Speculative Decoding and Inference Throughput

Improving Economics Through Token Prediction and Verification

Author: Phil Marshall, PhD

Publish Date: June 2026

Speculative decoding is an inference optimization technique that improves throughput by using a smaller, faster draft model to predict several tokens ahead of a larger target model. Rather than requiring the target model to generate every token sequentially, the draft model proposes a short sequence of candidate tokens. The target model then verifies those candidates in parallel, accepting the tokens that are consistent with its own output distribution and regenerating any tokens that are rejected.

The goal is not to bypass the larger model, but to reduce the amount of sequential decoding work it must perform. The larger model remains in the serving path and still determines the final output.

Acceptance Rate Is the Key Variable

The key performance variable is the acceptance rate. This measures the share of draft-model tokens accepted by the target model without regeneration. For example, an 80% acceptance rate means that eight out of every ten proposed tokens are accepted. This reduces the amount of sequential decoding work the larger model must perform. Higher acceptance rates increase effective throughput, reduce latency, and lower serving cost per generated token.

A simple analogy is a highly optimistic assistant who writes several sentences ahead, while a careful senior editor quickly scans the draft and corrects only the parts that do not match the intended output.

© Copyright Tolaga Research, 2026

Acceptance rate depends on how well the draft model can anticipate the target model's likely next tokens. If the draft model is well-matched to the target model and the workload is predictable, more proposed tokens are accepted. If the draft model is poorly matched or the output is open-ended, more tokens are rejected and the benefit declines.

Designing the Draft Model for High Acceptance

The draft model is usually selected, fine-tuned, or purpose-built to produce token predictions that closely match the target model's likely outputs. In the simplest case, it may be a smaller model from the same family, using a compatible tokenizer and similar output style, which helps maximize the acceptance rate.

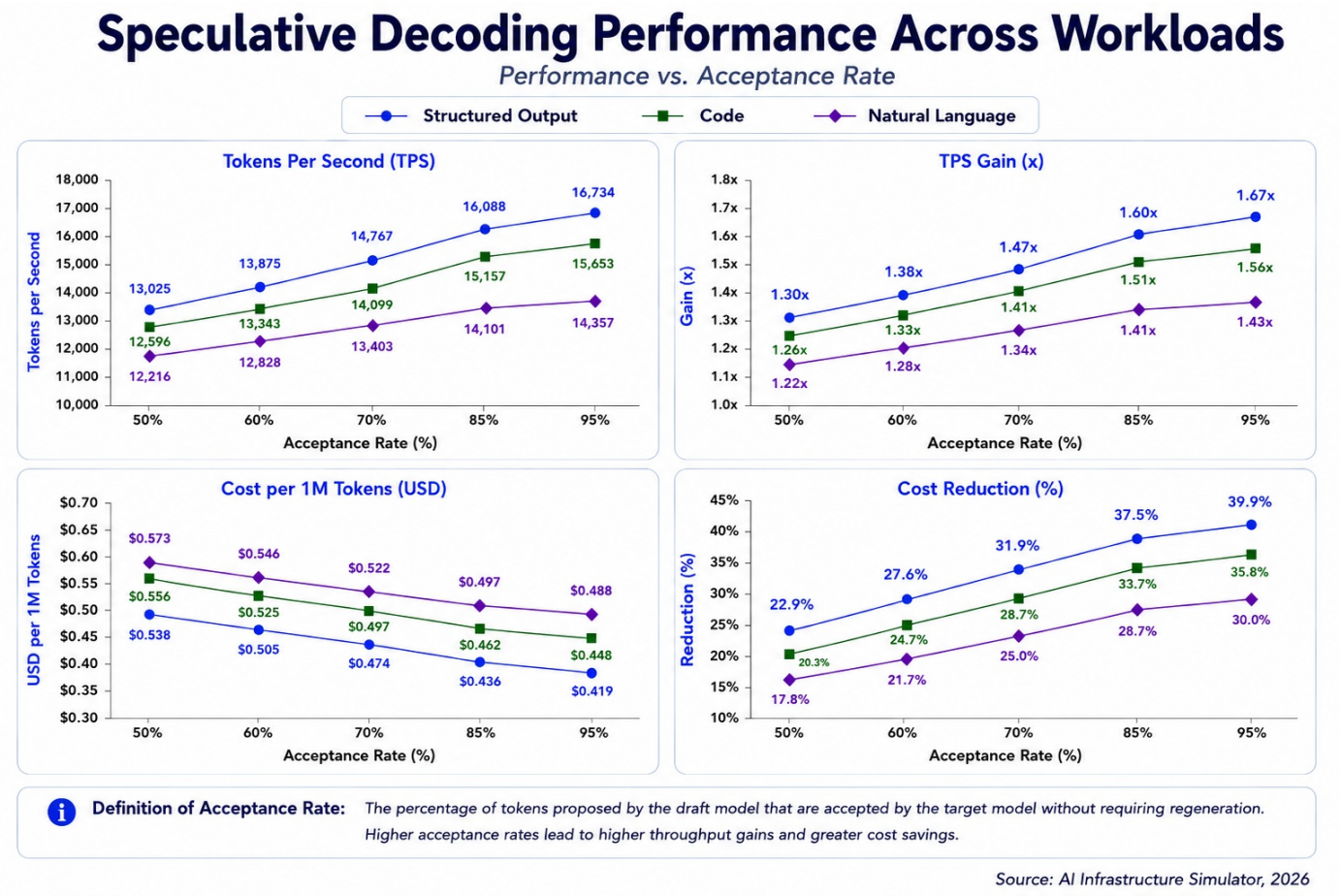
In other cases, the draft model may be fine-tuned on outputs generated by the target model, so that it learns to approximate the target model's behavior for a specific workload. A third approach is to train a draft model specifically for speculative decoding, where the objective is not broad standalone capability but fast and accurate short-horizon token prediction.

