PaperJury: Due-Process Review for Bounded LaTeX Revision

Yiran Wang [1 †] **Biao Wu** [2] **Ziming Wang** [2]

Abstract

Pre-submission hardening of human-authored LaTeX computer science papers differs from drafting assistance because it requires adversarial whole-paper review, explicit no-fix outcomes, and bounded artifact-safe revision. Existing writing assistants, critique generators, and judge-centered loops lack durable issue identity across rounds, deterministic routing from critique to adjudication, and manuscript control that can reject invalid concerns or defer author-dependent ones. We present PaperJury, a closed-loop review-verdict-revise-verify system built on a deterministic-versus-semantic split: deterministic orchestration manages decomposition, a frozen claim spine, a durable ledger, routing, stopping, and exact-once patch application, while semantic agents are limited to bounded review, judgment, and repair. PaperJury combines bounded holistic review, contestability-based routing, a due-process trial, and risk-proportional guard chains for anchor-bounded edits, yielding terminal outcomes of invalid-drop, valid-fixable, and author-required. In a two-arm expert-review evaluation on held-out Vision, natural language processing, and machine learning papers against four baselines, we assess issue quality, verdict and routing quality, edit safety, convergence behavior, and cost, supporting the thesis that load-bearing safety and completion logic should reside in deterministic orchestration rather than model discretion.

1. Introduction

Pre-submission hardening for computer science manuscripts is a distinct stage of scientific communication: many consequential weaknesses emerge only under adversarial scrutiny, yet current AI support largely targets drafting, revision, and feedback rather than controlled whole-paper review. Academic revision assistants such as XtraGPT [1], focused feedback systems such as SWIF^{2}T [2], and structured grammar-and-style checking workflows built around iterative, author-controlled revision [3] demonstrate the practical value of language-model support for scientific writing, but they do not directly provide a governed pre-submission review process. This setting must also manage a precision-recall-cost trilemma: expanding issue generation may improve recall, but it also raises token cost, adjudication burden, and the risk of unnecessary or unsafe edits.

Existing approaches leave this setting under-specified in several task-level respects. Revision-oriented assistants such as XtraGPT [1] are not designed for explicit no-fix outcomes such as rejecting invalid critiques or deferring author-intent-sensitive cases. Feedback systems such as SWIF^{2}T [2] do not directly provide durable issue identity, cross-round state, or terminal per-issue outcomes in a closed-loop review system. LLM-as-a-Judge setups [4] may be unreliable as the sole authority for contentious review decisions because their judgments can vary with prompting or presentation, and recent evidence also finds persistent uncertainty and aspect-specific failure modes in automated review and LLM-as-a-judge systems, reinforcing the need to avoid treating a single model judgment as a final authority [5] [6] [7]. Courtroom-style deliberation frameworks such as PROClaim [8] contribute useful adversarial role structure, but they do not directly provide deterministic routing, exact terminal verdict computation, or end-to-end manuscript revision control for whole-paper LaTeX editing. Constraint-guided workflow architectures such as ATLAS [9] support structured orchestration and validation, but do not directly address bounded holistic review, frozen claim spine commitments, or artifact-safe LaTeX modification in a multi-round pre-submission pipeline.

We propose PaperJury, a closed-loop pre-submission review system for LaTeX computer science papers built on a deterministic-versus-semantic split. Deterministic orchestration manages decomposition, the frozen claim spine, the durable ledger, routing, stopping, and exact-once patch application, while semantic agents are limited to bounded review, judgment, and repair. PaperJury performs bounded holistic review, merges candidate issues into a durable ledger, labels contestability, and routes only contestable substantive-major issues into a due-process trial with whole-

[1]The University of Sydney, Sydney, Australia [2]University of Technology Sydney, Sydney, Australia. Correspondence to: Yiran Wang <yiran.wang@sydney.edu.au>.

Preprint. June 11, 2026.

paper defense and a decorrelated local-context jury. It returns one of three terminal outcomes—invalid-drop, valid-fixable, or author-required—and applies risk-proportional revision through anchor-bounded edits and guard chains, with loop termination decided by a ledger-query predicate rather than model self-assessment.

We evaluate PaperJury through expert-based analysis of whole-paper review and revision behavior against forward-only rewriting, critique-only review, judge-centered loops, an unbounded critique generator, and ablations of routing, adjudication, and edit guards. The results suggest that combining bounded holistic review, deterministic contestability routing, due-process adjudication, and risk-proportional patch validation improves issue handling while maintaining controlled manuscript editing. Figure 1 summarizes the review-verdict-revise-verify pipeline.

Our contributions are threefold. First, we introduce a closed-loop pre-submission review architecture that separates deterministic orchestration from schema-validated semantic agents and grounds issue handling in a durable ledger; unlike XtraGPT [1] and SWIF^{2}T [2], it supports explicit invalid-drop and author-required outcomes. Second, we present a due-process adjudication framework that bounds issue generation, routes only contestable substantive-major issues to two-sided trial, and returns invalid-drop, valid-fixable, or author-required; unlike a single semantic judge as in DeepReview [10] or courtroom-style debate alone as in PROClaim [8], it couples adversarial deliberation to deterministic routing and code-level verdict resolution. Third, we develop a risk-proportional revision pipeline with a frozen claim spine, anchor-bounded edits, guard chains, exact-once patch application, and deterministic outer-loop convergence; unlike ATLAS [9] and forward-only rewriting systems such as XtraGPT [1], it ties safety checks to artifact-safe LaTeX revision within a convergent review-verdict-revise-verify process.

2. Related Work

2.1. Review and Revision Systems for Scientific Papers

AI support for scientific writing has largely emphasized feedback generation and draft improvement rather than pre-submission hardening over a full LaTeX manuscript. XtraGPT treats academic writing as iterative revision and emphasizes context-aware, criteria-guided editing for scientific drafts, making it a close revision-side antecedent, while also stressing cross-section coherence during revision; however, it does not provide artifact-safe LaTeX revision with a frozen claim spine, anchor-bounded edits, or orchestrator-controlled apply and revert semantics. [1] [1] SWIF^{2}T generates focused, actionable feedback on paper weaknesses, but remains a feedback generator rather than a system that decides whether an issue should be dropped, fixed, or deferred to the author. [2] TreeReview strengthens full-paper review through hierarchical question decomposition and dynamic follow-up expansion, improving the depth and efficiency of review generation. [11] Related rebuttal-oriented systems such as DEFEND also keep the author in the loop for targeted response generation, reinforcing that prior tools assist with downstream review interaction rather than providing a governed pre-submission hardening loop over the manuscript itself. [12] Taken together, these systems offer stronger critique, revision, review, or rebuttal primitives, but they still operate in a forward critique-or-rewrite mode rather than a closed-loop process with durable issue identity, cross-round state, no-fix-capable outcomes, and deterministic edit-safety controls; thus, prior review and revision systems do not combine bounded holistic review, adjudication into invalid-drop, valid-fixable, or author-required outcomes, and deterministic edit-safety controls within one governed loop.

