeScript, Objective-C, Kotlin, etc.



Comprehensive coverage



Improve the security and quality of software by complying with global security compliances guides and standard guides

DWE, OWASP, CERT, MISRA C/C++, BSSC, C/C++, I4C C++ and more





Fast & accurate analysis

Minimize analysis time and increase work efficiency by utilizing advanced features, including real code-based remediation guides and issue filtering.



Key Features

Analyze source code for security vulnerabilities quickly and accurately.
Fundamentally address cyberattacks with secure coding

 <h3>Powerful analysis</h3> <ul style="list-style-type: none">MVC structure analysis, associated file analysis, and analysis of function call relationship in various levels.Incremental analysis: Minimize analysis time by only analyzing newly added, modified files and their associated filesSupport analysis via GUI, CLI, and Plugin	 <h3>Advanced manageability</h3> <ul style="list-style-type: none">Issue navigator to track vulnerabilities from their origin to actual codeAutomated real source code connection guideAutomated classification of vulnerabilities
--	---

Sparrow
The Early Bird



RO SAEC center
Research On Software Analysis for Error-free Computing
소프트웨어 무결성 연구센터 (SEEC)



, thanks to AI.

Analysis Scalability: Our Story Upon Abstract Interpretation



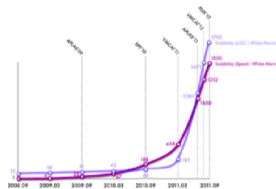
Soundness

1 Million LoC

Scalability

Precision

**General Sparse
Analysis Framework**



(TOPLAS'14, PLDI'12)

Analysis Scalability: Our Story Upon Abstract Interpretation

- ▶ Idea: exploit the semantic sparsity
 - ▶ spatial & temporal sparsities

Analysis Scalability: Our Story Upon Abstract Interpretation

- ▶ Idea: exploit the semantic sparsity
 - ▶ spatial & temporal sparsities
- ▶ Sparse analysis framework
 - ▶ **general** & precision-preserving

$$\begin{array}{ccc} F^\# : D^\# \rightarrow D^\# & \xrightarrow{\text{sparsify}} & F_s^\# : D^\# \rightarrow D^\# \\ \text{fix}F^\# & \stackrel{\text{still}}{=} & \text{fix}F_s^\# \end{array}$$

(TOPLAS'14, PLDI'12)

Analysis Scalability: Our Story Upon Abstract Interpretation

- ▶ Idea: exploit the semantic sparsity
 - ▶ spatial & temporal sparsities
- ▶ Sparse analysis framework
 - ▶ general & precision-preserving

$$\begin{array}{ccc} F^\# : D^\# \rightarrow D^\# & \xrightarrow{\text{sparsify}} & F_s^\# : D^\# \rightarrow D^\# \\ \text{fix}F^\# & \stackrel{\text{still}}{=} & \text{fix}F_s^\# \end{array}$$

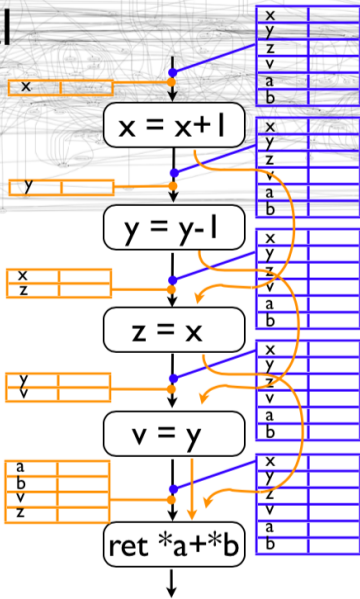
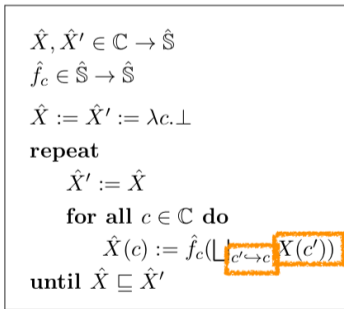
(TOPLAS'14, PLDI'12)

A review:

“the theoretical result is very general. It could be applied to many other analyses. PLDI papers have been accepted that were simply instances of this framework.”

