Outcome Logic: A Unifying Foundation for Correctness and Incorrectness Reasoning

Noam Zilberstein  
Cornell University

Derek Dreyer  
MPI-SWS

Alexandra Silva  
Cornell University
“Program correctness and incorrectness are two sides of the same coin.”

— Peter O’Hearn [2020]
Can a single program logic handle correctness and incorrectness?
What is Incorrectness?

• True positives
  Reported bugs should actually be possible

• Under-approximation
  Find bugs without inspecting the entire program
int* x = malloc(sizeof(int));
*x = 1;

Malloc is nondeterministic, may return null

Dereference may segfault
Incorrectness + Hoare Logic

\{true\}
\int* x = \text{malloc}(\text{sizeof}(\text{int}));
*x = 1;
\{(\text{ok} : x \mapsto 1) \lor (\text{er} : x = \text{null})\}

Does this spec characterize the bug?  
No! We don't know if the error is reachable.
Any valid start state...
Some valid start state...

...ends up outside the post
If the program is deterministic...

...shrink P to only include the bad start state
If the program is nondeterministic...

...we need to isolate the bad end state
\[ P \xrightarrow{\sigma} [C] \xrightarrow{\tau_1, \tau_2, \ldots, \tau_n} Q \]
\[ C \]
Outcome Logic

\[ \models \langle \varphi \rangle \ C \ \langle \psi \rangle \iff \forall S . \ S \models \varphi \implies [C](S) \models \psi \]

Pre and post satisfied by SETS of states
Outcome Assertions

\[ \varphi ::= \top \]

\[ \varphi \land \psi \]

\[ \varphi \lor \psi \]

\[ \vdots \]

\[ \varphi \oplus \psi \]

\[ P \]

\[ S \models \varphi \oplus \psi \iff \exists S_1, S_2. S = S_1 \cup S_2 \]

\[ \text{and } S_1 \models \varphi \]

\[ \text{and } S_2 \models \psi \]

\[ S \models P \iff S \neq \emptyset \text{ and } S \subseteq P \]
Outcome Logic and Incorrectness

\[
\langle \text{ok} : \text{true} \rangle
\]

\[
\text{int}^* \ x = \text{malloc} (\text{sizeof} (\text{int})); \quad \star x = 1; \\
\langle (\text{ok} : x \mapsto 1) \oplus (\text{er} : x = \text{null}) \rangle
\]

But this outcome is irrelevant

This outcome must be reachable
Dropping Outcomes

\langle \text{ok: true} \rangle

\text{int* } x = \text{malloc(}\text{sizeof(int)}\text{);} \\
x = 1; \\
\langle (\text{er: } x = \text{null}) \oplus T \rangle

Still reachable

But we dropped the extra info
Outcome Logic and Incorrectness

∀ ⟨φ⟩ C ⟨ψ⟩

iff

∃φ′ ⇒ φ. sat(φ′) and ⊨ ⟨φ′⟩ C ⟨¬ψ⟩

Any "correctness" specification...

...can be disproven in Outcome Logic
Incorrectness Logic [O’Hearn 2019]

Running C in any state...
\[ \langle \text{true} \rangle \ C \ ((\text{er} : x = \text{null}) \oplus T) \]
... might segfault

Any crash where \( x \) is null...
\[ [\text{true}] \ C \ [\text{er} : x = \text{null}] \]
... is reachable from some start state

Manifest errors:
Which start states force the bug to appear? [Le et al. 2022]
How can this spec be false?
Option 1: something “bad” sometimes occurs
Option 2: something “good” never occurs
Can a single program logic handle correctness and incorrectness...

...with computational effects?
\[ \models \langle \varphi \rangle \ C \ \langle \psi \rangle \ \text{iff} \ \forall m \in M\Sigma. \ m \models \varphi \ \Rightarrow \ [C](m) \models \psi \]

\[
M \text{ is a monad (with some extra properties)}
\]
Probabilistic Programs

Network is unreliable, may drop message

Program succeeds 99% of the time

\[
\langle \text{true} \rangle
\]

\[
\text{int } x = \text{ping}(192.0.2.1);
\]

\[
\langle \Pr[x = \text{ok}] = 99\% \oplus \Pr[x = \text{er}] = 1\% \rangle
\]
“Program correctness and incorrectness are two sides of the same coin.”

— Peter O’Hearn [2020]
Program correctness and incorrectness are two usages of the same program logic.
Conclusion

*Can a single program logic handle correctness and incorrectness?*

**Incorrectness Reasoning**
- True positives
- Under-approximation

**Outcome Logic**
- Semantics parametric on *monadic* representation of effects
- Any false triple can be disproven
- Outcome Logic can identify *more* types of bugs than IL
- *Manifest errors*: it’s useful to know which start states force a bug