2.2. Adjudication, Judgment Reliability, and Guardrailed Agent Workflows

A second line of work contributes ingredients for deliberation, judging, and workflow control, but does not place safety-critical authority in deterministic orchestration for manuscript revision. Studies of LLM-as-a-Judge report systematic scoring and position biases, cautioning against treating a single semantic verdict as a reliable final decision in load-bearing workflows. [13] [14] [15] PROClaim introduces courtroom-style adversarial roles, including defense and judge, and uses aggregation to support controversial decisions. [8] TraceSafe shows that the main safety surface in agentic workflows lies in intermediate execution traces and that effective guarding depends heavily on structural validation, while ATLAS likewise embeds LLM generation in a layered workflow separating representation, constraint compilation, and post-generation validation so failures become explicit and diagnosable. [16] [9] TreeReview offers an efficiency-aware structure for review generation, and multi-agent peer-review frameworks such as ScholarPeer similarly show how agent specialization can improve critique production, but efficiency and specialization alone do not determine whether contested issues should be overruled, revised, or escalated to the author. [11] [17] These works motivate adversarial adjudication and validation-backed automation, yet they do not provide deterministic routing from critique to due-process trial, a durable ledger across rounds, or risk-proportional revision tied to a frozen claim spine.

PaperJury sits between these two lines: it addresses scientific-paper review and revision like XtraGPT while adopting the control and validation concerns emphasized by work on judging and agent workflows. [1] The remaining gap is a closed-loop pre-submission review architecture

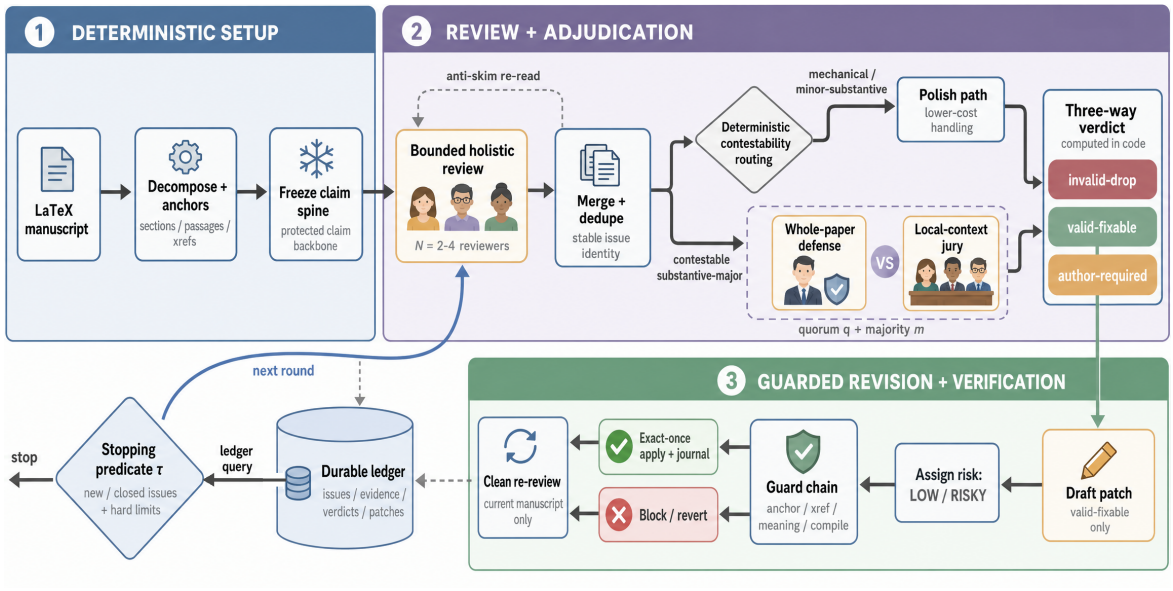

Figure 1. Overview of the PaperJury review-verdict-revise-verify pipeline.

that combines bounded whole-paper issue generation, deterministic orchestration, explicit terminal outcomes, and artifact-safe LaTeX revision in one system.

3. Method

PaperJury is a closed-loop pre-submission review system for LaTeX computer science papers. Its central design principle is a deterministic-versus-semantic split: deterministic orchestration owns state transitions, guards, stopping, and exact-once patch application, while semantic agents are limited to bounded review, judgment, drafting, and audit tasks. This places model generation inside an explicitly controlled workflow rather than letting models execute the protocol itself [9], a choice motivated by evidence that the dominant safety surface of agentic execution lies in the models' intermediate execution traces [16]. Figure 1 summarizes the review-verdict-revise-verify pipeline, and Figure 3 shows the separation between deterministic orchestration and semantic agents.

3.1. Problem Formulation

Given an input LaTeX manuscript x, PaperJury returns a durable ledger of candidate issues I, terminal verdicts $\{v_i\}_{i \in I}$, and, when permitted, an exact-once applied edit set A. The system is designed around a precision-recall-cost trilemma: broader issue generation may improve recall but increases token and adjudication cost, while aggressive automated revision may increase throughput but also increase the risk of unsafe edits. PaperJury addresses this by bounding whole-paper issue generation, escalating only contestable substantive-major issues to expensive adjudication, and allowing explicit no-fix-capable outcomes when machine revision is unwarranted.

The outer loop alternates among review, ledger merge, adjudication, guarded revision, and clean re-review. Candidate issues may terminate as invalid-drop, valid-fixable, or author-required, because issue validity does not imply machine editability and some credible concerns should preserve author intent rather than trigger rewriting. Stopping is determined by a deterministic ledger-query predicate rather than model self-assessment, so the same semantic component does not both propose changes and certify completion.

