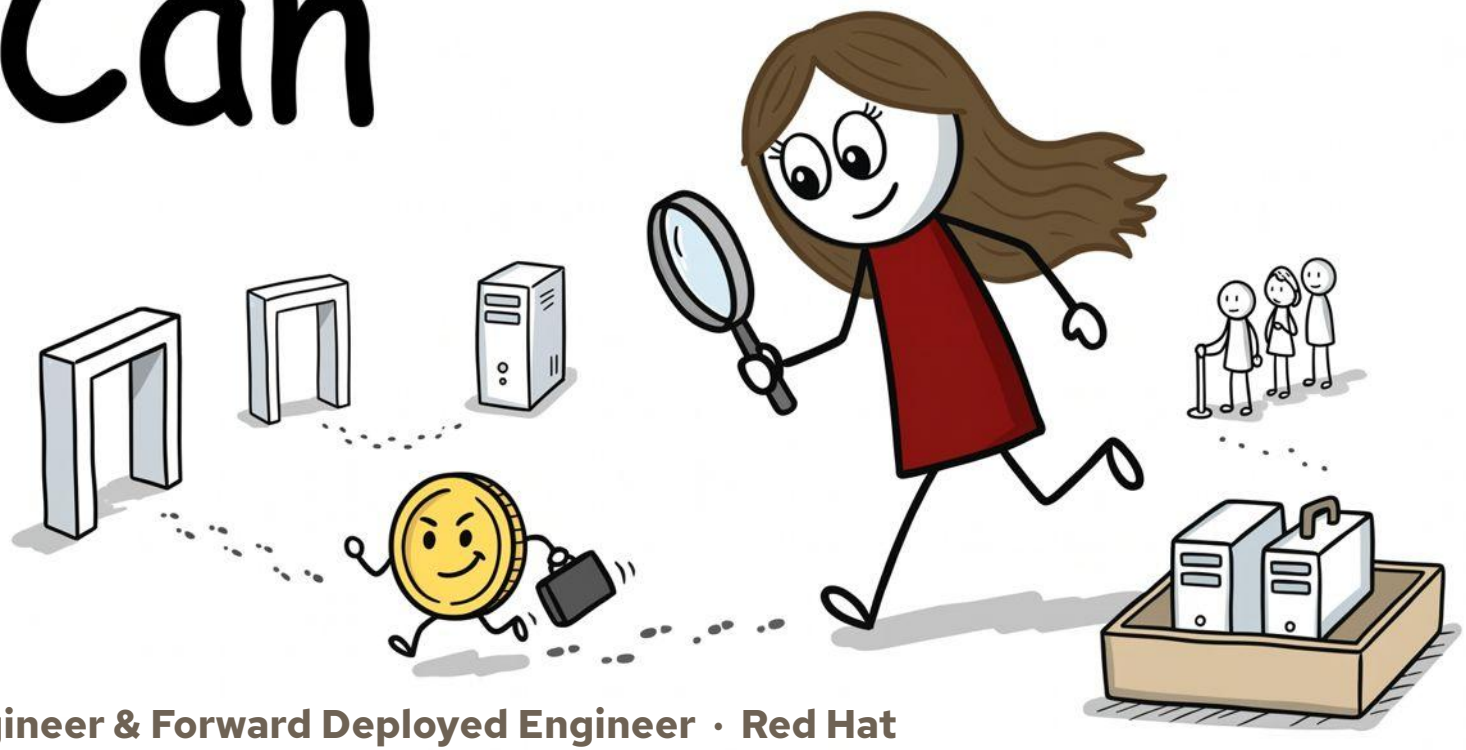
Cache Me If You Can



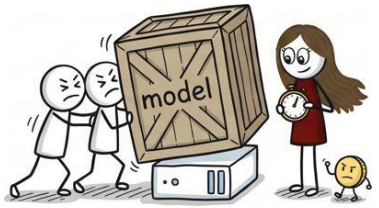
LLM Inference with llm-d

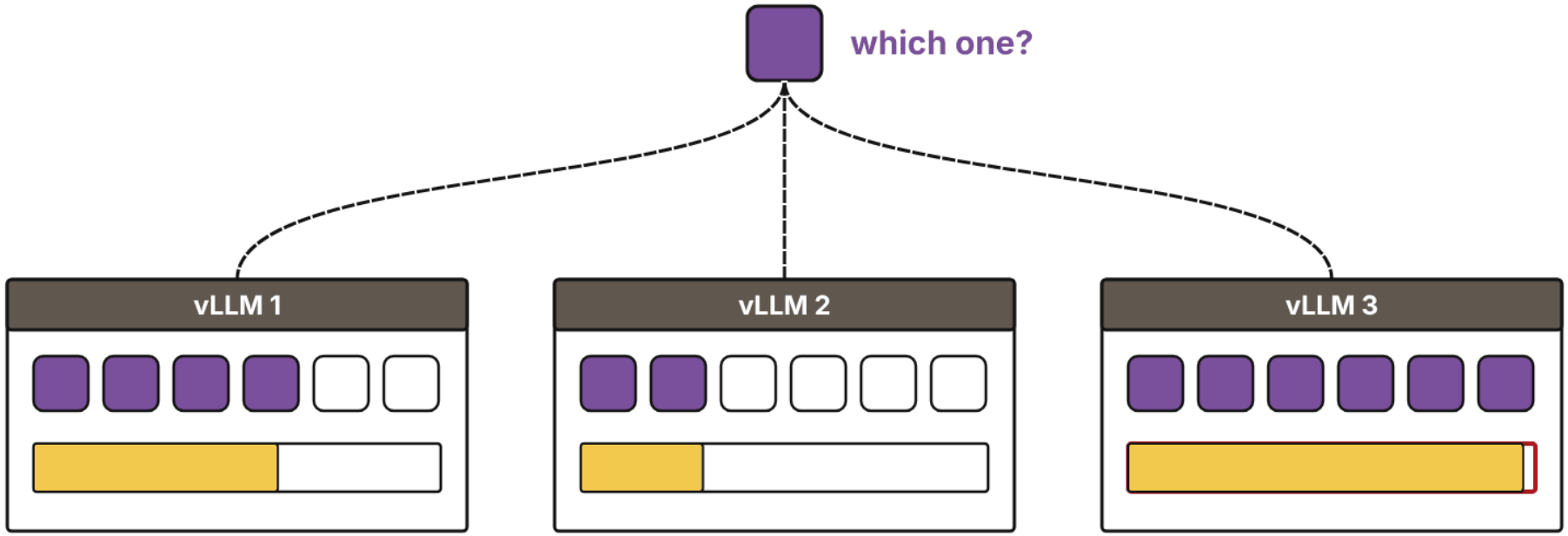
Alexa Griffith · Senior Principal Software Engineer & Forward Deployed Engineer · Red Hat

Where this started

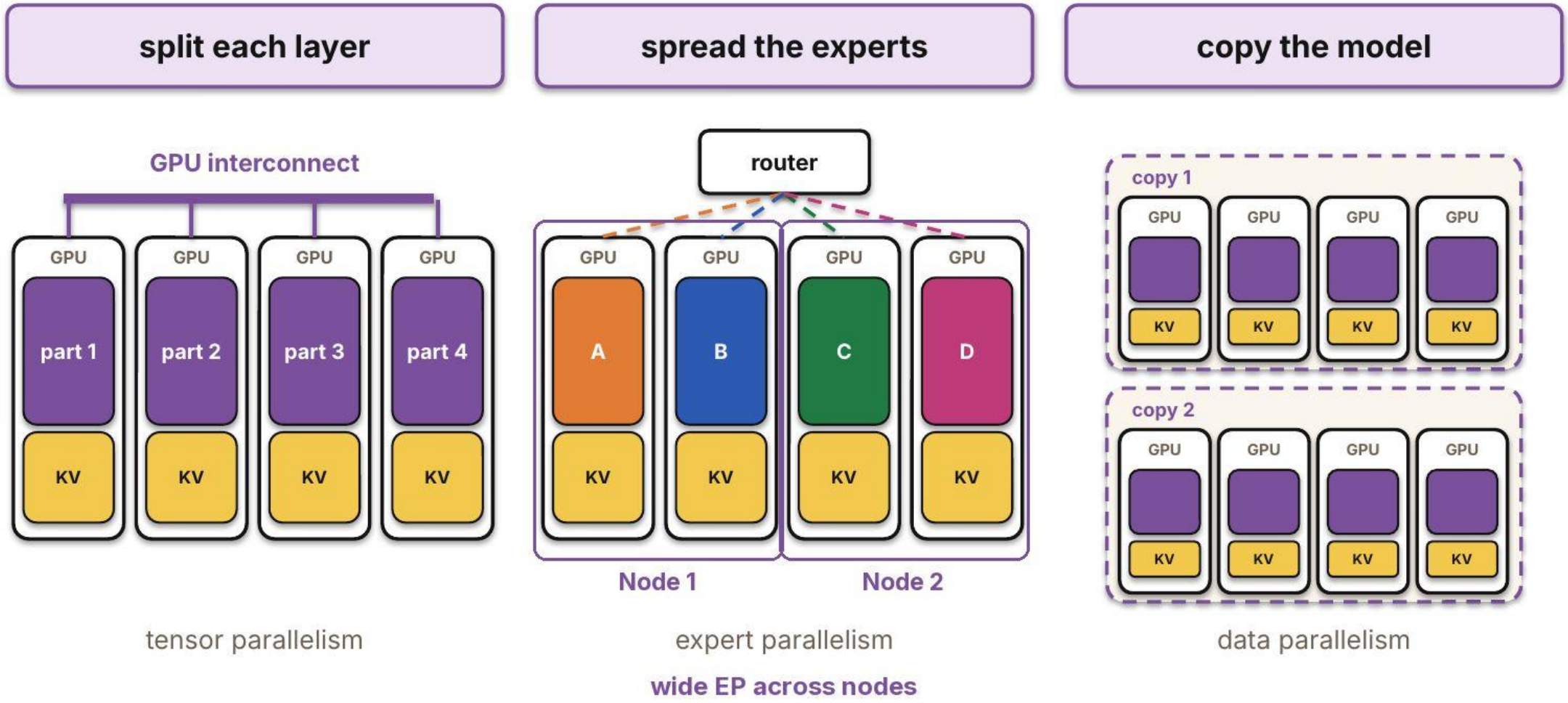


Scaling inference





Scaling inference



LLM workloads



HTTP assumptions that LLM requests break

multi-turn

the same growing prefix
is sent again and again

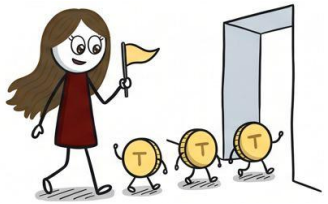
slow

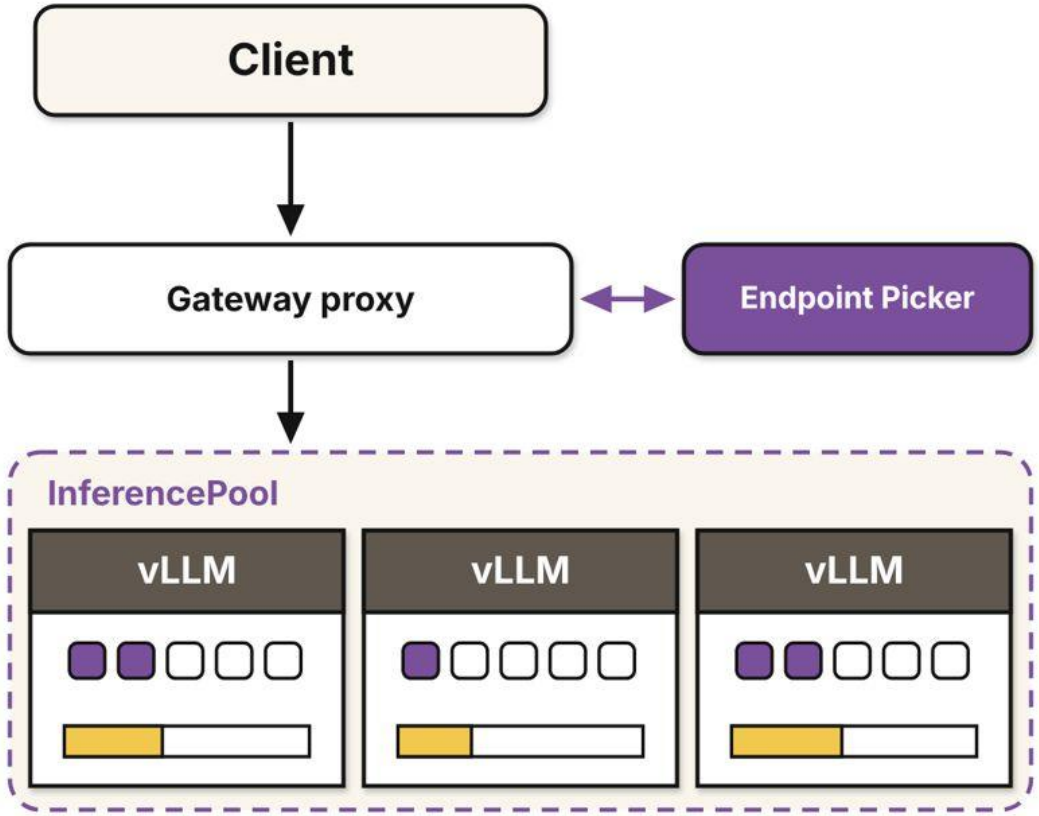
one request can stream
tokens for over a minute

