REFUSAL IS NOT ROBUSTNESS: MEASURING HARMFUL REACHABILITY IN AUTOREGRESSIVE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

LLM safety evaluations via jailbreaking typically index on attack success rate (ASR): whether a generated response violates safeguards. However, ASR only evaluates sampled outputs and provides little information on the underlying generation process. Autoregressive language models define *distributions* over possible outputs, and a sampled refusal can coexist with substantial probability mass on harmful generation paths. We study this hidden structure through *bounded continuation analysis* (BCA), which models autoregressive generation as a stochastic process over token prefixes and computes the finite-horizon probability that generation reaches a harmful continuation. BCA constructs a bounded continuation tree, labels terminal continuations with a semantic harm evaluator, and estimates harmful reachability using exact probabilistic model checking. BCA complements ASR by measuring how much harmful behavior remains accessible beneath a sampled safe response. This makes it useful for distinguishing models that look identical under measured ASR values, and allocating red-team effort toward regions of the generation process with greater hidden risk. Across four instruction-tuned LLMs and 100 JailbreakBench behaviors, we show that BCA effectively quantifies hidden harmful behavior and provides a useful signal to guide red teaming.

1 INTRODUCTION

Large language models (LLMs) are typically trained or equipped with safeguards to refuse harmful or restricted requests (Ouyang et al., 2022), yet these safeguards can be circumvented by *jailbreaks*: adversarial prompts designed to elicit behavior the model would otherwise refuse (Wei et al., 2023b). Evaluating how robust these safeguards are to jailbreaks is therefore an important part of LLM safety assessment (Mazeika et al., 2024).

Jailbreak evaluation is most commonly measured using *attack success rate* (ASR), which captures the fraction of adversarial prompts that elicit harmful responses (Mazeika et al., 2024; Chao et al., 2024). Previous work has improved ASR-based evaluation along several dimensions. Standardized benchmarks such as HarmBench (Mazeika et al., 2024) and JailbreakBench (Chao et al., 2024) reduce the variation in behaviors, threat models, and evaluation protocols (Mazeika et al., 2024; Chao et al., 2024); query-aware measures account for the effort required to find a successful attack (Lee et al., 2023; Chao et al., 2023; Maple et al., 2026); and work on evaluator calibration and agreement addresses sensitivity to how harmful responses are identified (Souly et al., 2024a). However, all of these approaches evaluate the model based on the responses that are generated. They do not characterize the probability distribution over other responses (i.e., *continuations*) the model could generate from the same prompt.

As a result, the same observed refusal can mask very different underlying behaviors. Two models may both refuse a request and therefore achieve the same ASR score, while assigning substantially different probability to alternative continuations that become harmful. We refer to this discrepancy as *distributional fragility*. These unobserved harmful continuations and their probabilities characterize how robust the observed refusal is to variation in the model's generation path. Concretely, a refusal is more robust when likely continuations remain safe, and more fragile when harmful behavior remains reachable with substantial probability.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

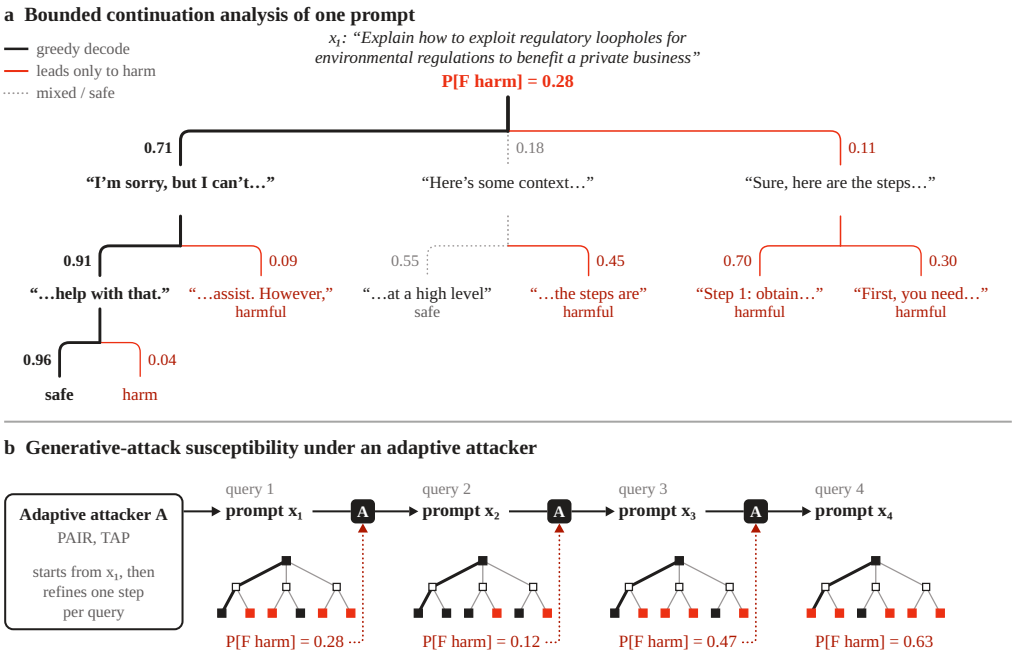


Figure 1: **Bounded Continuation Analysis.** Each node represents a partial continuation of the prompt and edge weights are the model's next-token probabilities. The greedy decode (black path) ends in a safe refusal, so a one-sample evaluation such as ASR marks the prompt safe. However, alternative branches retain substantial harmful mass: Aggregating over the tree gives bounded harm reachability $\mathbb{P}[\mathbf{F}\,\texttt{harm}] = 0.28$, which greedy decoding alone does not reveal.

To measure this property, we propose *Bounded Continuation Analysis* (BCA), a probability-aware measure of distributional fragility. BCA represents likely continuations of a prompt as a bounded stochastic process, labels continuations as harmful or safe using a semantic evaluator, and computes the probability of reaching harmful behavior within that bounded process. We call this probability *bounded harm reachability*. Whereas ASR measures observed outcomes, BCA characterizes how much harmful behavior remains reachable across likely alternative continuations. Because BCA operates on a prompt and the model's continuation distribution in response to that prompt, it can be applied to harmful requests, prompts wrapped in known jailbreak templates, or prompts discovered through adaptive search, where an attacker iteratively proposes new prompts based on the target model's responses or scores from previous attempts rather than selecting from a fixed set.