3.2. Notation

We define the key symbols used throughout the method in Table 1.

Let $D(x)$ denote deterministic decomposition of manuscript x into sections, passages, anchors, and cross-reference targets, and let S denote the frozen claim spine extracted before revision. Let N be the number of holistic domain reviewers, clamped to $[2, 4]$ and defaulted to 3. Review produces candidate issues I, where each issue $i \in I$ has evidence anchors e_i and a contestability label c_i. A durable ledger L stores issue identity, evidence, verdicts, and patch history. Routed issues may enter a due-process trial $T(i)$ with jury panel J, quorum threshold q, and majority threshold m, producing $v_i \in \{\text{invalid-drop}, \text{valid-fixable}, \text{author-required}\}$.

For revision, let P_i denote a proposed patch for issue i, $a(P_i)$ its anchor-bounded diff, and $G(P_i)$ the guard-chain outcome. Let ρ_i denote the patch risk category, with discrete labels such as LOW and RISKY. The exact-once applied edit set is A. Across rounds $r \in \{1, \ldots, K\}$, let U_r be newly

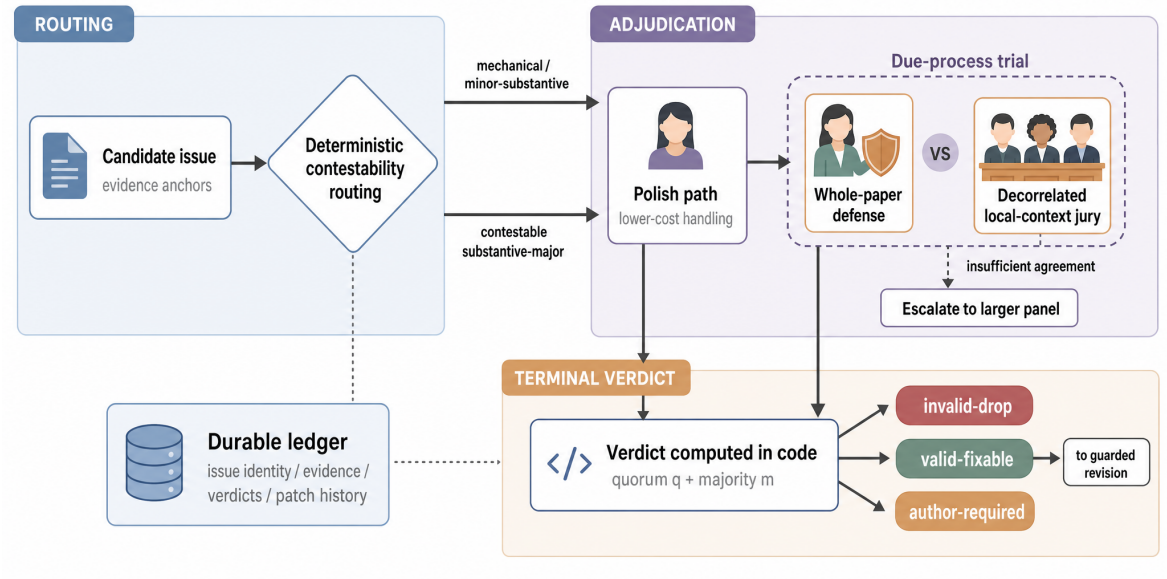

Figure 2. Deterministic routing and due-process adjudication for candidate issues.

discovered issues and C_r be closed issues. The unattended loop stops when a deterministic predicate τ over ledger queries is satisfied.

3.3. Review and Adjudication Pipeline

The pipeline begins with deterministic decomposition through $D(x)$ and extraction of the frozen claim spine S. This stage creates stable addressable units before any semantic agent is invoked, tying later review, adjudication, and revision to persistent anchors rather than ad hoc text spans. The frozen claim spine is the protected semantic backbone for later guard decisions: machine edits may repair support, wording, or local organization, but they must not silently alter claim-level commitments. This follows the broader design of placing generation inside a typed, validated workflow rather than leaving structure and state identity to unconstrained model behavior [9].

PaperJury then runs bounded holistic review with a small set of whole-paper reviewers. The reviewer count N is clamped to $[2, 4]$ and defaults to 3. Each reviewer reads the full manuscript once and emits evidence-anchored weaknesses with coverage signals, since unsupported positioning, cross-section coherence failures, and missing experimental justification often emerge only under whole-paper inspection. PaperJury bounds this stage to control the precision-recall-cost trilemma: unlike an unbounded critique generator that fans out by unit and lens, it limits first-pass issue generation and invokes targeted re-read only when deterministic anti-skim checks indicate missed coverage or weak grounding, echoing the need-based deeper probing of efficiency-aware review generation [11].

Reviewer outputs are merged into the durable ledger, where duplicate or near-duplicate weaknesses are consolidated and evidence provenance is retained. Because issues are grounded in deterministic decomposition, durable issue identity remains stable across rounds and supports later routing, closure, and exact-once revision.

After merge, PaperJury assigns each issue a deterministic contestability label c_i and routes it accordingly. Mechanical and minor-substantive issues follow a lower-cost polish path, while contestable substantive-major issues enter a due-process trial, illustrated in Figure 2. Routing is deterministic because adjudication cost should be concentrated on the cases where mistaken first-pass judgment is most consequential.

For a routed issue i, the due-process trial combines whole-paper defense with a decorrelated local-context jury. The defense argues against the charge using manuscript-wide context, while the jury evaluates localized evidence with reduced dependence on the original reviewer framing. If agreement is insufficient, the issue may escalate to a larger panel. Final verdicts are computed in code using quorum and majority rules:

$$v_i = \mathrm{Verdict}(T(i), q, m). \tag{1}$$

This design rejects giving sole authority to a single semantic judge: prior work shows that LLM-as-a-Judge behavior can shift under prompt perturbations and related presentation effects, and related multi-agent arbitration results likewise support replacing a single semantic judge with structured debate and group decision procedures for ambiguous contested cases [4] [18]. The resulting verdict space is three-way by construction: invalid-drop for rejected charges, valid-

