Small steps on the default kernel: more turns at 32 streams, a step every eight seconds above

The mixed load on the default kernel with room for 2,048 tokens in an engine step instead of 8,192. At 32 streams the engine makes three steps in four seconds. Above that the long prompt being read still fills every step: no more than 32 requests run while the load is on.

Qwen3.8-27B · one AMD Instinct MI300X · vLLM 0.27.1 **Run** 20261125T214036Z-series-batch2k
Changed 2,048-token engine steps instead of 8,192, against the series run
Load 70% short, 25% medium, 5% long prompts; 8, 32, 128, 256 streams

If a step that carries a long prompt's bite is slow, make the bite smaller. I gave the engine step room for 2,048 tokens instead of 8,192 and played the same mixed plan. At 32 streams the engine made 0.76 steps a second and the server wrote 18.8 answer tokens a second. Above that it made a step every 8 to 11 seconds, and the prompt being read still took all the room: until the plan stopped sending, no more than 32 requests ran.

0.76
engine steps a second at 32 streams; 0.12 at 128, 0.09 at 256

18.8
answer tokens a second at 32 streams; 8.9 with the long-prompt cap instead, the like-for-like pair

32
requests running at most at 128 and 256 streams, as with large steps

181.3 s
until the first long prompt's first token; the same prompt took 83.9 s with large steps

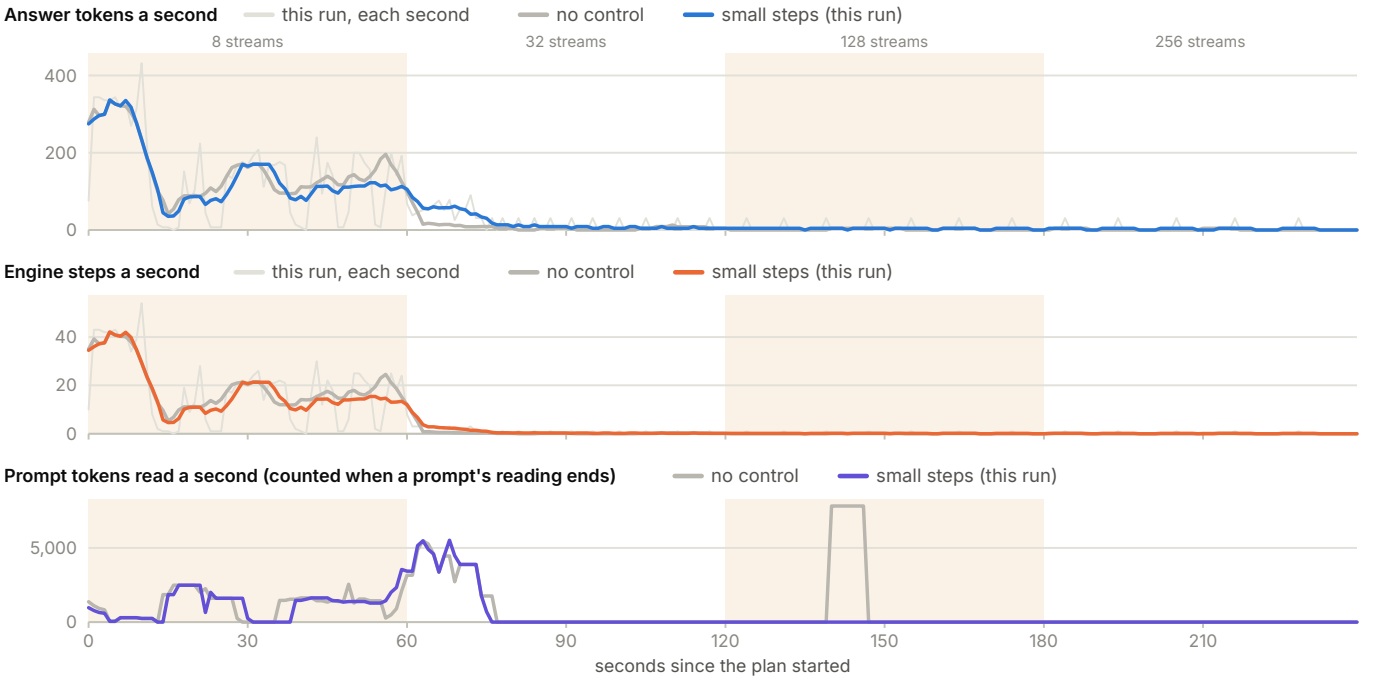


Figure 1 The four loaded minutes. Colour: this run, with its single seconds thin and grey behind the 7-second centred mean. Grey line: the same plan on the same kernel without the control. Small steps give more engine steps and more answers in the second minute. In the third and fourth both runs lie flat.
Source: plan-mixed-overlap/timeline.json of this run and of 20261125T201346Z-series