, thanks to AI.

Spatial & Temporal Localizations



Analysis Scalability: Our Story Upon Abstract Interpretation



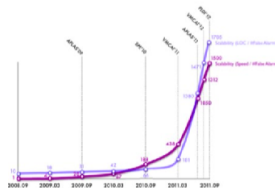
Soundness

1 Million LoC

Scalability

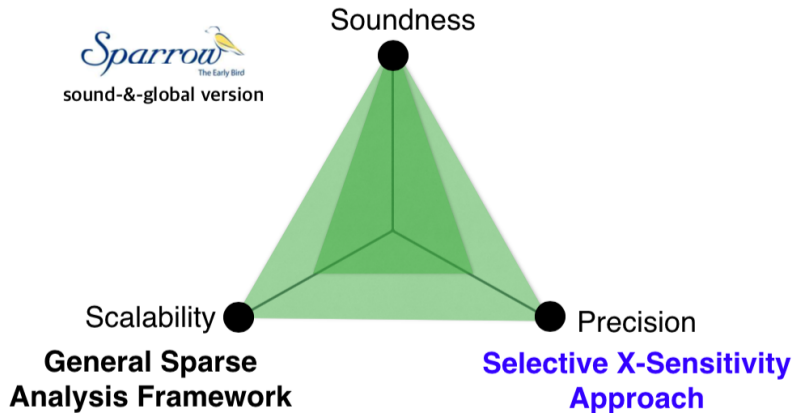
Precision

**General Sparse
Analysis Framework**



(TOPLAS'14, PLDI'12)

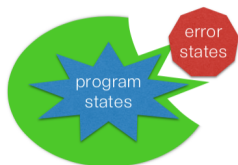
Analysis Precision: Our Story Upon Abstract Interpretation



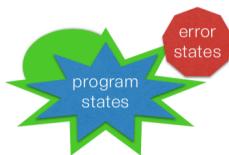
(TOPLAS'14, PLDI'12)

(TOPLAS'18, TOPLAS'16, PLDI'14)

Analysis Precision: Our Story Upon Abstract Interpretation



vs.



our method: **24%** / **28%**

3-CFA: **24%** / **1300%**

- ▶ Selective flow- or context-sensitivity
- ▶ i.e., sensitivity only when it would pay off
 - ▶ by (design) pre-analysis with 2-element domain (thanks to AI)
 - ▶ by (train) machine learning (Bayesian opt.) for linear score ftn
 - ▶ side cost: design < train (thanks to AI)

(TOPLAS'18, TOPLAS'16, OOPSLA'16, PLDI'14)

Program Synthesis: Our Story Upon Abstract Interpretation (PLDI'23, PLDI'20)

Inductive synthesis

- ▶ input: a grammar + input-output pairs
- ▶ output: a “right” program in the grammar

Example

- ▶ input

$$\begin{aligned} S &\rightarrow x \mid 0001_2 \mid S \gg S \\ &\quad \mid S \wedge S \mid S \vee S \mid S \oplus S \\ &\quad \mid S + S \mid S \times S \mid S / S \end{aligned}$$

$$f(1011_2) = 0011_2$$

- ▶ output

$$f(x) = ((x + 0001_2) \oplus x) \gg 0001_2$$

Automatic Synthesis Examples

► input

$$\begin{array}{l} S \rightarrow n \mid x \mid S \wedge S \mid S \vee S \mid S \oplus S \\ \quad \mid S \gg S \mid S \ggg S \mid S \ll S \mid S \lll S \\ \quad \mid -S \mid S + S \mid S \times S \mid S / S \end{array}$$

and 5-20 i/o pairs

► output (in 0.k sec - 35 min)

Automatic Synthesis Examples

- ▶ input

$$\begin{array}{l} S \rightarrow n \mid x \mid S \wedge S \mid S \vee S \mid S \oplus S \\ \quad \mid S \gg S \mid S \ggg S \mid S \ll S \mid S \lll S \\ \quad \mid -S \mid S + S \mid S \times S \mid S / S \end{array}$$

and 5-20 i/o pairs

- ▶ output (in 0.k sec - 35 min)