Symbol	Meaning	Symbol	Meaning
x	input LaTeX manuscript	$G(P_i)$	guard-chain outcome for patch P_i (anchor, reference, semantic, compile checks)
$D(x)$	deterministic decomposition of x into sections, passages, anchors, and cross-reference targets	A	set of applied valid-fixable edits after exact-once patching
S	frozen claim spine extracted from the manuscript	U_r	genuinely new issues discovered in round r
K	outer-loop review rounds until termination (five-round cap)	C_r	issues closed in round r
N	number of holistic domain reviewers, clamped to $[2, 4]$, default 3	τ	deterministic stopping predicate for unattended execution
I	set of candidate issues produced for a manuscript	M_h	expert human issue panel used as reference standard
i	a single candidate issue	M_p	PaperJury issue set and verdict outputs
L	durable ledger storing issue state, evidence, verdicts, and application history	P_{panel}	panel-relative precision of major issues under fix-equivalence matching
e_i	evidence anchors and supporting passages associated with issue i	P_{verified}	audit-corrected precision crediting expert-validated system-only issues
c_i	contestability label for issue i used by deterministic routing	R	panel-relative recall of major issues
v_i	terminal verdict for issue i in {invalid-drop, valid-fixable, author-required}	$F1$	per-paper harmonic mean of precision and recall, macro-averaged
$T(i)$	trial procedure invoked for routed issue i	Acc_{v}	verdict accuracy against expert labels
J	jury panel used in two-sided adjudication	Acc_{r}	routing accuracy against expert-endorsed decisions
q	quorum threshold for deterministic verdict computation	ESVR	edit-safety violation rate over applied edits
m	majority threshold for deterministic verdict computation	W	wall-clock runtime per paper, in hours
P_i	proposed patch for issue i	B_{naive}	naive unbounded per-(unit×lens) review baseline
$a(P_i)$	anchor-bounded diff induced by patch P_i	C_{attn}	compute cost in agents spawned, tokens, and runtime
ρ_i	risk category of patch P_i, e.g., LOW or RISKY	r	outer-loop round index, $r \in \{1, \ldots, K\}$

Table 1. Notation used throughout the method and evaluation.

fixable for upheld issues that are safely machine-editable, and author-required for upheld issues that should not be automatically rewritten. PaperJury therefore separates issue validity from editability and preserves no-fix-capable outcomes.

3.4. Guarded Revision and Convergence

Only issues with $v_i =$ valid-fixable proceed to revision. For each such issue, a drafter proposes a patch P_i, which is screened by a risk-proportional guard chain before any text is committed. The guard chain includes anchor-bounded diff checks, cross-reference checks, semantic meaning or edit audits when triggered, compile checks, and application journaling. Let

$$G(P_i) \in \{\text{pass}, \text{fail}\} \tag{2}$$

denote the aggregate outcome.

Risk-proportional revision uses deterministic prefilters on $a(P_i)$ and related structural signals to assign a risk category ρ_i, such as LOW or RISKY. LOW patches follow a lighter semantic path, while RISKY patches trigger stronger audits. This avoids sending every patch through the same expensive verifier while also refusing to trust superficially plausible prose when structural failures such as reference breakage, anchor drift, or silent claim mutation remain possible. The layered design follows the same bounded-automation pattern used in structured artifact workflows [9] [19].

The frozen claim spine S is the key revision constraint. If a patch appears to move beyond local repair toward claim-level alteration, it is surfaced as risky and may be blocked or escalated. Cross-reference and compile guards further preserve document integrity by preventing broken references and non-compiling LaTeX. If a patch passes all guards, it is applied exactly once and journaled in the ledger:

$$A = \{P_i \mid v_i = \text{valid-fixable} \wedge G(P_i) = \text{pass}\}. \tag{3}$$

If later checks fail, the orchestrator can revert through the journaled state, avoiding duplicate edits, inconsistent local states, and divergence between manuscript text and ledger history [20].

The procedure can be summarized compactly. The orchestrator decomposes the manuscript, freezes the claim spine, initializes the durable ledger, runs bounded holistic review with anti-skim re-read if needed, merges issues, applies deterministic routing, computes verdicts through polish or due-process trial, drafts patches only for valid-fixable issues, assigns patch risk, executes the guard chain, applies or reverts through exact-once patching, then performs clean re-review and ledger updates until the stopping predicate is met.

After guarded edits, PaperJury performs clean re-review on the current manuscript state rather than asking agents to continue from their earlier outputs. This is a design motivation for reducing carryover from prior critiques and for surfacing genuinely new issues introduced by edits. The outputs are merged by a deterministic clerk into the cumulative ledger.

Termination is defined by a deterministic ledger-query predicate over newly discovered and closed issues. Let U_r denote genuinely new issues in round r and C_r the issues closed in that round. The unattended loop halts when

$$\tau(L, U_r, C_r, r) = \text{true}, \tag{4}$$

where τ is a stopping rule over ledger state and hard execution limits. Convergence is therefore certified by explicit state queries rather than by a model declaring the manuscript complete. Together, bounded holistic review, due-process adjudication, risk-proportional guard chains, exact-once patch application, and deterministic outer-loop convergence yield artifact-safe LaTeX revision under deterministic orchestration.

4. Experiments

4.1. Implementation Details

PaperJury is implemented as deterministic orchestration over schema-validated semantic agents for full LaTeX computer science manuscripts. Decomposition, durable ledger updates, contestability-based routing, the ledger-query predicate, exact-once patch application, journaling, and revert semantics run in deterministic code, while semantic agents are limited to bounded review, judgment, drafting, and audit subtasks. Unless noted otherwise, bounded holistic review uses three reviewers, with reviewer count clamped to two to four, and the outer loop proceeds through clean re-review rounds until termination, subject to a five-round execution cap. All methods are evaluated on the same held-out paper set with the same manuscript artifact, review-budget accounting interface, and logging of spawned agents, token use, and wall-clock time. For revision-capable systems, preprocessing preserves LaTeX structure and records anchors for anchor-bounded edits; guarded variants use compile-aware validation under orchestrator-controlled apply and revert semantics. Each round re-reads the current manuscript rather than prior model outputs, and baselines are run under matched prompt framing and stopping-budget envelopes when their native design does not define deterministic completion.