MEASURED

At 128 and 256 streams the server wrote 3.7 and 2.7 answer tokens a second, with 96 and 224 requests waiting at most. With large steps it wrote 1.6 and 1.1. In the two minutes after the last request was sent small steps did better: 107.4 answer tokens a second against 14.2.

KEEP IN MIND

One setting differs from the grey run. At 8 streams both were sent the same 21 requests. After that a closed loop sends each server its own: at 32 streams this run was sent one long prompt and the grey run two, so I write no ratio there. At that stair this run pairs with the other control, which was also sent one.

The machine, and the part that was changed

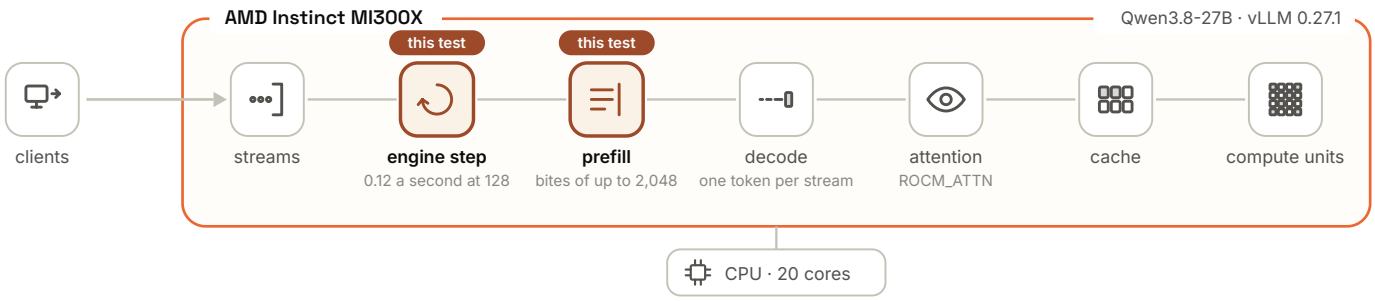


Figure 2 The card as every document in this series draws it. An engine step carries one new token for every stream that is answering and a bite of a prompt that is being read. This run gives the step a quarter of the room.
Source: FACTS.json; summary.json config and plans.plan-mixed-overlap.stages[stage=mix-128]

What differs between the two runs

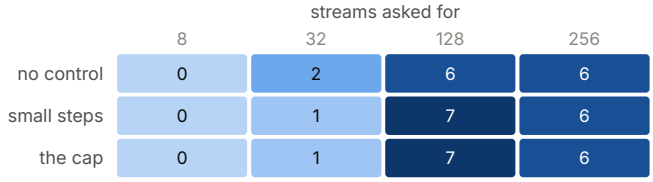
Room in one engine step	2,048 tokens here; 8,192 in the run before
Long-prompt cap	none in both
Attention kernel	ROCM_ATTN in both, by vLLM's own preference order
Cache	1,831,905 token slots here; 1,826,476 before
The run before	20261125T201346Z-series

The plan

1. Four stages of 60 seconds: 8, 32, 128, then 256 requests kept open. A new one starts only when one ends.
2. Each request draws its prompt from a mix: 70% short (256 to 2,000 tokens), 25% medium (4,000 to 16,000), 5% long (32,000 to 64,000); answers of 64 to 1,024 tokens.
3. Then 120 seconds with no new requests; what is still open after that is closed by the load generator.

