GATE-OVERRIDE: COST AMPLIFICATION ATTACKS ON ADAPTIVE REASONING MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Adaptive reasoning models save computation by choosing when to answer directly and when to reason at length. This choice also gives an attacker a way to increase computation on questions the model already answers correctly. We present GATE-OVERRIDE, an audit that tests this risk by linking each request's executed reasoning mode, answer correctness, and generation cost. Under white-box access, we optimize a shared 10-token suffix and insert its token IDs before the assistant-generation marker. Across five model–task settings spanning 1.5B–15B parameters, the resulting attacks change reasoning modes and increase the cost of correct answers. On AdaptThink-1.5B/GSM8K, the suffix switches 75 of 84 direct-answer requests to reasoning. On Phi-4-RV-15B/MMLU-Pro, it switches 93 of 100, versus zero for either random suffix; 48 of 52 complete, originally correct pairs remain correct while switching routes and costing at least 1.5 times as much. Our paired metrics reveal cost differences hidden by equal route-switch rates and accuracy. Forcing the same input through each route attributes most extra computation to the route change. Re-optimizing against a hardened route head restores substantial attack success, and matched serving experiments show added latency for clean requests. These findings make reasoning-mode selection a concrete target for resource-security evaluation.

1 INTRODUCTION

An adaptive reasoning model can answer a simple question directly and spend more time on a difficult one. This ability promises to make reasoning models cheaper and faster without giving up accuracy (Fang et al., 2025; Zhang et al., 2025; Tu et al., 2025; Aneja et al., 2026). It also creates a security question: can an attacker make the model choose expensive reasoning when its direct answer is already correct? In a shared service, the extra work may consume capacity needed by other users.

The decision about how much to think is part of the model's *compute control plane*: the mechanisms that select a reasoning mode and allocate computation. Because these mechanisms read the prompt, untrusted content can affect both the answer and the resources used to produce it. Existing work increases reasoning overhead while retaining correctness (Kumar et al., 2025), sends requests to more expensive models through external routers (Shafran et al., 2025; Tang et al., 2026), and changes reasoning modes through token-level triggers (Yang et al., 2026). We study the connection between internal route changes and the cost of answering the *same question correctly*.

Measuring this connection requires more than counting route changes. A model may prefer a reasoning token without actually following that route. A route change may add little cost, or it may make a previously correct answer wrong. Even the combination of route-switch rate and overall accuracy leaves a gap: it does not identify which originally correct requests became more expensive. Our experiments make this gap concrete. Two attacks can switch every eligible route and both achieve 93% accuracy, yet produce sharply different extra costs on requests that remain correct.

We introduce GATE-OVERRIDE to connect these observations at the request level (Figure 1). We optimize one 10-token suffix for each model–task–template setting under white-box access, insert its exact token IDs before a fixed assistant-generation marker, and freeze it before testing on held-out prompts. The audit then records the route the model actually executes, checks the answer, and compares generation cost with the clean run. Two paired metrics measure how often correct answers become more expensive and how much additional cost they incur. Finally, we force short and

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.

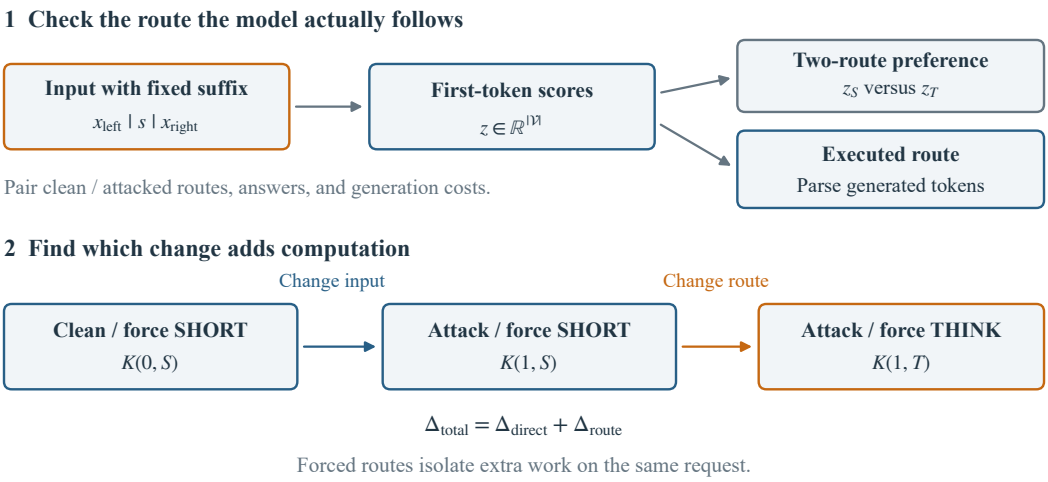


Figure 1: How GATE-OVERRIDE follows an attack from input to cost. Top: compare candidate scores, decode from the full vocabulary, and parse the generated trajectory to determine the executed route. Bottom: keep the short route fixed while changing the input, then keep the attacked input fixed while changing the route. These comparisons separate input-related and route-related costs.

reasoning routes on the same input to separate the cost of the input change from the cost of the route change.

The results show that correct answers can conceal substantial resource abuse. On Phi-4-RV-15B across four MMLU-Pro domains, the optimized suffix switches 93/100 routes, whereas both random suffixes switch 0/100. Among 52 complete pairs that were correct before attack, 48 switch routes, stay correct, and use at least 1.5 times the generated tokens. Forced-route comparisons locate most excess computation in the reasoning-route change. The effect also survives attacker adaptation to a trained route head and reaches clean users in a controlled shared service.

Contributions. **(1) A concrete compute-control attack.** Shared 10-token suffixes induce executed route changes and increase correct-answer cost in five independently optimized settings spanning 1.5B–15B models. **(2) An audit that connects control, correctness, and cost.** Paired metrics expose cost differences hidden by route-switch rate and accuracy; forced routes identify where the extra work arises. **(3) Evidence under adaptation and resource sharing.** Against the same hardened GSM8K head, re-optimization raises the flip rate from 42.5% to 76.3%. Matched serving workloads measure the additional latency imposed on clean requests.

2 RELATED WORK

Attacks on inference cost. OverThink increases reasoning overhead while retaining correctness (Kumar et al., 2025). Other attacks extend reasoning or generation (Si et al., 2025; Zhu et al., 2026; Liu et al., 2026a;b; Li et al., 2026), while external-router attacks redirect requests across models or services (Shafran et al., 2025; Tang et al., 2026). Our contribution is to connect a route change *within one model* to the same request's correctness and cost, and to test the source of that cost through forced routes.

Adaptive computation and reasoning. Conditional computation, adaptive computation time, sparse routing, and early exit allocate work according to the input (Bengio et al., 2015; Graves, 2016; Shazeer et al., 2017; Teerapittayanon et al., 2016). Adaptive reasoning learns when to use a longer reasoning process or a larger token budget (Fang et al., 2025; Zhang et al., 2025; Tu et al., 2025; Lou et al., 2025; Han et al., 2025; Lyu et al., 2025; Lin et al., 2025; Yang et al., 2026). We examine this allocation decision under adversarial input, measuring whether the extra computation improves benchmark correctness.

Suffix search and adaptive evaluation. Universal suffixes and transferable triggers provide methods for optimizing discrete inputs (Zou et al., 2023; Liu et al., 2024; Wang et al., 2024a; Xu & Wang, 2024; Huang et al., 2025; Rajeev et al., 2025). Prompt-injection benchmarks consider additional input access settings (Zhan et al., 2024; Ramakrishna et al., 2024). We use suffix search as a probe of compute control, compare objectives at matched candidate-forward counts, and re-optimize after training a route head, following adaptive attack evaluation principles (Athalye et al., 2018; Tramèr et al., 2020).