- ▶ $\text{abs}(x) : 8b = (x \ll (x \gg x)) \vee (x \times (x \gg x))$
- ▶ $\text{sign}(x) : 8b = ((-x) \ggg 7) \vee (x \gg x)$

Automatic Synthesis Examples

- ▶ input

$$\begin{array}{l} S \rightarrow n \mid x \mid S \wedge S \mid S \vee S \mid S \oplus S \\ \mid S \gg S \mid S \ggg S \mid S \ll S \mid S \lll S \\ \mid -S \mid S + S \mid S \times S \mid S / S \end{array}$$

and 5-20 i/o pairs

- ▶ output (in 0.k sec - 35 min)

- ▶ $\text{abs}(x) : 8b = (x \ll (x \gg x)) \vee (x \times (x \gg x))$
- ▶ $\text{sign}(x) : 8b = ((-x) \ggg 7) \vee (x \gg x)$
- ▶ $\text{averageFloor}(x, y) : u8b = ((x \oplus y) \ggg 1) + (x \wedge y)$
- ▶ $\text{averageCeil}(x, y) : u8b = (x \vee y) - ((x \oplus y) \ggg 1)$

Automatic Synthesis Examples

- ▶ input

$$\begin{array}{l} S \rightarrow n \mid x \mid S \wedge S \mid S \vee S \mid S \oplus S \\ \quad \mid S \ggg S \mid S \ggg S \mid S \lll S \mid S \lll S \\ \quad \mid -S \mid S + S \mid S \times S \mid S / S \end{array}$$

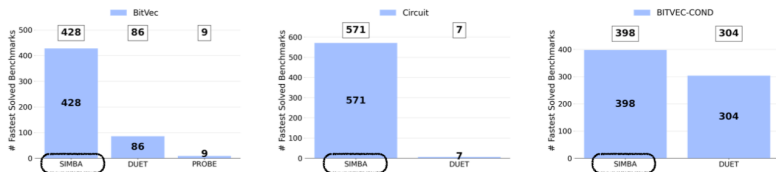
and 5-20 i/o pairs

- ▶ output (in 0.k sec - 35 min)

- ▶ $\text{abs}(x) : 8b = (x \lll (x \ggg x)) \vee (x \times (x \ggg x))$
- ▶ $\text{sign}(x) : 8b = ((-x) \ggg 7) \vee (x \ggg x)$
- ▶ $\text{averageFloor}(x, y) : u8b = ((x \oplus y) \ggg 1) + (x \wedge y)$
- ▶ $\text{averageCeil}(x, y) : u8b = (x \vee y) - ((x \oplus y) \ggg 1)$
- ▶ $\text{deObfus}(x, y) : 64b =$
 $((x \vee y) \wedge (-x \vee -(y \times y))) - (x \vee y \vee (-x \vee -(y \times y)))$

Synthesis Performance

as of 2023



of the fastest-solved problems for each domain

thanks to bi-directional reduced-product AI with finite refinements

Synthesis Key: Pruning by AI

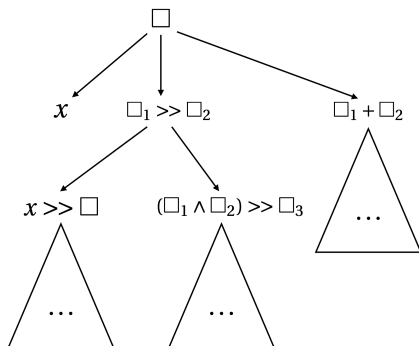
$$\begin{array}{l} S \rightarrow x \mid 0001_2 \mid S \ggg S \\ \quad \mid S \wedge S \mid S \vee S \mid S \oplus S \\ \quad \mid S + S \mid S \times S \mid S/S \end{array}$$

$$f(1011_2) = 0011_2$$

Synthesis Key: Pruning by AI

$$\begin{array}{l} S \rightarrow x \mid 0001_2 \mid S \gg S \\ \mid S \wedge S \mid S \vee S \mid S \oplus S \\ \mid S + S \mid S \times S \mid S / S \end{array}$$