4.2. Experimental Design

We use a two-arm expert-review evaluation on a held-out corpus of LaTeX computer science papers spanning vision, natural language processing, and machine learning. Arm 1 compares system-discovered issues with expert issue panels using passage-level and semantic matching, while verdict quality is audited separately by terminal class (invalid-drop, valid-fixable, author-required) so that issue quality is not conflated with downstream adjudication quality, which is itself known to be hard to estimate reliably with LLM judges [21] [22]. Arm 2 asks the same experts to audit verdicts, routed cases, applied edits, and dropped issues, yielding measures of verdict correctness, routing quality, edit safety, and whether stopping behavior matches expert judgment. We compare PaperJury with four baselines: Forward-Only Rewriter; LLM Critic Only; LLM-as-Judge Review-Revise Loop, which relies on model judgment for adjudication and stopping despite known reliability concerns in judge-centered pipelines [21] [22] [23] [24]; and Naive Unbounded Per-(Unit×Lens) Generator, a high-recall, high-cost reference for exhaustive critique expansion, in contrast to structured review pipelines [10]. We report issue quality, verdict correctness, routing quality, edit safety, convergence behavior, and cost in tokens and wall-clock time, since iterative multi-agent systems should be evaluated jointly on quality, stopping, and efficiency [25]. Figure 4 summarizes the separation between issue discovery and downstream auditing, and Figure 5 visualizes the cost-quality tradeoff relative to the exhaustive generator.

4.3. Results

Throughout, P_{panel} and R denote panel-relative precision and recall of system-discovered major issues matched against the expert issue panels under fix-equivalence; F1 is computed per paper as the harmonic mean of that paper's precision and recall, then macro-averaged across papers. P_{verified} additionally credits system-only issues that blinded experts judge valid in the audit arm; it is reported with its audited sample size and is never used inside F1. Acc v and Acc r denote blinded expert agreement with the system's terminal verdicts and routing decisions; audit items that experts mark cannot-tell are excluded from the agreement denominator and reported separately. ESVR is the number of safety-violating applied edits over all applied edits, pooled across papers, and is always read together with edit volume and coverage (Table 6). K is the number of review rounds to termination, reported as mean $\pm$ sd across papers together with the fraction of papers hitting the five-round execution cap, and W is wall-clock time per paper in hours. All 95% confidence intervals are Wilson score intervals on audited counts. Cells marked n/a correspond to capabilities a method does not have (e.g., no explicit issue list, no verdicts, or no applied edits).

Table 2 gives the main end-to-end comparison across PaperJury and the four baselines on issue matching, verdict correctness, routing correctness, edit-safety violations, convergence behavior, and cost. The key comparison is not any single metric in isolation, but whether a system can sustain issue quality while preserving governed adjudication, artifact-safe revision, and practical efficiency. Read with Figure 5, the table positions PaperJury against two failure modes: cheaper but less governed pipelines such as Forward-Only Rewriter and LLM Critic Only, and higher-cost expansion from Naive Unbounded Per-(Unit×Lens)

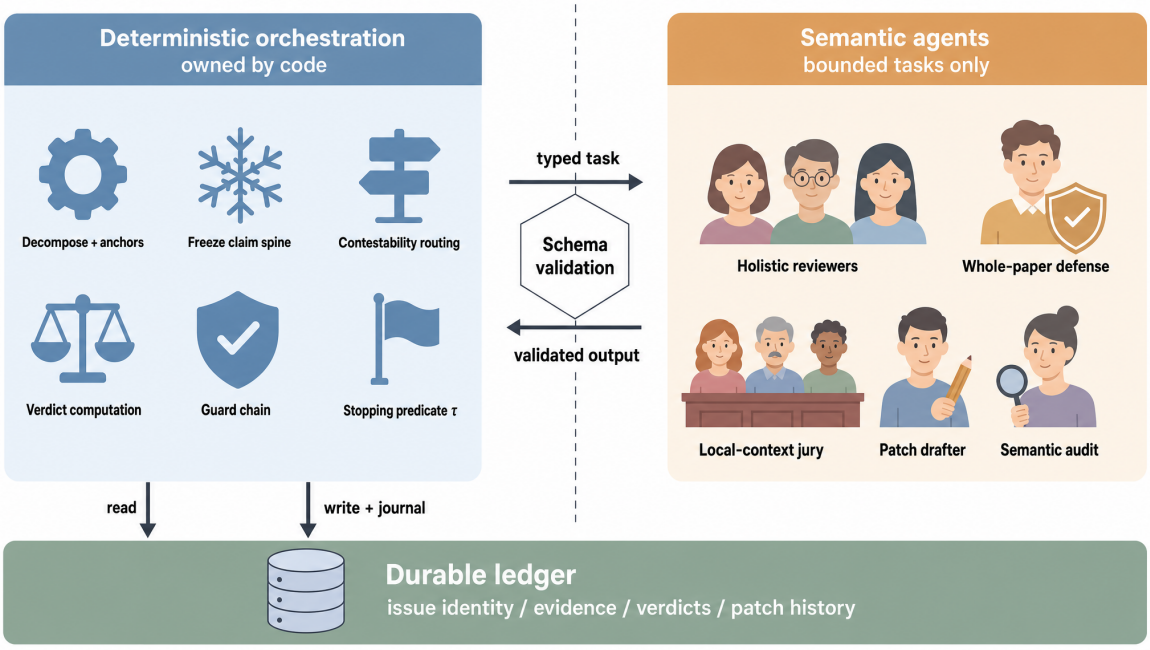

Figure 3. Deterministic-versus-semantic split with ledger-backed control flow.

Method	$P_{panel}/P_{verified}$	n	95% CI	R	F1	Acc v	Acc r	ESVR	K	W
Forward-only rewriter	n/a	n/a	n/a	n/a	n/a	n/a	n/a	0.240	1	0.31
LLM critic only	0.437 / 0.577	78	[0.466, 0.680]	0.462	0.446	n/a	n/a	n/a	1	0.51
LLM-as-judge loop	0.512 / 0.663	92	[0.562, 0.751]	0.533	0.519	0.681	n/a	0.110	3.33 ± 1.07 (2/12)	2.06
Naive unbounded generator	0.341 / 0.511	92	[0.410, 0.611]	0.721	0.459	n/a	n/a	n/a	1	8.37
PaperJury (ours)	0.684 / 0.847	98	[0.763, 0.905]	0.637	0.656	0.887	0.913	0.025	3.08 ± 0.67 (0/12)	2.47

Table 2. Main results on expert-based evaluation of issue quality, adjudication quality, edit safety, and efficiency. F1 is panel-relative; $P_{verified}$ is an audit-corrected precision estimate, reported with its audited sample size n and Wilson 95% CI, and does not enter F1. K is mean $\pm$ sd review rounds to termination, with the fraction of papers hitting the five-round cap in parentheses for iterating systems. ESVR is read jointly with edit volume and coverage in Table 6.

