Interaction-Aware Patch Assessment for Multi-Fault Automated Program Repair

Omar I. Al-Bataineh
Gran Sasso Science Institute (GSSI), L'Aquila, Italy

ASE NIER 2025

Motivation

Patch overfitting persists in automated program repair (APR) because validation relies on incomplete test suites. However, real-world programs often contain multiple interacting faults.

- Conventional APR assumes independence — this leads to incorrect judgments.
- A patch that passes tests may still fail once hidden faults are revealed.

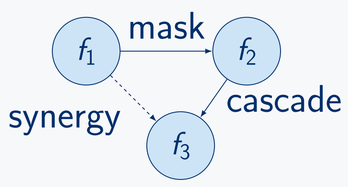
Goal: Assess patches with explicit awareness of fault interactions.

Why Multi-Fault APR is Hard

Masking: One fault suppresses another's effect.

Cascading: Fixing one enables another.

Synergy: Two faults jointly cause new failures.



Faults form a network of behavioral dependencies, not isolated bugs.

Research Question

How can patch correctness be assessed when faults interact?

- How do interactions among faults distort validation outcomes?
- What strategies can mitigate overfitting in multi-fault program repair?

Fault Interaction Model

Let program P contain faults $F = \{f_1, \ldots, f_n\}$. A fault interaction exists if repairing f_i affects f_j.

$$\mathcal{I}(f_i, f_j, P, T) = \langle b_i, v_i, b_j, v_j \rangle$$

b: behavioral change, v: visibility or reachability.

- Independence — no mutual effect.
- Masking — one fault hides another.
- Cascading — one enables the other.
- Synergy — joint activation causes new failure.

Patch Assessment Strategies

A taxonomy of interaction-aware patch evaluation.

Strategy	Concept	Use Case
Independent	Assess each patch separately	Single-fault
Fault-Aware	Evaluate with coexisting faults in context	Masking
Incremental	Sequentially validate patches	Cascading
Collaborative	Jointly test interacting patches	Synergy
Data-Driven	Predict quality via models	Unknown/ambiguous

Each strategy targets a distinct fault interaction pattern, balancing evaluation depth, computational cost, and robustness against overfitting.

Mapping Interactions to Strategies

Interaction Type	Recommended Assessment
Independence	Fault-aware independent
Masking	Fault-aware / Collaborative
Cascading	Incremental / Collaborative
Synergy	Collaborative
Ambiguous	Predictive, data-driven

This mapping guides practitioners in selecting the most suitable evaluation strategy based on the observed type of fault interaction.

Illustrative Example

Masked error activation between two faults.

f_1: Improper early exit in `process_input()` unintentionally hides f_2.

f_2: Division-by-zero in `calculate_ratio()` remains unreachable.

Traditional assessment: rejects the patch for f_1 as invalid (false negative).

Fault-aware assessment: recognizes that fixing f_1 reveals f_2, which helps reduce false negatives.

Collaborative assessment: jointly validates both patches for holistic correctness.

Key Insight

Evaluating patches in isolation **misrepresents correctness** when faults interact.

Interaction-aware assessment:

- Prevents premature rejection of valid partial fixes.
- Identifies hidden or synergistic faults early.
- Enables principled, stepwise multi-fault repair.

Conclusion & Outlook

- We present the first **formal foundation for interaction-aware patch assessment**.
- Our taxonomy of **assessment strategies** bridges theory and practice, guiding when to evaluate patches independently or collaboratively.
- **Next step:** automatically infer the interaction relation $\mathcal{I}$ through combined static, dynamic, and AI-guided analysis—closing the loop between detection and repair.