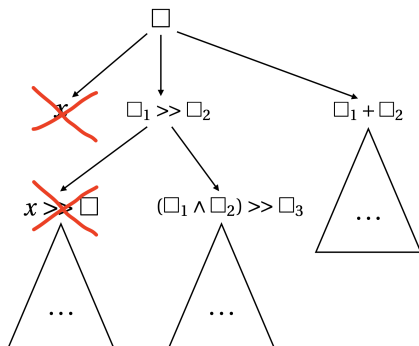
$$f(1011_2) = 0011_2$$



Synthesis Key: Pruning by AI

$$\begin{array}{l} S \rightarrow x \mid 0001_2 \mid S \gg S \\ \quad \mid S \wedge S \mid S \vee S \mid S \oplus S \\ \quad \mid S + S \mid S \times S \mid S / S \end{array}$$

$$f(1011_2) = 0011_2$$



because $\exists \square : f(1011_2) = (1011_2 \gg \square) = 0011_2$

Synthesis Pruning by AI: Forward Analysis

Candidate partial program

$$f(x) = x \vee \square$$

for

$$f(1011_2) = 0011_2$$

Forward



x	\mapsto	1011
\square	\mapsto	TTTT
$x \vee \square$	\mapsto	1T11 \neq 0011

Infeasible
output

Synthesis Pruning by AI: Forward Analysis Limitation

Candidate partial program

$$f(x) = x \oplus \square$$

for

$$f(1011_2) = 0011_2$$

Forward



x	\mapsto	1011
\square	\mapsto	TTTT
$x \oplus \square$	\mapsto	TTTT \cong 0011

Nothing
to do

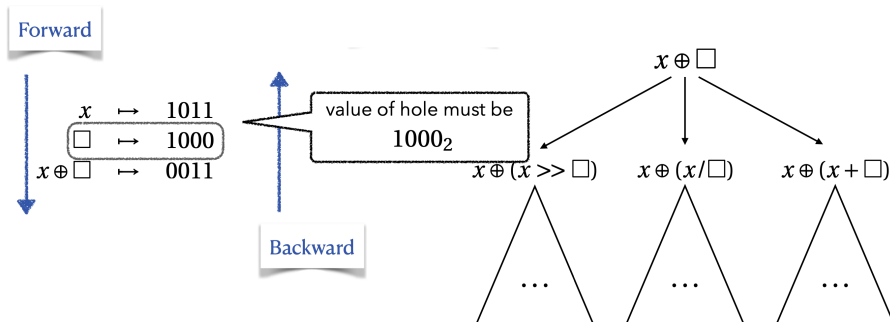
Synthesis Pruning by AI: Backward Analysis

Candidate partial program

$$f(x) = x \oplus \square$$

for

$$f(1011_2) = 0011_2$$



Synthesis Pruning by AI: Backward Analysis

Candidate partial program

$$f(x) = x \oplus \square$$

for

$$f(1011_2) = 0011_2$$

Forward



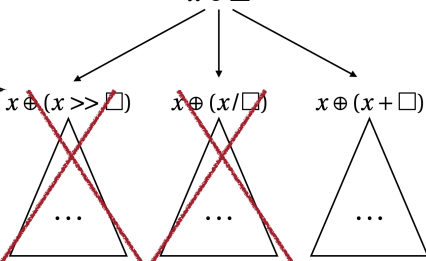
$x \mapsto 10$
 $\square \mapsto 10$
 $x \oplus \square \mapsto 00$

