Episode Book

EPISODE #6384

Why LLM Output Tokens Cost More Than Input

PUBLISHED	31 October 2026 at 15:45 UTC
DURATION	23:25
LISTEN	myweirdprompts.com/episode/blended-token-pricing-workloads-49591/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	anthropic/claude-sonnet-5.5
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.26
SOURCES	16 cited by the research (listed at the end)
BOOK GENERATED	1 November 2026 at 07:41 UTC

SYNOPSIS

Provider rate cards charge three to five times more for output tokens than input tokens, driven by the physics of memory bandwidth bottlenecks during decode. Comparing models using industry-standard three-to-one blends misrepresents actual costs, especially as agentic harnesses and retrieval-augmented generation skew workloads anywhere from twenty-two-to-one to over a hundred-to-one. Analysis explores roofline limits, hidden reasoning tokens, benchmark weighting, and why a single blended rate hides the primary cost drivers.

TRANSCRIPT



Corn

The second number is always the bigger one. Every rate card, every vendor, input on the left, output on the right, and the right one is a multiple of the left.



Herman

Three to five times, usually. Which is the whole reason Daniel's asking.



Corn

Right, so here's his version. Comparing models on a two-number card is dishonest unless you know your own input and output split. His idea is to turn that split into a single workload multiplier. If your traffic is eighty percent input and twenty percent output, you weight the two prices that way and get one blended number per model. He wants to know whether anyone has formalised that beyond the fixed three-to-one blend that benchmark sites publish. He also wants to know what real workloads look like, from chat to RAG to coding agents. And can the formula absorb prompt caching, so you end up with an effective cost per million tokens computed from your own logs, not a vendor's headline?



Herman

There's a lot of material on this, and most of it contradicts itself.



Corn

Naturally. Start with why two prices exist, because I'd assumed margin.



Herman

Most people do. It's not margin, it's physics. Serving a model has two phases. Prefill reads your entire prompt in one batched pass. All those tokens are processed in parallel, so the chip is compute-bound and busy. Decode then emits one token at a time, and each token is a full forward pass that has to stream every weight out of memory.



Corn

So the first phase is a sprint and the second is a very long queue at one door.



Herman

Here's the number that made it click for me. A seventy-billion-parameter dense model at BF16 on an H100 does prefill on a two-thousand-token prompt at roughly twenty-eight hundred prompt tokens a second. Decode on the same card tops out around ninety-six tokens a second per sequence. And compute utilisation during decode is about one point four percent.



Corn

I've napped with a higher duty cycle.



Herman

The chip is nearly idle in compute terms and completely saturated in memory terms. That's what you're paying for when you pay for output.



Corn

Though a rate card is a menu, not a quote. Two models can rank in opposite orders depending on your traffic, and the card won't tell you which order is yours.



Herman

Right. And that's why collapsing it into one number is tempting, and why several people have tried.



Corn

Before that, I want the roofline. You've got the look of a man with a balance point.



Herman

I do, and it's a good one. MAPIKA dot dev formalises it. Cost per token goes as two times the parameter count over how well you use the hardware. For a toy seventy-billion model on B200-class specs, that's a hundred and forty gigaflops per token, which is about sixty-two microseconds of compute. But reading a hundred and forty gigabytes of weights at eight terabytes a second takes seventeen and a half milliseconds.



Corn

So the arithmetic takes sixty-two microseconds and the fetching takes seventeen thousand.



Herman

Roughly two hundred and eighty times longer, and that's the balance point. You'd need a batch of about two hundred and eighty-one sequences decoding together before compute became the limit. Real decode batches sit far below that, so every token pays for a full sweep of the weights, shared among too few riders.



Corn

So it's a bus that has to drive the whole route every time, whether it carries three passengers or three hundred.



Herman

That's a fair picture. The fare per passenger drops as the bus fills, but the provider can't always fill it, because the people riding are waiting on their own answers and won't hold still for a bigger batch.



Corn

Fine, but you said the cost ratio is a hardware ratio. Then why is every card exactly five times?



Herman

Good catch, because MAPIKA says the same thing about itself. The line is that the roofline explains a cost pressure, not the whole card. Demand, margins, capacity, product strategy all move the posted price. The physics sets a floor and a direction. It doesn't set the number.