non-uniform

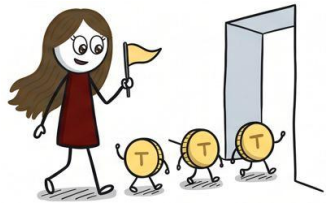
from about 1k tokens
to 100k+ tokens

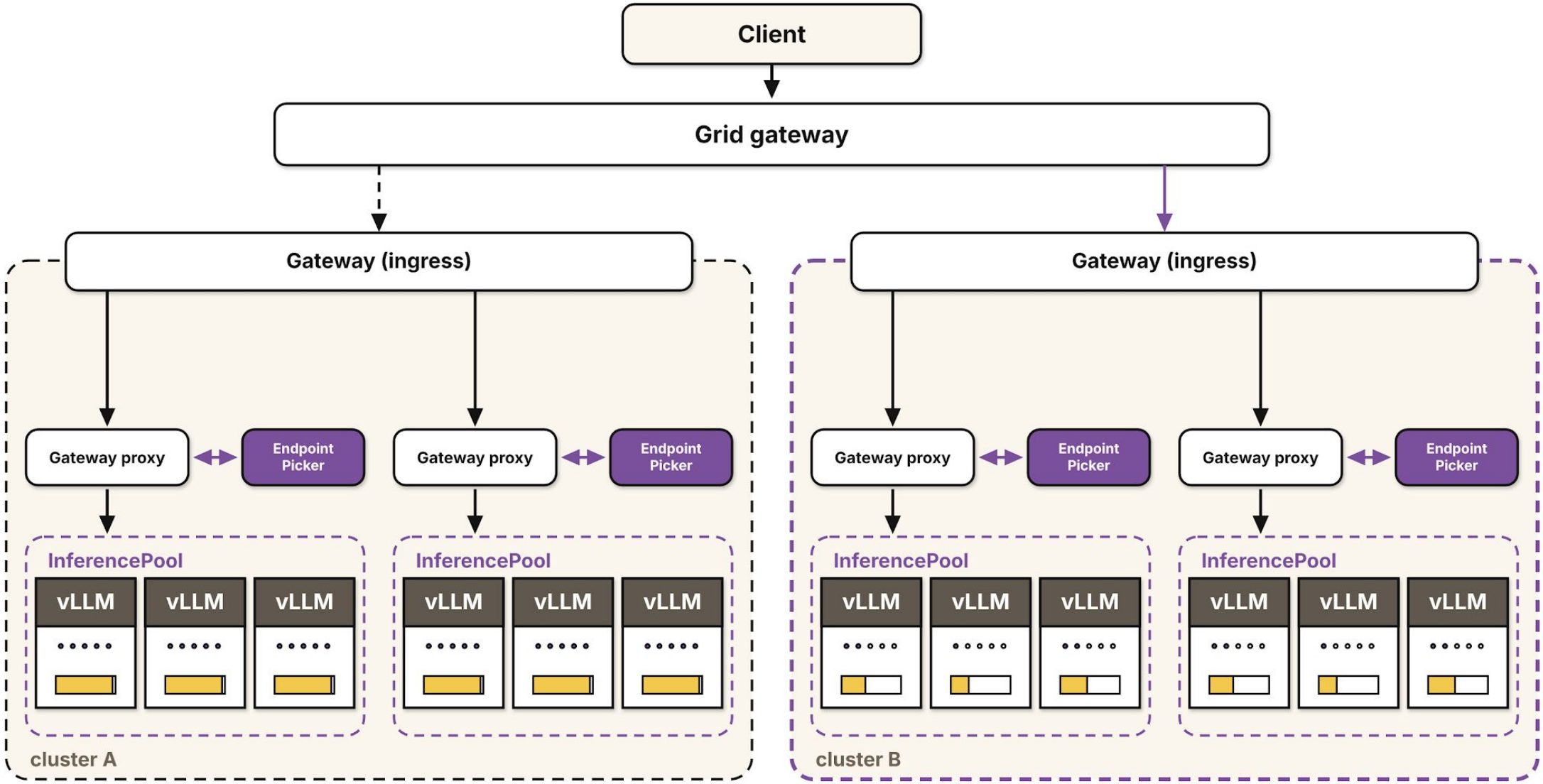
Where llm-d fits





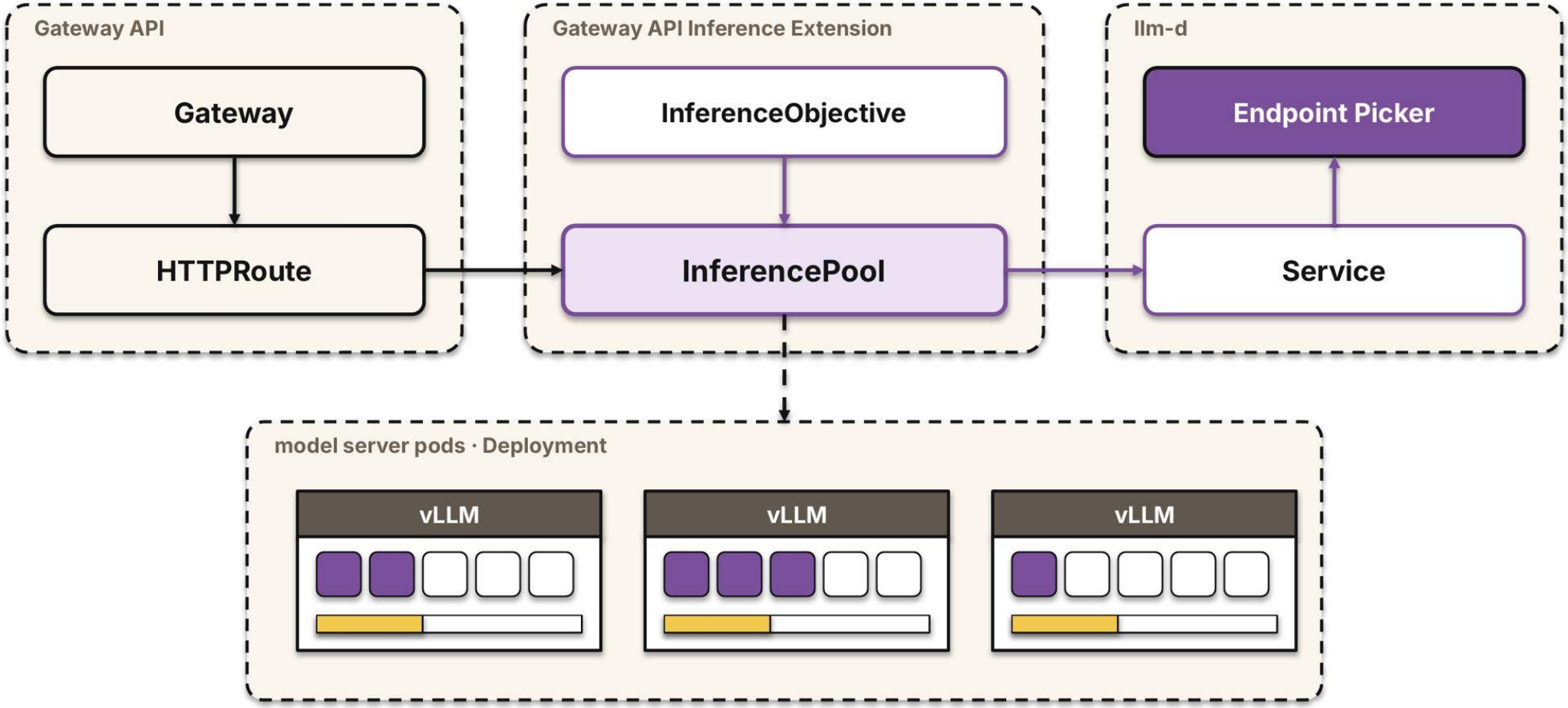
Where llm-d fits



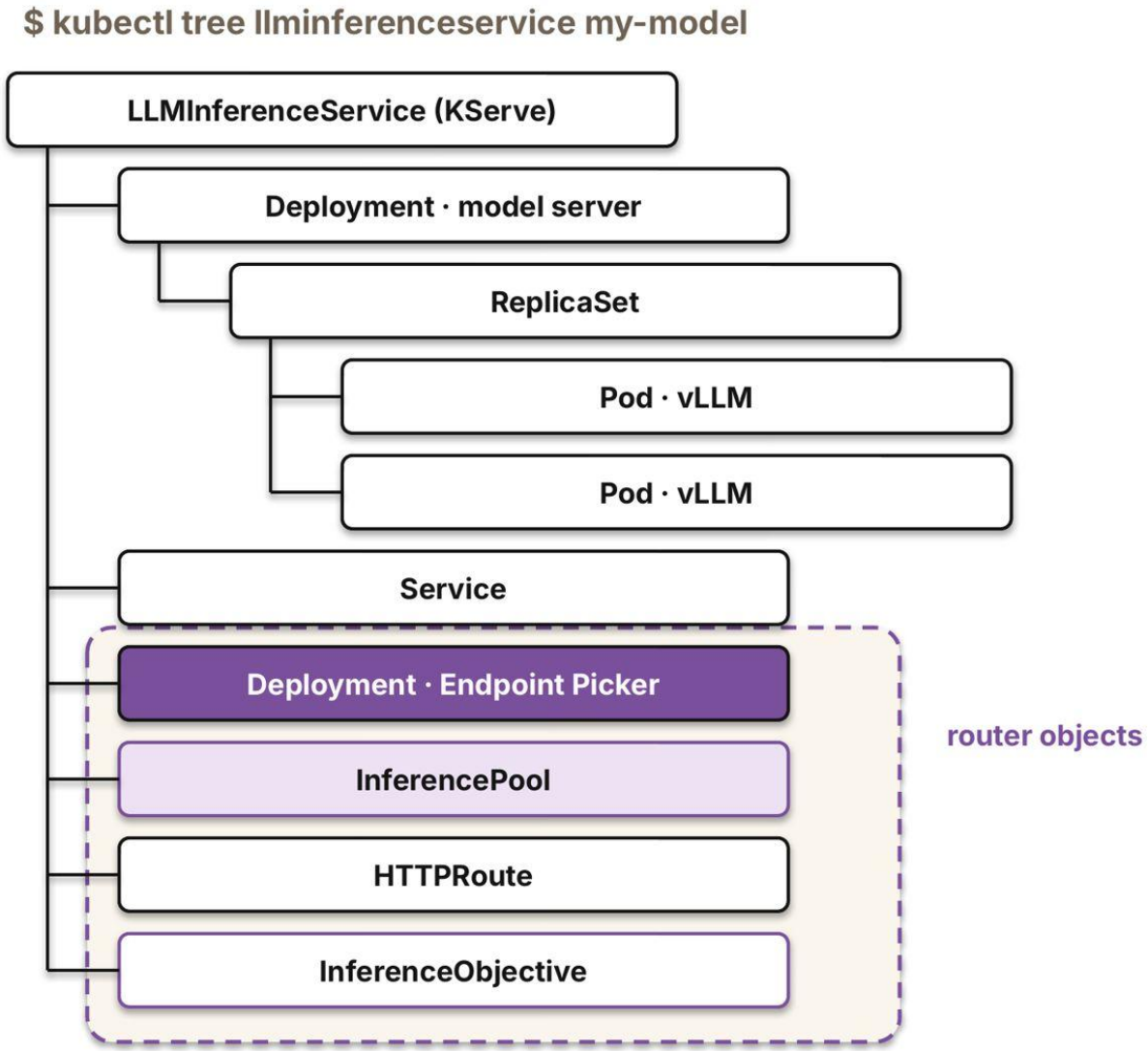


llm-d resources



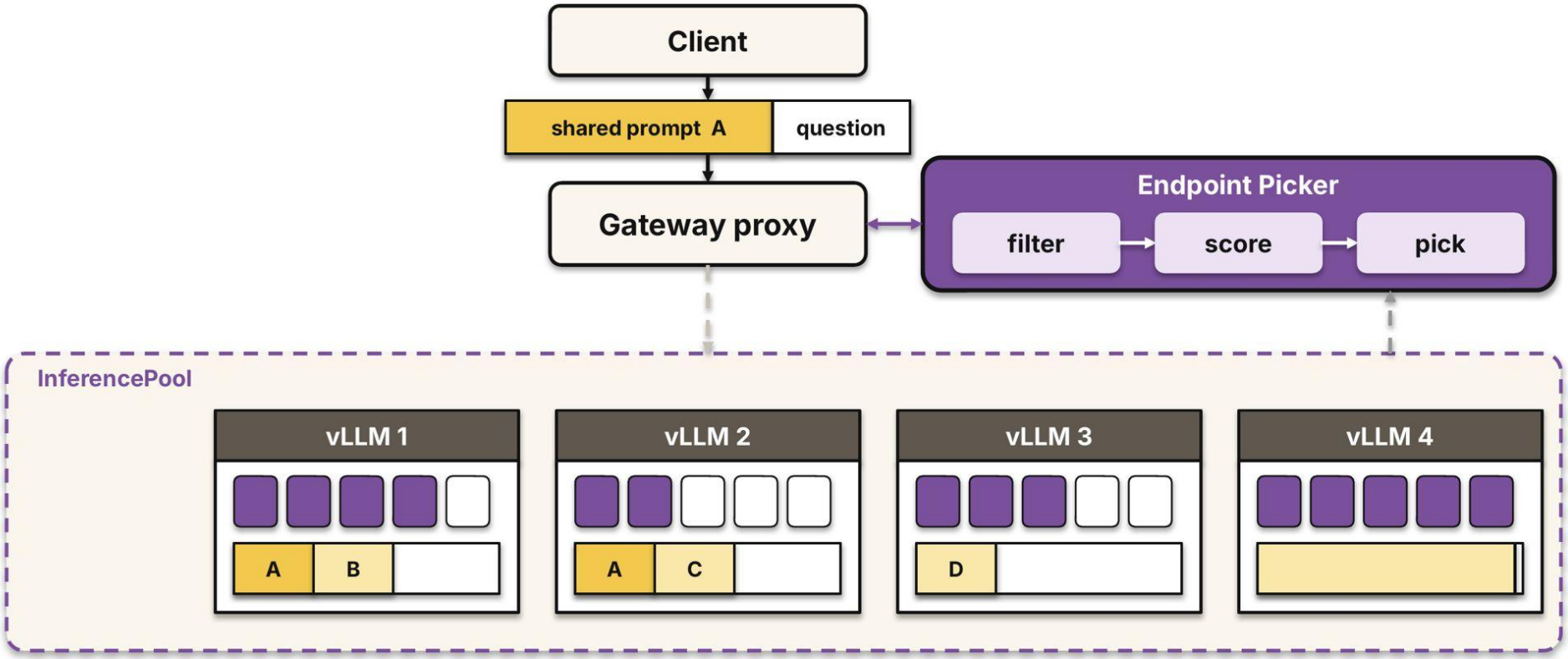


llm-d resources



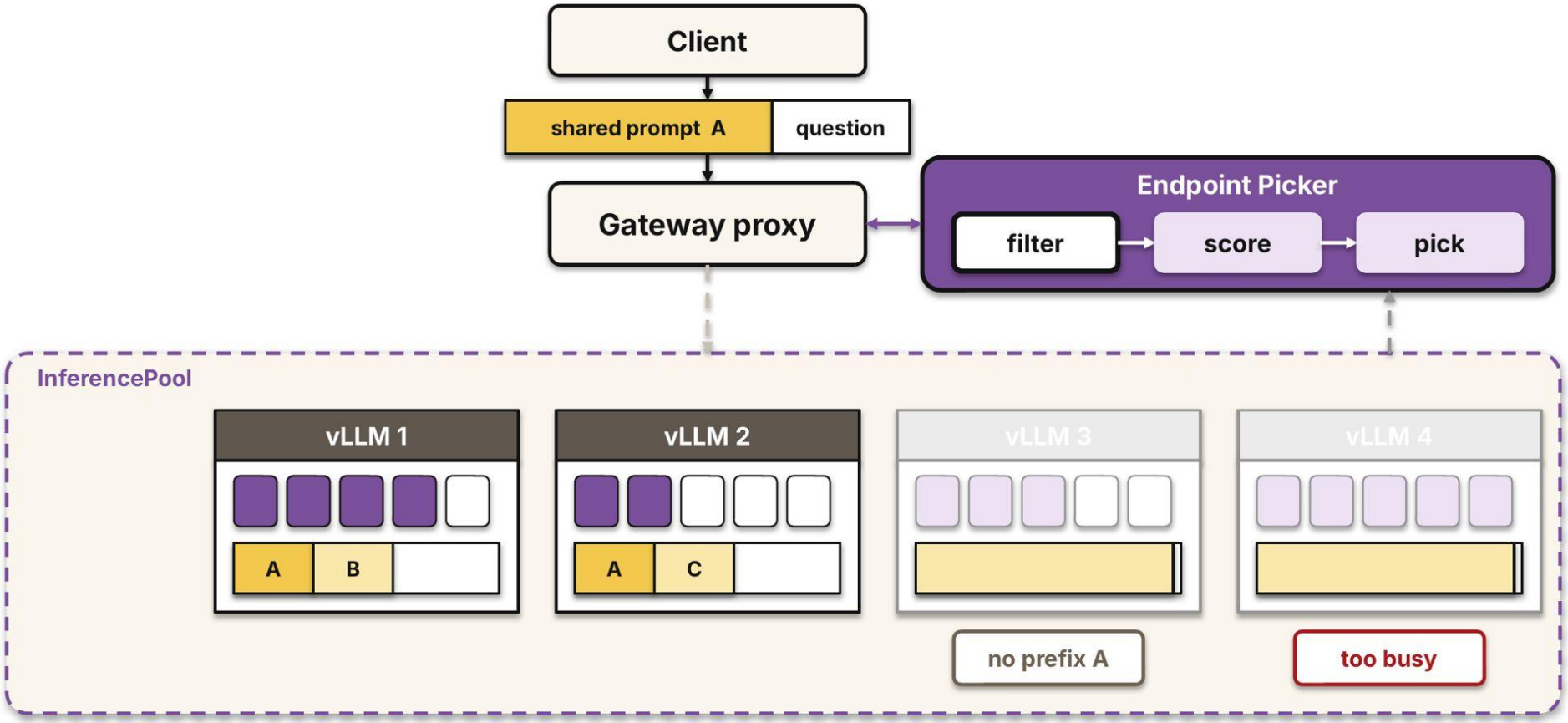
How the Endpoint Picker works





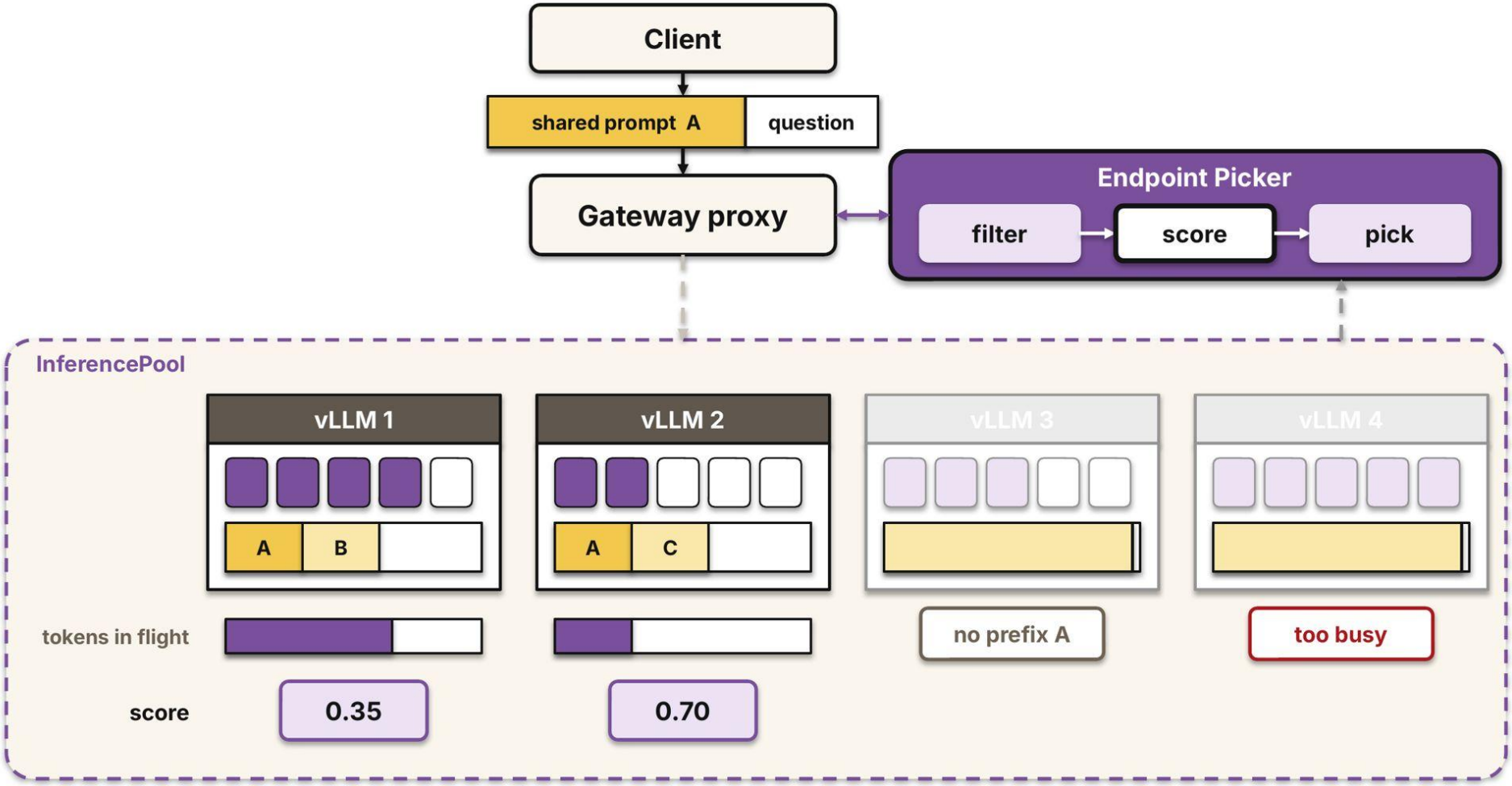
How the Endpoint Picker works





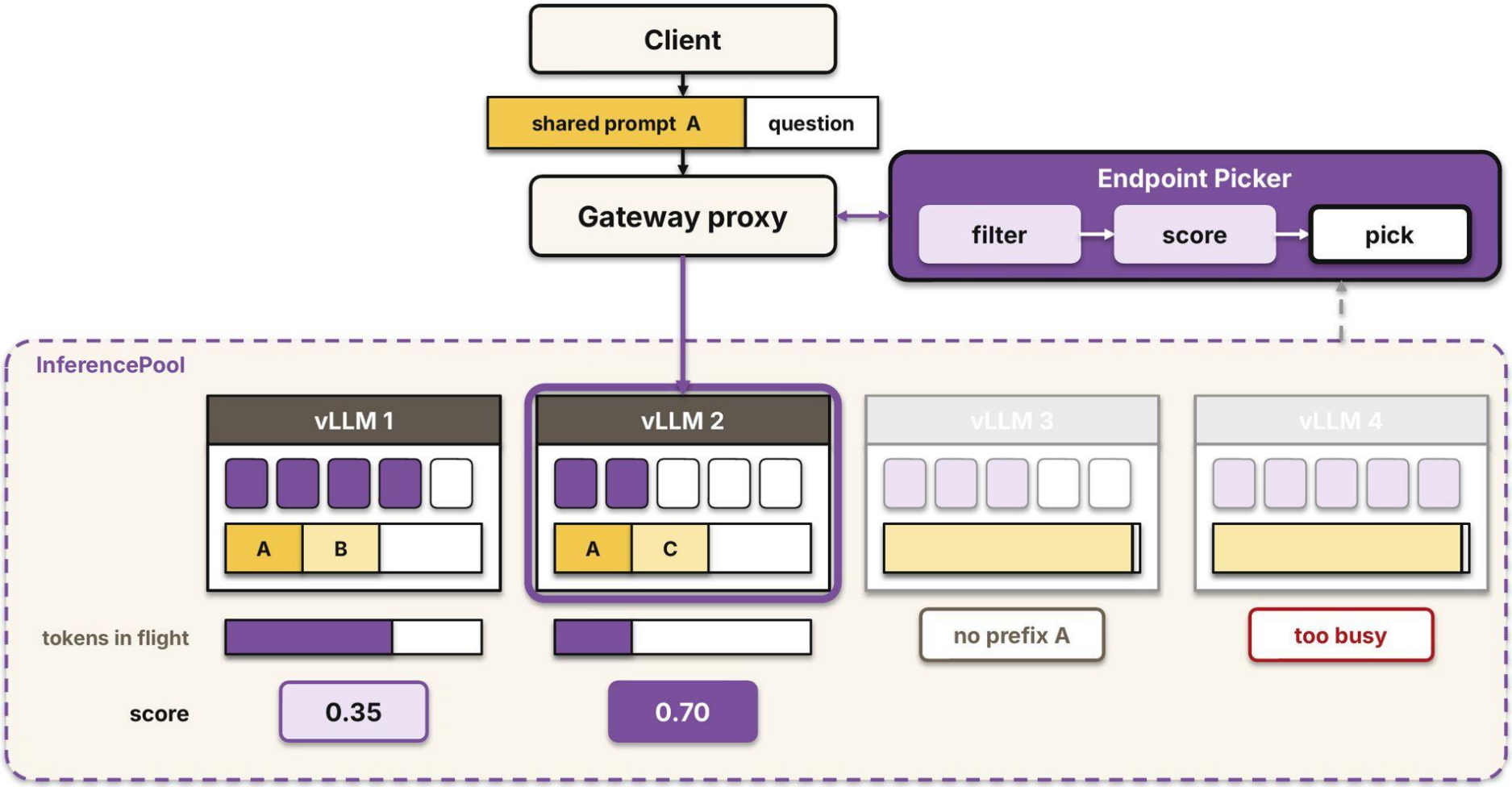
How the Endpoint Picker works





How the Endpoint Picker works





Configuring the Endpoint Picker



flowControl:

saturationDetector	fairnessPolicyRef	orderingPolicyRef	usageLimitPolicyPluginRef
utilization-detector	**global-strict-fairness-policy**	**fcfs-ordering-policy**	**static-usage-limit-policy**
concurrency-detector	round-robin-fairness-policy	edf-ordering-policy	priority-holdback-policy
scraped-concurrency-detector	program-aware-fairness	slo-deadline-ordering-policy	soft-reflective-ceiling-policy
max-saturation-detector			