Resource controls and serving. Early exit, input pruning, compact reasoning, and decoding safeguards act at different stages of inference (Yang et al., 2025; Qiao et al., 2025; Aytes et al., 2025; Xuan et al., 2025; Xiong et al., 2025). Batching and memory management determine how requests share serving resources (Yu et al., 2022; Kwon et al., 2023; Zhong et al., 2024). We evaluate both a trained route head and the effect of attack traffic on clean-request latency and service throughput.

3 What the Audit Measures

3.1 Attacker access and the executed route

The attacker has offline white-box access and controls a suffix of $L = 10$ token IDs from a specified allowed vocabulary. Model weights, decoding settings, and the chat template stay fixed. For a tokenized prompt $x = (x_{\text{left}}, x_{\text{right}})$, where x_{right} contains the fixed assistant-generation marker and reasoning prefix, the attacked input is $(x_{\text{left}}, s, x_{\text{right}})$. We insert IDs directly, without decoding and retokenizing. One suffix is shared across test prompts within a model–task–template setting; each setting has its own optimization and validation split.

We separate three observations. *Candidate preference* compares the scores of two route tokens. *First-token class* checks which token wins ordinary decoding over the full vocabulary. *Executed route* parses the generated sequence as SHORT, THINK, or OTHER. The last observation determines attack success: beating one route token need not mean winning over the full vocabulary. AdaptThink uses `</think>` and `Alright`; Thinkless uses `<short>` and `<think>`; Phi-4 uses `<nothink>` and `<think>`. The parser checks the first-token marker and then the model's native opening and closing markers.

A forward flip occurs when the clean run executes SHORT and the attacked run executes THINK. Unmatched trajectories remain OTHER. We determine eligibility from the clean run and pair routes, answers, and costs by request ID. Thus every reported flip represents an observed execution change on a known clean input.

3.2 The frequency and size of extra cost on correct answers

Let N be the number of test prompts, E the set of clean SHORT requests, $F \subseteq E$ the set that flips to THINK, and $C \subseteq E$ the requests answered correctly before attack. The executed flip rate is $\text{Exec-FlipASR} = |F|/|E|$. To measure cost, we use C_{comp}: requests in C with complete clean and attacked generations, available costs, and positive clean cost. We call this the *cost cohort*. Table set symbols abbreviate their cardinalities.

For request i, let y_i be the benchmark answer, $\hat{y}_i^a$ the attacked answer, and K_i^c, K_i^a the clean and attacked costs. Define $G = \{i \in C_{\text{comp}} \cap F : \hat{y}_i^a = y_i\}$, the complete pairs that flip and remain correct. We measure two complementary quantities:

$$\text{UP-CHR@}\gamma = \frac{1}{|C_{\text{comp}}|} \sum_{i \in G} \mathbf{1}[K_i^a / K_i^c \geq \gamma], \tag{1}$$

$$\text{UECI}_K = \frac{1}{|C_{\text{comp}}|} \sum_{i \in G} [K_i^a / K_i^c - 1]_+. \tag{2}$$

Here $[u]_+ = \max(0, u)$. The utility-preserving cost hit rate, UP-CHR@γ, asks: *what fraction of originally correct, complete requests switches routes, stays correct, and costs at least γ times as much?* Utility-preserving expected cost inflation, UECI, measures the positive relative excess from these correct-answer flips, averaged over the entire cost cohort. Non-flips, attacked errors, and requests with no cost increase contribute zero. In both metrics, utility preservation means benchmark correctness.

For a simple illustration, consider ten complete, originally correct requests. If five switch routes, stay correct, and double their cost while the rest contribute zero, UP-CHR@1.5 is 50% and UECI is 0.5. The latter means 50% average positive relative excess across the ten requests. It is an excess-cost measure, not a cost multiplier. Auxiliary CCA reports cost-ratio quantiles among all complete flips, without requiring correct answers.

The two paired metrics describe the frequency and size of the same effect. On a fixed nonempty cohort, $\mathrm{UECI}_K = \int_1^\infty \text{UP-CHR@}\gamma \, d\gamma$; Appendix B gives the proof and threshold bound. Default cost K_{tok} counts generated tokens. Latency is reported separately. Requests that reach the generation cap remain in all-prompt accuracy and cap rates but are excluded from complete-pair cost metrics. Each condition reports its own cost cohort; paired differences between conditions are recomputed on the intersection of their cohorts. These cost estimates describe requests that finish within the stated cap.

4 GATE-OVERRIDE: SEARCH AND CONTROLLED COMPARISONS

4.1 FINDING ONE SUFFIX THAT WORKS ACROSS PROMPTS

We search for a suffix that makes the model choose the reasoning route. Let $\mathcal{D}_{\mathrm{opt}}$ contain the optimization prompts, z_S, z_T be the two route-candidate logits, t_T the target reasoning token, and $\mathcal{V}$ the vocabulary. Candidate margin compares the two route candidates; full-vocabulary route margin compares the target with every other token:

$$\mathcal{L}_{\mathrm{pair}}(s) = \frac{1}{|\mathcal{D}_{\mathrm{opt}}|} \sum_i \mathrm{softplus}(z_S - z_T), \tag{3}$$

$$\mathcal{L}_{\mathrm{route}}(s) = \frac{1}{|\mathcal{D}_{\mathrm{opt}}|} \sum_i \left[\tau \log \sum_{v \neq t_T} e^{z_v/\tau} - z_{t_T} \right]. \tag{4}$$

Logits are evaluated at the attacked prompt's routing position; we set $\tau = 1$. Minimizing the route objective pushes the reasoning token above competing vocabulary tokens. For one input, a negative route-margin term guarantees that this token has the largest logit (Boyd & Vandenberghe, 2004). Generated trajectories establish whether it produces the intended route. This is a search surrogate; the audit's outcome remains the executed route and paired answer cost.

At each complete suffix sweep, gradients shortlist three allowed replacements for each of the ten positions. We evaluate the shortlisted discrete candidates directly and update the suffix. A sweep uses one batched backward call and 30 batched candidate forwards. The objective study uses four optimization prompts, eight validation prompts, five initializations, and $B \in \{5, 20, 100\}$ sweeps. Checkpoints are retained by validation objective; the final suffix is chosen by validation executed flips, with objective as a tie-breaker. Test prompts do not enter selection.

The controls isolate what the search is optimizing. A hybrid combines the two margins; think cross-entropy targets the reasoning token; delayed EOS and delayed reasoning close discourage early stopping; direct length uses a teacher-forced length objective. Random-coordinate search evaluates randomly proposed replacements with the same candidate-forward count. It is distinct from inserting an unoptimized random suffix. At B sweeps, each gradient method uses B backward calls and $30B$ candidate forwards; random coordinate has no backward calls. Validation calls are additional, so matched candidate counts do not mean equal total compute.

4.2 SEPARATING THE COST OF THE INPUT FROM THE COST OF THE ROUTE

A longer answer after attack could come from the added input, the selected route, or both. We distinguish these sources by running the same request with explicit short and reasoning prefixes. Let $K_i(a, r)$ be the cost under input $a \in \{0, 1\}$ (clean or attacked) and forced route $r \in \{S, T\}$ (SHORT or THINK). Starting from the clean short route, we change one factor at a time:

$$\begin{aligned} \Delta_i^{\mathrm{total}} &= K_i(1, T) - K_i(0, S), \\ \Delta_i^{\mathrm{direct}} &= K_i(1, S) - K_i(0, S), \\ \Delta_i^{\mathrm{route}} &= K_i(1, T) - K_i(1, S). \end{aligned} \tag{5}$$

The direct term measures what the suffix changes while the short route stays fixed. The route term measures the extra work from choosing reasoning on the same attacked input. We prepend the selected route token and continue greedily; that token counts toward generated length and the cap. These prefix-controlled comparisons (Pearl, 2001) satisfy $\Delta_i^{\text{total}} = \Delta_i^{\text{direct}} + \Delta_i^{\text{route}}$ for every request. They locate the cost under an explicit intervention, without assuming that the unforced model would choose the same route.