prune infeasible parts
 $x \gg \square \neq 1000_2$
 x / \square

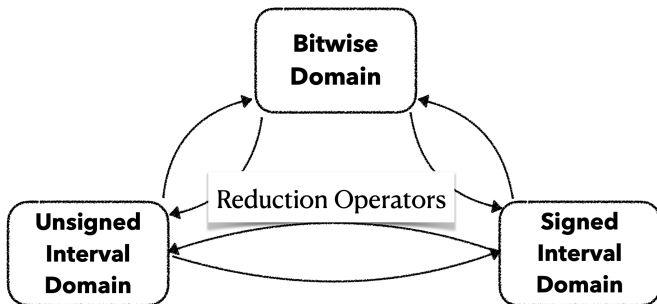
Backward



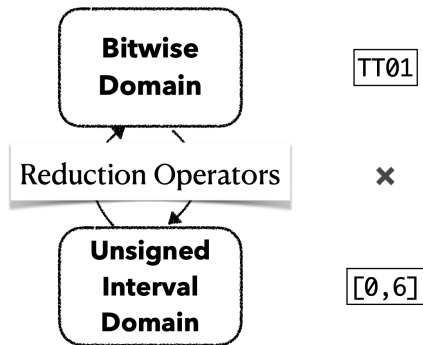
$x \oplus \square$



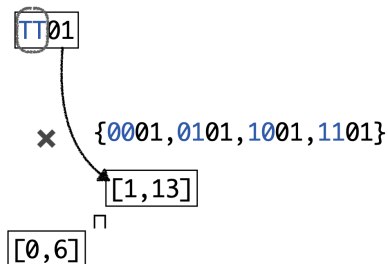
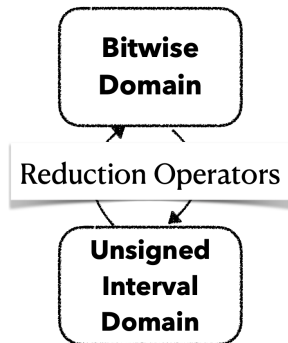
Synthesis Pruning by AI: Reduced Product for Precision & Cost



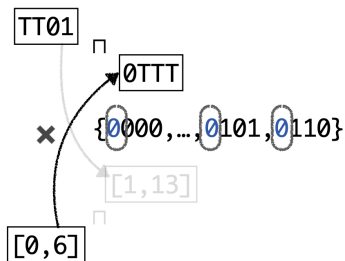
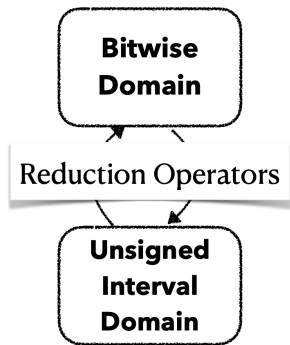
Synthesis Pruning by AI: Reduced Product for Precision & Cost



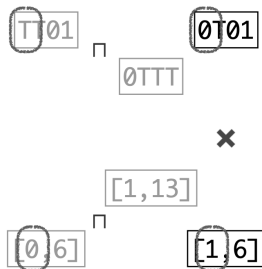
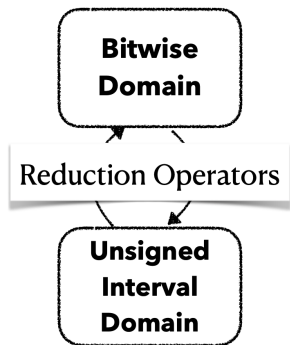
Synthesis Pruning by AI: Reduced Product for Precision & Cost



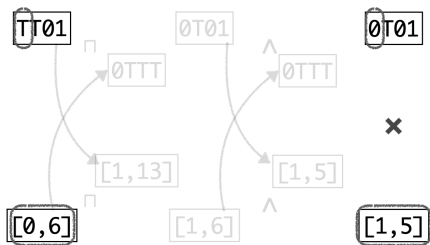
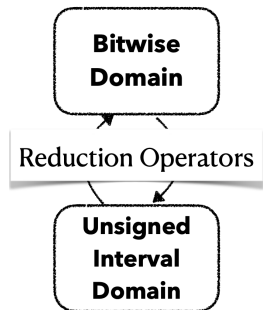
Synthesis Pruning by AI: Reduced Product for Precision & Cost



Synthesis Pruning by AI: Reduced Product for Precision & Cost



Synthesis Pruning by AI: Reduced Product for Precision & Cost



Synthesis Pruning by AI: Finite Refinements for Precision


Candidate partial program

$$f(x) = x \wedge (x \times \square)$$

for

$$f(1011_2) = 0011_2$$

Forward



x	\mapsto	1011
\square	\mapsto	TTTT
$x \times \square$	\mapsto	TTTT
$x \wedge (x \times \square)$	\mapsto	TTTT \cong 0011

Synthesis Pruning by AI: Finite Refinements for Precision

Candidate partial program

$$f(x) = x \wedge (x \times \square)$$

for

$$f(1011_2) = 0011_2$$

Forward



x	\mapsto	1011
\square	\mapsto	TTT1
$x \times \square$	\mapsto	0T11
$x \wedge (x \times \square)$	\mapsto	0011

Can we do better?



