

COMPILER DESIGN LAB · REVIEW 3 (FINAL VIVA)

AuxFlow: Analysis, Optimization, & Empirical Evaluation

Cycle Classification, Build Planning, Runtime Verification, and 15-Member Benchmark Corpus Results

Student: Upayan Mazumder (24BDS0367)

Course: Compiler Design Lab

Milestone: Review 3 — Full 16-Phase Implementation & Evaluation Complete

End-to-End Compiler Pipeline for Multi-Pass Documents

PHASES 1–7: FRONT-END

- **Lexer & Catcodes**
16 TeX category codes, single-pass scanner, precise source spans.
- **Recursive Descent Parser**
Argument signatures, error recovery, multi-file AST forest.
- **Symbol Tables & Checks**
2-pass declaration/resolution, zero-engine diagnostics.
- **Auxiliary Reference Graph**
Fact instances with exists/value split and lagged typed edges.

PHASES 8–11: ANALYSIS

- **Iterative Tarjan SCC**
Condensation DAG computation without interpreter stack limits.
- **Taint Analysis**
Worklist reachability from condition reads to CONTENT_VOLUME.
- **Stable-Pass Recurrence**
Derives exact pass numbers P for fact stabilization.
- **4-State Classification**
CERTIFIED, BENIGN_CYCLIC, AMPLIFYING, UNKNOWN.

PHASES 12–16: EVALUATION

- **Build Planner Optimizer**
Dead-file/tool elimination, run-count derivation, safety invariant.
- **Runtime Verification**
Canonical normalization of auxiliary records into fixpoint sets.
- **15-Member Corpus**
M1–M8 non-convergence mechanisms covering all declared gates.
- **7 Baselines Measured**
1050 containerized runs prove Gate 1 and Gate 2 (EC1) empirically.

Detecting Amplifying Feedback Without Full Simulation

THE TAINT-SEED MECHANISM

- Core Question: Can a cross-reference value influence how much content the document generates?
- Taint Seeds: Facts whose values are read inside loop conditions or bounds (`\forloop`, `\whiledo`, `\ifthenelse`).
- Forward Worklist: Taint propagates monotonically along value-carrying edges (push-at-most-once invariant).
- STRUCTURAL Edge Insertion: Each tainted fact gets a STRUCTURAL edge `f -> CONTENT_VOLUME` with its verbatim source span.
- Cycle Decision: If any SCC contains a STRUCTURAL edge, the document is classified AMPLIFYING.

FOUR-STATE CLASSIFICATION DECISION

[CERTIFIED (Bound = $P + 1$ passes)]

Provably acyclic dependency structure. Standard documents settle within derived bound.

[BENIGN_CYCLIC (Value-only cycle)]

Cycles exist in value edges (e.g. section lists) but no content quantity changes.

[AMPLIFYING (Refused by default)]

A cross-reference controls content growth. High divergence risk. Requires `--force`.

[UNKNOWN (Conservative fallback)]

Contains unmodelled packages/macros. Falls back to default 3-run schedule.

Safe Build Optimization & Sound Fixpoint Comparison

BUILD PLANNER PASSES (PHASE 12)

- **1. Dead-File Elimination**

Scans and spell-checks only files reachable in the inclusion graph. Unused chapters skipped.

- **2. Dead-Tool Elimination**

Omits BibTeX unless both citation facts and bibliography commands exist.

- **3. Run-Count Derivation**

CERTIFIED plans emit exactly $P+1$ engine passes; non-certified emit conservative floor (3 runs).

- **4. Safety Invariant**

Asserts non-certified executable plans have at least 3 engine runs before emission.

RUNTIME VERIFIER (PHASE 13)

- **Canonical Normalization**

Parses .aux, .toc, .lof, .lot, .bbl into order-independent record tuples.

- **Documented Exclusion List**

Only \relax is excluded (boiler-plate). Unmodelled commands stay observable.

- **Sound Fixpoint vs Warnings**

Compares normalized state $A_k == A_{k-1}$ directly, avoiding unsound engine warnings.

- **Bounded Extension & Misses**

Extends execution while state changes up to cap; logs PredictionMiss when exceeding certified bound.

Systematic Coverage of Multi-Pass Convergence Mechanisms

THE 15-MEMBER CORPUS (bench/corpus/b01_plain .. b15_ibid)

ID	MECHANISM / PURPOSE	EXPECTED	OBSERVED (D)	STATUS
b01_plain	Plain prose with no auxiliary state producers/consumers	CERTIFIED	CERTIFIED	PASS
b02_backref	Backward cross-reference (label then ref)	CERTIFIED	CERTIFIED	PASS
b03_fwdref	Forward cross-reference (ref then label)	CERTIFIED	CERTIFIED	PASS
b04_chain	Contents extent shifts page before reference	CERTIFIED	CERTIFIED	PASS
b05_toc	Contents list without cross-reference uses	CERTIFIED	CERTIFIED	PASS
b06_bib	BibTeX tool edge with generated-list cascade	CERTIFIED	CERTIFIED	PASS
b07_footnote	Footnote placement feedback via resolved ref width	UNKNOWN	UNKNOWN	PASS
b08_cycle	M1: Loop bounded by forward page ref (Gaspar 2014)	AMPLIFYING	AMPLIFYING	PASS
b09_thesis	Realistic multi-chapter thesis (TOC/LOF/LOT, bib, unreach)	CERTIFIED	CERTIFIED	PASS
b10_random	M8: Nondeterministic pdfTeX random auxiliary state	UNKNOWN	UNKNOWN	PASS
b11_width	M3: Reference width changes line-break (SE 30674)	CERTIFIED*	CERTIFIED	PASS (decl.)
b12_float	M4: Top float placement and Roman page number width	CERTIFIED*	CERTIFIED	PASS (decl.)
b13_varioref	M6: varioref dynamic text moves across page break	CERTIFIED*	CERTIFIED	PASS (decl.)
b14_ifthen	M2: ifthenelse conditional font sizing on pageref	AMPLIFYING	AMPLIFYING	PASS
b15_ibid	M5: biblatex authoryear-icomp same-page ibid feedback	UNKNOWN	UNKNOWN	PASS