Over successful flips with complete interventions, the *route-attributable fraction* (RAF) is

$$\text{RAF} = \frac{\sum_i \Delta_i^{\text{route}}}{\sum_i \Delta_i^{\text{total}}} = 1 - \frac{\sum_i \Delta_i^{\text{direct}}}{\sum_i \Delta_i^{\text{total}}}, \qquad \sum_i \Delta_i^{\text{total}} > 0. \tag{6}$$

RAF near 1 places nearly all aggregate excess in the route comparison. It can exceed 1 when the suffix reduces cost under forced SHORT, offsetting part of the reasoning-route increase. RAF sums token or latency differences; UECI instead averages relative excess on originally correct requests.

We also ask how often the costlier route provides no improvement in benchmark correctness. The *dominated-override request rate* (DORR) counts requests in E that flip, cost more under forced THINK than forced SHORT, and are no more correct under THINK. Both-correct and both-incorrect pairs count as equal correctness. If k requests meet these conditions and U contains flips with incomplete interventions, then

$$\text{DORR} \in [k/|E|,\ (k + |U|)/|E|]. \tag{7}$$

The endpoints treat all incomplete cases as nonqualifying or qualifying. These bounds describe missing intervention outcomes; confidence intervals describe sampling uncertainty.

4.3 Evaluating a hardened route head after attacker adaptation

We train a linear route head on the normalized hidden state just before routing, keeping the backbone frozen. Clean-input forced routes provide the training label: use the only correct route, or the cheaper route when both are correct. Capped pairs and pairs with neither route correct are excluded. Each prompt contributes one clean and eight augmented views: the original optimized suffix, three random lengths, natural text, a lexical cue, route margin, and hybrid. All share the clean label. Training combines clean cross-entropy, augmented cross-entropy with weight 0.8, and route-posterior consistency with weight 0.15.

We then compare two attackers against this same trained head. The fixed attacker reuses the original suffix. The adaptive attacker searches again over three seeds and up to 100 sweeps, minimizing $\text{softplus}(q_S - q_T)$ for the head's route logits. The head and decoding remain unchanged, so the comparison measures what the attacker recovers by adapting to the deployed policy (Athalye et al., 2018; Tramèr et al., 2020).

5 Experiments

5.1 Models, tasks, and reporting

We study five settings: AdaptThink-1.5B/7B and Thinkless-1.5B on GSM8K (Cobbe et al., 2021); AdaptThink-1.5B on MATH-500 (Hendrycks et al., 2021); and Phi-4-RV-15B (Aneja et al., 2026) on MMLU-Pro (Wang et al., 2024b). Phi-4 evaluation uses 25 prompts each from business, history, law, and psychology. Each setting has its own optimized shared suffix; the study tests within-setting control, not suffix transfer between models.

Core and objective experiments use greedy bfloat16 decoding and a 2,048-generated-token cap; head and serving experiments use 512. Main evaluations contain 100 prompts; compact random controls use the stated 50 prompts. The core GSM8K test set has 84 clean-short requests. A separate objective-comparison test set has 98, and comparisons stay within each test set. Phi-4 uses disjoint 20/4/8/100 pilot/optimization/validation/test prompts. After head training, 80 GSM8K and 95 MATH-500 requests are clean-short. GSM8K head evaluation is training-disjoint; the MATH-500 head evaluation shares 57 prompt IDs with training.

We report Wilson 95% intervals for binary proportions (Wilson, 1927; Brown et al., 2001) and bootstrap intervals for continuous metrics (Efron, 1979). Paired cost differences use common

complete cohorts; five-seed summaries resample seeds and prompts. Serving summaries give three-run means and sample standard deviations. Table 1 reports cost-cohort sizes, cap rates, and all-prompt accuracy together, so the cost estimates can be read alongside completion. CCA p95 requires at least 20 complete flips.

5.2 SHORT SUFFIXES RAISE THE COST OF CORRECT ANSWERS

The core AdaptThink-1.5B/GSM8K suffix switches 75/84 clean-short requests to reasoning (Table 1), or 89.3% [80.9,94.3]. Its flip rate is 33.3 percentage points above the lexical cue and 78.6–82.2 points above the two random suffixes. Natural text and a reasoning cue produce zero and one flip, respectively.

Table 1: Executed route changes and correct-answer cost. AT denotes AdaptThink. F/E counts flips among clean-short requests; k/C_{comp} counts correct-answer flips costing at least 1.5 times as much, over the complete cost cohort. UECI uses that same denominator. Accuracy is a fraction; cap rates use all prompts. MATH-500 and Thinkless random controls have 50 prompts; other rows have 100. Each setting is optimized separately.

Model/task	Input	F/E	k/C_{comp}	UECI	Acc. $c \to a$	Cap (%)
AT-1.5B/GSM8K	optimized	75/84	51/72	1.193	.73→.70	3
	lexical cue	47/84	36/72	1.058	.73→.82	5
	random-1	9/84	6/63	.293	.73→.70	2
AT-7B/GSM8K	optimized	29/100	20/84	.484	.84→.81	0
	random	3/100	2/84	.077	.84→.84	0
AT-1.5B/MATH-500	optimized	69/100	40/79	1.034	.81→.81	5
	random	15/50	4/39	.198	.78→.74	0
Thinkless-1.5B/GSM8K	optimized	99/99	87/90	1.781	.92→.95	0
	random	50/50	41/44	1.829	.92→.90	2
Phi-4-RV-15B/ MMLU-Pro	optimized	93/100	48/52	23.281	.58→.61	12
	random-0	0/100	0/58	0.000	.58→.49	0

The effect remains when we require correctness before and after attack. On core GSM8K, 51 of 72 cost-cohort requests flip, remain correct, and use at least 1.5 times the generated tokens: UPCHR@1.5 is 70.8% [59.5,80.1]. UECI is 1.193 [0.942,1.464], or 119.3% average positive relative excess under Equation (2). All-prompt accuracy is 73% before attack and 70% after attack, with 3% cap hits. The lexical cue increases accuracy to 82% while producing UECI 1.058. Better aggregate accuracy can therefore coexist with extra cost on already-correct requests.

Phi-4-RV-15B provides an independent model and task setting. Its optimized suffix switches 93/100 routes (93.0% [86.3,96.6]), versus 0/100 for either random suffix and 1/100 for a reasoning cue. Among 52 complete, originally correct pairs, 48 flip, remain correct, and cost at least 1.5 times as much (92.3% [81.8,97.0]). UECI is 23.281 [12.450,35.812], while accuracy changes from 58% to 61% and 12% of requests hit the cap. Cost ratios over complete flips have median 11.87 and p95 248.36; short clean answers make long attacked generations especially expensive in relative terms. Flip rates range from 84% to 100% across the four domains.

The other settings show how route sensitivity varies across models and tasks. Independently optimized suffixes yield 29.0% flips on AdaptThink-7B/GSM8K and 69.0% on AdaptThink-1.5B/MATH-500. On Thinkless, both optimized and random suffixes reach 100%. This last case makes cost measurement especially useful: once route changes saturate, the flip rate cannot distinguish their resource consequences.