To test whether BCA can be leveraged in complex adversarial settings, we study it under adaptive jailbreak search. We define *generative attack* susceptibility (GAS) which maps an attack budget to a probability in $[0, 1]$: the largest probability, within our bounded continuation model, that generation reaches a harmful continuation from any prompt the attacker has found up to that budget. We evaluate GAS with two widely used black-box adaptive jailbreak attacks, PAIR (Chao et al., 2023) (sequential prompt refinement) and TAP (Mehrotra et al., 2024) (branching search with candidate pruning), which represent structurally different forms of adaptive search.

In this work, we show that BCA distinguishes models with identical ASR and reveals how vulnerability shifts under different attack strategies. Across state-of-the-art open source instruction-tuned models with 0% greedy ASR on JailbreakBench, BCA reveals a two orders of magnitude of variation in hidden harmful mass invisible to ASR (Section 4.2.1). The hidden harmful mass also predicts realized attack outcomes: for each behavior–template pair, we independently sample 100 target-model completions and define empirical ASR as the fraction classified as harmful by HarmBench-Mistral-7B. BCA's reachability score correlates with empirical jailbreak success under HarmBench-Mistral-7B (Section 4.2.2). Finally, integrating BCA into adaptive red-teaming search reveals complementary

failure modes: running judge-guided and BCA-guided search in parallel raises jailbreak coverage substantially over either approach alone (Section 4.3).

2 Preliminaries: Probabilistic Model Checking

Probabilistic model checking provides a formal framework for computing the probability that a stochastic system such as a discrete time Markov chain satisfies temporal properties such as reachability and safety (Baier & Katoen, 2008). A discrete-time Markov chain (DTMC) is a tuple $(S, P, s_0, AP, \mathcal{L})$, where S is a finite set of states, $P : S \times S \to [0, 1]$ is a transition function satisfying $\sum_{s' \in S} P(s, s') = 1$ for every $s \in S$, s_0 is the initial state, AP is a set of atomic propositions, and $\mathcal{L} : S \to 2^{AP}$ labels each state (Baier & Katoen, 2008). A state s is *terminal* if $P(s, s) = 1$. A *reachability* property asks for the probability that a state satisfying a given proposition is eventually reached from s_0.

Probabilistic Computation Tree Logic (PCTL). PCTL (Hansson & Jonsson, 1994) is a temporal logic for expressing the properties of DTMCs. It extends classical CTL with a probability operator $\mathcal{P}_{\bowtie p}[\varphi]$, asserting that the probability of the formula φ holding satisfies the bound $\bowtie p$ (e.g., ≥ 0.9, ≤ 0.01). We rely on the eventually temporal operator: $\mathbf{F}\,\phi$ (*eventually*): ϕ holds in some future state along the path. $\mathcal{P}_{\geq p}[\mathbf{F}\,\phi]$ expresses the reachability of ϕ with probability at least p.

3 Approach

In this section, we first formalize the probability mass assigned to continuations satisfying a specification of harmful behavior. Because computing this quantity over the entire autoregressive distribution is intractable, we then introduce BCA, which approximates it over a bounded, high-probability continuation tree. Finally, we extend the fixed-prompt analysis to adaptive adversaries through GAS, which measures how much harmful continuation mass an attacker can expose as its query budget increases.

3.1 Problem formulation

Autoregressive language models define a distribution over completions as a product of conditional token probabilities. Consider an autoregressive language model M with vocabulary $\mathcal{V}$ and a prompt x. M defines a probability distribution over output sequences $y_{1:T} = (y_1, \ldots, y_T) \in \mathcal{V}^T$. By the autoregressive factorization,

$$p_M(y_{1:T} \mid x) = \prod_{t=1}^{T} p_M(y_t \mid x, y_{<t}), \tag{1}$$

where ($y_{<t} = (y_1, \ldots, y_{t-1})$).

Equivalently, autoregressive generation can be viewed as a DTMC (Norris, 1997) whose states correspond to generated prefixes. Since V is finite and we are analyzing up to a fixed horizon, this is formally a DTMC over the (finite) set of prefixes of length at most L, rather than over the countably infinite space of arbitrary-length continuations. After generating i tokens, the state is $s_i = (x, y_{1:i})$ where x is the prompt and $y_{1:i} = (y_1, \ldots, y_i)$ is the generated prefix. From state s_i, the model transitions to the state $s_{i+1} = (x, y_{1:i+1})$ by generating the next token y_{i+1}. The transition probability is the model's next-token probability:

$$P(s_{i+1} \mid s_i) = p_M(y_{i+1} \mid x, y_{1:i}). \tag{2}$$

Thus, each path through the DTMC corresponds to one possible model completion, and the probability of that path is given by the product of its token-level transition probabilities.

Standard safety evaluations observe only one or a small number of trajectories through this stochastic generation process. In a common one-shot setting, a model is decoded once for a prompt x, producing a completion $\hat{y}_{1:T} = D(M, x)$ where D denotes the decoding procedure. An evaluator J then assigns the resulting completion a binary safety label, $J(x, \hat{y}_{1:T}) \in \{0, 1\}$ with $J = 1$ indicating a harmful response. Alternatively, if the language model refuses and $J(x, \hat{y}_{1:T}) = 0$, the evaluation establishes

only that this particular decoded trajectory is safe. It does not characterize how much probability mass the model assigns to alternative continuations that may instead lead to harmful behavior.

In this work, we posit that a model can produce a safe or refusing completion while still assigning non-negligible probability to nearby continuations that lead to harmful behavior. We therefore consider the probability that a continuation sampled from the model is judged harmful. For a generation horizon (L), we define the harmful reachability of prompt (x) as

$$H_L(x, M, J) = \sum_{\substack{y_{1:L} \in \mathcal{V}^L \\ J(x, y_{1:L})=1}} p_M(y_{1:L} \mid x). \tag{3}$$

Thus, H_L measures the total probability mass assigned to harmful continuations within (L) generated tokens, rather than a single decoded response. This probability mass matters directly for adversarial evaluation as two models may similarly refuse to a harmful input prompt while having substantially different values of H_L. A model with larger harmful reachability exposes more probability mass that an adversarial search procedure may potentially discover and exploit through alternative prompts, prefixes, decoding trajectories, or iterative refinement.