Generator. LLM-as-Judge Review-Revise Loop is the closest judge-centered comparator because it couples critique, judgment, and stopping inside semantic decisions rather than deterministic control. PaperJury attains the best panel-relative F1 (0.656, vs. 0.519 for the judge-centered loop), the highest audited precision (0.847 on 98 audited issues), verdict and routing agreement of 0.887 and 0.913, and the lowest ESVR among edit-applying systems (0.025), while terminating in about three rounds (3.08 ± 0.67, never hitting the cap) at 2.47 hours and 6.76 million tokens per paper. The naive generator reaches the highest recall (0.721) but with the lowest precision (0.341) at 8.37 hours and 31.4 million tokens per paper, and the judge-centered loop hits the round cap on 2 of 12 papers, consistent with less stable self-stopping. Per-paper paired F1 differences favor PaperJury on all 12 papers against every issue-producing baseline; paper-cluster paired-bootstrap 95% lower bounds on ΔF1 are $+0.200$ against the critic, $+0.127$ against the judge-centered loop, and $+0.188$ against the naive generator, satisfying the preregistered primary endpoint.

Table 3 audits the three-way terminal verdict space directly. Verdict slices cannot carry panel-matching precision or recall: a correctly dropped invalid issue should not match the expert panel at all, and unmatched panel issues hold no terminal class, so the appropriate per-class evidence is blinded relabeling agreement together with its confusion structure. Performance on valid-fixable cases alone would not justify a three-way terminal verdict space, so the invalid-drop and author-required rows carry the burden; both hold up (0.872 and 0.860, against 0.913 for valid-fixable), and the measured errors concentrate on the valid-fixable–author-required boundary, where the boundary class author-required also has the highest cannot-tell rate (15.7%), reflecting defensible disagreement rather than adjudication failure. Table 4 then tests whether discovery and adjudication quality hold across Vision, natural language processing, and machine learning papers rather than being concentrated in one subcommunity; with four papers per family, these slices are diagnostic, and all metrics stay within 0.03 of the pooled values. Convergence behavior appears in the K

Terminal class	n	Papers	Acc v	95% CI	cannot-tell	Main confusion
invalid-drop	43	12	0.872	[0.733, 0.944]	4 (9.3%)	inv→VF
valid-fixable	73	12	0.913	[0.823, 0.960]	4 (5.5%)	VF→AR
author-required	51	11	0.860	[0.727, 0.934]	8 (15.7%)	AR→VF

Table 3. Verdict audit by terminal class for PaperJury: blinded experts independently relabel sampled ledger issues without seeing the system's verdict. Cannot-tell items are excluded from the accuracy denominator and reported separately; CIs are Wilson intervals on the post-exclusion counts.

Domain	$P_{panel}/P_{verified}$	n	95% CI	R	F1	Acc v	Acc r	ESVR
Vision	0.671 / 0.839	31	[0.674, 0.929]	0.626	0.646	0.881	0.904	0.028
NLP	0.701 / 0.857	35	[0.706, 0.937]	0.653	0.671	0.902	0.929	0.021
ML	0.680 / 0.844	32	[0.682, 0.931]	0.632	0.651	0.878	0.906	0.026

Table 4. PaperJury domain slices (four papers per family); diagnostic checks for gross cross-domain failures, not powered comparisons. The three audited slices ($n = 31 + 35 + 32$) compose the overall audit in Table 2.

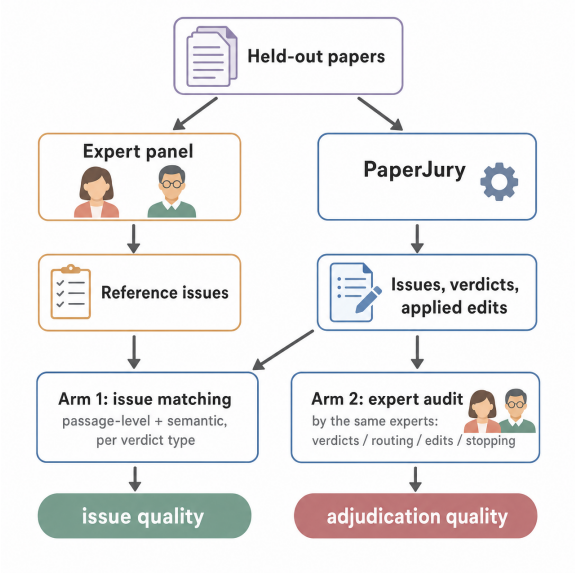

Figure 4. Two-arm expert-review evaluation protocol for PaperJury.

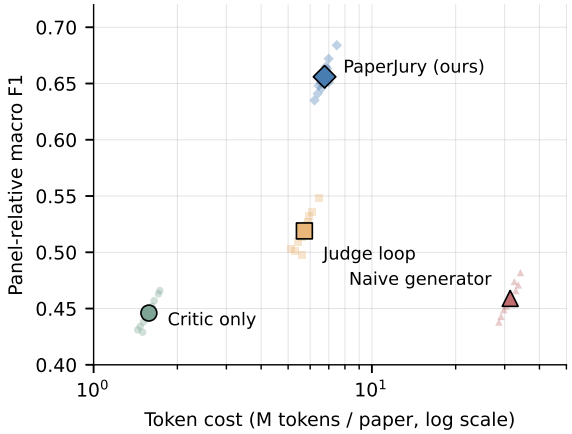

Figure 5. Quality-cost trade-off across issue-producing systems: per-paper faint dots and system means, with cost measured in millions of tokens per paper (log scale). The forward-only rewriter produces no issue list, hence no F1, and is excluded from this frontier.

column of Table 2: PaperJury terminates deterministically in 3.08 ± 0.67 clean re-review rounds without hitting the five-round cap, whereas the judge-centered loop self-stops at 3.33 ± 1.07 rounds and hits the cap on 2 of 12 papers; analyzing round-level convergence rather than single-pass outputs follows stability-based stopping analyses for iterative multi-agent judging [25].