Corn

And five times on every card is more marketing than physics.



Herman

Look at the current listings. Sonnet five point five is two dollars in, ten out. Opus five point five is four and twenty. Haiku is ten cents and fifty cents. Fable five point one is ten and fifty. All five times, to the cent. MAPIKA sampled Anthropic, OpenAI and DeepSeek and found the ratio sitting between three and five, and Multigrid says usually three to five and occasionally far more.



Corn

So the hardware says something like a bandwidth tax, and the price list says round number.



Herman

Both can be true. Now, blending. Multigrid gives the cleanest statement. Blended price equals the input price times the input fraction, plus the output price times the output fraction, and the fractions sum to one. Take a model at two-fifty in and ten out. On a four-to-one workload, the input fraction is point eight, and it blends to four dollars a million. On a one-to-one mix, the same model blends to six twenty-five.



Corn

Same model, same card, a fifty-six percent difference in what I'd pay.



Herman

Their line is, the model did not change price, the workload did. And that's the whole legitimacy argument. If you're comparing two models, you hold the weights fixed and rank them. It's a weighted average. It's not clever, but it's honest.



Corn

It's also something I'd have written on the back of a receipt. What I want to know is whether anyone's done it properly or whether everyone just picks three-to-one.



Herman

Mostly the latter. Convly's price-performance index uses three times input plus output, divided by four, and reports a hundred and fourteen times spread across frontier APIs, eighteen cents to twenty dollars a million. DataLLM Lab ranks ninety-seven models by seventy-five percent input plus twenty-five percent output and calls it the ratio they see in typical production traffic. CloudQuell uses the same formula and calls it typical of RAG and chat, and then concedes that your real ratio changes the order.



Corn

So the benchmark site publishes the ranking and then footnotes that the ranking is wrong for you.



Herman

In small print, yes. MyTokenTracker is a bit different. It's an index over a fixed basket with a fixed ratio, on purpose, so the index measures price movement rather than model mix drift. That's a legitimate use of a fixed ratio. You want the thermometer to stay the same thermometer.



Corn

And the parameterised ones?



Herman

Multigrid's formula page, which is manual. You plug in your own fractions. And AI slash COST publishes an agent-shaped blend, mostly input and most of that cached, and notes that Artificial Analysis uses seven parts cache hits to two parts uncached input to one part output. Their caveat is the one to carry. A blend is good for ranking models on one kind of traffic and not good for predicting your bill if your mix differs.



Corn

Multigrid has a warning too, didn't you say?



Herman

That a single blended number hides the number you're actually optimising. The blend is a ranking device. The second you use it as a forecast, you've thrown away the one variable you control, which is the split.



Corn

Hm. So my instinct is, don't optimise the blend, optimise the thing inside it. Cut output tokens first.



Herman

Multigrid says that almost verbatim. A thirty percent cut in output length is worth several times a thirty percent cut in input length. Corn, you're the one who defends the slow option. Would you take a model that thinks longer and costs more per token if it got the answer right?



Corn

Anything worth computing is worth waiting for. But I'd want it priced per answer, not per token, because a model that thinks for ten minutes is just billing me for its nap.



Herman

You'd know.



Corn

I bill for mine at a flat rate.



Herman

And that per-answer point is where part of this goes next. Multigrid has an example where an extraction job at ten dollars in and thirty out, four thousand tokens in and four hundred out, cost five point two cents. A reasoning model at one dollar in and four out, emitting three thousand hidden reasoning tokens, cost one point seven six cents. Per-token prices fell ten times and seven and a half times, and the bill fell three times.



Corn

That's a strange result. Prices fell by close to an order of magnitude and the saving is a third of that.



Herman

Because the hidden tokens ate most of the discount. Twelve thousand dollars' worth of per-token cleverness, if you like, spent on thinking nobody sees.



Corn

Hidden reasoning tokens bill as output, so the output share isn't whatever the fixed mix says it is.



Herman

And that's something no fixed blend sees.