5.3 EQUAL FLIP RATES AND ACCURACY CAN HIDE DIFFERENT COSTS

Figure 2 shows why the audit follows each request through all three outcomes. In the GSM8K objective comparison, candidate margin and think cross-entropy both flip 98/98 routes, but attacked

accuracy is 43% and 85%. Route control alone therefore does not establish preservation of task performance.

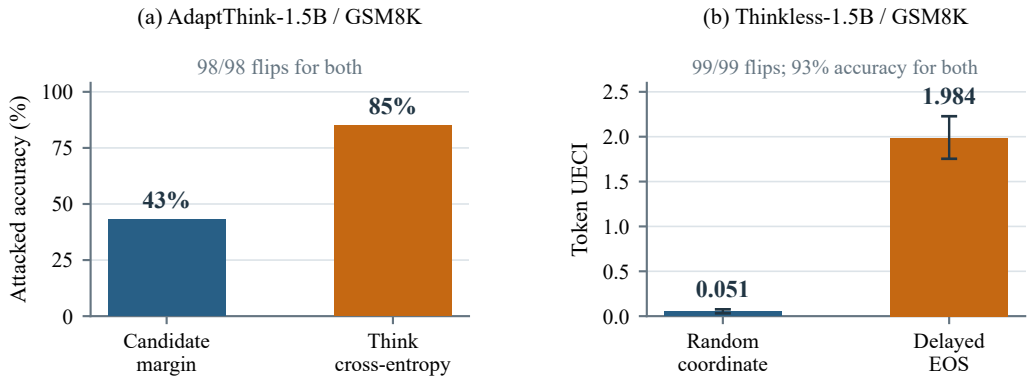


Figure 2: What aggregate success measures miss. (a) Both objectives flip 98/98 routes, but answer accuracy differs. (b) Both objectives flip 99/99 routes and achieve 93% accuracy, but correct-answer cost inflation differs. Panel (b) uses each condition's 92/89 complete cost samples and marginal 95% bootstrap intervals.

On Thinkless, random-coordinate search and delayed EOS both flip 99/99 routes and achieve 93% attacked accuracy. Yet UECI is 0.051 for random coordinate and 1.984 for delayed EOS, using their respective 92 and 89 complete cost samples. The cost-ratio medians are 1.000 and 2.764; UP-CHR@1.5 counts are 1/92 and 87/89. Equal routing and accuracy thus hide the distinction between changing a route at little additional cost and making correct answers much more expensive.

The paired metrics also distinguish cost inflation from answer damage. Direct length on AdaptThink/GSM8K gives a CCA median of 3.538, but attacked accuracy is 37% and UP-CHR@1.5 is 23/71. UECI assigns no contribution to attacked errors and retains the identity of originally correct requests. This makes its interpretation different from the length of an arbitrary attacked output.

Search objective and budget. At five sweeps, route margin reaches 98.8% flips on GSM8K, compared with 74.1% for random-coordinate search. The gap narrows with more sweeps, and the final MATH-500 budget favors random coordinate. On common complete cohorts, route margin's UECI difference is +0.382 (95% CI [+0.176, +0.623]) against random coordinate on MATH-500, +0.389 ([-0.163, +0.882]) against delayed EOS on GSM8K, and -0.397 ([-0.621, -0.137]) against delayed EOS on Thinkless. Search objectives therefore change early route control and final correct-answer cost differently; no single objective leads on every measure.

5.4 The reasoning-route change accounts for most extra work

Forced routes test the source of the additional cost (Table 2). Of the 75 core GSM8K flips, 73 have complete interventions. Token RAF is 0.989 [0.913,1.095], and latency RAF is 1.020 [0.938,1.137]. Under the specified prefixes, the route change accounts for about 98.9% of aggregate excess generated tokens; the direct-input share is about 1.1%.

Phi-4 makes the cost decomposition concrete. Across 84 complete interventions, clean forced-SHORT uses 92.4 tokens on average. Adding the suffix while keeping SHORT changes this by -8.7 tokens. Switching the attacked input from SHORT to THINK adds 812.6 tokens. The input change therefore offsets a small part of the much larger route cost, giving token RAF 1.011 [0.992,1.028]. This explains why RAF can exceed 1 without contradicting the additive accounting.

DORR checks whether the extra reasoning improves the answer. On core GSM8K, 61/84 eligible requests use a costlier forced reasoning route without improving correctness over forced SHORT. The two incomplete interventions give bounds of [72.6%,75.0%]. Phi-4 gives [78.0%,87.0%]. Together,

Table 2: Where the excess cost arises. RAF is the route comparison's share of aggregate excess over successful flips with complete interventions. DORR bounds count costlier reasoning routes with no correctness improvement, placing incomplete cases at either extreme. Complete counts define the RAF population; the difference from F in Table 1 gives the unknown count for DORR. A dash denotes unreported latency RAF.

Setting	complete	token RAF [95% CI]	latency RAF [95% CI]	DORR bounds (%)
AT-1.5B/GSM8K	73	.989 [.913,1.095]	1.020 [.938,1.137]	[72.6,75.0]
AT-7B/GSM8K	28	.987 [.850,1.106]	.863 [.704,1.010]	[25.0,26.0]
AT-1.5B/MATH-500	62	1.136 [.978,1.473]	1.119 [.978,1.412]	[59.0,66.0]
Thinkless-1.5B/GSM8K	98	1.000 [.982,1.017]	.998 [.981,1.014]	[92.9,93.9]
Phi-4-RV-15B/ MMLU-Pro	84	1.011 [.992,1.028]	–	[78.0,87.0]

the cost decomposition and the answer comparison show that an attacker can redirect computation toward a more expensive route with no observed correctness benefit on many requests.

5.5 Re-optimization restores attack success after head hardening

On training-disjoint GSM8K evaluation, the original fixed suffix flips 42.5% of eligible routes against the trained head. Re-optimizing against that same head raises the rate to 76.3% (Table 3). UECI rises from 0.412 to 0.827 on the respective complete cohorts. The increase comes from changing the attack while holding the trained policy and decoding fixed.

Table 3: Fixed and adaptive attacks against a trained route head, with a 512-token generation cap. Original retains the model's routing policy. Fixed and adaptive attacks share the hardened head; the latter searches against it again. GSM8K is training-disjoint; † MATH-500 shares 57 training IDs. Accuracy and cap rates use all 100 prompts; UECI uses each condition's complete cost cohort.

Data	System / attacker	Acc. $c \to a$	Exec-FlipASR [95% CI] (F/E)	UECI	Cap (%)
GSM	Original + fixed	.73→.70	89.3 [80.9,94.3] (75/84)	1.193	3
	Hardened + fixed	.71→.68	42.5 [32.3,53.4] (34/80)	.412	5
	Hardened + adaptive $B = 100$	.71→.69	76.3 [65.9,84.2] (61/80)	.827	4
MATH†	Original + fixed	.81→.81	69.0 [59.4,77.2] (69/100)	1.034	5
	Hardened + fixed	.79→.78	49.5 [39.6,59.4] (47/95)	.526	7
	Hardened + adaptive $B = 100$	.79→.78	65.3 [55.3,74.1] (62/95)	.718	6

The hardened GSM8K head achieves 71% clean accuracy, compared with 73% for original routing, and increases prompt-plus-generation length by 2.3%. Under fixed/adaptive attack, its accuracy is 68%/69% and cap rates are 5%/4%. Always choosing SHORT gives 60% accuracy; a 256-token cap gives 22% accuracy and 78% cap hits. The fixed suffix is included in training augmentation, so its result measures resistance to a known pattern. Re-optimization tests adaptation to the frozen head. The comparison shows why resource controls need both task-quality checks and an attacker that responds to the new policy.