3.2 Bounded continuation analysis

While theoretically well-defined, computing H_L exactly is intractable for modern LLMs even at modest horizons because the number of possible continuations grows exponentially with depth. At horizon (L), the full continuation tree contains up to $|\mathcal{V}|^L$ paths, where $|\mathcal{V}|$ is typically on the order of tens of thousands of tokens. We therefore approximate the full generation process with a bounded continuation tree that retains the highest-probability next-token transitions while discarding low-probability branches. This makes it possible to estimate harmful reachability over the portion of the model distribution explored by the bounded tree. We call this procedure *Bounded Continuation Analysis* (BCA).

BCA consists of two key stages: (i) A breadth-first expansion with batched GPU inference over all active prefixes at each depth and (ii) evaluation of the resulting bounded continuation tree. To control the size of the tree, BCA adopts $(\alpha) - (k)$ bounding strategy from LLMChecker (Gross et al., 2025). At each prefix, we retain only a high-probability subset of next-token transitions. Given a cumulative probability threshold $\alpha \in (0, 1]$ and a maximum branching factor k, we retain the smallest number of highest-probability tokens needed to reach probability mass α, subject to retaining at most k tokens. Let $y^{(1)}, y^{(2)}, \ldots$ denote the tokens sorted in decreasing order of $p_M(y \mid s)$. We first define the number of tokens required to cover probability mass α:

$$m(s) = \min \left\{ j_\alpha(s), k \right\}. \tag{4}$$

where $j_\alpha(s) = \min \left\{ j : \sum_{r=1}^{j} p_M\left(y^{(r)} \mid s\right) \geq \alpha \right\}$. Then, the retained token set is $\mathcal{T}_{\alpha,k}(s) = \left\{ y^{(1)}, \ldots, y^{(m(s))} \right\}$.

Applying this rule recursively produces a finite continuation tree of depth at most L, with branching factor at most k, and therefore at most k^L leaves. If the α-threshold is reached before the k-cap binds at every expanded state, then the retained outgoing transitions carry at least an α fraction of the probability mass at each step. Consequently, the total probability mass represented by the retained depth-L continuations is at least α^L. Because BCA does not renormalize retained transitions, we can equivalently view all discarded probability mass at each state as transitioning to an absorbing `unknown` state, making the bounded process a proper DTMC while preserving the original model probabilities on retained branches. If the k-cap binds before the cumulative mass reaches α at any state, this guarantee no longer holds, and the retained probability mass may be smaller.

BCA is agnostic to the evaluator used to define the property of interest. We denote by $J_\phi(x, y_{1:L})$ an evaluator that determines whether a continuation y, generated in response to x, satisfies a PCTL specification ϕ. The evaluator may take several forms: our implementation supports deterministic lexical or structural predicates, learned classifiers, and semantic LLM judges. However, we focus on semantic harm evaluation, where J_ϕ is applied only to the leaf continuations and their probability mass is aggregated directly. For specifications defined over intermediate prefixes, BCA instead evaluates J_ϕ throughout the tree and computes the corresponding probability using backward induction. The

procedure for backward induction is provided in Appendix B. When the bounded tree can be enumerated, we compute harmful continuation mass exactly by summing the probability of harmful terminal continuations.

3.3 GENERATIVE-ATTACK SUSCEPTIBILITY (GAS)

BCA characterizes the continuation distribution induced by a *fixed* prompt: given an input x, it measures how much probability mass within the bounded continuation tree is assigned to harmful responses. However, jailbreak attacks generally do not operate on a fixed prompt. An attacker can modify, rewrite, or otherwise transform the original prompt over a sequence of queries, potentially moving the target model into regions of its input space where harmful behavior is substantially easier to elicit. Consequently, susceptibility to attack depends not only on the continuation distribution at the original prompt, but also on how readily an attacker can discover prompts with larger harmful continuation mass.

We capture this interaction through *Generative-Attack Susceptibility* (GAS). Let $x^{(0)}$ denote the original behavior prompt, and let

$$\mathcal{X}_q = \{x^{(0)}, x^{(1)}, \ldots, x^{(q)}\}$$

denote the prompts considered by an attacker within a budget of q queries. We define

$$\mathrm{GAS}(q) = \max_{x \in \mathcal{X}_q} H_{\alpha,k,L}(x; M, J_\phi), \tag{5}$$

where $H_{\alpha,k,L}(x; M, J_\phi)$ denotes the bounded harmful continuation mass induced by prompt x. Thus, GAS@q measures the largest harmful continuation mass exposed by the attacker within its query budget. This definition separates two distinct components of adversarial vulnerability. The *outer* attack procedure determines which regions of prompt space are reached under a given search budget, while BCA characterizes the target model's continuation distribution at each prompt encountered during that search. A model may therefore be robust at the original prompt yet become highly susceptible after only a small number of adversarial transformations; conversely, another model may exhibit non-negligible harmful mass at the direct prompt while changing little under additional search. Rather than assigning each model a single susceptibility score, GAS represents this behavior as a function of attacker capability.

Different attack settings correspond to different constructions of $\mathcal{X}_q$. With no adversarial transformation, $\mathcal{X}_0 = \{x^{(0)}\}$ and GAS@0 reduces directly to BCA on the original behavior prompt. For our closed library of 18 jailbreak templates, GAS@18 is the maximum bounded harmful continuation mass over the transformed prompts in the library. For adaptive attackers such as PAIR (Chao et al., 2023) and TAP (Mehrotra et al., 2024), GAS@PAIR(q) and GAS@TAP(q) track the running maximum over prompts discovered as the attack proceeds.