Corn

And here's a voicemail from Opal in Tel Aviv.
Opal: Hi Corn, hi Herman, this is Opal from Tel Aviv, I am actually calling from the queue at the pharmacy on Ibn Gabirol, they are very slow here so I have time to think about your show. So today is about the tokens, the input ones and the output ones, and I wanted to ask, when you do the blending thing with the percentages, does it matter which way round you write the ratio, because I always get confused if it is input to output or output to input, and my cousin who works in insurance says ratios are always written the wrong way round on purpose. Also I once spent a whole weekend in two thousand and one trying to work out the average price of a falafel across the whole city, weighting by how much I liked each place, and I never finished it, so I am very glad someone is doing this properly.



Herman

Write input to output and label it, Opal, so four to one means four input for every one output, and your falafel average was a weighted blend of exactly this shape, just with tastier weights.



Corn

Now, the part Daniel wanted most. What does a real ratio look like? Because I'd bet chat is somewhere near three to one.



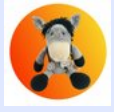
Herman

That's probably where the folklore came from, and I'll say upfront that nobody has published a measured per-workload table. What we have are fragments, and they pull in different directions. Three to one is asserted as typical of RAG and chat by one site and as typical production traffic by another. Someone on a forum cites four or five to one for general real-world mixes. Agentic harnesses run at twenty-two to ninety-eight to one.



Corn

Ninety-eight.



Herman

LiveBench publishes average input and output tokens per model in its cost snapshot. Claude Fable five point one is about five million input against fifty-two thousand output, ninety-eight to one. Sonnet five point five is eleven million against a hundred and fifty



Herman (continued)

thousand, seventy-five to one. GPT-6 Luna is fifty-two to one, Gemma four thirty-one-B is thirty-two, and Qwen three point eight Max is twenty-two. The least input-heavy model in the table is still about a hundred and twenty-one to one.



Corn

Hold on, that last one is least input-heavy at a hundred and twenty-one and Fable is ninety-eight? Those can't both be right.



Herman

No, you're right, I garbled that. The model with the highest ratio in the table is the one at a hundred and twenty-one, so that's the most input-heavy, not the least. Scratch my phrasing. The point stands, the whole table sits between twenty-two and the low hundreds.



Corn

Fine. And these are benchmark harnesses, so agents hammering a task with tool output and files.



Herman

Which is the point. They aren't chat, but they show how far real traffic can sit from three to one. Promptster's line is the one I like. The standard three-to-one blend describes an average workload, which is to say nobody's.



Corn

So let's reason it through, with the caveat that we're guessing from fragments. RAG?



Herman

Very input-heavy. Promptster's example is a pipeline stuffing eight thousand tokens of context to get a fifty-token answer. That's a hundred and sixty to one. The output rate barely matters to that bill. You could halve it and not notice.



Corn

Chat is the closest to balanced, though even there the history gets resent every turn, so input piles up.



Herman

Yes, my guess is chat is higher than three to one once conversations run long, but I wouldn't swear to it. Think of a twenty-turn conversation. Turn twenty resends all nineteen earlier exchanges, so the input on that single turn dwarfs the short reply coming back. Long-form generation is the reverse of RAG. A short brief and a long essay, so output dominates and the input rate is a rounding error. Summarisation is input-heavy, a long document in and a short summary out. Classification, I'd say, is near balanced or even output-light, a label out for a paragraph in, although it depends how much you stuff in the prompt as instructions.



Corn

And coding agents are the extreme end.



Herman

Cache reads make up eighty to ninety percent of tokens in a real agentic task, per one practitioner. An agent re-reads the same files and the same conversation every step, so almost everything is repeated input.



Corn

That has a consequence for three to one, doesn't it? If agents sit at twenty-two to ninety-eight, three to one is over-weighting output.



Herman

Systematically, and in a specific direction. It flatters models with cheap output and penalises models with cheap input. That's a bias baked into published rankings, not a neutral average.



Corn

So the rankings aren't neutral, they have an accent.



Herman

And caching makes it worse, because caching breaks the two-number model harder than output pricing does. Multigrid extends the formula. Effective input equals the input price times the uncached fraction, plus the input price times the cache discount times the cached fraction. At two-fifty a million, eighty percent cached, with cache reads at ten percent of full price, you get two-fifty times point two plus two-fifty times point one times point eight. Seventy cents a million.



Corn

Seventy-two percent off, from a feature nobody puts on a price chart.