Settings

Model	Qwen3.8-27B: 27.8 billion parameters, BF16 weights, 262,144 tokens of context
Card	one AMD Instinct MI300X: 304 compute units, 191.7 GiB of memory
Server	vLLM 0.27.1+rocm723, ROCm 7.2.3, PyTorch 2.11.0+gitd0c8b1f
Machine	a cloud GPU virtual machine: 20 virtual cores, Intel Xeon Platinum 8568Y+
This run	20261125T214036Z-series-batch2k, started 2026-11-25T21:40:49Z, server ready after 182 s



darker = more: long prompts sent in that minute

Figure 3 Long prompts sent in each minute, in the three runs on the default kernel. Rows that agree carried a like load in that minute. At 32 streams the two controls agree with each other, not with the run without one.
Source: plan-mixed-overlap/requests.jsonl of the three runs (stage, asked_in)

KEEP IN MIND

The plan is a closed loop: a new request is sent when one ends, so the moment each is sent depends on the server. Request number N has the same lengths in every run. "Ended inside the plan" is a fact about each run, not a ratio between runs.



Figure 4 What one engine step holds in the two runs. The time is computed: one second divided by the engine steps a second at 128 streams.
Source: each run's summary.json (config, plans.plan-mixed-overlap.stages[stage=mix-128].engine_steps_per_s); FACTS.json

Minute by minute, with and without the control

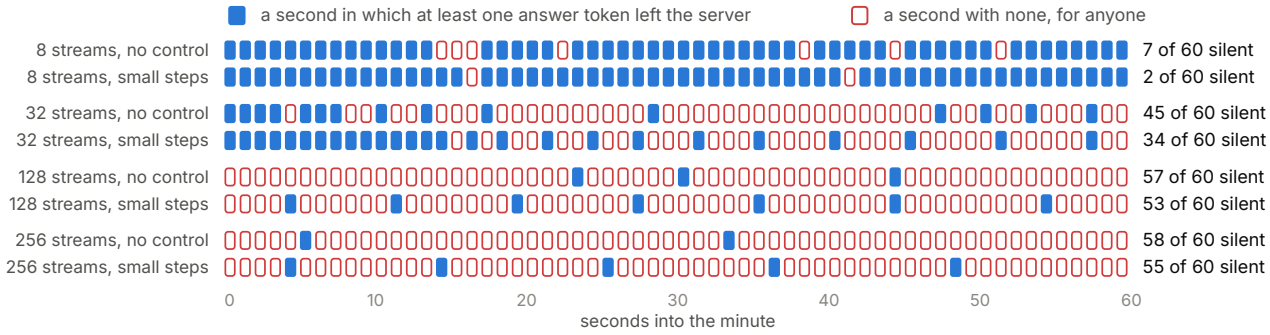


Figure 5 Every second of the four loaded minutes in both runs. The first minute is the same 21 requests in both. The rows at 32 streams are not like for like: one long prompt here, two in the other run. With small steps the second minute has more seconds with an answer token; the third and fourth hardly any.

Source: plan-mixed-overlap/timeline.json of both runs (output_tok_s per one-second row)

Answer tokens a second, all streams together

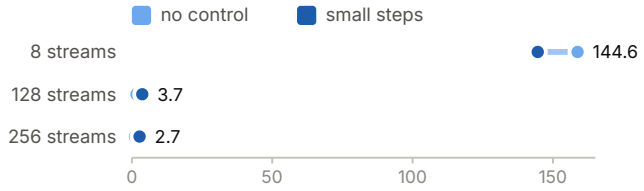


Figure 6 Answer tokens a second in the three minutes in which both runs were sent about as many long prompts.

Source: both runs' summary.json stages[].server_output_tok_s

Engine steps a second (log scale)

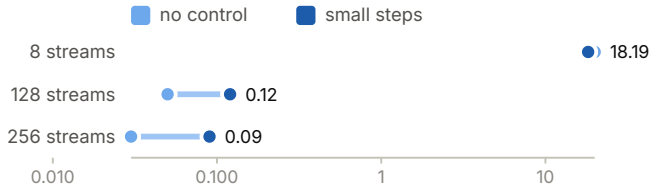


Figure 7 Engine steps a second in the same minutes. Every running stream gets one token per step.