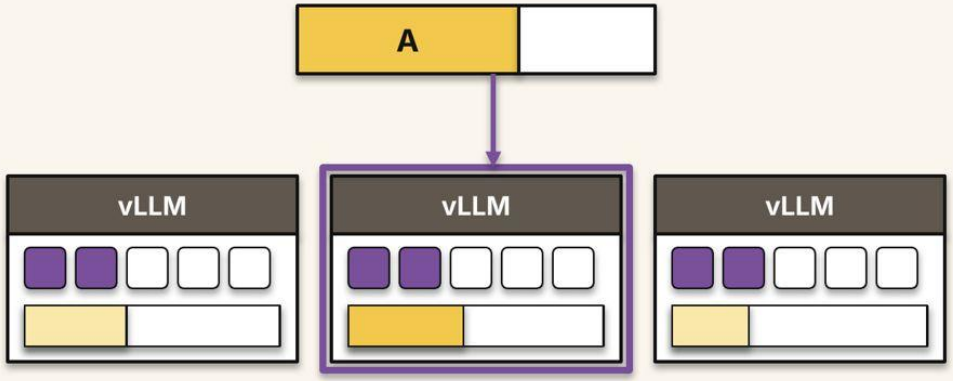
schedulingProfiles:

filters	scorers (weighted)		picker
prefix-cache-affinity-filter	**token-load-scorer**	precise-prefix-cache-scorer	**max-score-picker**
utilization-detector	prefix-cache-scorer	load-aware-scorer	random-picker
concurrency-detector + headroom	queue-scorer	lora-affinity-scorer	weighted-random-picker
prefill-filter	kv-cache-utilization-scorer	latency-scorer	
decode-filter			
label-selector-filter			
session-affinity-filter			

