Adaptive Compute for Language Model Reasoning

Test Author One[1], **Test Author Two**[1], **Test Author Three**[2], **Test Author Four**[1,2,†]
[1]Test Research Organization One, [2]Test University Two

Abstract

We study adaptive computation for language-model reasoning under a fixed inference budget. Our method allocates additional tokens only when intermediate confidence estimates indicate that a problem is unresolved. Across mathematical reasoning, code generation, and scientific question answering, adaptive allocation improves accuracy while reducing average compute. The results suggest that inference-time scaling can be made both more predictable and more efficient when budget decisions are treated as part of the reasoning policy.

Keywords: *language models, adaptive computation, reasoning*

Website: https://example.org/adaptive-compute
Contact: test.author@example.org

1. Introduction

Language models often receive the same inference budget for every prompt even though problem difficulty varies substantially. Easy examples may be solved in a few steps, while difficult examples benefit from verification, revision, or a larger search tree. A useful technical-report template should make this kind of argument easy to read: the first page establishes the contribution, and the remaining pages prioritize equations, figures, and experimental evidence.

We consider a policy that selects an inference budget $b(x)$ for prompt x. Given utility U and compute cost C, the policy solves

$$b^\star(x) = \arg\max_{b \in \mathcal{B}} \mathbb{E}[U(y, x) \mid b, x] - \lambda C(b), \tag{1}$$

where λ controls the accuracy–efficiency trade-off. This formulation is compatible with outcome-based reinforcement learning as well as process-level supervision. It also separates the question of how to reason from the question of how much computation a particular input warrants.

2. Method

The controller observes summary statistics from the current reasoning state and chooses whether to stop, verify, or continue. Training examples are grouped by difficulty, and the controller is optimized against held-out utility rather than raw response length. This prevents verbosity from becoming a proxy for quality.

† *Corresponding author*

Table 1 | Illustrative accuracy and token usage. Lower token counts are better.

Method	Accuracy (%)	Mean tokens
Fixed short budget	71.4	512
Fixed long budget	78.2	2048
Adaptive budget	**79.0**	**936**

2.1. Budget controller

At decision step t, the controller receives state s_t and samples action $a_t \in \{\text{stop}, \text{verify}, \text{continue}\}$. The return combines task reward with a linear token penalty. A clipped policy update limits abrupt changes in allocation behavior, while an independent value head estimates the remaining benefit of additional computation.

The example in Table 1 illustrates the desired operating point: adaptive allocation approaches the quality of a long fixed budget while using less than half as many tokens. The values are placeholders and are included only to demonstrate the inherited table and caption typography.

3. Relationship to prior work

The placeholder bibliography demonstrates author–year citations for scaling and compute-allocation studies (Test Researcher One and Test Researcher Two, 2024; Test Researcher Three and Test Researcher Four, 2025). Our focus is complementary: the trained model is held fixed while computation is allocated dynamically at inference time. A separate synthetic entry illustrates how an architecture reference appears in the template (Test Researcher Five and Test Researcher Six, 2023).

4. Conclusion

This sample demonstrates the intended division of responsibility. The neutral base class supplies the type system, mathematical fonts, page geometry, paragraph rhythm, section hierarchy, captions, and later-page headers. The companion package changes only the title composition and abstract presentation, using the same XCharter family throughout.

References

Test Researcher Five and Test Researcher Six. A placeholder attention architecture. In *Proceedings of the Synthetic Systems Workshop*, pages 25–32, 2023.

Test Researcher One and Test Researcher Two. A placeholder study of language-model scaling. *Journal of Synthetic Examples*, 1(1):1–12, 2024.

Test Researcher Three and Test Researcher Four. Synthetic methods for compute-aware training. *Transactions on Placeholder Methods*, 2(1):13–24, 2025.