Source: both runs' summary.json stages[].engine_steps_per_s

The exact record of both runs

	ANSWER TOKENS/S	PROMPT TOKENS/S	ENGINE STEPS/S	SECONDS WITH NO ANSWER TOKEN	RUNNING AT MOST	WAITING AT MOST	CACHE AT MOST, %	FIRST TOKEN, MEDIAN S	REQUESTS SENT	CARD, W
8 STREAMS										
no control	158.8	1,066	19.91	7 of 60	8	0	3.7	0.69	21	413
small steps	144.6	1,033	18.19	2 of 60	8	0	3.7	0.49	21	405
32 STREAMS										
no control	6.3	1,104	0.24	45 of 60	32	18	13.5	5.57	25	505
small steps	18.8	1,060	0.76	34 of 60	32	19	12.0	5.64	24	432
128 STREAMS										
no control	1.6	943	0.05	57 of 60	32	96	14.9	≥ 211.93	96	366
small steps	3.7	0	0.12	53 of 60	32	96	12.8	≥ 172.38	96	312
256 STREAMS										
no control	1.1	0	0.03	58 of 60	32	224	15.8	≥ 179.91	128	303
small steps	2.7	0	0.09	55 of 60	32	224	13.3	≥ 179.90	128	305
DRAIN: NO NEW REQUESTS										
no control	14.2	2,204	0.24	93 of 121	96	224	37.8	-	0	359
small steps	107.4	3,000	1.41	20 of 121	111	223	41.6	-	0	460

From each run's `summary.json`; seconds with no answer token are counted from its `timeline.json`. Tinted rows are this run. A stage row spans 59 s of its 60 s. The server counts a prompt's tokens when its reading ends. Running and waiting are the server's gauges, which move only when a step ends: with steps this long they are often old.

Like for like at 32 streams: the two controls, one long prompt each

AT 32 STREAMS	ANSWER TOKENS/S	ENGINE STEPS/S	SECONDS WITH NO ANSWER TOKEN	FIRST TOKEN, MEDIAN S	RUNNING AT MOST
small steps	18.8	0.76	34 of 60	5.64	32
the cap	8.9	0.33	40 of 60	4.31	32

Four times the steps for the same prompt, and nothing else gets in

	ROOM IN A STEP, TOKENS	CAP ON ONE PROMPT PER STEP	THE PROMPT, TOKENS	SENT AT SECOND	ITS FIRST ANSWER TOKEN AFTER, S	ENGINE STEPS IN BETWEEN	ANSWER TOKENS WRITTEN MEANWHILE, FOR EVERYONE	THAT IS, A SECOND
no control	8,192	none	54,698	60	83.9	15	395	4.7
small steps	2,048	none	54,698	60	181.3	60	1,591	8.8
the cap	8,192	2,048	54,698	60	119.0	33	1,091	9.2

The first long prompt of the plan is the same request in every run and was sent in the same second. Steps and answer tokens are counted from each run's `timeline.json` between its sending and its first token (computed). The tinted row is this run.

Seconds per engine step: no control

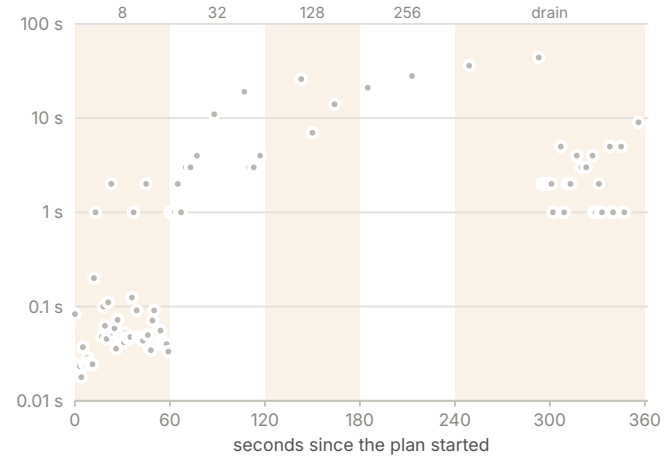


Figure 8 Without the control: from the moment a long prompt is read, a step takes seconds, then tens of seconds.
Source: timeline.json of 20261125T201346Z-series (steps_s)