Modern variants push this idea further by moving beyond simple sequential drafting. Techniques such as Medusa add multiple prediction heads directly to the target model for tree-based speculation, while methods such as Eagle use more sophisticated drafting strategies to explore multiple possible continuations in parallel. These approaches can improve acceptance rates and reduce overhead compared with traditional draft-model approaches.

Simulated Throughput Gains

The results shown in the chart illustrate the importance of acceptance rate. At a 50% acceptance rate, throughput improves by approximately 1.22x relative to conventional decoding. At 70% acceptance, the improvement rises to approximately 1.41x. At 85% acceptance, throughput reaches around 1.56x, and at 95% acceptance, the improvement increases to approximately 1.67x.

© Copyright Tolaga Research, 2026



The gains are meaningful even at moderate acceptance levels, but they are strongest when the draft model can reliably anticipate the target model's output.

Reported Results Versus End-to-End Modeled Gains

The modeled results should be interpreted as end-to-end serving gains, not as the maximum possible decode-stage speedup. This distinction matters when comparing the simulator results with reported production implementations. For example, IBM has reported speculative decoding speedups in the 2x to 3x range using purpose-built speculator models and optimized serving integration.

The simulator's decode-stage assumptions are broadly aligned with that range. In the modeled scenarios, maximum speculative decode speedup is capped at 3.0x for structured outputs, 2.5x for code, and 2.0x for natural language. However, the reported TPS gain peaks at approximately 1.67x because the acceleration is applied only to the decode-compute portion of the inference request.

This difference does not imply that speculative decoding is underperforming. It reflects the translation of decode-stage acceleration into full serving-system economics. Prefill, target-model verification, memory movement, communication, batching, queueing, and other inference-server overheads continue to limit end-to-end throughput even when most draft tokens are accepted.

© Copyright Tolaga Research, 2026

Why the Gains Saturate

Performance gains can be large, but they are not unlimited. Speculative decoding accelerates generation by reducing the amount of sequential decoding performed by the target model, not by bypassing the target model entirely. Even when many draft tokens are accepted, the larger model must still verify them.

As a result, throughput improves with acceptance rate but eventually saturates once verification overhead, draft-model cost, memory movement, and serving-stack constraints become the limiting factors.

Lessons from Production Implementations

Recent implementations show that speculative decoding is moving from research into production inference engineering. IBM's work on purpose-built speculator models highlights the value of training draft models specifically for short-horizon token prediction. Medusa takes a different path by adding multiple prediction heads directly to the target model, reducing sequential decoding without requiring a separate draft model.

The production challenge, however, is not limited to model design. vLLM shows that speculative decoding must be integrated into the inference server itself, including scheduling, batching, KV-cache management, and rejected-token handling. It is not simply a model wrapper that can be added on top of an existing serving stack.

AWS experiments using vLLM on Trainium-based infrastructure also point to the importance of workload structure. The strongest benefits appear in settings where prompts and outputs are more predictable, because constrained formats tend to improve acceptance rates. Red Hat’s Speculators project points to another adoption path: tooling for training and packaging custom draft models that are better matched to enterprise workloads.

Taken together, these examples show that speculative decoding works best when the draft strategy, workload, and serving stack are optimized together.

Where Speculative Decoding Works Best

Speculative decoding is highly workload dependent. It works best when the next tokens are relatively predictable and the draft model can propose continuations that the target model is likely to accept. Structured outputs, such as JSON, extraction, function calls, and templated responses, are strong candidates because they operate within narrower formatting and content constraints.

© Copyright Tolaga Research, 2026

Code generation can also perform well, especially for completion-style workloads where syntax, indentation, and local context provide strong guidance for the next-token distribution. By contrast, open-ended natural language tasks are harder because there are many plausible ways to continue a response, reducing the likelihood that draft tokens will match the target model's preferred output.

The best deployment candidates therefore have three characteristics: predictable outputs, low speculation overhead, and an inference stack that can efficiently manage batching, verification, rejected tokens, and KV-cache behavior. If these conditions are not met, the cost of drafting and verification can offset the gain from accepted tokens.

For enterprises, the strongest use cases are likely to be predictable, high-volume workflows rather than generic chatbots. Structured extraction, code completion, function calling, configuration generation, and templated document production are especially attractive because they combine repeatable prompt patterns with outputs that are easier to verify efficiently.