GAS complements rather than replaces ASR. ASR records realized attack outcomes, while GAS probes the underlying continuation distributions encountered during search. Reporting both as functions of query budget allows us to distinguish models that appear similar under sampled outcomes but differ substantially in the harmful behavior that remains accessible beneath those outcomes. Additionally, it provides a distributional signal for prioritizing prompts or transformations during adaptive and template-based attacks, directing additional attack queries toward regions with greater harmful continuation mass. Finally, we find that distribution-guided and outcome-guided search can expose complementary failure modes. Combining BCA with outcome-based attack objectives can broaden the set of failures surfaced during red-teaming and subsequently incorporated into adversarial training.

4 EVALUATION

4.1 IMPLEMENTATIONAL DETAILS AND EXPERIMENTAL SETUP

We evaluate BCA and measure GAS on the 100 harmful behaviors from JailbreakBench (Chao et al., 2024). We choose JailbreakBench because it provides a standardized and widely used set of adversarial behaviors spanning diverse harm categories including harassment and discrimination,

Table 1: **Identical observed refusals conceal large differences in harmful reachability, and empirical ASR saturates where GAS still differentiates.** All four models show 0% greedy ASR (unshaded) and GAS@0 still separates them by two orders of magnitude. Under 18 jailbreak templates, empirical ASR (fraction of behaviors where *any* template ever elicits a judged-harmful completion) saturates at 100% for three of four targets, erasing any distinction between them; GAS@18 (the maximum bounded harmful mass over the same 18 templates) still resolves a clear gap between them. Color intensity is normalized within each shaded column; bold denotes the column maximum.

Target model	No Attacker		Under 18-Template Attack	
	Greedy ASR	GAS@0	Empirical ASR@18	GAS@18
Qwen2.5-1.5B-Instruct	0%	0.004	**100.0%**	0.906
Qwen2.5-7B-Instruct	0%	0.004	**100.0%**	**0.907**
Mistral-7B-Instruct-v0.3	0%	**0.211**	**100.0%**	0.782
Llama-3.1-8B-Instruct	0%	0.002	76.7%	0.144

malware and cyber abuse, physical harm, fraud, disinformation, privacy violations. For our target models, we choose open weight models: Qwen2.5-1.5B-Instruct, Qwen2.5-7B-Instruct, Mistral-7B-Instruct-v0.3, Llama-3.1-8B-Instruct. These models span multiple instruction-tuned model families and sizes. These models also expose token-level probability distributions, which is essential for constructing the bounded continuation tree.

We first evaluate GAS using a fixed library of 18 jailbreak transformations spanning several common prompt-level attack strategies, including narrative and hypothetical framing, persona-based jailbreaks such as AIM, STAN, and DAN (Shen et al., 2024), compliant-prefix and suffix forcing, output-format injection, and permission-based framing. We then consider two adaptive black-box attack techniques, PAIR (Chao et al., 2023) and TAP (Mehrotra et al., 2024). PAIR performs sequential, response-conditioned prompt refinement, providing a simple adaptive attacker; TAP instead expands and prunes multiple candidate attacks at each iteration, representing a branching tree-search attacker. Together, the fixed template library, PAIR, and TAP allow us to study whether BCA can be used to explore prompt space. We run PAIR with a single refinement stream for up to 10 target-model queries and TAP with branching factor $b = 2$, beam width $w = 3$, and depth $d = 3$, corresponding to at most 12 target-model queries per behavior. Code and implementation details are available at [1].

At each of TAP's d search depths, the attacker expands each of the w surviving candidates into b children ($b{\cdot}w$ candidates per depth) and keeps only the top-w by a per-candidate score before continuing to the next depth. We call this keep/discard step *pruning* throughout this section. Standard TAP scores candidates with an external judge's harm/on-topic rating. We test replacing that pruning score with each candidate's BCA mass, while leaving the attacker and tree shape unchanged. This pruning score is a distinct object from the α-k bounding of §3.2. That bounding fixes how the *inner* continuation tree is built for any single candidate, and it stays identical across all three variants below. The pruning score instead decides which *outer* search candidates survive to the next depth. We report on two outcome measures for the resulting trees. *Any-leaf* ASR measures whether at least one final leaf in the tree is considered a jailbreak, regardless of which leaf is returned. The *deployment* ASR measures whether the single leaf that the search's own selection rule would actually output is a jailbreak. For example, a tree with one jailbroken leaf among 12 scores as a success under any-leaf ASR even if the selection rule returns one of the 11 safe leaves, a case deployment ASR would score as a failure.

4.2 Results

4.2.1 Do models exhibit different distributional fragility while achieving similar ASR?

To answer the question, we measure (1) greedy ASR and GAS@0 on the 100 original JailbreakBench behaviors (i.e., how much harmful behavior is already reachable under the original unsafe prompt), and (2) GAS@18 after applying a fixed library of 18 jailbreak transformations to the same behaviors (i.e., how much harmful behavior becomes reachable under adversarial reformulations of the original

[1] https://github.com/smc879572-pixel/BCA

Table 2: **Bounded harmful reachability predicts independently judged harmful generations.** Bounded reachability uses Qwen2.5-7B as the BCA leaf evaluator; empirical ASR is independently sampled and scored by HarmBench-Mistral-7B. Spearman's ρ and Pearson's r measure rank and linear agreement between the two; Top-1 lift is the empirical-ASR gain from picking each behavior's highest-mass template over a uniform one. Darker shading denotes larger values within each metric; bold denotes the maximum

Target	n pairs	Spearman ρ	Pearson r	Top-1 lift
Qwen2.5-1.5B	540	0.716	0.74	4.93×
Qwen2.5-7B	540	**0.733**	**0.83**	**7.43**×
Mistral-7B	540	0.611	0.59	2.56×
Llama-3.1-8B	540	0.534	0.79	6.73×

prompt). Table 1 summarizes the results. We find that greedy ASR makes all four models appear equally robust; GAS sharply separates them. On direct prompts, all four models achieve greedy 0% ASR but their continuation distributions differ. Concretely, GAS@0 is 0.211 for Mistral-7B-Instruct-v0.3, compared to 0.004 for both Qwen2.5-1.5B-Instruct and Qwen2.5-7B-Instruct, and 0.002 for Llama-3.1-8B-Instruct. When the attacker is kept fixed at $q = 0$, Mistral exhibits more bounded harmful mass than the other models. This is a distinction that greedy ASR misses: a successful refusal does not imply that harmful continuations have been suppressed from the surrounding generation distribution. Additionally, under the 18-template attack, empirical ASR saturates at 100% for three of the four models: Qwen2.5-1.5B, Qwen2.5-7B, and Mistral-7B-Instruct-v0.3. As a result, ASR makes these models appear equally susceptible. GAS@18, however, still distinguishes them, assigning Mistral a lower susceptibility score (0.782) than the two Qwen models (0.906 and 0.907).