* Declared false negatives: predicted in advance as limitations of the line-breaking abstraction.

1050 Containerized Runs Across 7 Compilation Baselines

ORACLE COMPARISON (EXECUTIONS TO CERTIFY)

BASELINE PERFORMANCE SUMMARY:

- Baseline A (Fixed Schedule = 3 runs):
Silently wrong on all 6 non-converging docs.
- Baseline B (Engine Warnings):
Unsound! Declares CONVERGED after 1 run on b10_random (never warns on random changes), misses TOC shifts with no ref on b05_toc.
- Baselines C / C' / C'' (State Checksums):
Sound dynamic fixpoint detection.
C costs N+1 runs; C' and C'' cost N runs.
- Baseline D (AuxFlow Static Alone):
Zero engine runs! Classifies in <10ms.
- Baseline E (Static Plan + Verification):
Combines static speed with dynamic safety.
Correctly catches b11-b13 false negatives at cap=6 instead of failing silently.

GATES & EXTERNAL TOOL CROSS-CHECKS

GATE 1 (Classification Separation): PASS

- b08 correctly AMPLIFYING with 3 named facts.
- b05 correctly CERTIFIED (no false alarm).
- b11-b13 matched declared false negatives.

GATE 2 (Empirical Claim EC1): PASS

- Predicted N $\geq$ Ground Truth N* for all certified docs.
- 0 undeclared bound failures across all 15 members.

SHIPPING TOOL CROSS-CHECKS (RQ6):

- latexmk 4.79: Fails to converge on b08, b10, b11-b14.
- latexrun: Fails to converge on b08, b10, b11-b15.
--> Independently confirms b11-b13 non-convergence is real, not an AuxFlow artifact!
- Typst Compiler: Refuses b08_cycle with 'error: loop seems to be infinite'.

Verifying b08_cycle & b11_width via bench/demo.sh

DEMONSTRATION 1: b08_cycle (AMPLIFYING)

```
$ auxflow classify bench/corpus/b08_cycle/main.tex
auxflow: AMPLIFYING CYCLE -- may not converge.
cycle (3 facts):
  PAGeref_USE(end:generated) (main.tex:5:32)
    -> CONTENT_VOLUME (main.tex:8:3)
      STRUCTURAL: reaches content construct
    -> LABEL_PAGE(end:generated) (main.tex:10:29)
      -> PAGeref_USE(end:generated)
[Refuses to emit plan without --force]

$ auxflow run --force main.tex
Pass 1 -> 1 page
Pass 2 -> 2 pages
Pass 3 -> 3 pages ... Pass 6 -> 6 pages
Status: cap_exceeded (Monotone divergence!)
```

DEMONSTRATION 2: b11_width (DECLARED MISCLASSIFICATION)

```
$ auxflow classify bench/corpus/b11_width/main.tex
auxflow: CERTIFIED -- 2 runs
[Acyclic ARG: no STRUCTURAL edges found]

THE DECLARED EXPERIMENT:
expected.json:
  declared_false_negative: true
  mechanism: M3 ref width changes line break

GROUND TRUTH REALITY:
Real engine state hashes across 8 runs:
  Run 1: add8c2c1 (page 1)
  Run 2: 72beb0e4 (page 2)
  Run 3: add8c2c1 (page 1)
  Run 4: 72beb0e4 (page 2) ... period-2 forever!

Conclusion: Predicted misclassification is confirmed!
```

Summary of Completed Compiler Deliverables

PROJECT METRICS & VERIFICATION

- ✓ 16 Implementation Phases Complete: From Lexer to Final Evaluation.
- ✓ 75 GitHub Issues & 7 Milestones Closed: 58/58 sub-issues linked and reconciled.
- ✓ Quality Gate Scores: 245 unit tests passed, strict mypy type check clean, ruff lint/format clean.
- ✓ Plan Validation: 0.0 defects, 100% quality score on `validate_plan.py`.
- ✓ Fast Performance: Static classification in <10ms, including multi-chapter thesis.

KEY RESEARCH CONTRIBUTIONS

★ Instance-Level Reference IR

A formal dependency graph (ARG) with an exists/value split that keeps ordinary multi-pass documents acyclic.

★ Taint-Based Amplification Analysis

Detects monotone divergence risks statically before execution, producing source-located diagnostics.

★ Empirical Soundness Proof

Demonstrates that engine warnings are unsound, and that runtime verification rescues static false negatives.

★ Transparent Falsification

Predicts its own limitations (M3, M4, M6) in advance rather than disguising them as unexpected errors.