Seconds per engine step: small steps

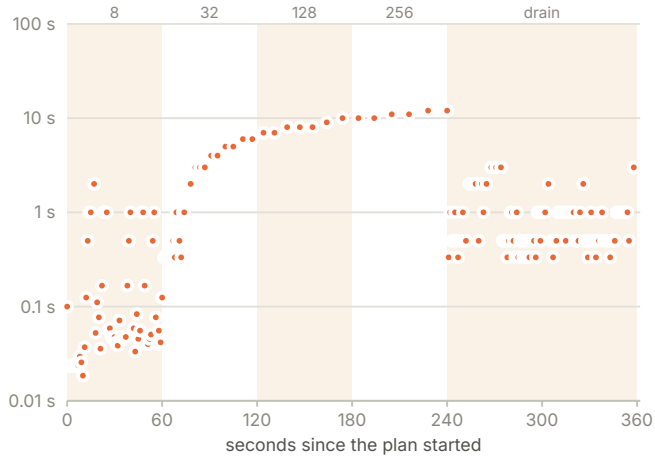


Figure 9 With small steps: while the first long prompt is read, from second 73 to 241, each step takes as long as the one before or longer, from 1 second to 12. In the drain a step takes half a second at the median. Computed from the step counter: one second divided by the steps in it, or the seconds since the last step.
Source: plan-mixed-overlap/timeline.json (steps_s)

MEASURED

The same prompt of 54,698 tokens, sent in the same second: with small steps its first token came after 181.3 s, with large steps after 83.9 s. Meanwhile the engine made four times the steps, each as long as the one before or longer, and the server wrote about twice the answer tokens a second for everyone: still under ten.

COMPUTED

A prompt of 54,698 tokens is 26.7 bites of 2,048 and 6.7 bites of 8,192. Its reading took 2.1 times as long.

READING, NOT MEASUREMENT

My reading: a smaller step gives the answering streams more turns, but the prompt being read needs four times as many of them, and on this kernel each still takes long. While it is read it takes all of every step, so nothing else is let in: the server never ran more than 32 requests before the drain.

KEEP IN MIND

The size of a bite is the server's setting. Its "tokens per step" counter is not the bite: it counts a whole prompt in the step that finishes reading it.

What the clients saw

Every request: no control

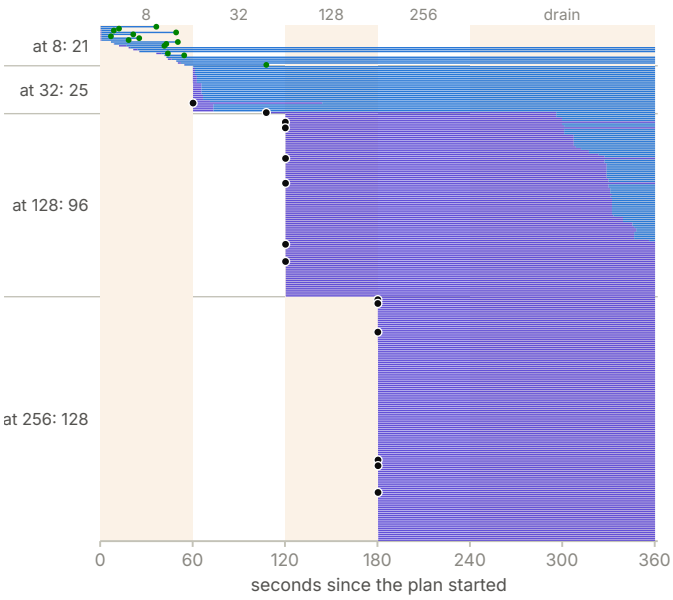


Figure 10 Every request of the run without the control as one line, in the order sent. Violet: no answer token yet. Blue: answering. Green dot: ended well. Black dot: a long prompt.
Source: requests.jsonl of 20261125T201346Z-series