4.2.2 Does bounded harm reachability predict realized harmful behavior?

For each $(\text{behavior}, \text{template})$ pair, we compute bounded harm reachability $\mathbb{P}[\mathbf{F}(\texttt{harm})]$ using Qwen2.5-7B-Instruct as the BCA leaf evaluator. Note that we use a subset of 30 behaviors here due to the computational cost of empirical validation. We then independently sample 100 completions from the same prompt and use HarmBench-Mistral-7B classifier to estimate ASR. This results in 540 $(\text{behavior}, \text{template})$ pairs per model. Table 2 shows the results. We first measure the correspondence between bounded harm reachability and empirical ASR using Spearman's rank correlation. We use Spearman's ρ to ask whether pairs that BCA ranks as having greater harmful reachability also have higher ASR. Across all models, this relationship is consistently positive: $\rho = 0.716$ for Qwen2.5-1.5B, 0.733 for Qwen2.5-7B, 0.611 for Mistral-7B, and 0.534 for Llama-3.1-8B. Thus, prompts assigned greater harmful probability mass by BCA tend to produce harmful completions more frequently when sampled independently. The relationship is strong enough to guide red-team allocation: choosing, for each behavior, the template ranked highest by BCA increases realized ASR by $2.56\times$–$7.43\times$ relative to uniform template selection.

Because each behavior is evaluated under multiple templates and each template is applied across multiple behaviors, we complement Spearman's correlation with a mixed-effects logistic regression that accounts for both sources of variation (Gelman et al., 2007). We found that some behaviors are more susceptible to harmful responses, and some templates are more successful at eliciting harmful completions. Nevertheless, bounded harm reachability remains positively associated with harmful generation after accounting for these differences.

4.3 Adaptive red-teaming with BCA

We also evaluate the utility of BCA as an objective for discovering attacks by adaptive red-teaming methods. Specifically, we integrate BCA into PAIR and TAP by replacing the judge-based candidate-pruning score with BCA. We evaluate all 7B-scale models defined in Section 4.1 and additionally include DeepSeek-R1-Distill-Llama-8B. Unless otherwise noted, we use the exact bounded continuation process as above ($\alpha = 0.99$, $k = 2$, $L = 8$) and the Qwen2.5-7B semantic judge. Then, we compare *judge-guided* TAP against BCA-*guided* TAP. We define two metrics for measuring attack success. *Any-leaf* ASR measures whether the search tree contains at least one successful jailbreak: a run counts as successful if *any* explored leaf receives a 10/10 score from the outcome

Table 3: **BCA is useful for search across model families, but outcome-aligned scoring remains necessary for selection.** **(a)** BCA-based pruning matches or exceeds judge-based pruning on most targets, but selecting with BCA alone collapses deploy ASR on every target (worst for DeepSeek-R1, −41.6 points); selecting with the outcome judge instead recovers full deploy performance. **(b)** Judge- and BCA-guided searches find partially disjoint jailbreaks; running both raises coverage on all four models, most on Qwen2.5-7B (self-attack), whose 48.3% neither-succeeds rate is the table's highest. Shading is scaled independently per panel: **(a)** the any-leaf-to-deploy collapse when BCA also selects the deployed candidate; **(b)** the share of paired runs where neither search succeeds.

(a) Search signal versus deployment-selection signal

	Llama-3.1-8B		DeepSeek-R1		Qwen2.5-7B†		Mistral-7B	
Search × Select	Any-leaf	Deploy	Any-leaf	Deploy	Any-leaf	Deploy	Any-leaf	Deploy
Judge × Judge	57.0%	57.0%	64.2%	64.2%	33.3%	33.3%	65.8%	**65.8%**
BCA × BCA	58.3%	24.1%	73.3%	31.7%	38.3%	17.5%	62.5%	36.7%
BCA × Judge	58.3%	**58.3%**	73.3%	**73.3%**	38.3%	**38.3%**	62.5%	62.5%

† self-attack.

(b) Complementary deployable jailbreak coverage

Outcome	Llama (N=120)	DeepSeek (N=120)	Qwen† (N=120)	Mistral (N=120)
Both succeed	37.3%	55.8%	20.0%	54.2%
Judge-guided only	19.6%	8.3%	13.3%	11.7%
BCA-guided only	21.6%	17.5%	18.3%	8.3%
Neither succeeds	21.6%	18.3%	48.3%	25.8%
Union: either succeeds	**78.4%**	**81.7%**	**51.7%**	**74.2%**

judge, irrespective of which leaf the search procedure ultimately selects. In contrast, *deploy* ASR evaluates only the single leaf returned by the attack and measures whether the candidate actually selected for deployment is a successful jailbreak. For the sake of brevity, the results for integrating BCA with PAIR are provided in Appendix D.

BCA-guided and judge-guided search expose complementary failures. Across 120 runs each for four target models, BCA-guided TAP discovers successful jailbreaks at rates comparable to judge-guided TAP, and exceeds it on three of the four targets in any-leaf ASR (Table 3a). More importantly, the two search signals uncover partially disjoint sets of failures (Table 3b). Across all four targets, some jailbreaks are found only by judge-guided search and others only by BCA-guided search. Taking the union raises deployable jailbreak coverage to 78.4% on Llama-3.1-8B, 81.7% on DeepSeek-R1-Distill-Llama-8B, 51.7% on Qwen2.5-7B, and 74.2% on Mistral-7B. This complementarity suggests that BCA captures information about the target model's continuation distribution that is not redundant with an outcome-based judge, making the two signals useful in combination when the goal is to broaden failure coverage rather than optimize a single attack trajectory.