Backward

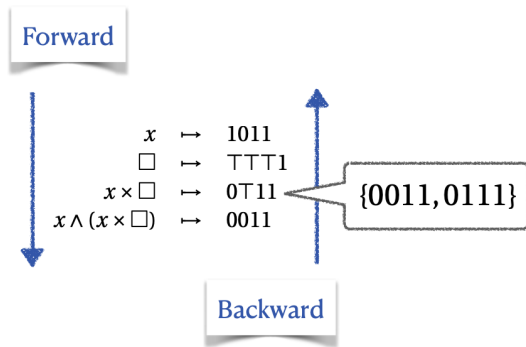
Synthesis Pruning by AI: Finite Refinements for Precision

Candidate partial program

$$f(x) = x \wedge (x \times \square)$$

for

$$f(1011_2) = 0011_2$$



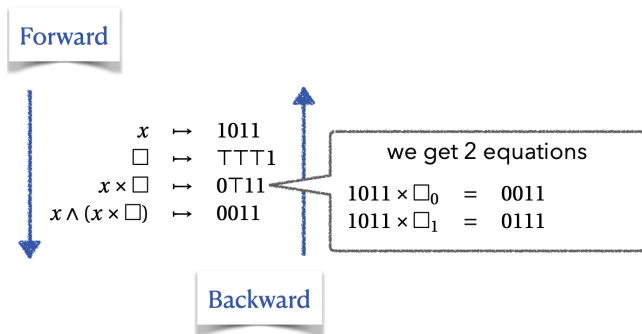
Synthesis Pruning by AI: Finite Refinements for Precision

Candidate partial program

$$f(x) = x \wedge (x \times \square)$$

for

$$f(1011_2) = 0011_2$$



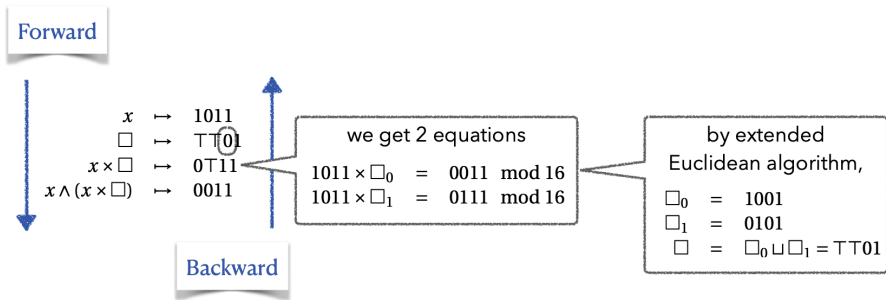
Synthesis Pruning by AI: Finite Refinements for Precision

Candidate partial program

$$f(x) = x \wedge (x \times \square)$$

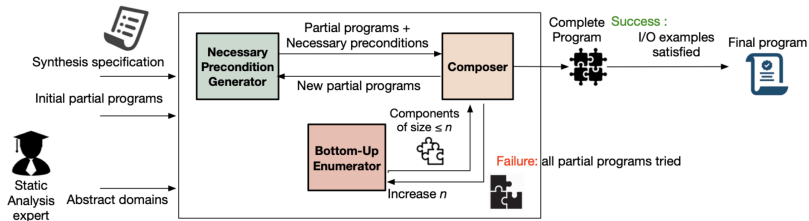
for

$$f(1011_2) = 0011_2$$



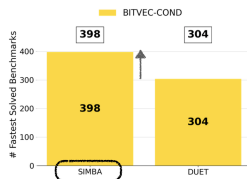
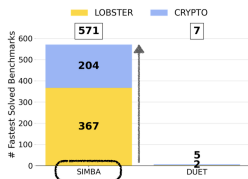
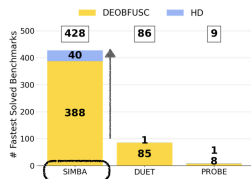
Built a Synthesis System: SIMBA