Herman

That's their phrasing, nearly. Now the write side, which most blends skip. A cache miss is priced at the cache-write rate when the vendor bills writes, because a miss is the request that writes the prefix again. So low-hit-rate traffic can cost more than no caching at all. The write premium is a real cost, not a free option.



Corn

So the discount has a deposit.



Herman

Look at Anthropic's card for Sonnet five point five. Two dollars in, ten out. Five-minute cache writes are two-fifty, one point two five times input. One-hour writes are four dollars, twice input. Hits and refreshes are ten cents, point oh five times input. Opus five point five is four and twenty, with writes at five and eight and hits at twenty cents. Fable is ten and fifty, with hits at twenty-five cents, which is two and a half percent of input.



Corn

Four meters on a two-number card.



Herman

Default time-to-live is five minutes, refreshed free on each use. And the clock counts streaming time. If a response takes four minutes to stream, the follow-up that reuses the prefix has to start within about a minute.



Corn

So slow answers cost you your cache. I find that unfair, personally.



Herman

Corn, would you pay double for the hour-long version, if your usage were bursty? I honestly can't tell which way you'd go.



Corn

Bursty usage with a premium for patience? I'd take the five-minute one and gamble that I'm back in time. Which is how everything I own got cold.



Herman

Then you'd take the loss. At low hit rates, the two-times write is the one that stings.



Corn

So what's the recipe? Daniel wants to compute it from his own logs.



Herman

It's short. Pull the four usage fields per request: input tokens, cache read tokens, cache creation tokens, output tokens. Multiply each by its own rate, sum, and divide by total tokens. That's your effective cost per million, and it's measured, not quoted. Then redo it monthly, because the mix drifts every time you add an agent.



Corn

And this is why the headline price won't match the bill.



Herman

Promptster ran eleven model configurations on identical tasks. The rate card predicted the wrong winner five times out of five. Kimi K3 charges fifteen dollars a million output, half of GPT-5.6 Sol's thirty, and cost fifty-four percent more because it emitted three point one times the tokens. Gemini three point six Flash charges more on both rates than GPT-5.6 Luna, and cost thirty-two percent less because Luna used one point nine times the tokens.



Corn

So the cheap model talked more.



Herman

Output counts on the same three questions ranged from two hundred and eighty-six to seventeen hundred and fifty-one. That's a sixfold spread, set against a three-to-five-times price ratio. Verbosity is a bigger lever than price. And repeat runs of Claude Opus five on the same battery varied twenty-six percent, which sets a floor. Any comparison that hinges on a gap under about twenty-five percent is unresolved.



Corn

So if I rerun the same prompt tomorrow, I might get a bill a quarter different, and that's the same model on the same day's price.



Herman

Which is why one run proves nothing. You want a battery, repeated, before you believe a ranking.



Corn

That reminds me of the old fight over transcription scores. A headline error rate told you very little about whether the transcript was any use.



Herman

Same gap, between a headline metric and the thing you care about. So I'd say the blend is a ranking tool and the effective cost is a log-derived measurement. They aren't the same object. AI slash COST says it plainly. If your cache hit rate is low, your answers long, or your prompts cross a long-context threshold, your effective rate will differ from any published blend, sometimes by a multiple.



Hilbert

Typical. I'd argue with the word. Nobody I know has a typical anything until they've counted.



Herman

Fair, Hilbert, though that's the three-to-one claim exactly. It's repeated until it sounds like a measurement.



Hilbert

Mr. Barak downstairs was quoted a copier lease this week. Per-page rate for black and white, a much higher one for colour. The salesman assumed three to one. I took Mr. Barak's receipts home, thirteen months of them, and counted. Eleven to one. He's been overpaying since the last lease. The counting is in a ring binder, section four, and I turn the pages with one claw because the rings weren't made for the rest of me.



Corn

Eleven to one, and the salesman's three was the number on his own sheet, so of course he picked it.



Herman

It's the same shape of error as a bad cache assumption. A wrong ratio on a two-rate card inflates the bill for years and nobody checks.



Hilbert

The copier went back to the lessor in March. I don't use the lease, I use the binder. I won't count my own model usage, mind. The logs are the vendor's logs, and I don't trust a man to count his own till.



Herman

Hilbert, you can export the usage fields per request from the API side, though. You'd be multiplying them yourself.