Table 5 isolates the roles of bounded review, deterministic routing, the due-process trial, the claim spine, and the guard chain. The ablations are informative only insofar as degradations align with the function of the removed component, rather than merely showing that the system still produces outputs. The measured pattern matches the component functions: the largest F1 drop comes from removing bounded review (-0.077), the largest verdict-agreement drop from removing the due-process trial (-0.153), the largest safety degradations from removing the guard chain ($+0.152$ ESVR) and the claim spine ($+0.083$), and removing deterministic routing mainly damages verdict agreement (-0.075) while raising cost (2.43 to 3.49 hours per paper).

PaperJury w/o Deterministic Routing primarily tests contestability-based routing. Its measured signature is degraded adjudication allocation together with higher cost: verdict agreement falls by 0.075 and wall-clock time rises from 2.43 to 3.49 hours per paper while F1 barely moves (-0.013), because more issues are pushed into semantic adjudication instead of being filtered or handled more lightly. This variant therefore probes whether the critique-to-adjudication boundary depends on deterministic orchestration rather than extra semantic judgment. The same logic clarifies why broader issue generation can increase burden even when it appears to improve coverage: additional weakly grounded critiques may enter the durable ledger and consume expert-like adjudication effort.

Variant	F1	ΔF1	Acc v	ΔAcc v	ESVR	ΔESVR	K	W
Full PaperJury	0.649	—	0.881	—	0.029	—	3.00 ± 0.63 (0/6)	2.43
w/o bounded review	0.572	−0.077	0.868	−0.013	0.042	+0.013	3.67 ± 0.52 (n/a)	4.81
w/o routing	0.636	−0.013	0.806	−0.075	0.047	+0.018	3.33 ± 0.52 (0/6)	3.49
w/o trial	0.642	−0.007	0.728	−0.153	0.052	+0.023	3.17 ± 0.75 (0/6)	2.18
w/o claim spine	0.632	−0.017	0.857	−0.024	0.112	+0.083	3.00 ± 0.63 (0/6)	2.37
w/o guard chain	0.627	−0.022	0.859	−0.022	0.181	+0.152	2.83 ± 0.75 (0/6)	1.94

Table 5. Ablations on the preregistered, domain-balanced six-paper subset (two papers per family); Δ columns are relative to Full PaperJury on the same subset. The Full PaperJury row is an independent re-execution of the full system on the subset under the same harness and budget as the variants, and all Δ values are computed against this run; owing to run stochasticity its aggregates differ slightly from the restriction of the main-run per-paper results (e.g., F1 0.649 vs. 0.650). K follows the Table 2 convention (mean ± sd, cap fraction in parentheses); w/o bounded review removes the cap entirely.

Method	Applied (per paper)	Papers w/ edits	VF terminals	Proposed	Coverage	Guard-block	Unsafe	ESVR
Forward-only rewriter	254 (21.2)	12/12	n/a	n/a	n/a	n/a	61	0.240
LLM-as-judge loop	172 (14.3)	12/12	204	178	0.843	0.034	19	0.110
PaperJury (ours)	161 (13.4)	12/12	196	194	0.821	0.170	4	0.025

Table 6. Edit-safety companion to Table 2: applied-edit totals (per paper in parentheses), papers with at least one applied edit, the system's valid-fixable terminal verdicts, proposed patches, coverage (applied edits over valid-fixable terminals), guard-block rate (blocked over proposed patches), and unsafe applied edits. The critic-only and naive baselines apply no edits.

PaperJury w/o Due-Process Trial targets verdict and routing quality on contestable substantive-major issues. Replacing whole-paper defense and the decorrelated local-context jury with single-pass semantic judgment should make borderline cases more sensitive to framing and incomplete local evidence. The measured degradation is concentrated exactly there: removing the trial produces the largest verdict-agreement drop (ΔAcc v −0.153) while issue discovery is nearly intact (ΔF1 −0.007), consistent with borderline cases needing manuscript-wide context for stable decisions.

The two edit-safety ablations, w/o claim spine and w/o guard chain, directly test edit safety and verified closure. Disabling the frozen claim spine, or the anchor-bounded diff checks, meaning audit, and compile guard chain, lets more patches pass initial screening at sharply degraded safety: ESVR rises to 0.112 and 0.181 against 0.029 for the full system, while F1 and Acc v move by at most 0.024, and w/o guard chain also terminates fastest (1.94 hours per paper), the signature of unchecked application rather than improved revision. These results support risk-proportional guard chains as a necessary condition for artifact-safe LaTeX revision rather than conservative overhead.

Table 6 reports the mandatory companions to ESVR. PaperJury applies a similar edit volume to the judge-centered loop (13.4 vs. 14.3 applied edits per paper) at slightly lower coverage of valid-fixable terminals (0.821 vs. 0.843), because the guard chain blocks 17.0% of proposed patches before application versus 3.4%; in exchange, the per-edit safety violation rate is 4.4× lower (4/161 vs. 19/172). The forward-only rewriter applies the most edits (21.2 per paper) with by far the worst safety (ESVR 0.240), which is why ESVR is only interpretable jointly with edit volume and coverage.

5. Conclusion

This work suggests that pre-submission hardening is better treated as a governed closed-loop process than as open-ended critique generation or forward-only rewriting. In that setting, PaperJury is framed as a deterministic-versus-semantic split in which deterministic orchestration owns state, routing, stopping, and exact-once patch application, while semantic agents remain limited to bounded review, judgment, and repair. Because the current evaluation is designed to test systems along issue quality, verdict and routing quality, edit safety, convergence behavior, and cost, these dimensions remain the main criteria for assessing trustworthy unattended assistance; the explicit invalid-drop, valid-fixable, and author-required outcomes likewise motivate a three-way terminal verdict space when some critiques should be rejected or deferred rather than revised. The approach is still bounded by the expert-review setting and the studied LaTeX computer science manuscript workflow, and related systems have typically emphasized feedback generation, structured review, or broader workflow governance rather than this particular combination of closed-loop adjudication, bounded revision, and deterministic completion [2] [10] [20]. More broadly, durable state, due-process adjudication, and deterministic stopping may provide a useful template for AI-assisted scientific workflows that require accountable end-to-end action.