5.6 Attack traffic adds latency for clean requests

We test whether the additional work affects other requests on a shared server. On one H100 PCIe with vLLM 0.10.2, we use a 512-generated-token cap and Poisson arrivals at 2 or 4 requests/s. Each run contains 100 requests, and each condition has three workload-order seeds. Gate-Override and random-suffix traffic reuse prompts, arrival times, and order at perturbed fractions $\alpha \in \{0, .05, .10, .20\}$. Latency is measured on successful clean requests; throughput uses all successful requests and includes arrival and final drain time.

At 4 requests/s and 20% perturbed traffic, mean clean p99 latency is 2.457 s with Gate-Override and 2.370 s with random suffixes, a 3.7% increase. Request throughput is 7.0% lower and token

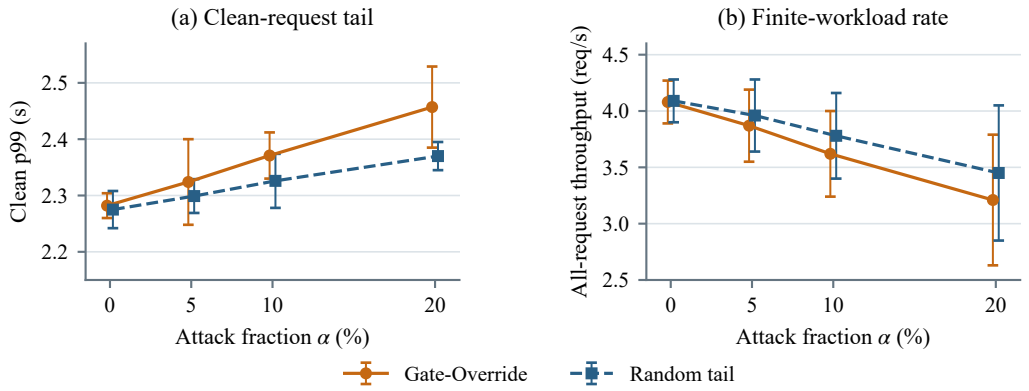


Figure 3: Effects on a shared service at the higher offered load. Gate-Override and random-suffix traffic use matched finite workloads. Points show three-run means; bars show sample standard deviations. Clean p99 measures the slowest tail of clean-request latency. Throughput counts all successful requests over the full run duration.

throughput is 5.4% lower. Relative to their own zero-perturbation baselines, clean p99 rises 7.7% and 4.2%, respectively. The matched comparison measures the extra effect of optimized attack traffic at the same load and fraction.

At that load and fraction, the three-run clean-p99 SDs are 0.072 s and 0.025 s for Gate-Override and random suffixes; cap rates are 11.0% and 7.0%. At 2 requests/s, clean p99 is 2.329 s versus 2.273 s. These are finite-workload measurements: each run's clean p99 is computed before averaging, and throughput includes final drain time. Across 4,800 requests, 2 fail. The observed service effect complements the request-level token audit; the two measure different resource consequences.

6 Discussion

What a resource-security audit should retain. Keep the input, executed route, answer, completion status, and cost linked by request. Route changes establish control; paired correctness and cost identify expensive correct answers; forced routes locate the added work. Testing an adapted attacker and matched shared workloads then connects this request-level mechanism to policy hardening and clean-user latency.

The input boundary being tested. The same suffix flips 72/80 requests at the query tail or before the assistant marker and 0/80 after the fully rendered prompt. This tests a specific token-ID access boundary, not a stealthy text-only API attack. The practical question is both how untrusted content reaches the route decision and how the selected route spends computation.

What counts as useful extra reasoning. Our utility measure is benchmark correctness. A task requiring detailed explanations may benefit from more reasoning even when the final answer is unchanged. Deployment audits can substitute the relevant quality measure while retaining request pairing, completion status, and cost. The central question remains whether added computation serves the task; here, many induced route changes add cost without an observed correctness gain.

7 Conclusion

Gate-Override shows how short shared suffixes increase computation on correct answers. Paired metrics expose costs hidden by route-switch rates and accuracy; forced routes locate most excess work in the reasoning-route change. Adaptive attacks and matched serving experiments connect this mechanism to policy hardening and clean-user latency. Adaptive reasoning security therefore requires tracking how a model spends computation alongside whether it answers correctly.

Ethics Statement

We study compute-resource abuse through offline open-model evaluation and a self-hosted controlled service environment. We do not test third-party online services or user data. Releasing route parsers, denominator definitions, and controlled suffix artifacts supports reproduction and has dual-use implications. Responsible release should include rate limiting, route logs, and input-provenance checks, with an explicit prohibition on unauthorized testing of third-party services.

Reproducibility Statement

The main text specifies the threat model, metric populations, search and intervention methods, evaluation splits, and serving comparisons. Appendix A adds implementation details, Appendix B proves metric identities, and Appendix C collects additional numerical settings. The accompanying source contains the reported aggregate data and scripts for rebuilding the numerical summaries and vector figures.

AI Use Statement

We used generative AI tools to assist with grammar checking and correction, as well as checking LaTeX formatting. The authors are responsible for the final text, claims, and artifacts.

References

Jyoti Aneja, Michael Harrison, Neel Joshi, Tyler LaBonte, John Langford, and Eduardo Salinas. Phi-4-reasoning-vision-15B technical report. *arXiv preprint arXiv:2603.03975*, 2026. doi: 10.48550/arXiv.2603.03975. URL https://arxiv.org/abs/2603.03975.

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 274–283, 2018. URL https://proceedings.mlr.press/v80/athalye18a.html.

Simon A. Aytes, Jinheon Baek, and Sung Ju Hwang. Sketch-of-thought: Efficient LLM reasoning with adaptive cognitive-inspired sketching. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 24296–24320, Suzhou, China, November 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.1236. URL https://aclanthology.org/2025.emnlp-main.1236/.

Emmanuel Bengio, Pierre-Luc Bacon, Joelle Pineau, and Doina Precup. Conditional computation in neural networks for faster models, 2015. URL https://arxiv.org/abs/1511.06297.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. ISBN 978-0-521-83378-3. doi: 10.1017/CBO9780511804441. URL https://web.stanford.edu/~boyd/cvxbook/.

Lawrence D. Brown, T. Tony Cai, and Anirban DasGupta. Interval estimation for a binomial proportion. *Statistical Science*, 16(2):101–133, 2001. doi: 10.1214/ss/1009213286.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. doi: 10.48550/arXiv.2110.14168. URL https://arxiv.org/abs/2110.14168.

Bradley Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1): 1–26, 1979. doi: 10.1214/aos/1176344552.

Gongfan Fang, Xinyin Ma, and Xinchao Wang. Thinkless: LLM learns when to think. In *Advances in Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=ariVQf0KZx. NeurIPS 2025 poster; arXiv:2505.13379.

Alex Graves. Adaptive computation time for recurrent neural networks, 2016. URL https://arxiv.org/abs/1603.08983.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. Token-budget-aware LLM reasoning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 24842–24855, Vienna, Austria, July 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-acl.1274. URL https://aclanthology.org/2025.findings-acl.1274/.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

David Huang, Avidan Shah, Alexandre Araujo, David Wagner, and Chawin Sitawarin. Stronger universal and transferable attacks by suppressing refusals. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 5850–5876, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.naacl-long.302. URL https://aclanthology.org/2025.naacl-long.302/.