- ▶ Synthesis from Inductive specification **M** powered by Bidirectional Abstract interpretation
- ▶ available <https://github.com/yhyoon/simba> (PLDI'23)

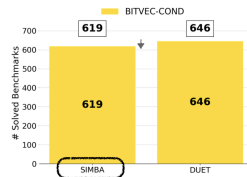
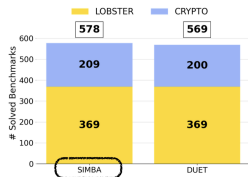
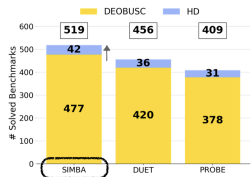


Synthesis Superb Performance

thanks to AI



of the fastest-solved problems for each domain



of the solved problems with 1h timeout for each domain

Our story upon abstract interpretation

- ▶ Analysis
 - ▶ drive to a “limit”: lots of compromises
 - ▶ commercialization: SparrowFasoo.com
 - ▶ drive to yet another “limit”: on sound & global analysis workbench
 - ▶ scalability: general sparse analysis framework
 - ▶ precision: selective x-sensitivity
- ▶ Synthesis
 - ▶ superb in performance
 - ▶ by simple bi-directional+reduced-product AI with finite refinements

Our story upon abstract interpretation

- ▶ Analysis
 - ▶ drive to a “limit”: lots of compromises
 - ▶ commercialization: SparrowFasoo.com
 - ▶ drive to yet another “limit”: on sound & global analysis workbench
 - ▶ scalability: general sparse analysis framework
 - ▶ precision: selective x-sensitivity
- ▶ Synthesis
 - ▶ superb in performance
 - ▶ by simple bi-directional+reduced-product AI with finite refinements

Without AI framework/concepts/vocabularies, we might have

Our story upon abstract interpretation

- ▶ Analysis
 - ▶ drive to a “limit”: lots of compromises
 - ▶ commercialization: SparrowFasoo.com
 - ▶ drive to yet another “limit”: on sound & global analysis workbench
 - ▶ scalability: general sparse analysis framework
 - ▶ precision: selective x-sensitivity
- ▶ Synthesis
 - ▶ superb in performance
 - ▶ by simple bi-directional+reduced-product AI with finite refinements

Without AI framework/concepts/vocabularies, we might have

- ▶ failed to conceive & develop our methods.
- ▶ been too scared of the sheer complexities of the reality.

Our story upon abstract interpretation

- ▶ Analysis
 - ▶ drive to a “limit”: lots of compromises
 - ▶ commercialization: SparrowFasoo.com
 - ▶ drive to yet another “limit”: on sound & global analysis workbench
 - ▶ scalability: general sparse analysis framework
 - ▶ precision: selective x-sensitivity
- ▶ Synthesis
 - ▶ superb in performance
 - ▶ by simple bi-directional+reduced-product AI with finite refinements

Without AI framework/concepts/vocabularies, we might have

- ▶ failed to conceive & develop our methods.
- ▶ been too scared of the sheer complexities of the reality.
- ▶ no understanding about what we were doing.
- ▶ failed to communicate our methods.

Our story upon abstract interpretation

- ▶ Analysis
 - ▶ drive to a “limit”: lots of compromises
 - ▶ commercialization: SparrowFasoo.com
 - ▶ drive to yet another “limit”: on sound & global analysis workbench
 - ▶ scalability: general sparse analysis framework
 - ▶ precision: selective x-sensitivity
- ▶ Synthesis
 - ▶ superb in performance
 - ▶ by simple bi-directional+reduced-product AI with finite refinements

Without AI framework/concepts/vocabularies, we might have

- ▶ failed to conceive & develop our methods.
- ▶ been too scared of the sheer complexities of the reality.
- ▶ no understanding about what we were doing.
- ▶ failed to communicate our methods.

Thank you