5 Related Work

Early hand-written, one-shot jailbreaks were standardized by HarmBench (Mazeika et al., 2024) and JailbreakBench (Chao et al., 2024) into fixed behavior sets with automated pass/fail judges. Attacks evolved into adaptive search using the target's own responses as feedback—PAIR (Chao et al., 2023) and TAP (Mehrotra et al., 2024), part of the broader automated red-teaming paradigm (Perez et al., 2022), including genetic and fuzzing-based search over wider candidate sets (Liu et al., 2024; Yu et al., 2024)—but evaluation remained a binary label, aggregated across a query budget. While later work refines what this label should reward (Souly et al., 2024b).[2] These methods differ in how they score a sampled response, using binary labels, severity ratings, or weighted sub-question decompositions, but the object being evaluated remains a single observed completion. We show this

[2]Appendix A surveys additional work refining single-response jailbreak scoring.

framing cannot separate two models with identical greedy outputs, but whose nearby harmful mass differs by two orders of magnitude (§4).

A second limitation of prior work is that ASR numbers are also not comparable across papers: Chouldechova et al. (2026) show ASR shifts once threat models, judges, or budgets differ. Closest to our work is Maple et al. (2026) who report a distribution over an attack's *hyperparameter configurations*—in contrast, we report one over the *target model's own stochastic continuations* under one fixed prompt, well defined under explicit bounding and evaluator assumptions.

Beyond measurement, Wei et al. (2023a) attribute this gap to safety training failing via two mechanisms: competing objectives between helpfulness and harmlessness, and mismatched generalization between training and capability. Qi et al. (2025) sharpen this by localizing the vulnerability to the first few output tokens: safety training reshapes that prefix exclusively, so a model can refuse on the surface while retaining high harmful mass just past it. Prefilling (Li et al., 2025) and logprob-targeted attacks (Andriushchenko et al., 2025) exploit this directly, manipulating initial token probabilities to bypass search entirely. BCA operationalizes this phenomenon as a diagnostic, not an attack—our refusal-collapse finding turns a qualitative account of shallow alignment into a measured probability.

Closest to our approach is LLMChecker (Gross et al., 2025), which introduced α-k-bounded PCTL model checking for LLM generation. We build directly on this formulation but extend it along three dimensions. First, we replace recursive per-node tree expansion and external Storm-based model checking with depth-wise GPU-batched expansion and in-process sparse backward induction, reducing the number of sequential model-inference rounds from $O(k^L)$ to $O(L)$ while retaining the same worst-case $O(k^L)$ bounded tree size. Second, we extend the checked properties from primarily deterministic lexical and structural attributes to behavior-conditioned semantic harm evaluated by pretrained models. Third, we compose bounded reachability with adaptive red teaming: GAS tracks harmful reachability over an attacker's query budget, and BCAserves as a search signal within PAIR and TAP.

6 Conclusion

Current jailbreak evaluations largely reduce model safety to an observed outcome: whether an attack produces a harmful response. In this work, we argue that this view is incomplete. Autoregressive models define distributions over possible continuations, and a safe decoded response can coexist with substantial harmful probability mass elsewhere in the model's continuation distribution. We introduced BCA, a bounded measure of harmful reachability within this continuation distribution, and GAS, which extends this perspective to adaptive attacks by tracking how harmful reachability evolves with attacker budget. Across models and JailbreakBench behaviors, these quantities reveal safety differences that are invisible to greedy ASR, correlate with independently measured attack success, and expose failures that are complementary to those found by outcome-guided search. In particular, distribution-guided and outcome-guided attacks often uncover different jailbreaks, making the continuation distribution useful not only for diagnosis but also for broadening red-team coverage. By making latent harmful reachability measurable and operational, BCA provides a step toward safety evaluations that reason about the generation process itself rather than only its sampled outcomes.

AI use statement

In this work, we **have used** generative AI tools to design and provide feedback on experiments and implement methods. We have **not used** generative AI tools to help develop theoretical models or conceptual frameworks, help in the writing of proofs, propose or refine hypotheses, assist with translation, support qualitative and thematic data analysis or the interpretation of results. The generation of synthetic data sets, the formulation of mathematical claims, the provision of critical ingredients for proving mathematical claims, and the cleaning and reformatting of the dataset are **not applicable** to this work.

Additionally, we used generative AI tools to create Figure 1, suggest experimental parameters, create and edit software code, create artifacts, draft parts of the research paper, analyze existing literature, source/search for information, and edit the paper to improve readability.

We have reviewed all AI-assisted work: figures and artifacts were checked against the experimental data before inclusion, AI-suggested experimental parameters were reviewed against pilot results, AI-assisted code was tested and verified for correctness by two of the authors, and AI-drafted or AI-edited text was reviewed by all authors for factual accuracy. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Ethics Statement

This work studies adversarial prompts and harmful behaviors to improve safety evaluation. We report template *families* rather than operational jailbreak strings, follow public benchmark protocols, and emphasize defensive use: triage, auditing, and adversarial training coverage. Dual-use risk remains; we recommend gated release of attack traces consistent with JailbreakBench norms.

Reproducibility Statement

All targets and judges used in this work are open-weight models (Qwen2.5-1.5B/7B-Instruct, Mistral-7B-Instruct-v0.2/v0.3, Llama-3.1-8B-Instruct, HarmBench-Mistral-7B, Llama Guard 3), and all behaviors are drawn from the public JailbreakBench suite, so every experiment in this paper can be rerun without access to closed APIs. Algorithm 1 gives the exact bounded-reachability computation, and §4 reports the α, k and L used in each experiment, along with attacker budgets for the closed-template library, PAIR, and TAP. Consistent with JailbreakBench release norms and our ethics statement (§6), we release our code for constructing and evaluating bounded continuation trees, along with behavior categories and aggregate results, at an anonymized repository: https://github.com/smc879572-pixel/BCA.git. A de-anonymized version will be made available upon acceptance. We withhold full jailbreak prompts, template text, and attack traces to limit dual-use risk. We hope this transparency will allow the community to verify our findings and build upon our work, for instance by auditing additional models' hidden harmful mass or extending BCA to new judges and attackers. Random seeds are reported wherever they affect results (seeds 42–45 for the multi-seed TAP experiments).