Well-lit paths



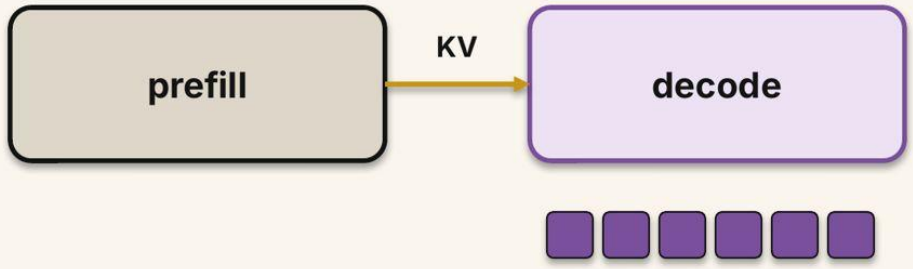
intelligent routing



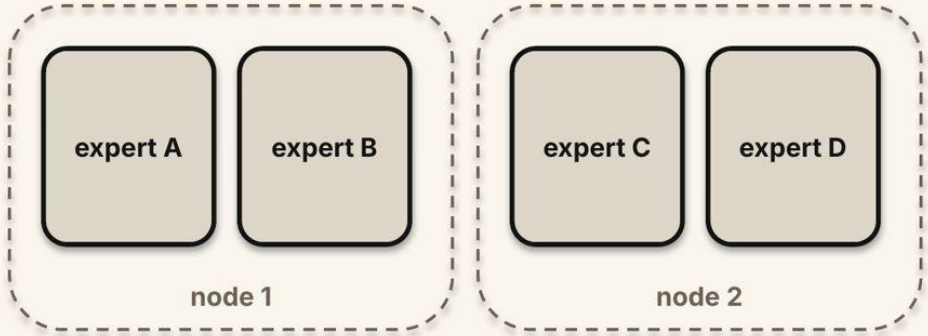
KV cache management



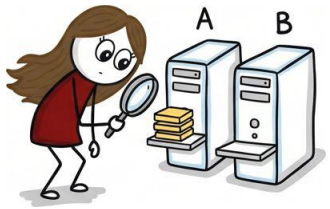
large models · P/D

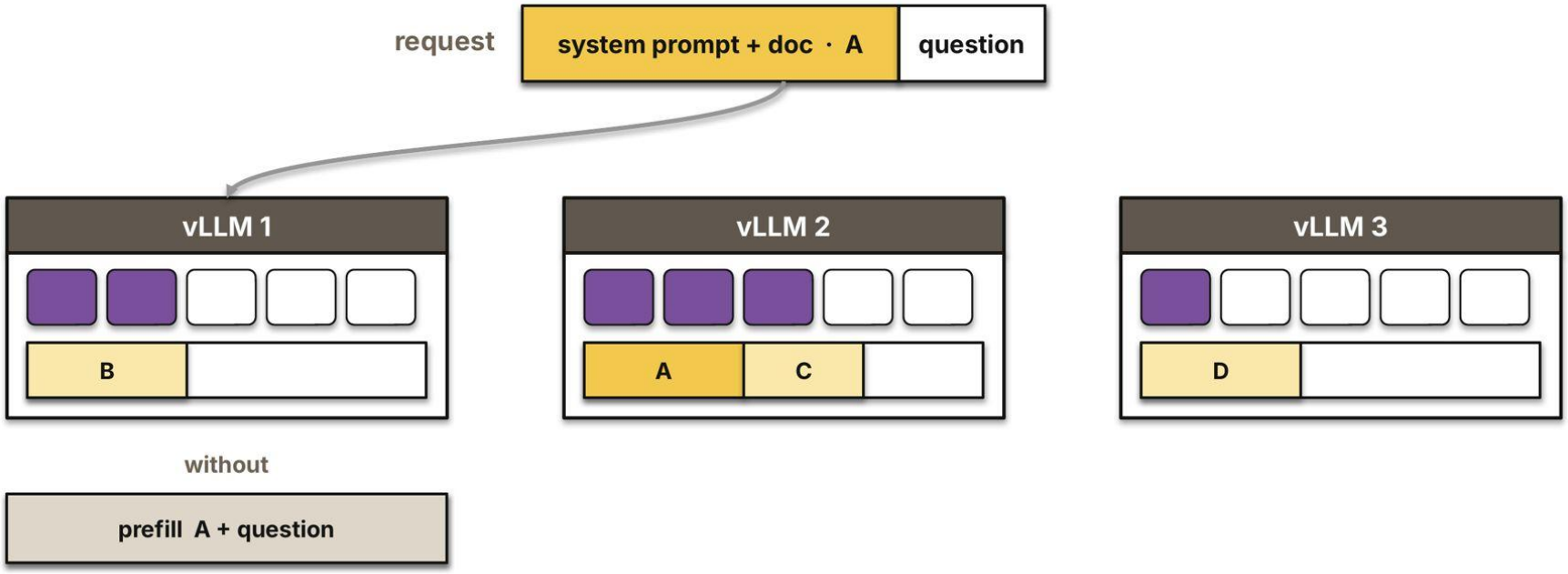


large models · expert parallelism

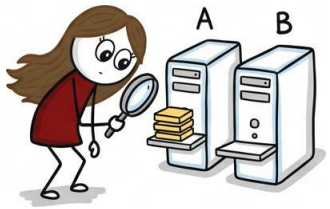


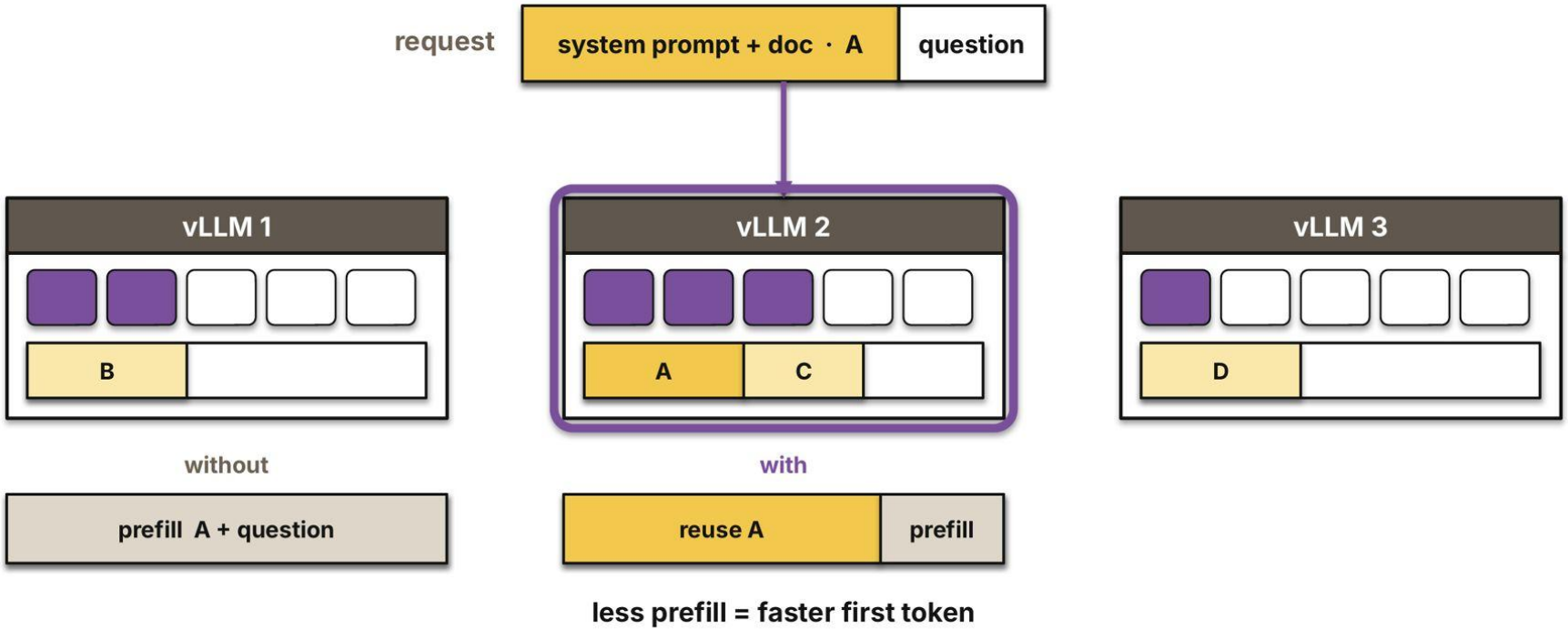
Intelligent routing



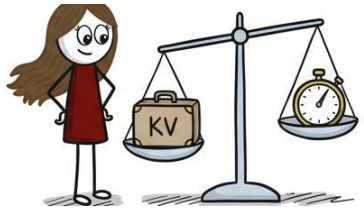


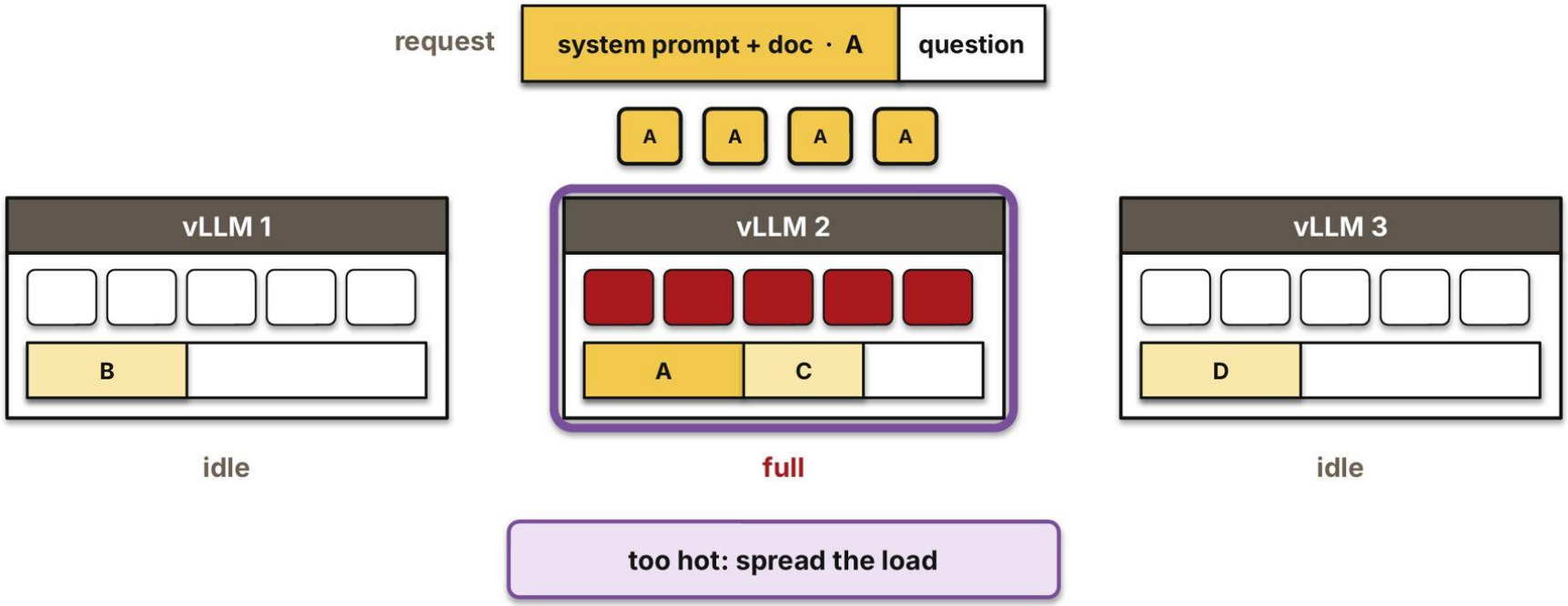
Intelligent routing