Abhinav Kumar, Jaechul Roh, Ali Naseh, Marzena Karpinska, Mohit Iyyer, Amir Houmansadr, and Eugene Bagdasarian. OverThink: Slowdown attacks on reasoning LLMs, 2025. URL https://arxiv.org/abs/2502.02542.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, 2023. doi: 10.1145/3600006.3613165. URL https://arxiv.org/abs/2309.06180.

Xingyu Li, Xiaolei Liu, Cheng Liu, Yixiao Xu, Kangyi Ding, Bangzhou Xin, and Jia-Li Yin. LoopLLM: Transferable energy-latency attacks in LLMs via repetitive generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026. URL https://ojs.aaai.org/index.php/AAAI/article/view/40445. Also available as arXiv:2511.07876.

Wenhao Lin, Qiong Wu, Shuaixiang Yang, Yiyi Zhou, Xiaoshuai Sun, and Rongrong Ji. Towards efficient chain-of-thought reasoning via adaptive-budgeting based policy optimization. OpenReview submission to ICLR 2026, withdrawn, 2025. URL https://openreview.net/forum?id=ji8OFKRYwX.

Hongfu Liu, Yuxi Xie, Ye Wang, and Michael Shieh. Advancing adversarial suffix transfer learning on aligned large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 7213–7224, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.409. URL https://aclanthology.org/2024.emnlp-main.409/.

Shuaitong Liu, Renjue Li, Lijia Yu, Lijun Zhang, Zhiming Liu, and Gaojie Jin. BadThink: Triggered overthinking attacks on chain-of-thought reasoning in large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 32141–32149, 2026a. doi: 10.1609/aaai.v40i38.40486. URL https://ojs.aaai.org/index.php/AAAI/article/view/40486.

Xiaogeng Liu, Xinyan Wang, Yechao Zhang, Sanjay Kariyappa, Chong Xiang, Muhao Chen, G. Edward Suh, and Chaowei Xiao. ReasoningBomb: A stealthy denial-of-service attack by inducing pathologically long reasoning in large reasoning models, 2026b. URL https://arxiv.org/abs/2602.00154.

Chenwei Lou, Zewei Sun, Xinnian Liang, Meng Qu, Wei Shen, Wenqi Wang, Yuntao Li, Qingping Yang, and Shuangzhi Wu. AdaCoT: Pareto-optimal adaptive chain-of-thought triggering via reinforcement learning. *arXiv preprint arXiv:2505.11896*, 2025. doi: 10.48550/arXiv.2505.11896. URL https://arxiv.org/abs/2505.11896.

Shangke Lyu, Linjuan Wu, Yuchen Yan, Xingyu Wu, Hao Li, Yongliang Shen, Peisheng Jiang, Weiming Lu, Jun Xiao, and Yueting Zhuang. Hierarchical budget policy optimization for adaptive reasoning. *arXiv preprint arXiv:2507.15844*, 2025. doi: 10.48550/arXiv.2507.15844. URL https://arxiv.org/abs/2507.15844.

Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947. doi: 10.1007/BF02295996.

Judea Pearl. Direct and indirect effects. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, pp. 411–420. Morgan Kaufmann, 2001. URL https://ftp.cs.ucla.edu/pub/stat_ser/R273.pdf.

Kangyu Qiao, Shaolei Zhang, and Yang Feng. IG-pruning: Input-guided block pruning for large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 10618–10629, Suzhou, China, November 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.537. URL https://aclanthology.org/2025.emnlp-main.537/.

Meghana Arakkal Rajeev, Rajkumar Ramamurthy, Prapti Trivedi, Vikas Yadav, Oluwanifemi Bamgbose, Sathwik Tejaswi Madhusudhan, James Zou, and Nazneen Rajani. Cats confuse reasoning LLM: Query-agnostic adversarial triggers for reasoning models. In *Proceedings of the Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=VrEPiN5WhM. COLM 2025; arXiv:2503.01781.

Anil Ramakrishna, Jimit Majmudar, Rahul Gupta, and Devamanyu Hazarika. LLM-PIEval: A benchmark for indirect prompt injection attacks in large language models. *NeurIPS 2024 Workshop on Adversarial Machine Learning*, 2024. URL https://www.amazon.science/publications/llm-pieval-a-benchmark-for-indirect-prompt-injection-attacks-in-large-language-models.

Avital Shafran, Roei Schuster, Thomas Ristenpart, and Vitaly Shmatikov. Rerouting LLM routers. In *Proceedings of the Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=U6C7odo5SX. arXiv:2501.01818.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017. URL https://arxiv.org/abs/1701.06538.

Wai Man Si, Mingjie Li, Michael Backes, and Yang Zhang. Excessive reasoning attack on reasoning LLMs, 2025. URL https://arxiv.org/abs/2506.14374.

Haochun Tang, Yuliang Yan, Jiahua Lu, Huaxiao Liu, and Enyan Dai. Route to Rome attack: Directing LLM routers to expensive models via adversarial suffix optimization. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 44333–44347. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.2051. URL https://aclanthology.org/2026.acl-long.2051/.

Surat Teerapittayanon, Bradley McDanel, and H. T. Kung. BranchyNet: Fast inference via early exiting from deep neural networks. In *23rd International Conference on Pattern Recognition*, pp. 2464–2469, 2016. doi: 10.1109/ICPR.2016.7900006.

Florian Tramèr, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/11f38f8ecd71867b42433548d1078e38-Abstract.html.

Songjun Tu, Jiahao Lin, Qichao Zhang, Xiangyu Tian, Linjing Li, Xiangyuan Lan, and Dongbin Zhao. Learning when to think: Shaping adaptive reasoning in R1-style models via multi-stage RL, 2025. URL https://arxiv.org/abs/2505.10832.

Hao Wang, Hao Li, Minlie Huang, and Lei Sha. ASETF: A novel method for jailbreak attack on LLMs through translate suffix embeddings. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 2697–2711, Miami, Florida, USA, November 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.157. URL https://aclanthology.org/2024.emnlp-main.157/.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-Pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*, 2024b. doi: 10.48550/arXiv.2406.01574. URL https://arxiv.org/abs/2406.01574.

Edwin B. Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927. doi: 10.1080/01621459.1927.10502953.

Chen Xiong, Xiangyu Qi, Pin-Yu Chen, and Tsung-Yi Ho. Defensive prompt patch: A robust and generalizable defense of large language models against jailbreak attacks. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 409–437, Vienna, Austria, July 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-acl.23. URL https://aclanthology.org/2025.findings-acl.23/.

Yue Xu and Wenjie Wang. LinkPrompt: Natural and universal adversarial attacks on prompt-based language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 6473–6486, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.360. URL https://aclanthology.org/2024.naacl-long.360/.

Zitao Xuan, Xiaofeng Mao, Da Chen, Xin Zhang, Yuhan Dong, and Jun Zhou. ShieldHead: Decoding-time safeguard for large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 18129–18143, Vienna, Austria, July 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-acl.932. URL https://aclanthology.org/2025.findings-acl.932/.

Chenxu Yang, Qingyi Si, Yongjie Duan, Zheliang Zhu, Chenyu Zhu, Qiaowei Li, Minghui Chen, Zheng Lin, and Weiping Wang. Dynamic early exit in reasoning models. *arXiv preprint arXiv:2504.15895*, 2025. doi: 10.48550/arXiv.2504.15895. URL https://arxiv.org/abs/2504.15895.

Van Yang, Shouren Wang, Debargha Ganguly, Xinpeng Li, Chaoda Song, Vikash Singh, Vipin Chaudhary, and Xiaotian Han. Mid-Think: Training-free intermediate-budget reasoning via token-level triggers. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 6024–6038. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.299. URL https://aclanthology.org/2026.findings-acl.299/.