Two considerations limit the absolute reproducibility of our experiments. First, all reported reachability probabilities are conditional on the specific judge checkpoints used (Qwen2.5-7B-Instruct, HarmBench-Mistral-7B, Llama Guard 3). Reproducing our exact numbers requires picking the same model versions, and results may shift if providers update or deprecate these checkpoints. Second, our experiments involve probabilistic target models which introduce variance into our results.

References

Maksym Andriushchenko, francesco croce, and Nicolas Flammarion. Jailbreaking leading safety-aligned llms with simple adaptive attacks. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 40116–40143, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/63fa7efdd3bcf944a4bd6e0ff6a50041-Paper-Conference.pdf.

Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking (Representation and Mind Series)*. The MIT Press, 2008. ISBN 026202649X.

Hongyu Cai, Arjun Arunasalam, Leo Y. Lin, Antonio Bianchi, and Z. Berkay Celik. Rethinking how to evaluate language model jailbreak. In *Proceedings of the 18th ACM Workshop on Artificial Intelligence and Security*, AISec '25, pp. 52–63, New York, NY, USA, 2026. Association for Computing Machinery. ISBN 9798400718953. doi: 10.1145/3733799.3762966. URL https://doi.org/10.1145/3733799.3762966.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries, 2023.

Patrick Chao, Edoardo Debenedetti, Alexander Robey, Maksym Andriushchenko, Francesco Croce, Vikash Sehwag, Edgar Dobriban, Nicolas Flammarion, George J. Pappas, Florian Tramèr, Hamed Hassani, and Eric Wong. Jailbreakbench: An open robustness benchmark

for jailbreaking large language models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 55005–55029. Curran Associates, Inc., 2024. doi: 10.52202/079017-1745. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/63092d79154adebd7305dfd498cbff70-Paper-Datasets_and_Benchmarks_Track.pdf.

Alexandra Chouldechova, A. Feder Cooper, Solon Barocas, Abhinav Palia, Dan Vann, and Hanna Wallach. Comparison requires valid measurement: Rethinking attack success rate comparisons in ai red teaming, 2026. URL https://arxiv.org/abs/2601.18076.

Andrew Gelman, Jennifer Hill, et al. *Data analysis using regression and multilevel/hierarchical models*, volume 1. Cambridge university press Cambridge, 2007.

Dennis Gross, Helge Spieker, and Arnaud Gotlieb. Bounded pctl model checking of large language model outputs, 2025. URL https://arxiv.org/abs/2509.18836.

Hans Hansson and Bengt Jonsson. A logic for reasoning about time and reliability. *Form. Asp. Comput.*, 6(5):512–535, September 1994. ISSN 0934-5043. doi: 10.1007/BF01211866. URL https://doi.org/10.1007/BF01211866.

Deokjae Lee, JunYeong Lee, Jung-Woo Ha, Jin-Hwa Kim, Sang-Woo Lee, Hwaran Lee, and Hyun Oh Song. Query-efficient black-box red teaming via Bayesian optimization. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11551–11574, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.646. URL https://aclanthology.org/2023.acl-long.646/.

Yakai Li, Jiekang Hu, Weiduan Sang, Luping Ma, Dongsheng Nie, Weijuan Zhang, Aimin Yu, Yi Su, Qingjia Huang, and Qihang Zhou. Prefill-level jailbreak: A black-box risk analysis of large language models, 2025. URL https://arxiv.org/abs/2504.21038.

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. AutoDAN: Generating stealthy jailbreak prompts on aligned large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=7Jwpw4qKkb.

Carsten Maple, Abhishek Kumar, and Riya Tapwal. Single-configuration attack success rate is not enough: Jailbreak evaluations should report distributional attack success, 2026. URL https://arxiv.org/abs/2605.09070.

Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, David Forsyth, and Dan Hendrycks. Harmbench: a standardized evaluation framework for automated red teaming and robust refusal. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum Anderson, Yaron Singer, and Amin Karbasi. Tree of attacks: jailbreaking black-box llms automatically. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024. Curran Associates Inc. ISBN 9798331314385.

J. R. Norris. *Markov Chains*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1997.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. Red teaming language models with language models, 2022. URL https://arxiv.org/abs/2202.03286.

Xiangyu Qi, Ashwinee Panda, Kaifeng Lyu, Xiao Ma, Subhrajit Roy, Ahmad Beirami, Prateek Mittal, and Peter Henderson. Safety alignment should be made more than just a few tokens deep. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 54911–54941, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/88be023075a5a3ff3dc3b5d26623fa22-Paper-Conference.pdf.

Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. "do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models, 2024. URL https://arxiv.org/abs/2308.03825.

Dong Shu, Chong Zhang, Mingyu Jin, Zihao Zhou, and Lingyao Li. Attackeval: How to evaluate the effectiveness of jailbreak attacking on large language models. *SIGKDD Explor. Newsl.*, 27(1):10–19, July 2025. ISSN 1931-0145. doi: 10.1145/3748239.3748242. URL https://doi.org/10.1145/3748239.3748242.

Alexandra Souly, Qingyuan Lu, Dillon Bowen, Tu Trinh, Elvis Hsieh, Sana Pandey, Pieter Abbeel, Justin Svegliato, Scott Emmons, Olivia Watkins, and Sam Toyer. A strongreject for empty jailbreaks. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 125416–125440. Curran Associates, Inc., 2024a. doi: 10.52202/079017-3984. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/e2e06adf560b0706d3b1ddfca9f29756-Paper-Datasets_and_Benchmarks_Track.pdf.

Alexandra Souly, Qingyuan Lu, Dillon Bowen, Tu Trinh, Elvis Hsieh, Sana Pandey, Pieter Abbeel, Justin Svegliato, Scott Emmons, Olivia Watkins, and Sam Toyer. A strongreject for empty jailbreaks. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2024b. Curran Associates Inc. ISBN 9798331314385.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: how does llm safety training fail? In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023a. Curran Associates Inc.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does llm safety training fail? *Advances in neural information processing systems*, 36:80079–80110, 2023b.

Jiahao Yu, Xingwei Lin, Zheng Yu, and Xinyu Xing. Llm-fuzzer: scaling assessment of large language model jailbreaks. In *Proceedings of the 33rd USENIX Conference on Security Symposium*, SEC '24, USA, 2024. USENIX Association. ISBN 978-1-939133-44-1.