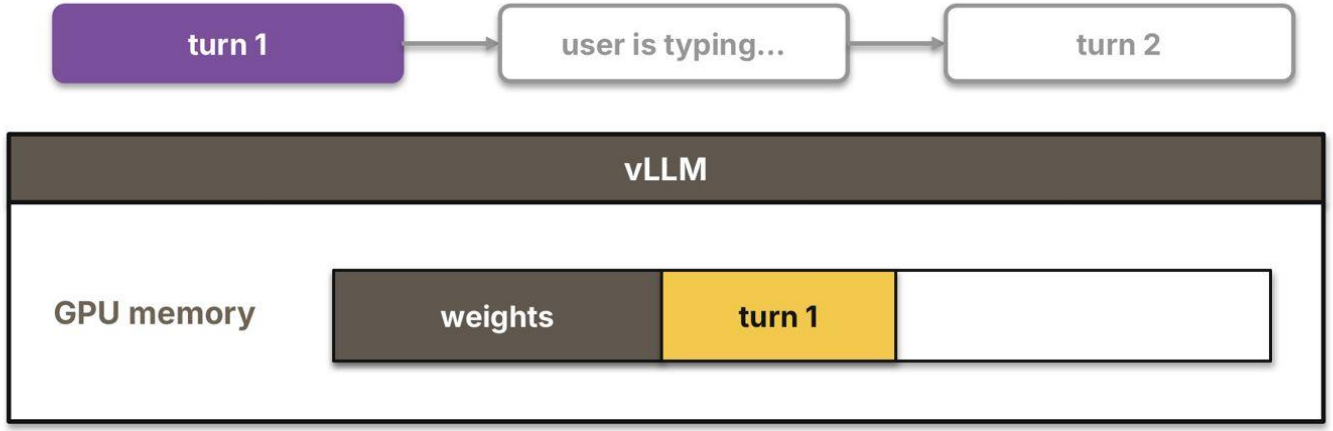
Intelligent routing





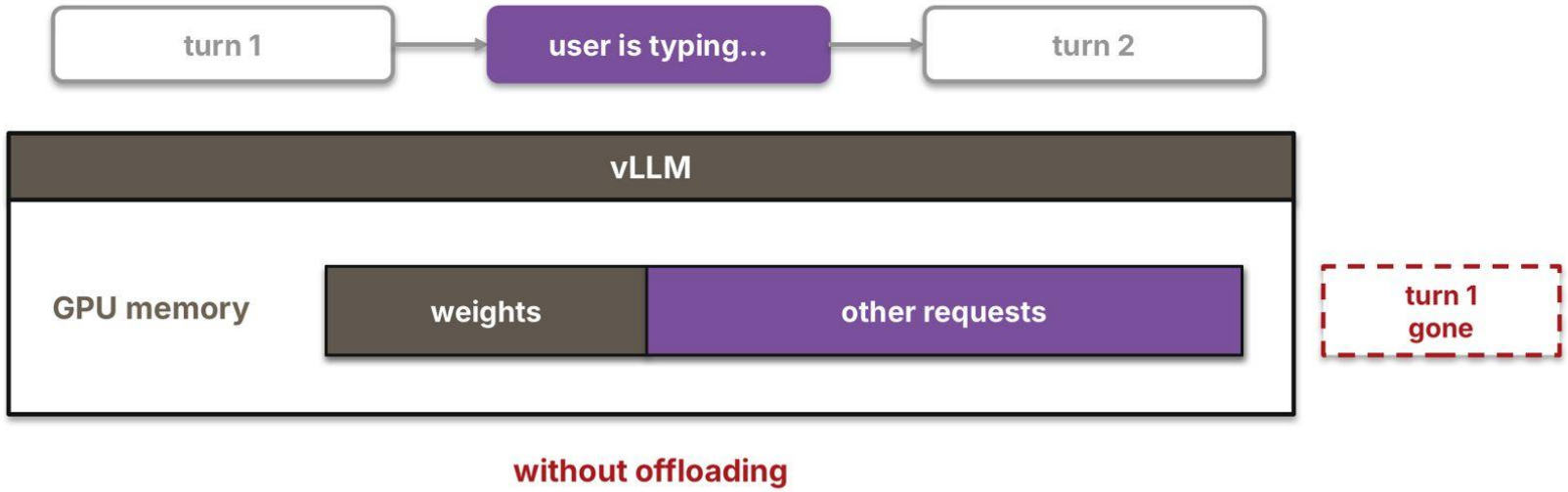
KV cache offloading





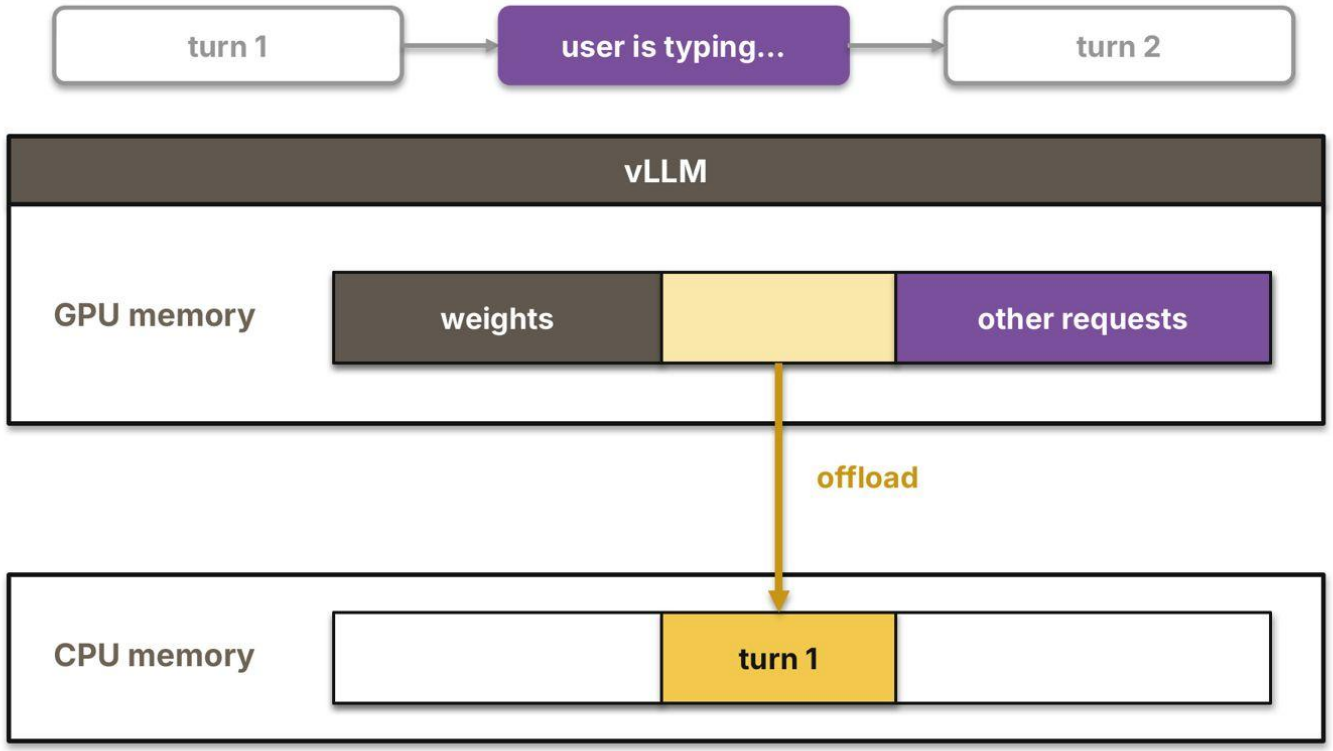
KV cache offloading





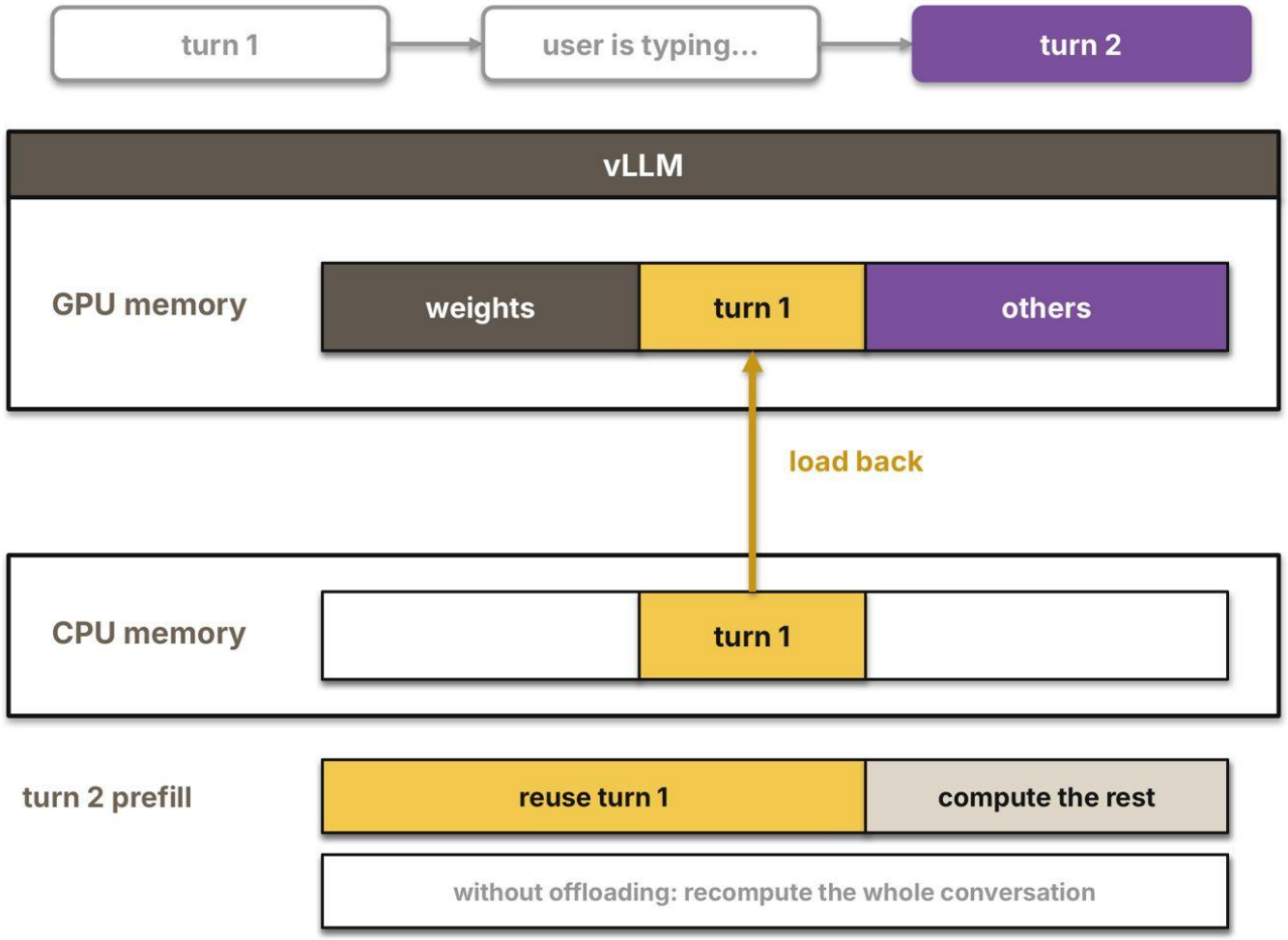
KV cache offloading



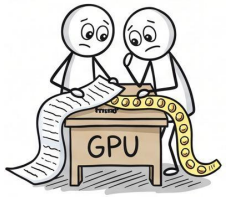


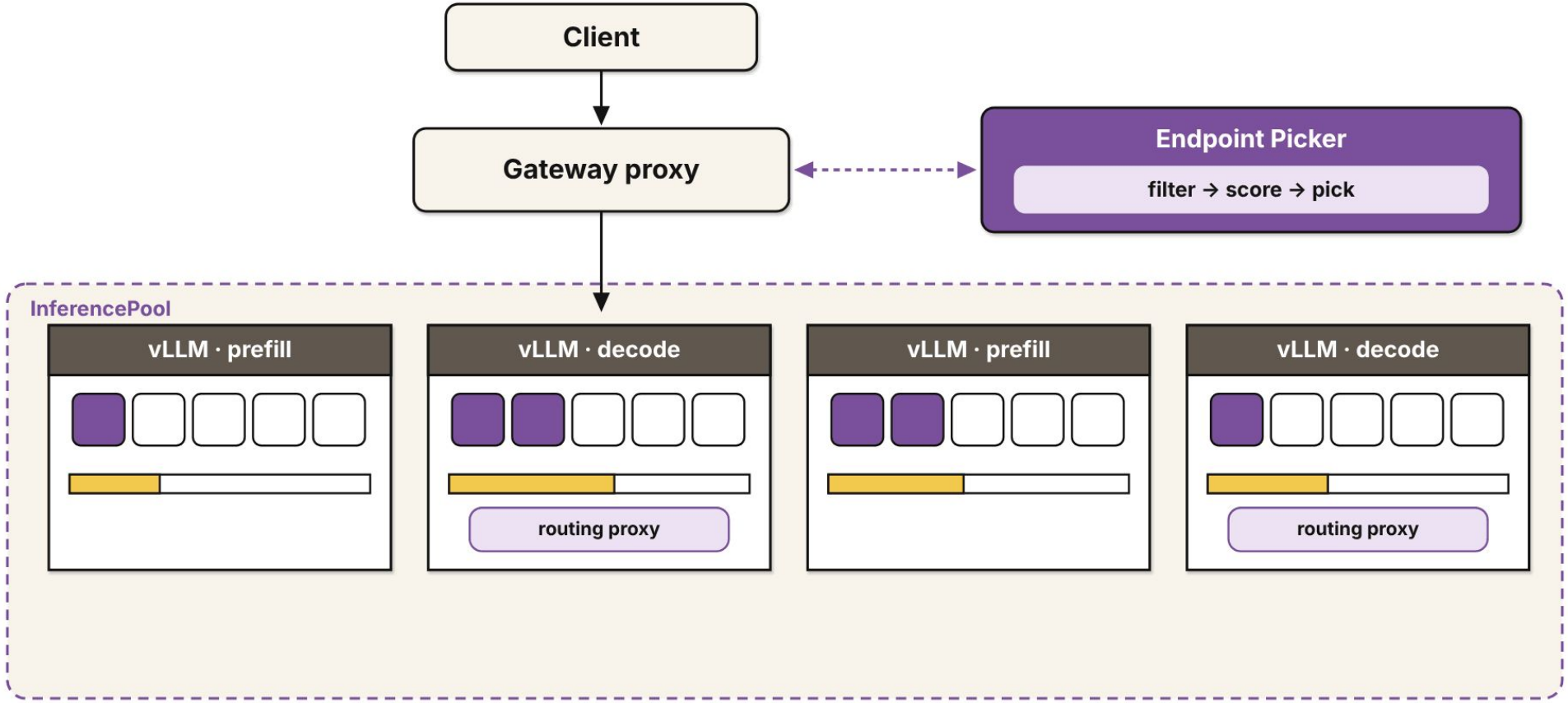
KV cache offloading





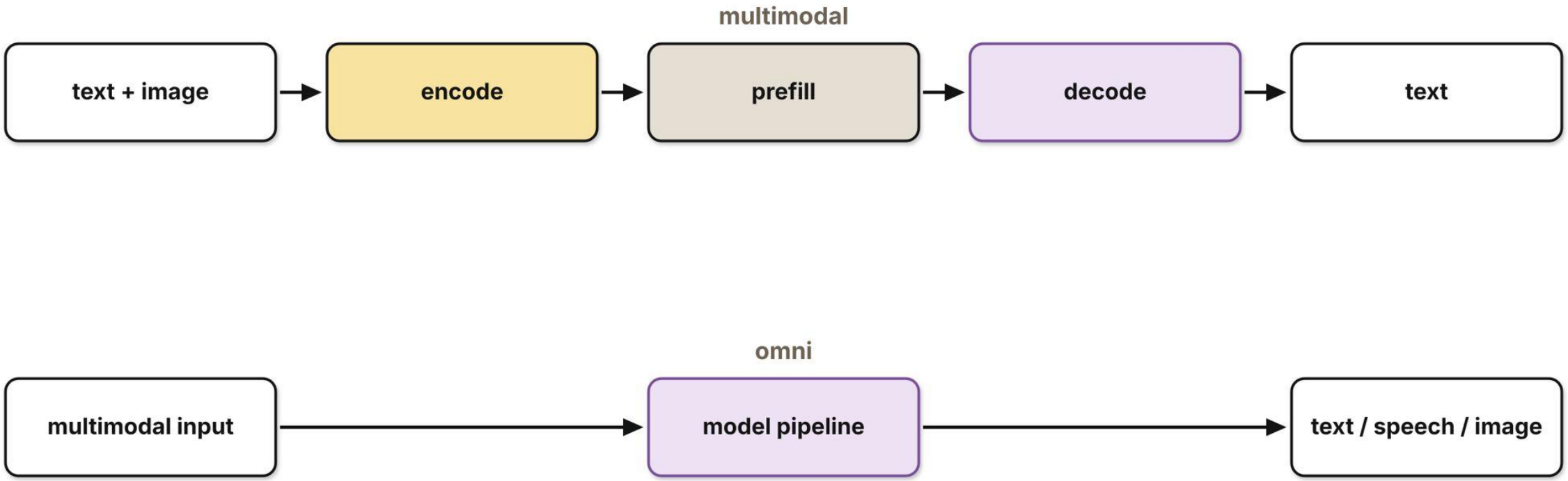
Disaggregated Serving





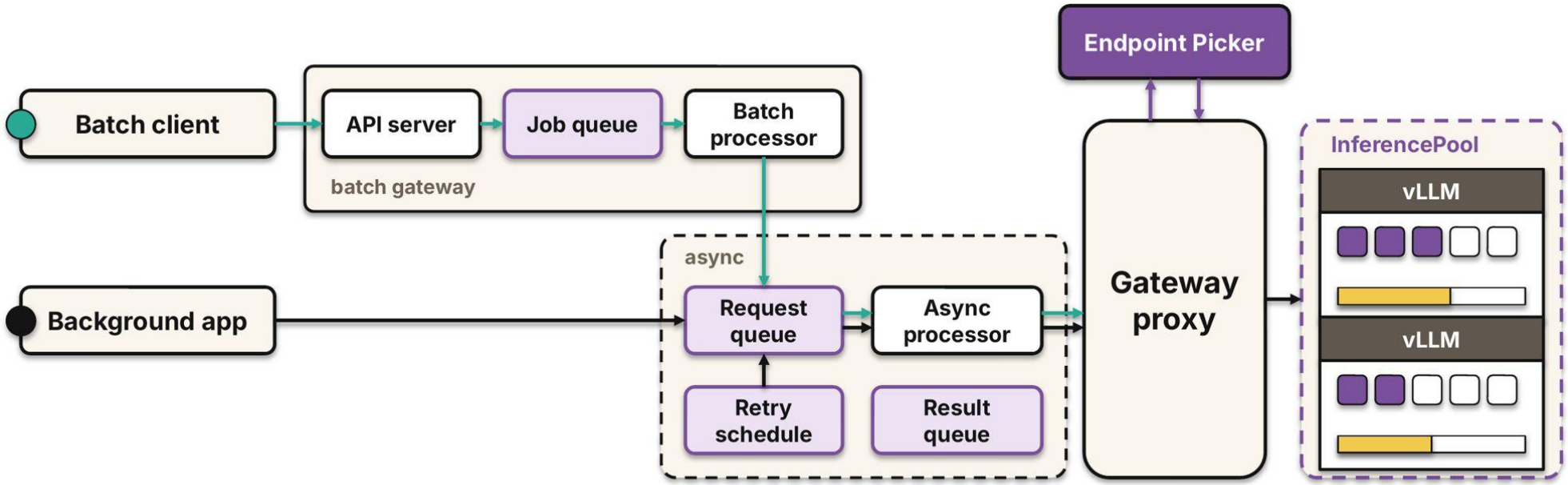
Remote prefill when needed; KV transfer adds network cost.