Hilbert

Herman said earlier the mix drifts monthly. I'm going to recount Mr. Barak's in December. Dalia loads his paper tray every morning at ten past eight, and she says the colour toner goes twice as fast as the black. Eleven to one is a hundred and forty-four pages of colour a month.



Corn

Which means the salesman's assumption is probably wrong in the other direction on the toner.



Herman

Well, the pages-per-cartridge differ, so possibly. The useful thing from the binder is that the number came from receipts, not from a sheet.



Corn

There's a version of Daniel's question where that's the entire answer. Count first, then argue.



Herman

And that matters for the closing, because there's another place the number gets lost. KickLLM found two independent public price sources disagreeing on the input price for the same first-party model, with the widest gap at two times on GPT-5.6 Sol. So even the card isn't always the card.



Corn

The one number I thought I had, the input price, has an argument going on inside it.



Herman

In at least one case, yes.



Corn

Here's what I'm left with. If three to one over-weights output, how much of the published ranking is just a ratio nobody measured? The hundred and fourteen times spread is real, but the order inside it may not be.



Herman

Caching makes effective cost a function of your architecture, not your model choice. The unit may be shifting from price per million tokens to price per task, and nobody has a standard for that yet. The model did not change price, the workload did, and the rate card was never what you were buying.



Corn

Thanks to our producer, Hilbert Flumingtop, for keeping the desk running.



Herman

If you want more of this, try episode nine, Benchmarking Custom ASR Tools - Beyond The WER; and episode five, Fine-Tuning ASR For Maximal Usability. This has been My Weird Prompts. If you've enjoyed it, a review helps more people find the show.



Corn

See you tomorrow.

SOURCES

What the research for this episode read before the script was written. Primary sources first.

1. **Anthropic docs (cache write/read multipliers, TTLs, per-model table) [primary]**
   https://docs.anthropic.com/en/docs/build-with-claude/prompt-caching
2. **Multigrid, updated 2026-08-03 (prefill/decode economics, 1.4% utilisation)**
   https://multigrid.ai/learn/input-vs-output-tokens
3. **Multigrid, updated 2026-08-04 (blended formula, caching extension, price-per-task)**
   https://multigrid.ai/learn/ai-price-history
4. **MAPIKA.DEV, published 2026-06-09, reviewed 2026-09-05 (roofline model, = K*/K)**
   https://mapika.dev/essays/the-price-is-the-physics
5. **AI//COST glossary (agent-shaped blend, Artificial Analysis 7:2:1, cache-write treatment)**
   https://aicost.tools/glossary/blended-rate/
6. **Promptster, 2026-08-19 (measured rate-card inversions, verbosity spread)**
   https://www.promptster.dev/blog/blended-token-cost-how-to-calculate
7. **LiveBench cost data, 2026-10-25 snapshot (avg input/output tokens per model)**
   https://livebench.ai/data/20261025/cost.csv
8. **OpenRouter listing, fetched 2026-10-31 (current list prices)**
   https://openrouter.ai/anthropic/claude-sonnet-5.5
9. **Convly AI Price-Performance Index 2026 (114× spread, 3:1 blend)**
   https://convly.ai/ai-price-performance-index-2026/
10. **DataLLM Lab, export 2026-06-12 (75/25 blend, 97 models)**
    https://www.datallmlab.com/blog/cheapest-llm-api-2026.html
11. **CloudQuell, prices as of 2026-10-12 (3:1 blend rationale)**
    https://www.cloudquell.com/blog/cheapest-llm-api/
12. **MyTokenTracker methodology (AI Cost Index, fixed-ratio basket)**
    https://mytokentracker.io/methodology

13. **KickLLM, 2026-08-24 (2× price disagreement on GPT-5.6 Sol)**
https://kickllm.com/research/llm-price-source-disagreement.html

14. **HN, 2026-05-19 (4:1**
5:1 real-world ratio critique)
https://news.ycombinator.com/item?id=48189375

15. **HN, 2026-07-24 (cache = 80**
90% of agentic tokens; Grok 4.5 cache price cut)
https://news.ycombinator.com/item?id=49040667

16. **HN, 2026-10-19 (DS vs GLM output-heavy workload comparison)**
https://news.ycombinator.com/item?id=50157697