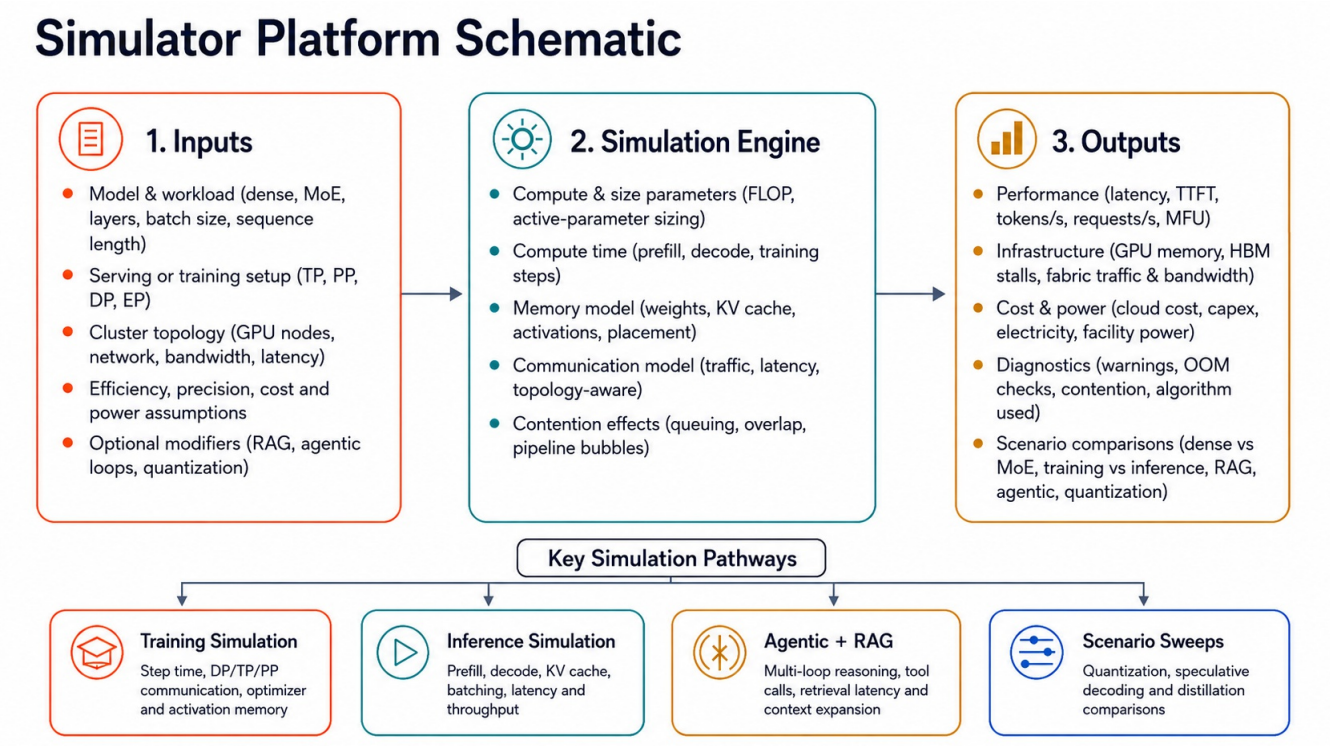
Where Speculative Decoding Improves Economics

Speculative decoding is best understood as part of a broader inference optimization stack alongside quantization, batching, caching, routing, and model-specific serving improvements. Its role is to reduce the cost of using a large model by allowing some of its sequential generation work to be anticipated by a faster drafting mechanism and then verified more efficiently by the target model.

For high-volume AI services, this distinction matters. Speculative decoding preserves the larger model in the serving path while improving its economics, making it especially relevant when quality requirements still justify using a frontier or near-frontier model. It does not replace the large model, but it can reduce the cost of using it when outputs are predictable, acceptance rates are high, and the serving stack is engineered to exploit accepted draft tokens.

© Copyright Tolaga Research, 2026

Tolaga AI Infrastructure Simulator



LEARN MORE

About the Author: Phil Marshall, PhD

Phil Marshall's work in AI began more than 30 years ago with Kohonen maps, discrete neural networks, machine learning, and fuzzy logic. More recently, his work has targeted AI-enabled solutions for the automotive, healthcare, manufacturing, and smart city sectors. Over the past decade, he has applied natural language processing and large language models to analyze online content and generate distinctive, data-driven industry research insights.

Marshall also brings deep experience in telecom, wireless, mobile, and networking technologies, fields he has worked in since the early 1990s. More recently, he has combined this infrastructure background with advanced software and modeling capabilities to develop Tolaga's AI Infrastructure Simulator, which is used to analyze the performance, cost, and scalability of emerging AI architectures.

Marshall is a dual citizen of the United States and New Zealand and currently resides in Auckland, New Zealand.

Contact: philip.marshall@tolaga.com

© Copyright Tolaga Research, 2026