Beyond text


multimodal
text + image
encode
prefill
decode
text
omni
multimodal input
model pipeline
text / speech / image

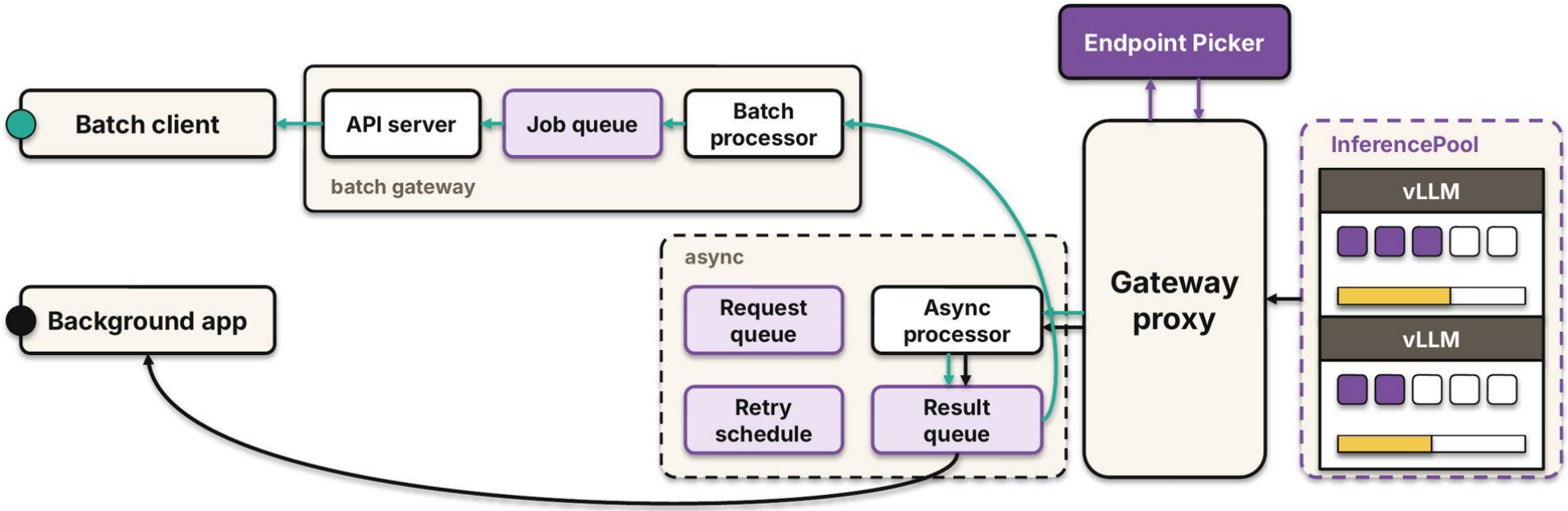
Async inference



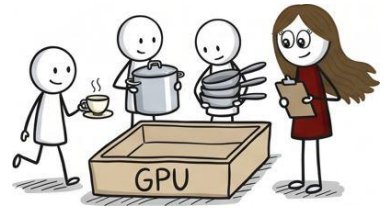


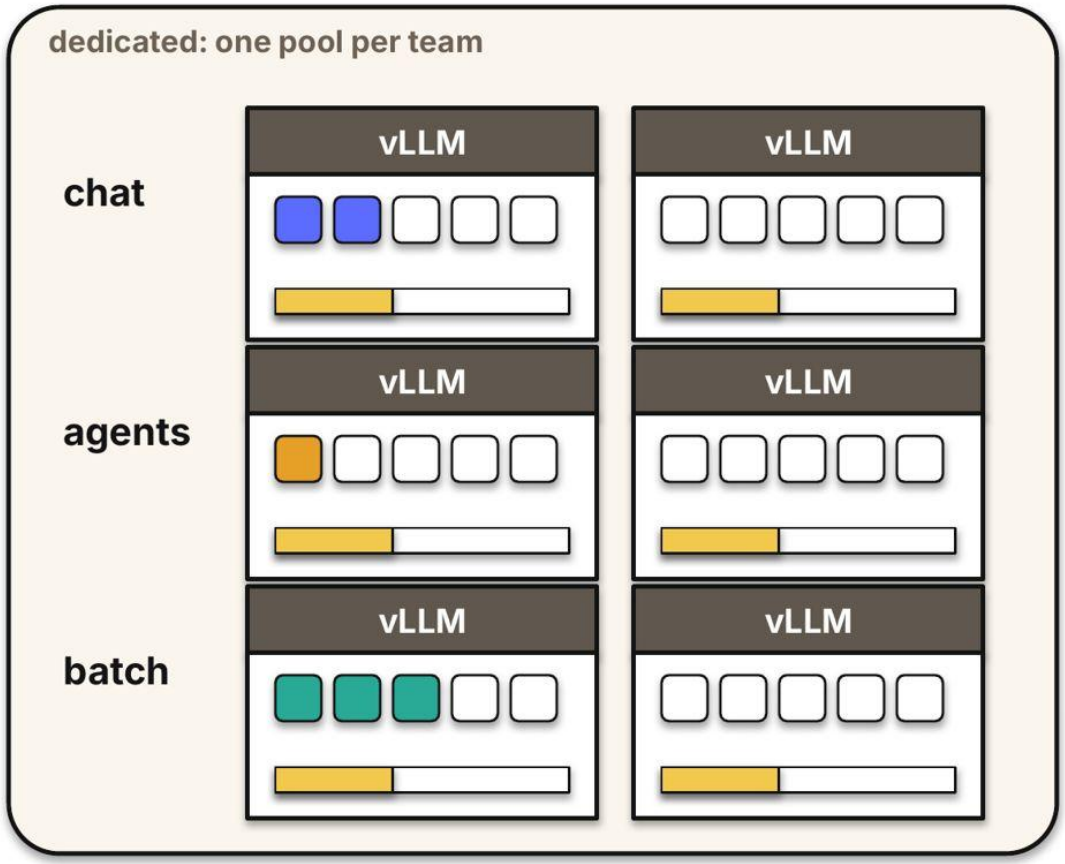
Async inference

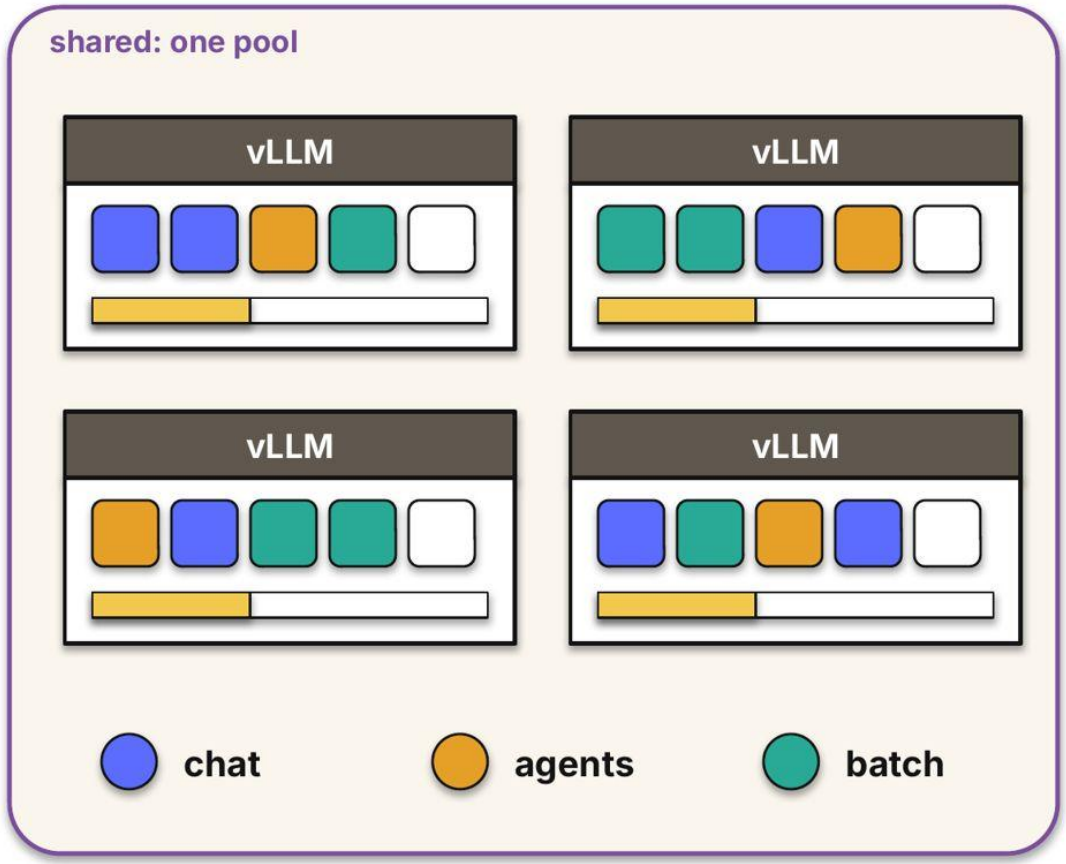




Sharing Inference with Flow Control

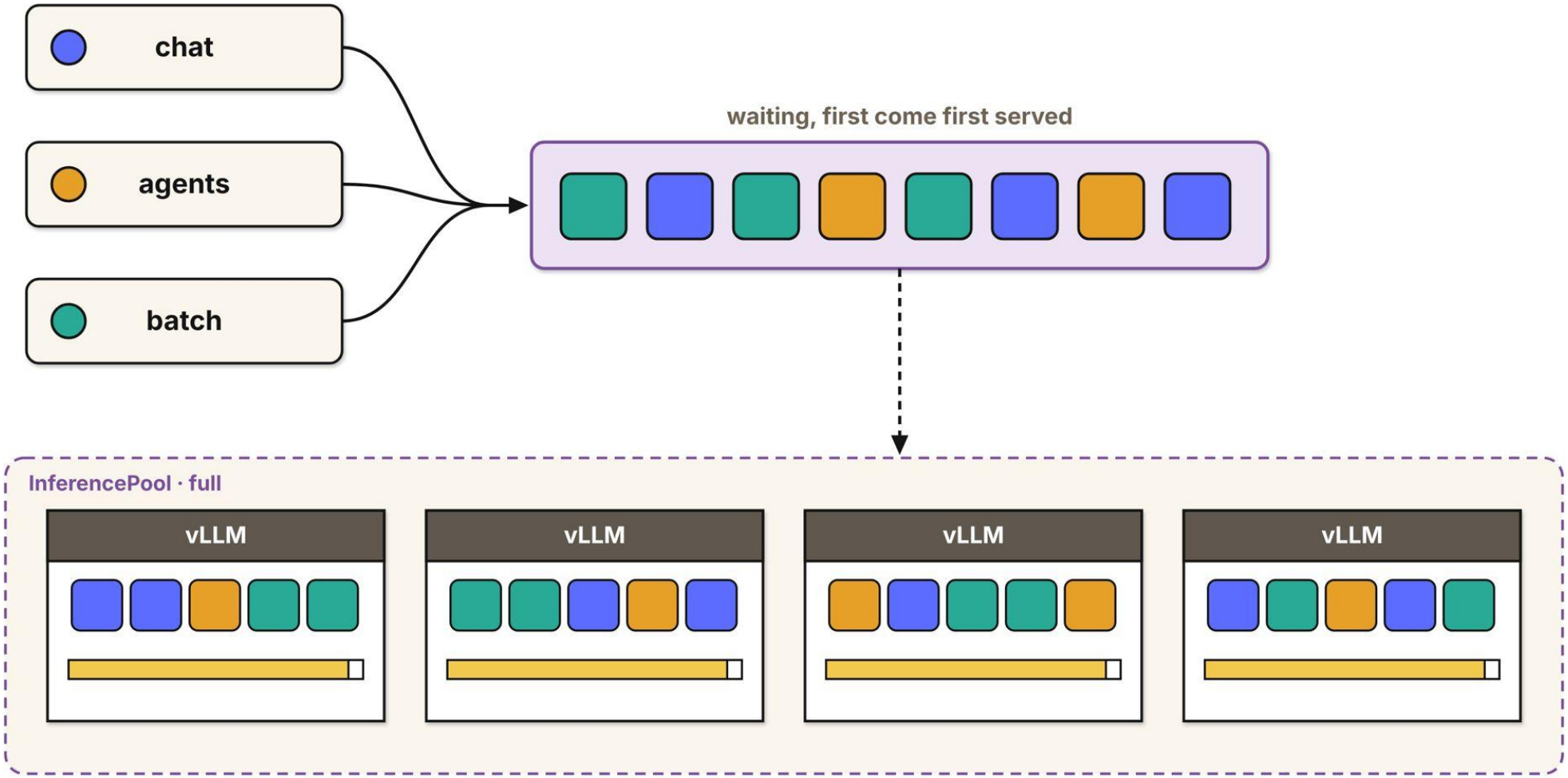






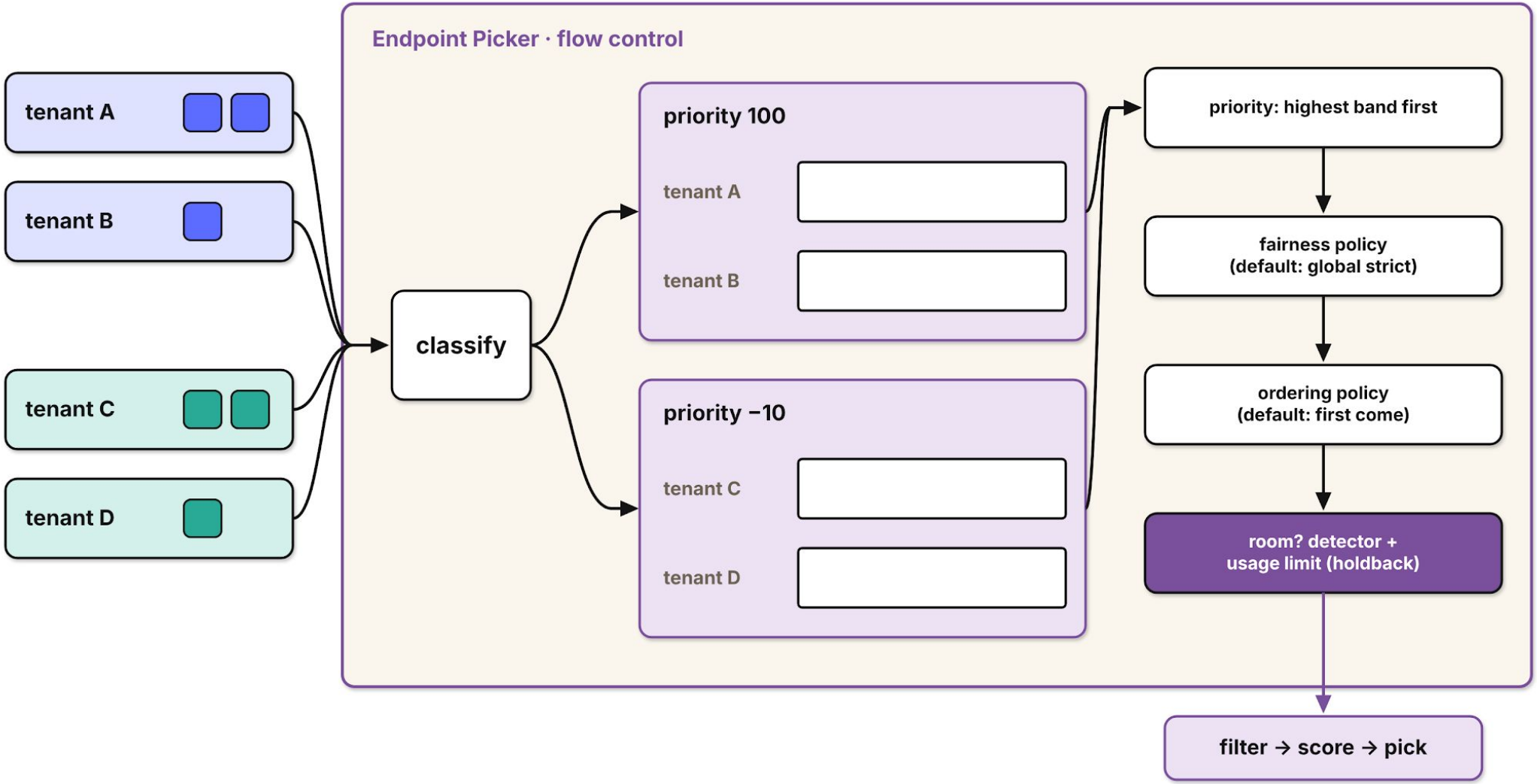
Sharing Inference with Flow Control



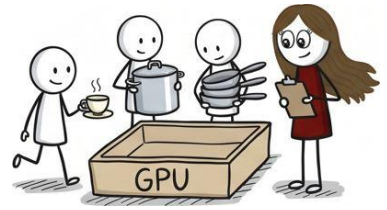


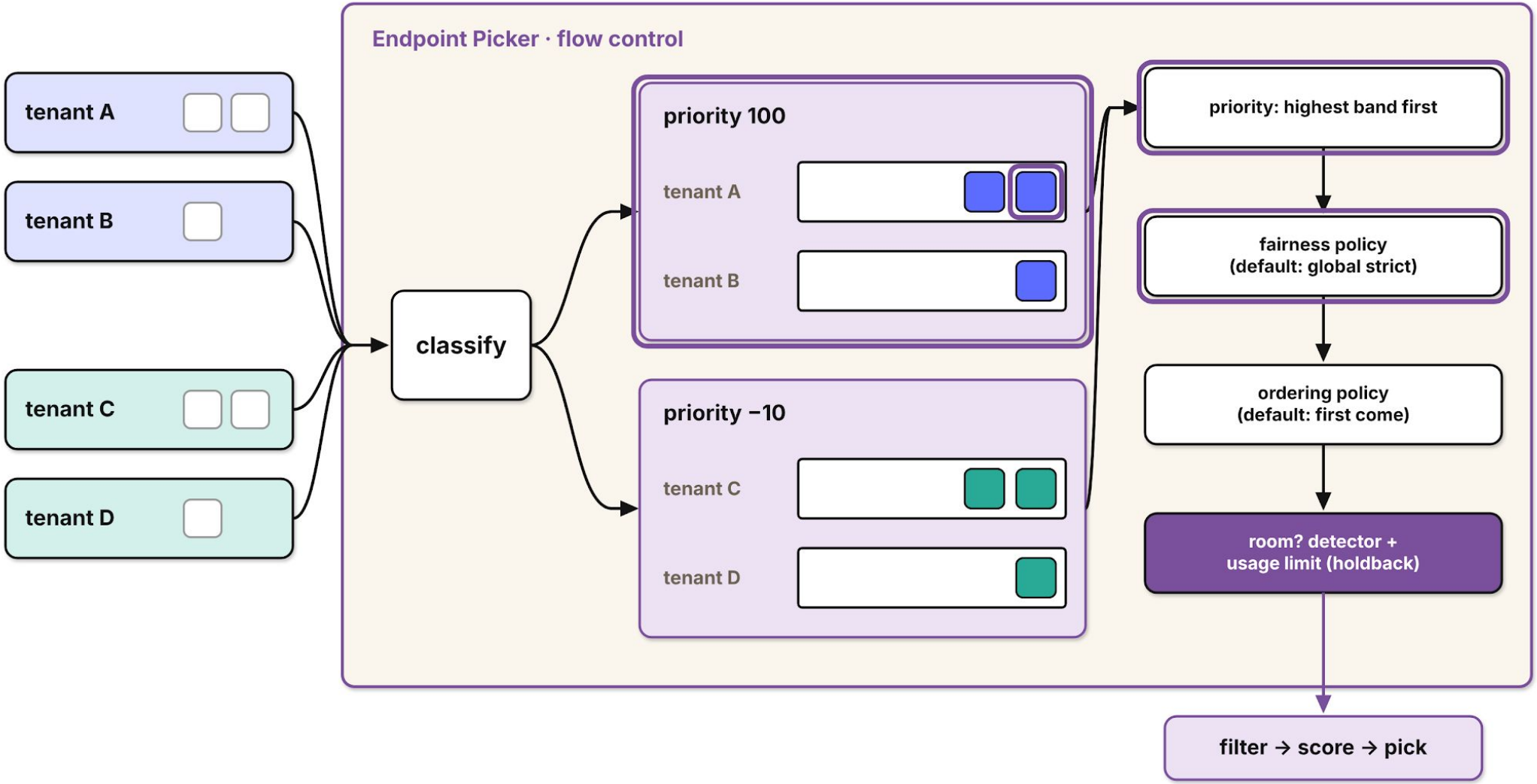
Sharing Inference with Flow Control



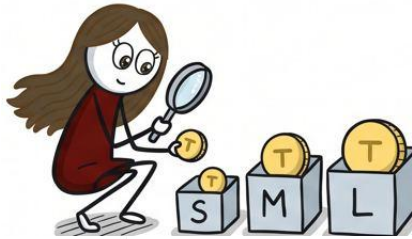


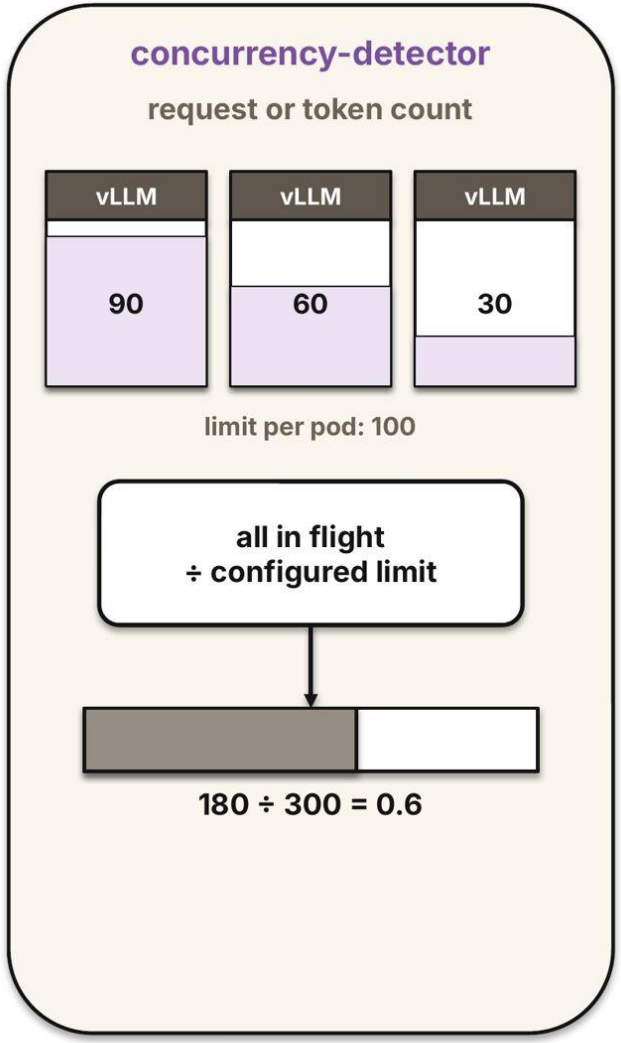
Sharing Inference with Flow Control



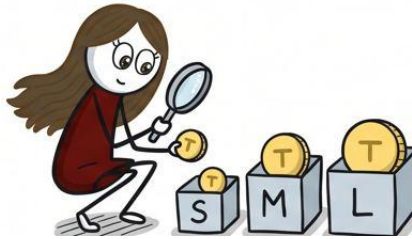


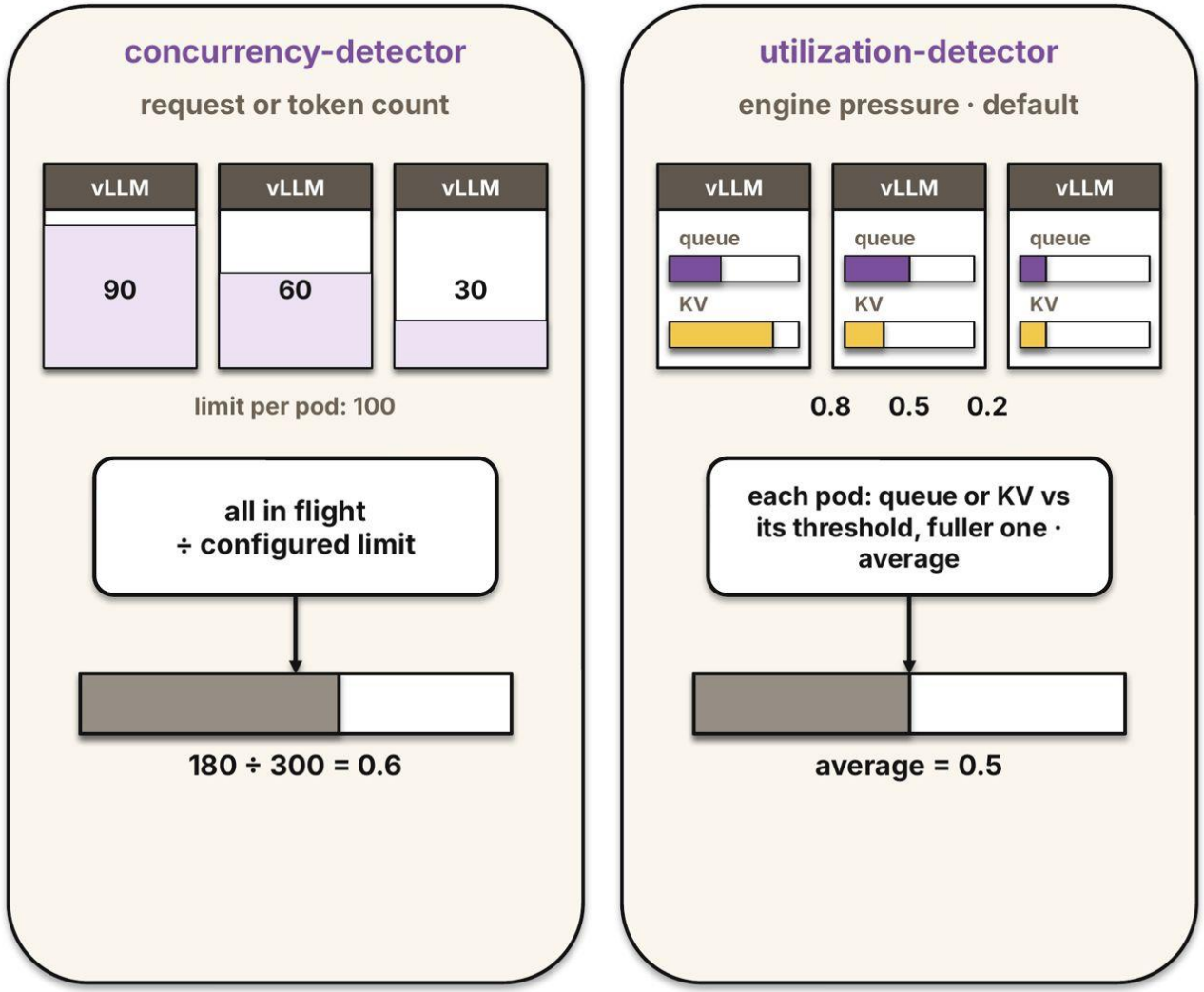
Detecting saturation



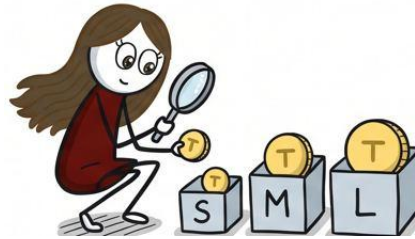


Detecting saturation



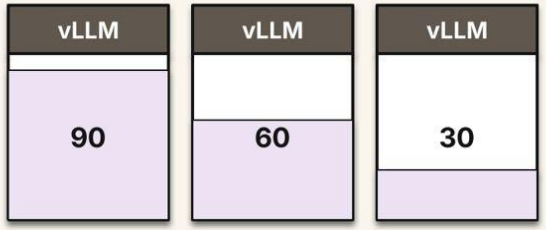


Detecting saturation



concurrency-detector

request or token count



limit per pod: 100

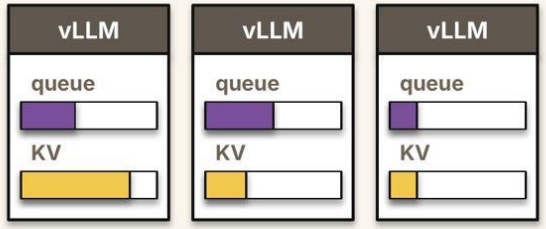
all in flight
÷ configured limit



180 ÷ 300 = 0.6

utilization-detector

engine pressure · default



0.8 0.5 0.2

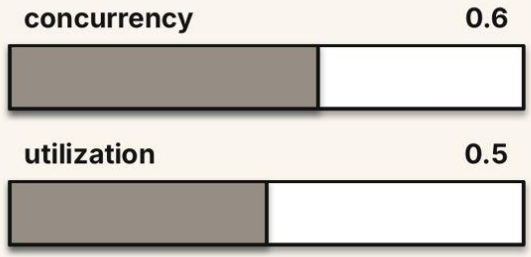
each pod: queue or KV vs its threshold, fuller one · average