Every request: small steps

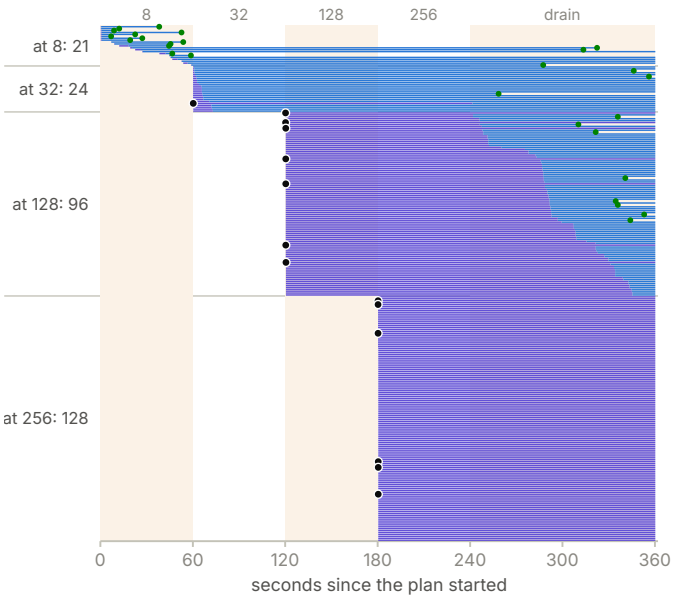


Figure 11 The same for this run. The requests sent at 32 streams answer a little faster, and a few of them end. Those sent at 256 streams wait as before.
Source: plan-mixed-overlap/requests.jsonl (t_sent, t_first, t_end, end)

	REQUESTS SENT	LONG PROMPTS AMONG THEM	FIRST TOKEN, MEDIAN S	LONGEST, S	ENDED INSIDE THE PLAN	CUT WHILE ANSWERING	CUT BEFORE A FIRST TOKEN
SENT AT 8 STREAMS							
no control	21	0	0.7	5.0	14	7	0
small steps	21	0	0.5	6.1	16	5	0
SENT AT 32 STREAMS							
no control	25	2	5.6	188.3	0	25	0
small steps	24	1	5.6	181.3	3	21	0
SENT AT 128 STREAMS							
no control	96	6	209.8	236.0	0	63	33
small steps	96	7	171.8	225.0	8	81	7
SENT AT 256 STREAMS							
no control	128	6	-	-	0	0	128
small steps	128	6	-	-	0	0	128
ALL REQUESTS OF THE PLAN							
no control	270	14	187.2	236.0	14	95	161
small steps	269	14	166.1	225.0	27	107	135

From each run's `requests.jsonl`. Tinted rows are this run. The median and the longest first-token time are over the requests that got a first token. The client takes the first piece of visible text as the first token, so with steps seconds apart a first-token time can be a step or two late.

MEASURED

With small steps 11 of the 248 requests sent after the first minute ended inside the plan. None of the 128 sent at 256 streams got a first token, and 7 of the 96 sent at 128 streams did not.

NOT A RESULT

Requests "cut at the end of the plan" (242 here) are not refusals and not failures: the load generator closed them when the plan's time was over. All 269 requests got HTTP 200.

