

Toward Static Analysis of Natural-Language Specifications

Hakjoo Oh
Korea University

SAS 2026, Future Trends in Static Analysis
October 8, 2026
Oakland, California

Past



- Humans wrote the code
 - "Program" meant code
 - Writing code was the bottleneck
 - Naturally, static analysis has targeted code

Now



- Developers write specs; coding agents write code
- Specs are the new “program”
- Developers write, review, and maintain specs, not code
- The bottleneck has moved to writing good specs

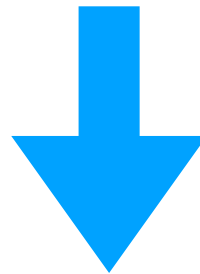
Proposal: Future static analysis should target specs

Example

```
def same_sign(a: int, b: int) -> bool:  
    """Return True if a and b have the same sign."""
```



Static Spec Analyzer



Determined behavior:

```
same_sign(+, +) = T  
same_sign(+, -) = F
```

...

Undetermined behavior:

```
same_sign(0, 0) = ?  
same_sign(0, +) = ?
```

...

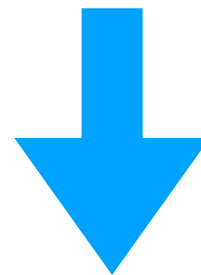
```
def same_sign(a: int, b: int) -> bool:
```

Fixed spec

```
    """Return True if a and b have the same sign.
    Zero has the same sign as zero, but not as any
    positive or negative integer."""
```



Static Spec Analyzer



Determined behavior:

`same_sign(+, +) = T`

`same_sign(+, -) = F`

`same_sign(0, 0) = T`

...

Undetermined behavior:

$\emptyset$

Approach

1. Use coding agents to sample multiple implementations

```
def same_sign(a: int, b: int) -> bool:  
    """Return True if a and b have the
```



```
# Z -- zero has a sign of its own  
return (a > 0) - (a < 0) == (b > 0) - (b < 0)
```

```
# P -- zero has the sign of the positives  
return (a >= 0) == (b >= 0)
```

```
# N -- zero has no sign, not even its own  
return a * b > 0
```

2. Analyze the programs with a sound static analyzer and join

Z				P			
	$b < 0$	$b = 0$	$b > 0$		$b < 0$	$b = 0$	$b > 0$
$a < 0$	T	F	F	$a < 0$	T	F	F
$a = 0$	F	T	F	$a = 0$	F	T	T
$a > 0$	F	F	T	$a > 0$	F	T	T

N				join			
	$b < 0$	$b = 0$	$b > 0$		$b < 0$	$b = 0$	$b > 0$
$a < 0$	T	F	F	$a < 0$	T	F	F
$a = 0$	F	F	F	$a = 0$	F	?	?
$a > 0$	F	F	T	$a > 0$	F	?	T

3. Identify the **determined** and **undetermined** parts

Statistical Soundness

Suppose the analysis proved a property φ of S : $\gamma(\widehat{\llbracket S \rrbracket}) \subseteq \varphi$.
What does it say about S ?

Theorem. If every intended meaning of S has probability $\geq \epsilon$ under the agent, then

$$\Pr[\llbracket S \rrbracket \subseteq \varphi] \geq 1 - \frac{(1 - \epsilon)^n}{\epsilon}$$

e.g., $\epsilon = 0.1$, $n = 60$: $\geq 98\%$

Summary

- Proposed **static analysis for natural-language specs**
 - “Programs” are becoming specs. Static analysis should target specs as well as code.
 - Proposed a simple approach: sample, analyze, join.
 - Statistical soundness under some assumptions.
- **Open questions:** what properties of specs to analyze? how to determine the right abstract domain for specs, how to effectively analyze multiple programs? ...

Thank you