average = 0.5

max-saturation-detector

example: combine with max



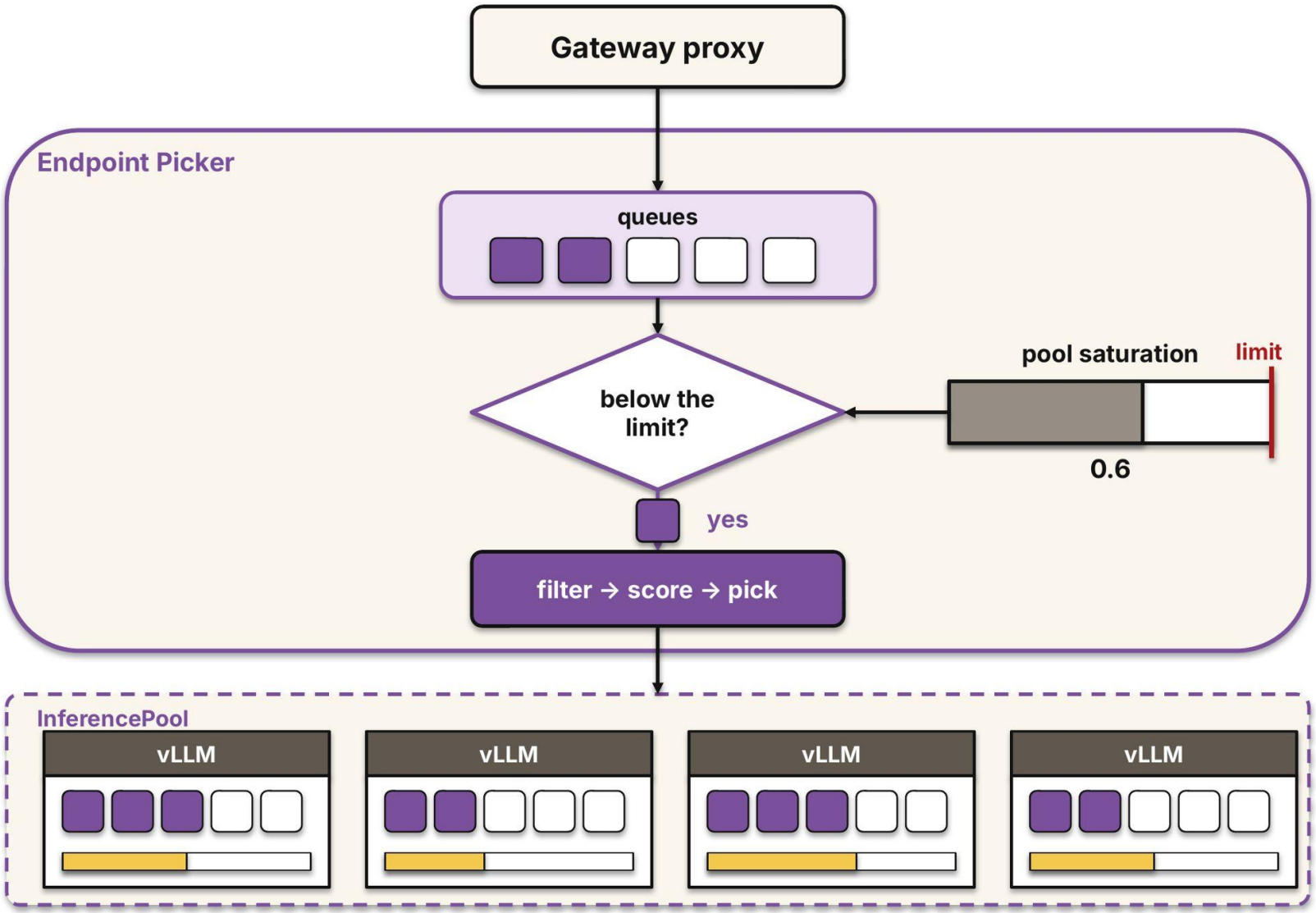
take the highest



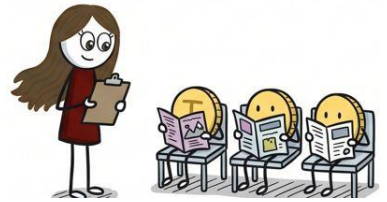
max = 0.6

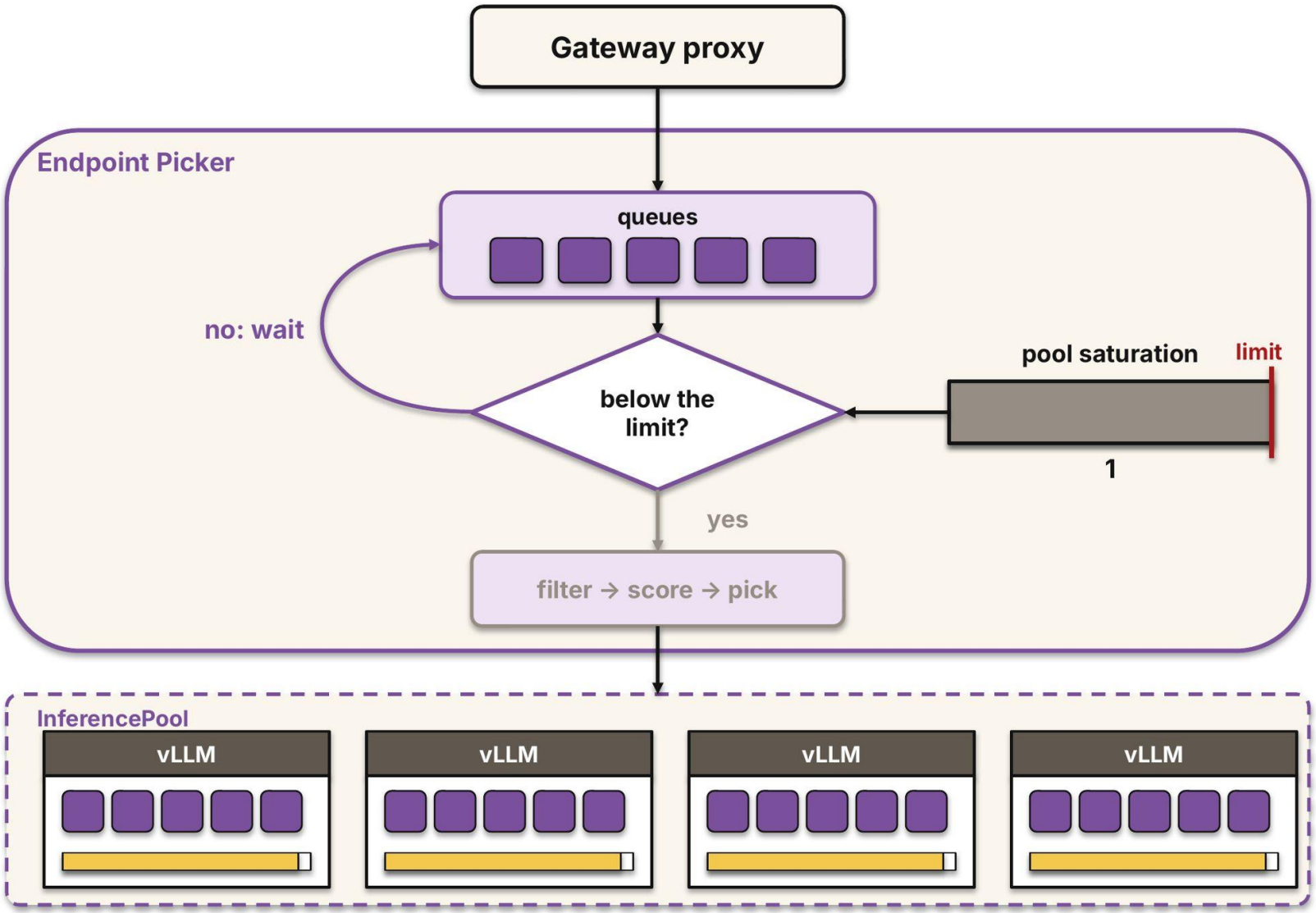
Admission limits and queues





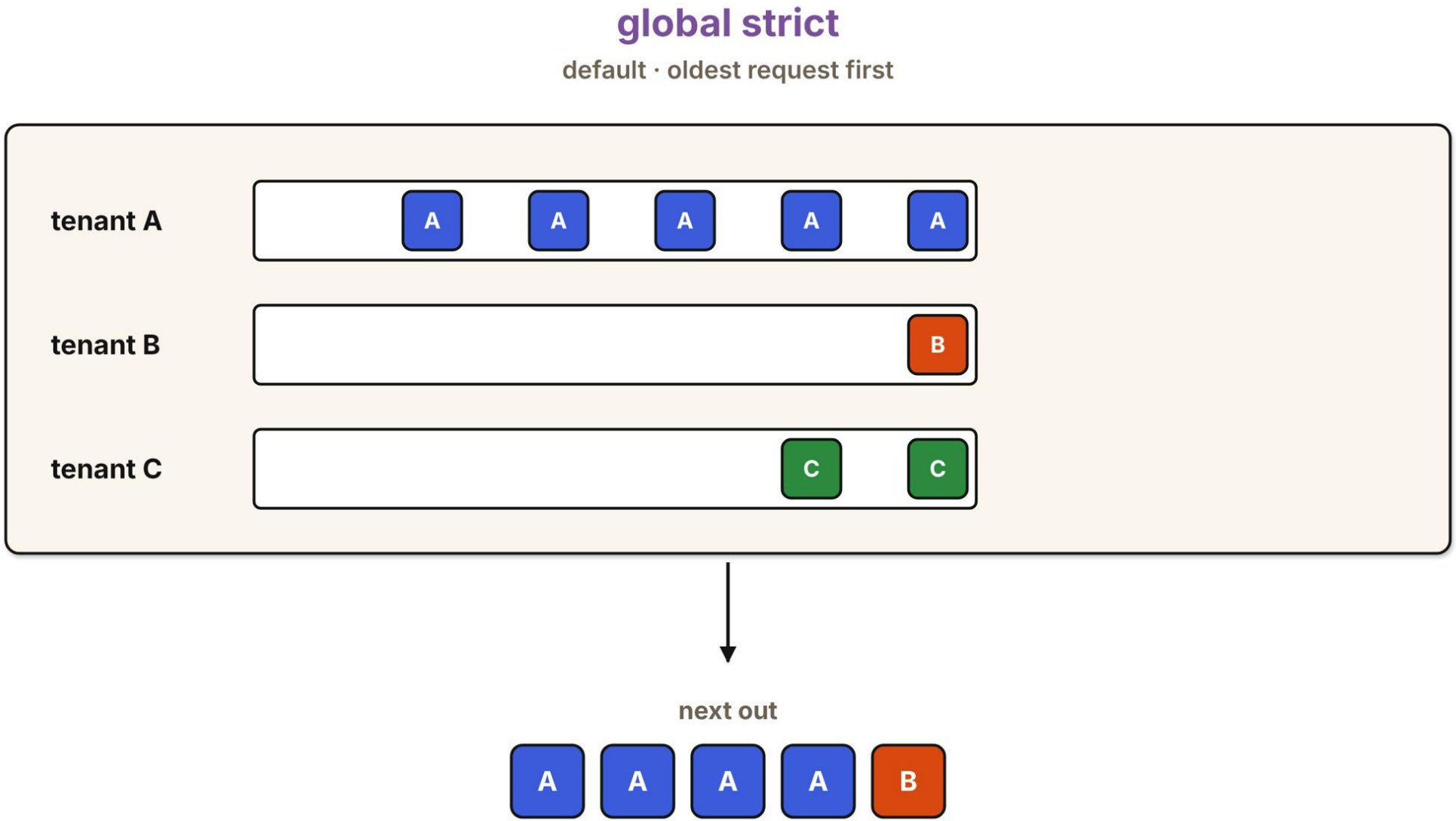
Admission limits and queues





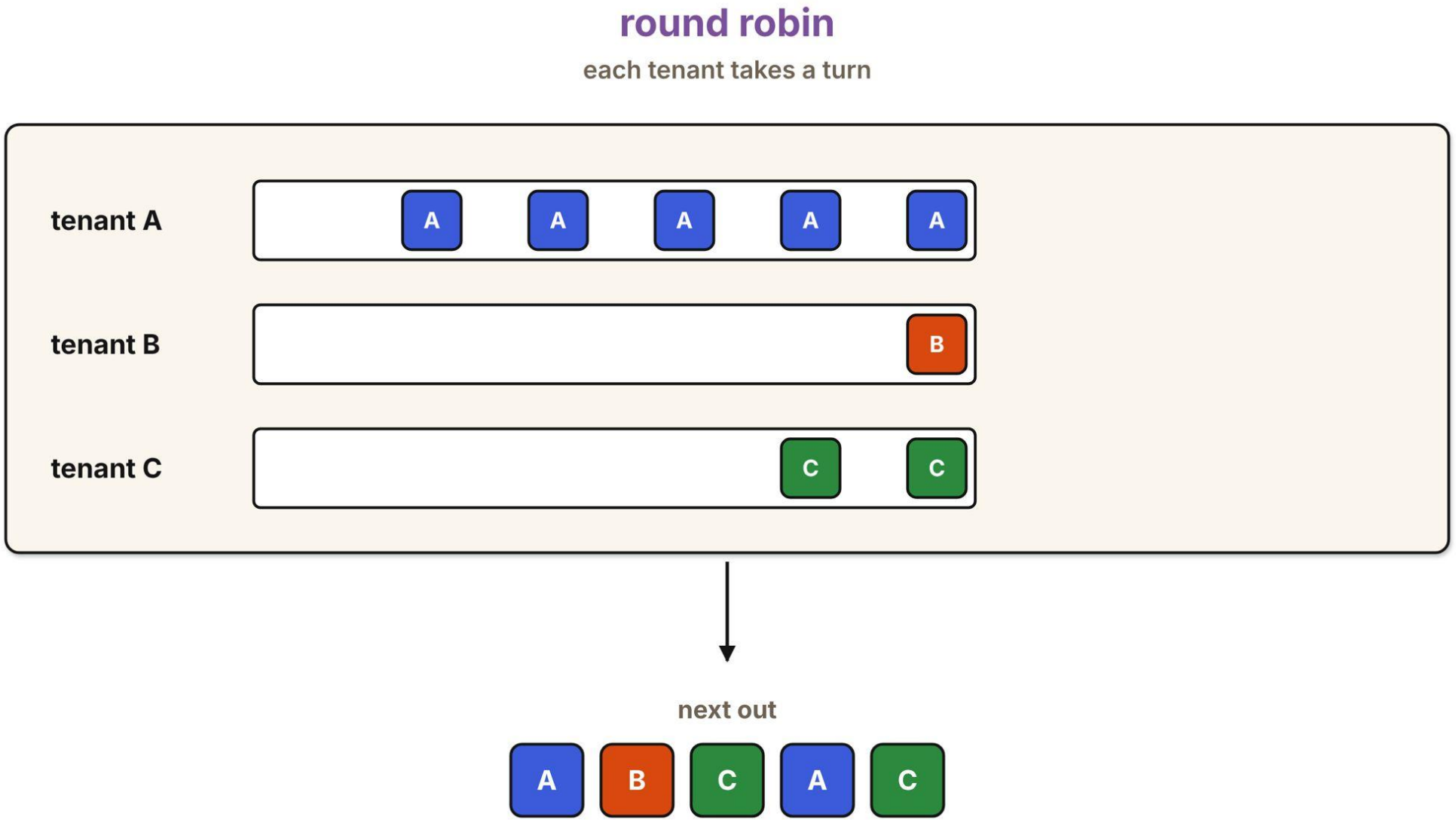
Fairness policies





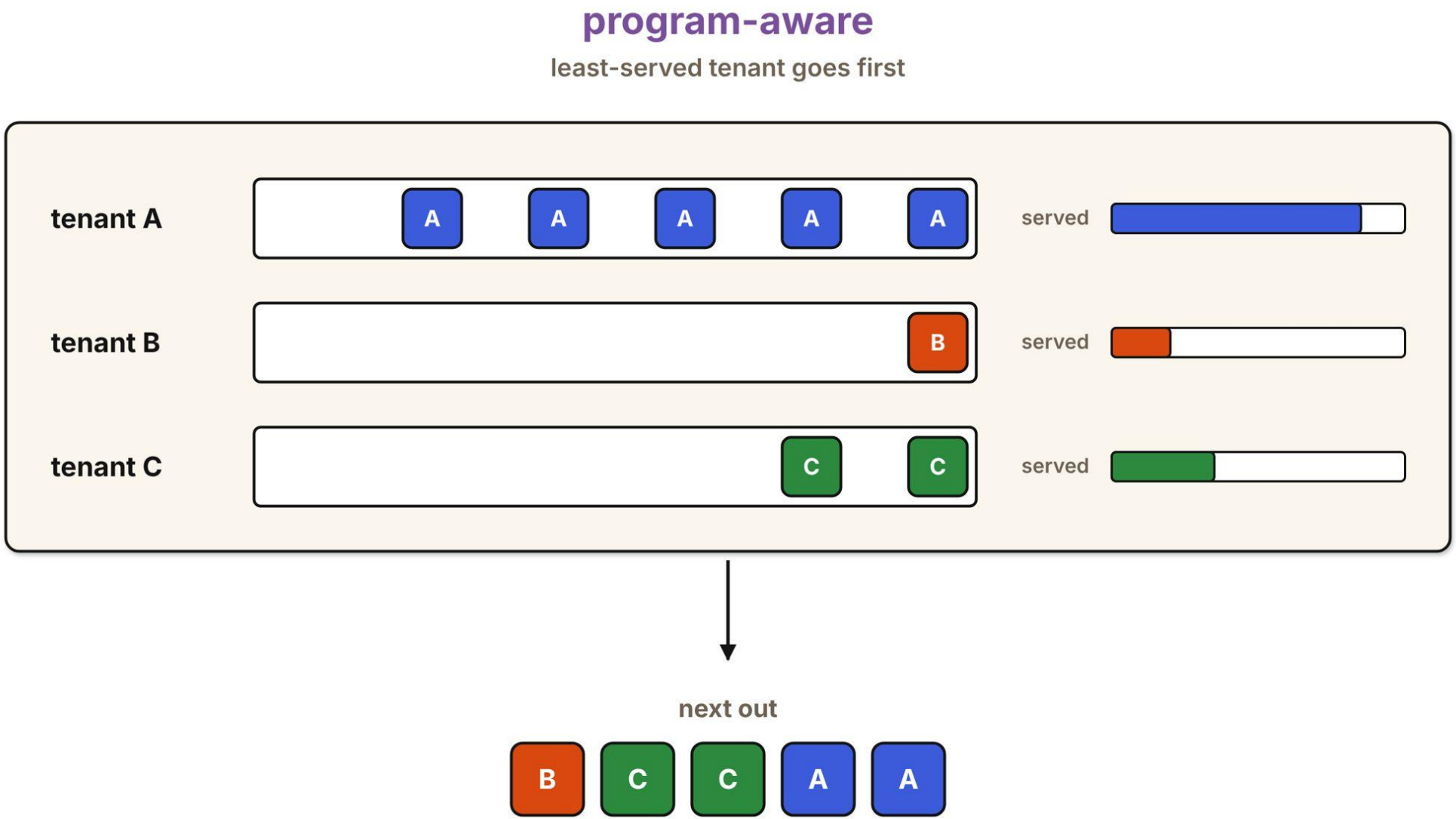
Fairness policies





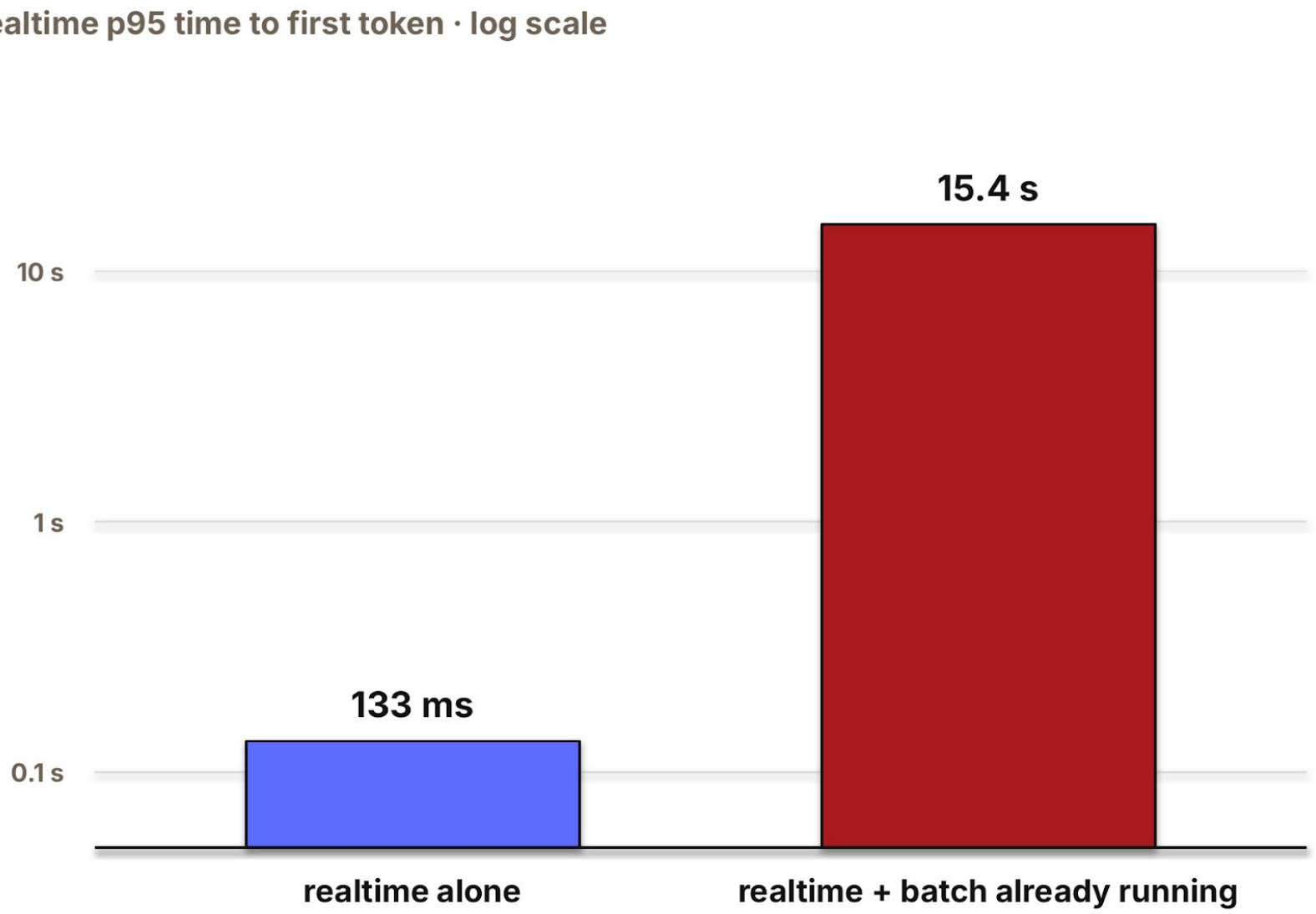
Fairness policies





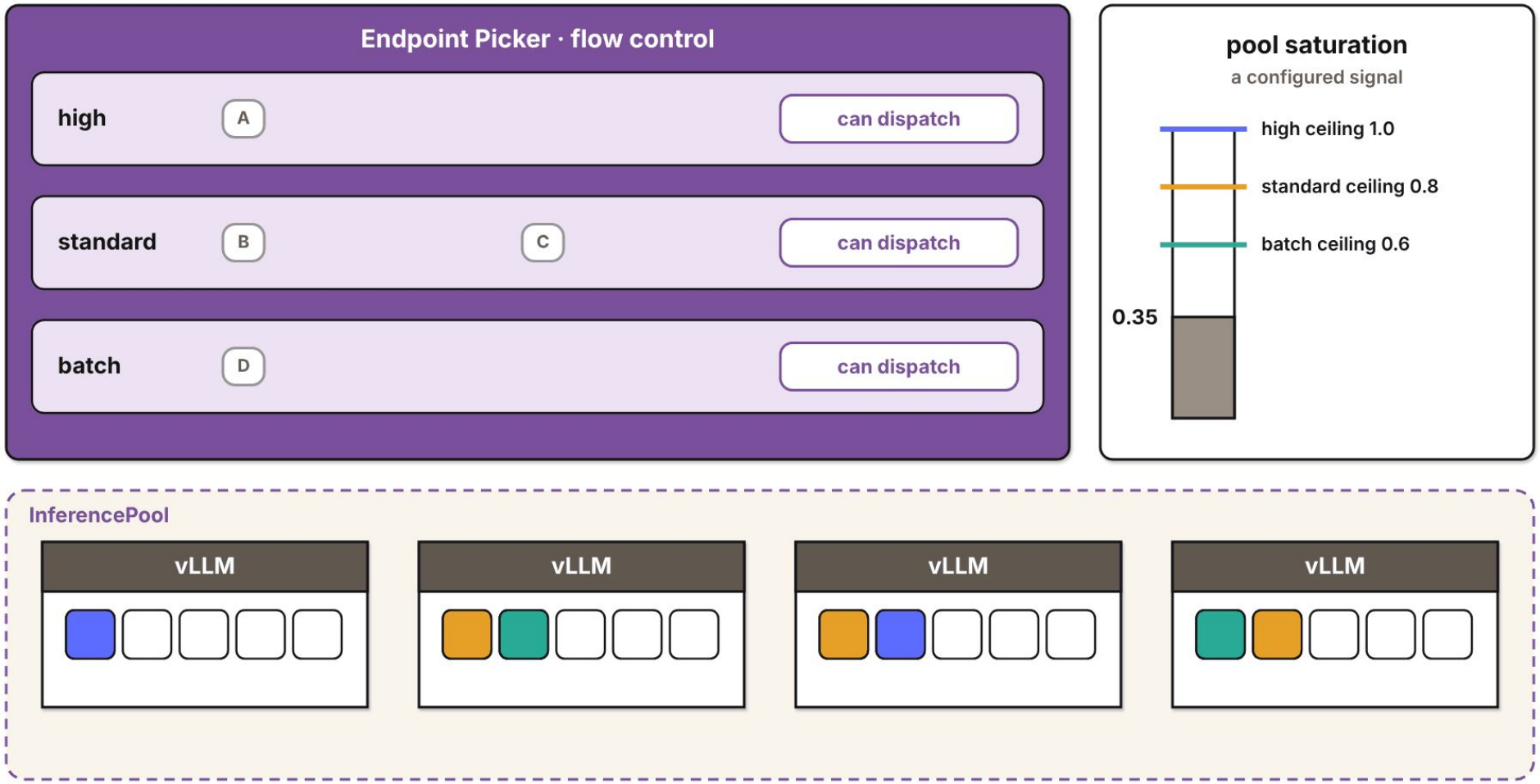
Priority holdback





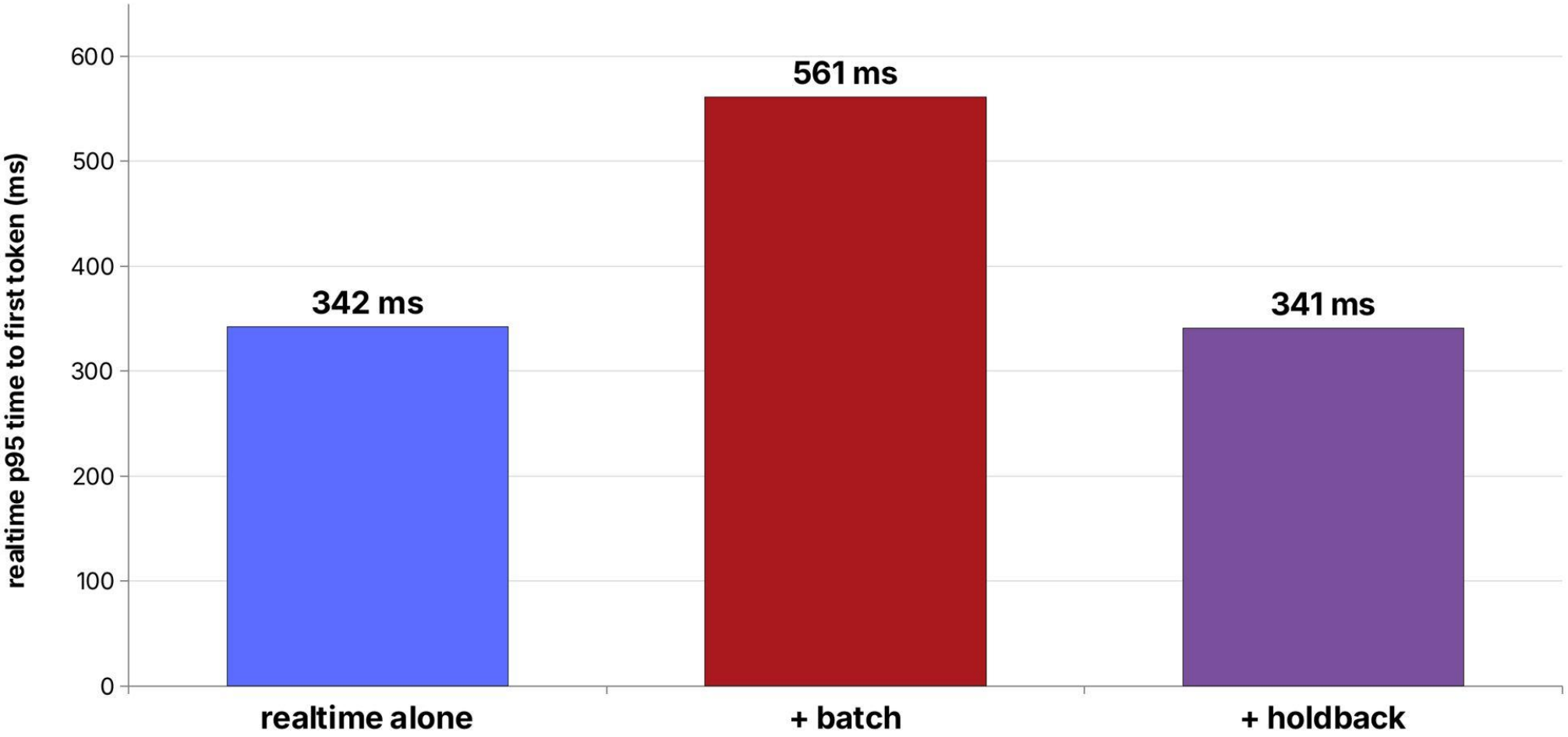
Priority holdback





Priority holdback

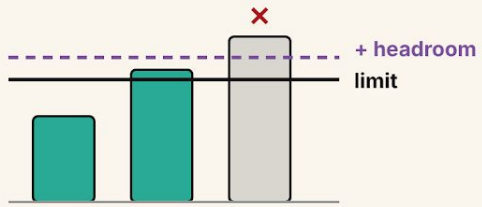




Also tested



Headroom



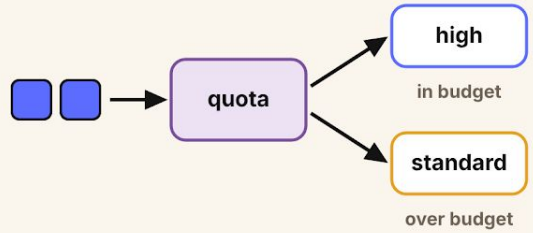
slack above each replica limit

SLO deadline ordering



tightest objective first

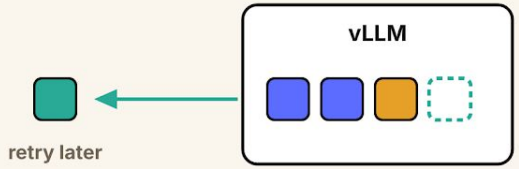
Quota-based priority



in-budget work goes first

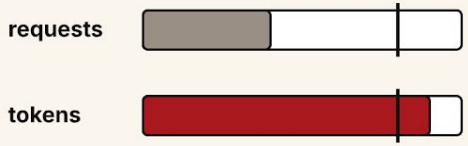
Batch eviction

experimental



free room from running batch

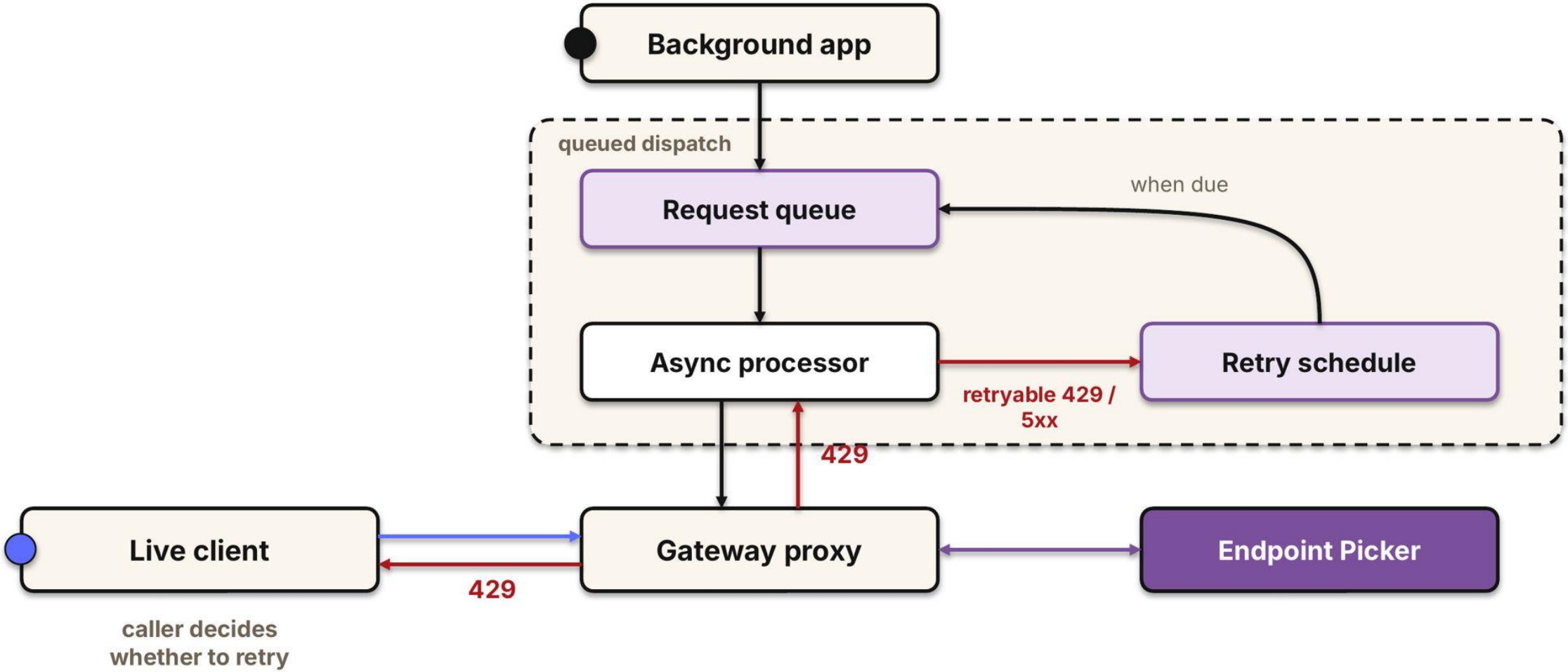
Hybrid admission (P/D)



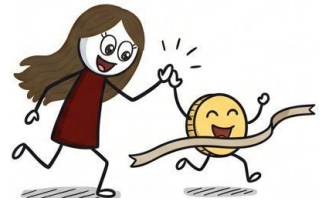
limit requests and tokens