Gyeong-In Yu, Joo Seong Jeong, Geon-Woo Kim, Soojeong Kim, and Byung-Gon Chun. Orca: A distributed serving system for Transformer-Based generative models. In *16th USENIX Symposium on Operating Systems Design and Implementation*, pp. 521–538. USENIX Association, 2022. URL https://www.usenix.org/conference/osdi22/presentation/yu.

Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. InjecAgent: Benchmarking indirect prompt injections in tool-integrated large language model agents. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 10471–10506, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.624. URL https://aclanthology.org/2024.findings-acl.624/.

Jiajie Zhang, Nianyi Lin, Lei Hou, Ling Feng, and Juanzi Li. AdaptThink: Reasoning models can learn when to think. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 3716–3730, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.184. URL https://aclanthology.org/2025.emnlp-main.184/.

Yinmin Zhong, Shengyu Liu, Junda Chen, Jianbo Hu, Yibo Zhu, Xuanzhe Liu, Xin Jin, and Hao Zhang. DistServe: Disaggregating prefill and decoding for goodput-optimized large language model serving. In *18th USENIX Symposium on Operating Systems Design and Implementation*, pp. 193–210. USENIX Association, 2024. URL https://www.usenix.org/conference/osdi24/presentation/zhong-yinmin.

Zhenhao Zhu, Yue Liu, Zhiwei Xu, Yingwei Ma, Hongcheng Gao, Nuo Chen, Yanpei Guo, Wenjie Qu, Huiying Xu, Zifeng Kang, Xinzhong Zhu, and Jiaheng Zhang. ExtendAttack: Attacking servers of LRMs via extending reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 35257–35265, 2026. doi: 10.1609/aaai.v40i41.40833. URL https://ojs.aaai.org/index.php/AAAI/article/view/40833.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023. doi: 10.48550/arXiv.2307.15043. URL https://arxiv.org/abs/2307.15043.

A Reproduction Details

A.1 Suffix search and route implementation

The main text defines the attack access, objectives, selection rule, and accounting units. Table 4 gives the implementation order. Reserved and control IDs are excluded from replacement candidates. The gradient is refreshed once per complete sweep; the incumbent token is not in the shortlist. Each update chooses the best rescored candidate within that shortlist, while validation selects the retained checkpoint. Thus the method does not require every coordinate update to lower the objective.

Table 4: Implementation of shared-suffix coordinate search. A candidate forward evaluates the four-example optimization batch. Validation calls are additional.

Step	Operation
1	Initialize $L = 10$ allowed token IDs with a fixed seed; preserve the model-specific rendered template.
2	Evaluate the batch objective on four optimization examples and backpropagate once through suffix embeddings.
3	Project coordinate gradients onto vocabulary embeddings; exclude reserved and control IDs; shortlist $k = 3$ candidates.
4	Sequentially rescore each coordinate's candidates on the current suffix and commit the best shortlisted token; refresh the gradient at the next complete sweep.
5	At budgets 5, 20, and 100, retain the best within-seed checkpoint by objective on eight validation examples; select the final suffix by validation executed flips, breaking ties by objective.
6	Freeze the suffix, generate the held-out test manifest once, and compute metrics from request-paired records.

The input builder tokenizes the rendered prompt once, then inserts the selected IDs without decoding and retokenizing. AdaptThink uses `</think>` (151649) for SHORT and `Alright` (71486) for THINK. Thinkless uses `<short>` (151650) and `<think>` (151648), respectively. Its matched objective suite includes an explicit reasoning instruction. Phi-4 uses native single-token `<nothink>` and `<think>` markers. Parsing checks the first generated route token and then the native opening/closing markers, retaining unmatched trajectories as OTHER. Forced-route records retain both the prepended token and the parsed continuation.

A.2 Phi-4 search configuration

The disjoint pilot/optimization/validation/test splits contain 20/4/8/100 MMLU-Pro prompts. The test set contains 25 prompts from each of business, history, law, and psychology. A 20-prompt pilot produces 20 SHORT trajectories at 80% accuracy. Three seeds are screened through $B = 20$; seed 2703 is then extended to $B = 100$ and selected after flipping 8/8 validation routes. The test set is not used for this decision. The selected suffix IDs are

$$[633, 2468, 38696, 2224, 9066, 74029, 2905, 5263, 45951, 4343].$$

The decoded display contains a replacement glyph, so these IDs, not a decoded string, specify the intervention. Greedy bfloat16 decoding uses the 2,048-generated-token cap. In all four Phi-4 conditions, candidate preference, first-token class, and parsed route agree, with no OTHER trajectories.

A.3 Route-head training

For clean and augmented route posteriors p_0, p_a and clean oracle label y^*, training minimizes

$$\mathcal{L}_{\text{head}} = \text{CE}(p_0, y^*) + 0.8\,\text{CE}(p_a, y^*) + 0.15\,D_{\text{KL}}(p_a \| p_0). \tag{8}$$

The backbone is frozen. The linear head reads the normalized last hidden state before routing. Full-batch AdamW uses learning rate 2×10^{-3}, weight decay 10^{-3}, at most 300 epochs, early-stopping patience 40, and feature-extraction batches of 32. Training cross-entropy is class-weighted; clean validation accuracy and loss are unweighted.

Each prompt contributes nine views: clean, original optimized suffix, random suffixes of lengths 4/10/16, natural text, lexical cue, route-margin suffix, and hybrid suffix. All share the clean-input oracle label. Complete forced-route pairs supply that label: choose the only correct route or the cheaper route when both are correct; omit capped pairs and pairs with neither answer correct.

GSM8K supplies 555 oracle-labeled prompts, split into 459 training and 96 validation examples. Training short/think labels number 390/69; validation labels number 76/20. MATH-500 supplies 228 prompts, split into 171/57, with short/think counts 111/60 and 38/19. The selected epochs are 7 and 9; validation route-label accuracies are 93.8% and 89.5%, respectively. These measure label prediction, not task-answer accuracy. GSM8K test IDs are disjoint from head training; the MATH-500 test set contains 57 head-training IDs, as marked in the main text.

Adaptive suffix search uses three seeds and $B \in \{5, 20, 100\}$, with $30B$ candidate forwards per seed and additional validation. At $B = 100$, selected validation flips are 6/7 on GSM8K and 5/6 on MATH-500. Fixed and adaptive attacks use the same trained head and 512-token generation cap. Original-policy cost summaries at that cap and core summaries at 2,048 tokens form their complete cohorts independently.

A.4 Serving and statistical details

Serving uses AdaptThink-1.5B-delta0.02 on one NVIDIA H100 PCIe with vLLM 0.10.2, bfloat16, maximum model length 4,096, a 512-generated-token cap, and 128 request workers. Sixteen traffic–fraction–load conditions produce 48 runs of 100 requests each. Gate-Override and random-suffix runs share prompts, arrival schedules, and workload order. Of 4,800 requests, 4,798 succeed; at high load and 20% perturbation, one Gate-Override request and one random-suffix request fail.

Clean latency quantiles use successful clean requests and are computed separately within each run. Request throughput divides all successful requests by elapsed run time; token throughput uses their generated-token total and the same duration. Timing includes arrivals and the final drain. Cap rates use each run's successful-request denominator. Table 6 reports means and sample SDs across three runs, not steady-state service estimates.

Binary intervals use Wilson's method. Continuous paired comparisons recompute both metrics on the common complete cohort and bootstrap paired requests; seed summaries resample seeds and prompts. Pairwise flip tests use exact McNemar tests (McNemar, 1947). DORR endpoints assign incomplete interventions to opposite outcomes; they are identification bounds rather than sampling confidence intervals.