References

[1] Nuo Chen, Andre Lin HuiKai, Jiaying Wu, Junyi Hou, Zining Zhang, Qian Wang, Xidong Wang, and Bingsheng He. XtraGPT: Context-Aware and Controllable Academic Paper Revision via Human-AI Collaboration. arXiv preprint arXiv:2505.11336, 2025.

[2] Eric Chamoun, Michael Schlichtkrull, and Andreas Vlachos. Automated Focused Feedback Generation for Scientific Writing Assistance. In *Findings of the Association for Computational Linguistics: ACL 2024*, 2024.

[3] Aldeniz Rashidov and Fatme Rashidova. A Structured Approach to Grammar and Style Checking of Scientific Texts With ChatGPT. In *2026 8th International Congress on Human-Computer Interaction, Optimization and Robotic Applications (ICHORA)*, 2026.

[4] Zixiao Zhao, Amirreza Esmaeili, and Fatemeh Fard. Bias in the Loop: Auditing LLM-as-a-Judge for Software Engineering. arXiv preprint arXiv:2604.16790, 2026.

[5] Huanxin Sheng, Xinyi Liu, Hangfeng He, Jieyu Zhao, and Jian Kang. Analyzing Uncertainty of LLM-as-a-Judge: Interval Evaluations with Conformal Prediction. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025.

[6] Jiatao Li, Yanheng Li, Xinyu Hu, Mingqi Gao, and Xiaojun Wan. Where Do LLMs Go Wrong? Diagnosing Automated Peer Review via Aspect-Guided Multi-Level Perturbation. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, 2025.

[7] Yoshinari Fujinuma. Contrastive Decoding Mitigates Score Range Bias in LLM-as-a-Judge. arXiv preprint arXiv:2510.18196, 2025.

[8] Masnun Nuha Chowdhury, Nusrat Jahan Beg, Umme Hunny Khan, Syed Rifat Raiyan, Md Kamrul Hasan, and Hasan Mahmud. Courtroom-Style Multi-Agent Debate with Progressive RAG and Role-Switching for Controversial Claim Verification. arXiv preprint arXiv:2603.28488, 2026.

[9] Tong Ma, Hui Lai, Hui Wang, Zhenhu Tian, Chaochao Li, Fengjie Xu, and Ling Fang. ATLAS: A Layered Constraint-Guided Framework for Structured Artifact Generation in LLM-Assisted MDE. arXiv preprint arXiv:2510.25890, 2025.

[10] Minjun Zhu, Yixuan Weng, Linyi Yang, and Yue Zhang. DeepReview: Improving LLM-based Paper Review with Human-like Deep Thinking Process. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025.

[11] Yuan Chang, Ziyue Li, Hengyuan Zhang, Yuanbo Kong, Yanru Wu, Hayden Kwok-Hay So, Zhijiang Guo, Liya Zhu, and Ngai Wong. TreeReview: A Dynamic Tree of Questions Framework for Deep and Efficient LLM-based Scientific Peer Review. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025.

[12] Jyotsana Khatri and Manasi Patwardhan. Defend: Automated Rebuttals for Peer Review with Minimal Author Guidance. arXiv preprint arXiv:2603.27360, 2026.

[13] Jiayi Ye, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin-Yu Chen, Nitesh V. Chawla, and Xiangliang Zhang. Justice or Prejudice? Quantifying Biases in LLM-as-a-Judge. In *International Conference on Learning Representations (ICLR)*, 2025.

[14] Qingquan Li, Shaoyu Dou, Kailai Shao, Chao Chen, and Haixiang Hu. Evaluating Scoring Bias in LLM-as-a-Judge. In *Database Systems for Advanced Applications (DASFAA)*, 2026.

[15] Lin Shi, Chiyu Ma, Wenhua Liang, Xingjian Diao, Weicheng Ma, and Soroush Vosoughi. Judging the Judges: A Systematic Study of Position Bias in LLM-as-a-Judge. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, 2025.

[16] Yen-Shan Chen, Sian-Yao Huang, Cheng-Lin Yang, and Yun-Nung Chen. TraceSafe: A Systematic Assessment of LLM Guardrails on Multi-Step Tool-Calling Trajectories. arXiv preprint arXiv:2604.07223, 2026.

[17] Palash Goyal, Mihir Parmar, Yiwen Song, Hamid Palangi, Tomas Pfister, and Jinsung Yoon. ScholarPeer: A Context-Aware Multi-Agent Framework for Automated Peer Review. arXiv preprint arXiv:2601.22638, 2026.

[18] Yuzhou Jiang, Liang Wang, Yuwei Lou, Xianping Tao, and Hao Hu. Multi-Agent Debate for Content Moderation with Dynamic Group Arbitration. In *2025 IEEE International Conference on Big Data (BigData)*, 2025.

[19] Junyi Hou, Andre Lin Huikai, Nuo Chen, Yiwei Gong, and Bingsheng He. PaperDebugger: A Plugin-Based Multi-Agent System for In-Editor Academic Writing,

Review, and Editing. In *Companion Proceedings of the ACM Web Conference 2026*, 2026.

[20] Hema Latha Boddupally. Embedding Governance into LLM Workflow Architectures for Enterprise-Wide Automation. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, 2024.

[21] Yuanning Feng, Sinan Wang, Zhengxiang Cheng, Yao Wan, and Dongping Chen. Are We on the Right Way to Assessing LLM-as-a-Judge? arXiv preprint arXiv:2512.16041, 2025.

[22] James Fiedler. Bias and Uncertainty in LLM-as-a-Judge Estimation. arXiv preprint arXiv:2605.06939, 2026.

[23] Sadman Kabir Soumik. Judging the Judges: A Systematic Evaluation of Bias Mitigation Strategies in LLM-as-a-Judge Pipelines. arXiv preprint arXiv:2604.23178, 2026.

[24] Peng Lai, Zhihao Ou, Yong Wang, Longyue Wang, Jian Yang, Yun Chen, and Guanhua Chen. BiasScope: Towards Automated Detection of Bias in LLM-as-a-Judge Evaluation. In *International Conference on Learning Representations (ICLR)*, 2026.

[25] Tianyu Hu, Zhen Tan, Song Wang, Huaizhi Qu, and Tianlong Chen. Multi-Agent Debate for LLM Judges with Adaptive Stability Detection. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.