The card waits here too

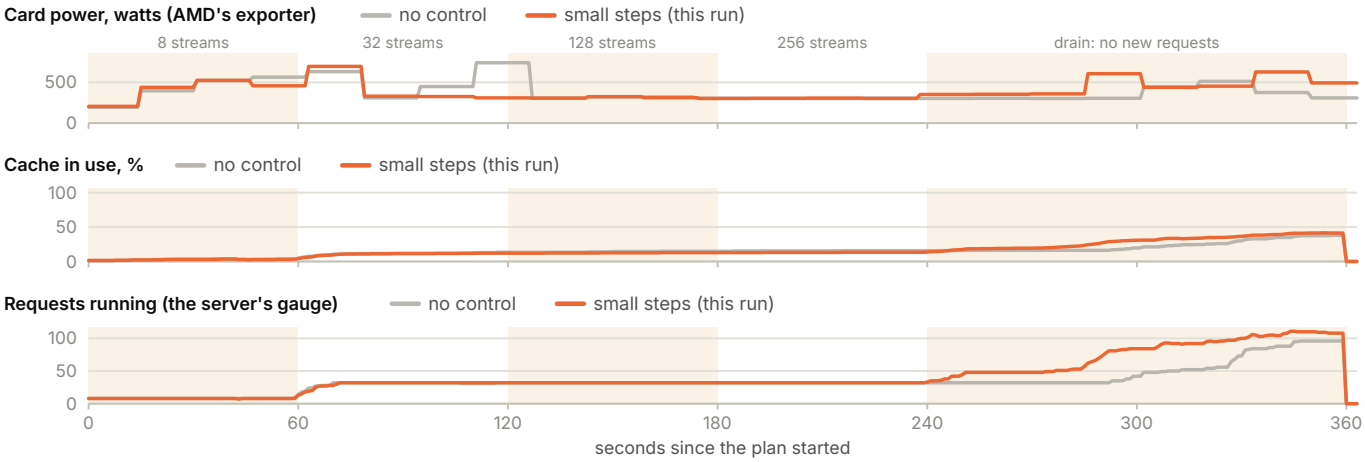


Figure 12 Both runs over the whole plan. Grey: without the control. Colour: this run. Power is AMD's device exporter, which refreshes every 11 to 16 s; cache and running are the server's gauges.
Source: plan-mixed-overlap/timeline.json of both runs (watts, cache_percent, running)

MEASURED

At 128 and 256 streams the card drew 312 and 305 W, with its matrix units 0.2% and 0.2% in use. With large steps: 366 and 303 W.

MEASURED

In the drain the engine made 1.41 steps a second with small steps and 0.24 with large ones, and up to 111 requests ran.

What I take from it

- ● At 32 streams small steps give 0.76 engine steps a second and 18.8 answer tokens a second; the cap, in the like-for-like minute, 0.33 and 8.9.
- ● Above 32 streams the engine makes a step every eight to eleven seconds and, while the load is on, runs no more than 32 requests.
- ● The long prompt itself is read more slowly: 181.3 s instead of 83.9 for the same prompt.
- ◌ Smaller steps mean more turns for the streams and more steps for the prompt; on this kernel each step still takes long.

Not results, and limits

- At 32 streams this run and the run without the control were not sent the same long prompts; no ratio is written there.
- A closed loop: "ended inside the plan" is a fact about each run. The bench has a second plan with the same requests at the same moments for every server.
- Default attention kernel throughout. Prompts are random tokens; nothing here is about the quality of an answer.

● measured: read from a run file · ○ computed: arithmetic on measured values · ◑ estimate: vLLM's, from the model's shape · ◌ my reading: interpretation, not measurement

Words used

Engine step
one pass of the model over a batch: one new token for every decoding stream, plus a bite of prefill.

Chunked prefill
a long prompt is read in bites across several engine steps, at most `max_num_batched_tokens` (8,192 here by default) per step for all requests together.

Every number in this document was read from these files at build time (sha256, first 12 characters):

`FACTS.json` 160f5ca0d495 · `GLOSSARY.md` 98323be796af · `plans/mixed-overlap.json` b86c1f4ef044 · `runs/20261125T201346Z-series/plan-mixed-overlap/requests.jsonl` ec7778099e60 · `runs/20261125T201346Z-series/plan-mixed-overlap/t0` f79227400789 · `runs/20261125T201346Z-series/plan-mixed-overlap/timeline.json` 328dd803edea · `runs/20261125T201346Z-series/summary.json` aab3940eb12e · `runs/20261125T213149Z-series-longprefill2k/plan-mixed-overlap/requests.jsonl` 774120396827 · `runs/20261125T213149Z-series-longprefill2k/plan-mixed-overlap/t0` c945be467c45 · `runs/20261125T213149Z-series-longprefill2k/plan-mixed-overlap/timeline.json` 37662cd0f155 · `runs/20261125T213149Z-series-longprefill2k/summary.json` 110822c67acb · `runs/20261125T214036Z-series-batch2k/plan-mixed-overlap/requests.jsonl` d80dc5271519 · `runs/20261125T214036Z-series-batch2k/plan-mixed-overlap/t0` fffb5e8db961 · `runs/20261125T214036Z-series-batch2k/plan-mixed-overlap/timeline.json` ced017b86f78 · `runs/20261125T214036Z-series-batch2k/summary.json` 3d4fa8438e60

90 values bound by path, 12 figures drawn from data. built 2026-11-26 02:39Z, explainer-pdf-system v2.

CREATED AND DIRECTED BY



Peter Styk

HPC & AI Platform Engineer

GPU inference, data and performance

linkedin.com/in/peterstyk · github.com/styk-tv

Questions about a run? Connect on LinkedIn.

LinkedIn