Aggregate sources. The source package retains reported aggregates in `paper_data.json`. `build_paper_data.py` and `make_paper_figures.py` rebuild table fragments, repeated-value macros, and vector figures. The complete aggregate source is retained even where the manuscript omits a secondary analysis. These files reproduce the reported summaries; model-level reruns additionally require the checkpoint revisions, exact templates, split manifests, selected suffixes, paired generations, and serving traces.

B Metric Identities

The main-text metrics can be used directly from their definitions. This section derives their exact relationship and the route objective's first-token condition.

Lemma 1 (Threshold and expectation identity)**.** *On a fixed, finite, nonempty complete cost cohort with positive clean costs, write $H_K(\gamma) = \text{UP-CHR@}\gamma$ as a fraction. Then*

$$\text{UECI}_K = \int_1^\infty H_K(\gamma)\, d\gamma, \qquad (\gamma - 1)H_K(\gamma) \le \text{UECI}_K \quad (\gamma > 1). \tag{9}$$

Proof. For a cost ratio $r \ge 0$, $\int_1^\infty \mathbf{1}[r \ge t]\, dt = [r - 1]_+$. Sum this identity over G and divide by $|C_{\text{comp}}|$. Since H_K is nonincreasing, $(\gamma - 1)H_K(\gamma) \le \int_1^\gamma H_K(t)\, dt \le \text{UECI}_K$. □

Lemma 2 (Full-vocabulary sufficient condition)**.** *For $|\mathcal{V}| \geq 2$, let $m_T = \max_{v \neq t_T}(z_v - z_{t_T})$ and $g_\tau = \tau \log \sum_{v \neq t_T} e^{(z_v - z_{t_T})/\tau}$. For $\tau > 0$,*

$$m_T \leq g_\tau \leq m_T + \tau \log(|\mathcal{V}| - 1). \tag{10}$$

Thus $g_\tau < 0$ implies that t_T has a strictly larger logit than every other token.

Proof. Factor $e^{m_T/\tau}$ out of the sum. The remaining sum lies between 1 and $|\mathcal{V}| - 1$. Taking its logarithm and multiplying by τ gives the bounds (Boyd & Vandenberghe, 2004). □

At $\tau = 1$, target-token cross-entropy is $\mathrm{CE}_T = \log(1 + e^{g_1})$ and $\nabla \mathrm{CE}_T = \sigma(g_1) \nabla g_1$, where σ is the logistic function. Averaging cross-entropy rather than route margin therefore weights examples by $\sigma(g_1)$. Both target the first-token decision; executed trajectories and paired costs supply the audit outcomes.

C Supplementary Numerical Results

The main text contains the comparisons and evidence needed for the conclusions. The following two tables collect additional objective settings and serving conditions in a compact form.

Table 5: Validation-selected objective comparisons. AT is AdaptThink-1.5B. All settings use 100 prompts and a 2,048-token cap. F/E counts executed flips; k/C_{comp} counts correct-answer flips at ≥ 1.5 times cost over the complete cost cohort. UECI uses that same cohort; paired differences in the main text instead use the intersection of two cohorts. Accuracy is a fraction and caps are percentages.

Setting	Objective	F/E	k/C_{comp}	UECI [95% CI]	Acc. $c \to a$	Cap (%)
AT/GSM8K	Candidate margin	98/98	38/77	.730 [.551,.914]	.79→.43	0
	Think CE	98/98	70/75	1.568 [1.393,1.751]	.79→.85	4
	Route margin	96/98	61/73	1.853 [1.544,2.181]	.79→.73	6
	Hybrid	96/98	65/75	1.277 [1.132,1.423]	.79→.80	2
	Delayed EOS	63/98	47/77	1.475 [1.079,2.025]	.79→.80	2
	Delayed close	88/98	49/77	1.001 [.813,1.206]	.79→.68	1
	Direct length	81/98	23/71	1.140 [.743,1.643]	.79→.37	8
	Random coordinate	96/98	68/77	1.223 [1.066,1.397]	.79→.78	1
AT/MATH-500	Candidate margin	89/98	51/76	.955 [.771,1.154]	.77→.77	5
	Think CE	91/98	51/76	1.085 [.874,1.326]	.77→.83	5
	Route margin	98/98	51/72	1.136 [.906,1.400]	.77→.76	9
	Hybrid	94/98	33/55	1.060 [.787,1.383]	.77→.61	32
	Random coordinate	98/98	34/59	.838 [.631,1.071]	.77→.63	26
Thinkless/GSM8K	Candidate margin	99/99	85/89	1.616 [1.399,1.868]	.92→.95	1
	Think CE	99/99	83/88	1.823 [1.613,2.041]	.92→.90	4
	Route margin	99/99	85/89	1.567 [1.354,1.803]	.92→.93	3
	Hybrid	99/99	85/89	1.832 [1.617,2.061]	.92→.91	2
	Delayed EOS	99/99	87/89	1.984 [1.754,2.228]	.92→.93	3
	Direct length	99/99	83/86	1.919 [1.663,2.240]	.92→.95	1
	Random coordinate	99/99	1/92	.051 [.032,.076]	.92→.93	0

Table 6: All serving conditions. Gate denotes Gate-Override; Random uses unoptimized random suffixes. Clean p99 uses successful clean requests; throughput and cap rates use all successful requests. Values are three-run means ± sample SD. Each run has 100 arrivals; throughput includes arrival and final drain time.

Load (req/s)	Traffic	α (%)	Clean p99 (s)	Req/s	Tok/s	Cap (%)
4	Gate	0	$2.282 \pm .022$	$4.08 \pm .19$	1582 ± 75	3.0 ± 0.0
	Gate	5	$2.324 \pm .076$	$3.87 \pm .32$	1510 ± 121	4.7 ± 0.6
	Gate	10	$2.371 \pm .041$	$3.62 \pm .38$	1425 ± 159	6.7 ± 0.6
	Gate	20	$2.457 \pm .072$	$3.21 \pm .58$	1288 ± 237	11.0 ± 0.9
	Random	0	$2.275 \pm .033$	$4.09 \pm .19$	1587 ± 72	3.0 ± 0.0
	Random	5	$2.299 \pm .030$	$3.96 \pm .32$	1542 ± 124	3.7 ± 0.6
	Random	10	$2.326 \pm .048$	$3.78 \pm .38$	1479 ± 148	4.7 ± 0.6
	Random	20	$2.370 \pm .025$	$3.45 \pm .60$	1362 ± 242	7.0 ± 0.0
2	Gate	0	$2.208 \pm .021$	$2.01 \pm .17$	772 ± 62	3.0 ± 0.0
	Gate	5	$2.241 \pm .027$	$1.95 \pm .40$	759 ± 152	4.3 ± 0.6
	Gate	10	$2.278 \pm .054$	$1.88 \pm .11$	731 ± 45	5.7 ± 0.6
	Gate	20	$2.329 \pm .058$	$1.62 \pm .20$	639 ± 91	9.7 ± 0.6
	Random	0	$2.198 \pm .043$	$2.02 \pm .17$	775 ± 65	3.0 ± 0.0
	Random	5	$2.218 \pm .039$	$1.97 \pm .40$	764 ± 158	3.7 ± 0.6
	Random	10	$2.249 \pm .023$	$1.92 \pm .10$	748 ± 41	4.3 ± 0.6
	Random	20	$2.273 \pm .030$	$1.68 \pm .21$	668 ± 76	6.0 ± 0.0