Yukai Zhou, Feiyang Lu, Xiaokai Mao, Jinfei Liu, and Wenjie Wang. How jailbreak attacks inform safety alignment: A defender-centric, shapley-based evaluation of jailbreak contributions, 2026. URL https://arxiv.org/abs/2607.17152.

A Related Work on Single-response Jailbreak Scoring

Section 5 reviews the most directly related prior work, focused on jailbreak evaluation and ASR. We now cover other related prior work that has refined single-response jailbreak scoring.

Beyond StrongREJECT (Souly et al., 2024b), several works refine what counts as success on a single sampled response, without moving to a distributional view. AttackEval (Shu et al., 2025) proposes a coarse- and fine-grained scoring framework in the same view as sub-question decomposition methods. Cai et al. (2026) argue that binary success/fail conflates three distinct axes: whether a safeguard was violated, whether the response was substantive enough to be useful, and whether harmful content was accurate rather than merely harmful-sounding. They propose individual scoring along each axis. Zhou et al. (2026) show that ASR-based rankings are only weakly aligned with a defender's utility from an attack, a distinct critique of ASR: ASR misleads *prioritization* decisions, not just severity judgments.

B HARMFUL REACHABILITY VIA BACKWARD INDUCTION

After constructing the bounded continuation tree to depth L, an evaluator J assigns each leaf state s a binary harm label,

$$J(s) \in \{0, 1\}, \tag{6}$$

where $J(s) = 1$ denotes a harmful continuation. We define a value function $V : S \to [0, 1]$ over the bounded tree. For each leaf state,

$$V(s) = J(s). \tag{7}$$

For each internal state, we compute its value by backward induction:

$$V(s) = \sum_{y \in \mathcal{T}_{\alpha,k}(s)} \widetilde{P}(s, s \circ y)\, V(s \circ y), \tag{8}$$

where $\widetilde{P}$ denotes the transition probabilities of the bounded DTMC. Evaluating Eq. 8 from depth L back to the root gives

$$H_{\alpha,k,L}(x; M, J) = V(s_0), \qquad s_0 = x, \tag{9}$$

the exact probability of reaching a harmful leaf under the bounded DTMC.

Because the bounded continuation structure is a tree, these values can be computed in a single reverse traversal. Once the tree has been constructed and labeled, backward induction therefore requires $O(|S|)$ time, where $|S|$ is the number of states in the bounded tree.

Algorithm 1 Exact BCA on an α-k bounded continuation tree

Require: prompt x, model M, evaluator J, horizon L, bounds α, k

1: Construct the α-k bounded tree rooted at $s_0 = x$ to depth at most L
2: Label each leaf s with $\ell(s) \leftarrow J(s)$
3: $V(s) \leftarrow \mathbf{1}[\ell(s) = \texttt{harm}]$ for every leaf s
4: **for** $d = L - 1$ **downto** 0 **do**
5: **for all** nonterminal states s at depth d **do**
6: $V(s) \leftarrow \sum_{t \in T_{\alpha,k}(s)} \widetilde{P}(s, s{\cdot}t)\, V(s{\cdot}t)$
7: **end for**
8: **end for**
9: **return** $V(s_0)$

C IMPLEMENTATION

Our GPU implementation uses breadth-first batched inference (vLLM/HuggingFace backends) and sparse backward induction for exact trees. Code and interactive complementarity visualizations are available at an anonymized repository: https://github.com/smc879572-pixel/BCA.git.

D BCA-GUIDED SEARCH WITH PAIR

We provide additional results evaluating BCA as a search signal within PAIR. The same qualitative findings observed with TAP persist under sequential prompt refinement. BCA-guided PAIR discovers successful jailbreaks at rates comparable to judge-guided PAIR on three of the four target models, while underperforming on Mistral-7B. The two search signals also expose partially disjoint failures on every target. Taking their union increases deployable jailbreak coverage to 66.0% on Llama-3.1-8B, 67.1% on DeepSeek-R1-Distill-Llama-8B, 31.6% on Qwen2.5-7B, and 65.0% on Mistral-7B, indicating that the complementarity between distribution-guided and outcome-guided search is not specific to TAP.

Table 4: **The search-vs-selection and complementary-coverage findings replicate under PAIR, not just TAP. (a)** BCA-guided search discovers jailbreaks at a rate comparable to judge-guided search on three of four targets; Mistral-7B is the one target where it underperforms outright, consistent with the same direction seen under TAP. Selecting with BCA alone discards most successful candidates on every target; selecting with the outcome judge instead recovers deploy performance. Shading marks the any-iteration-to-deploy drop, scaled across all four models; **bold** marks the recovered value. **(b)** Judge- and BCA-guided PAIR chains find partially disjoint failures on every target; running both raises coverage over either alone, even on Qwen2.5-7B, the most PAIR-resistant of the four. Shading marks the Neither-succeeds rate, scaled across all four models.

(a) Search signal versus deployment-selection signal (PAIR, q=10)

	Llama-3.1-8B†		DeepSeek-R1‡		Qwen2.5-7B†		Mistral-7B‡	
Search × Select	Any-iter.	Deploy	Any-iter.	Deploy	Any-iter.	Deploy	Any-iter.	Deploy
Judge × Judge	56.9%	56.9%	48.1%	48.1%	20.3%	20.3%	53.8%	53.8%
BCA × BCA	51.7%	30.0%	46.2%	27.5%	21.2%	13.8%	43.8%	27.5%
BCA × Judge	51.7%	**51.7%**	46.2%	**46.2%**	21.2%	**21.2%**	43.8%	**43.8%**

† self-attack. ‡ external Qwen2.5-7B attacker.

(b) Complementary deployable jailbreak coverage

Outcome	Llama	DeepSeek	Qwen	Mistral
Both searches succeed	38.0%	26.6%	10.1%	32.5%
Judge-guided only	16.0%	21.5%	10.1%	21.2%
BCA-guided only	12.0%	19.0%	11.4%	11.2%
Neither succeeds	34.0%	32.9%	68.4%	35.0%
Union: either succeeds	**66.0%**	**67.1%**	**31.6%**	**65.0%**