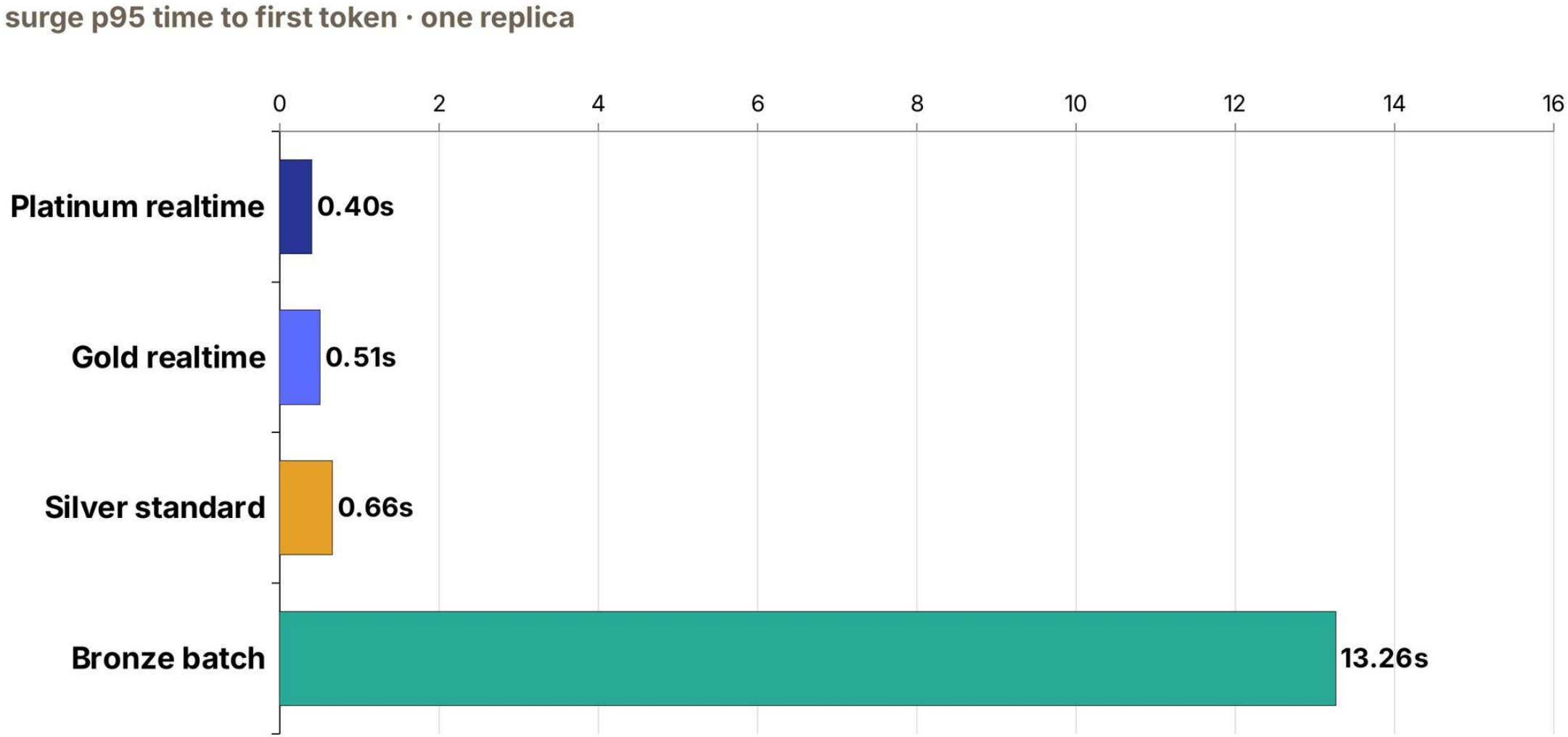
Overload handling





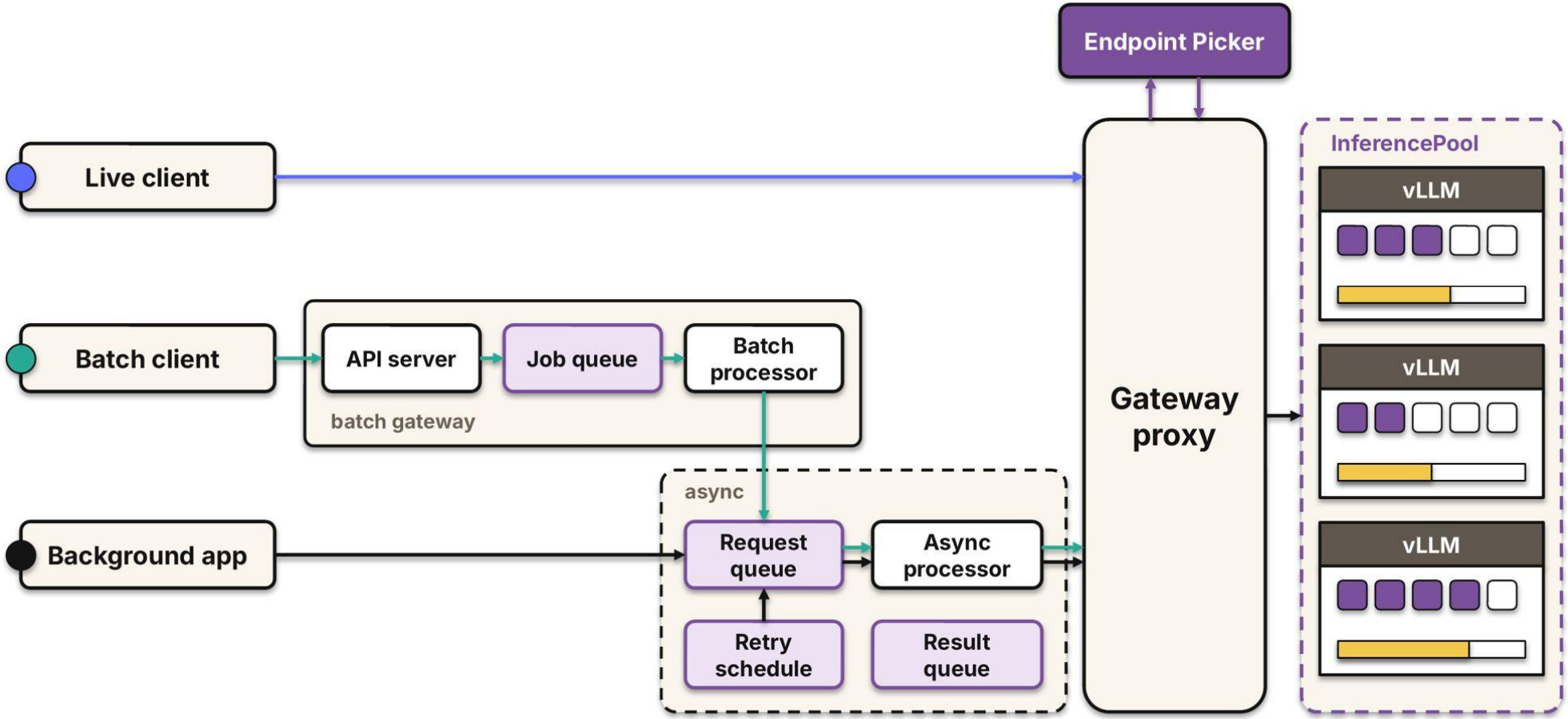
Benchmarking flow control





Putting it all together





llm-d metrics

Metric	What it shows	Why look
`llm_d_epp_flow_control_queue_size`	Requests waiting in the Endpoint Picker, by priority	Is work queuing before it reaches vLLM?
`llm_d_epp_flow_control_request_queue_duration_seconds`	How long requests waited to be dispatched	How much of time to first token is admission wait?
`llm_d_epp_flow_control_pool_saturation`	The configured saturation signal	Is flow control holding work back?
`llm_d_epp_prefix_indexer_hit_ratio`	Router's estimate of prefix reuse	Is prefix-aware routing finding warm pods?
`vllm:prefix_cache_hits / queries`	Actual cache reuse inside vLLM	Does the estimate match real reuse?
`vllm:num_requests_waiting / running`	Queueing inside the engine	Is work piling up past admission?
`vllm:kv_cache_usage_perc, num_preemptions`	KV memory pressure	Are requests being preempted and recomputed?

Replay a run

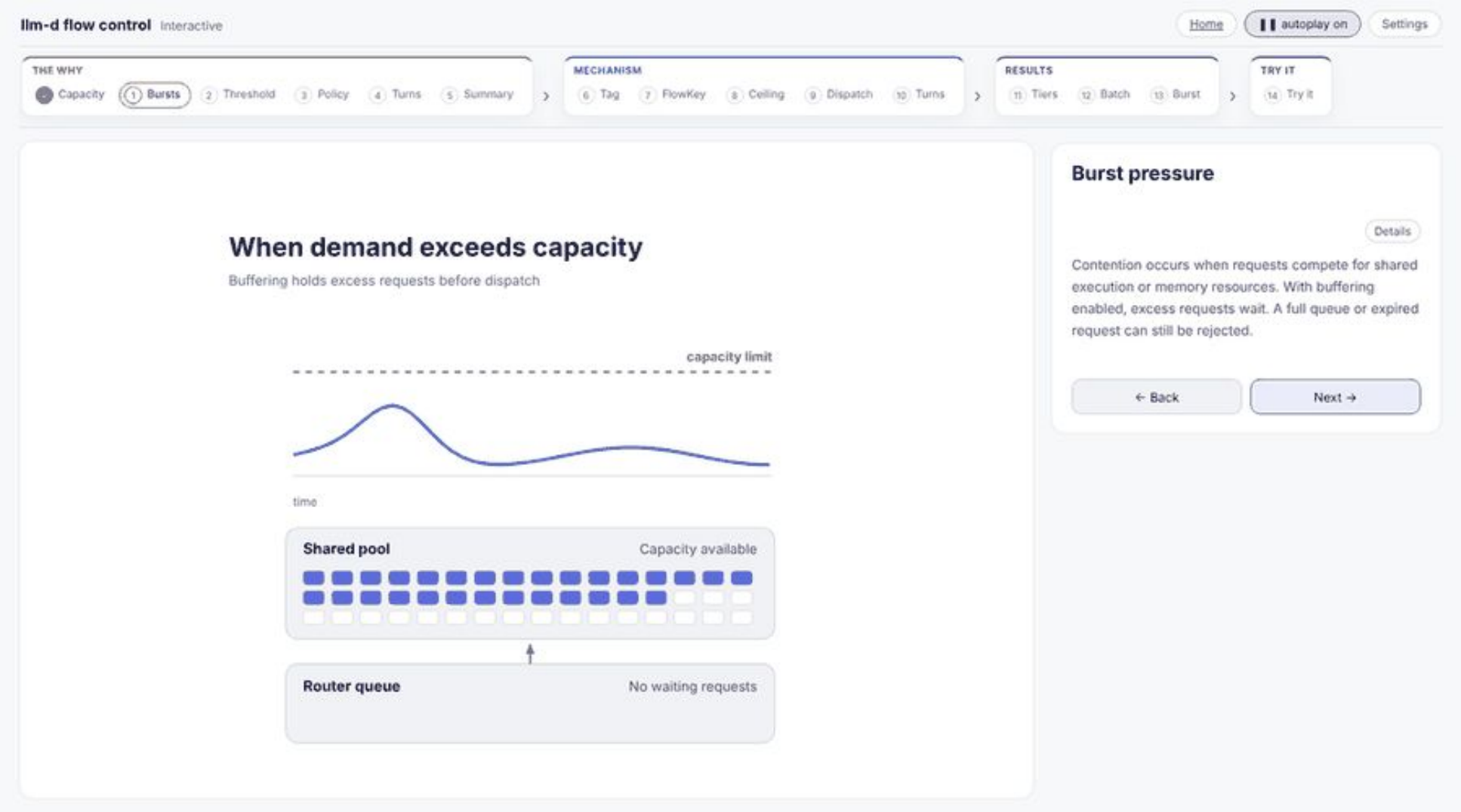


Flight Recorder · synthetic demo

Inference decision map

What you observe	What to consider	What to measure
Repeated prefixes are being recomputed	**Prefix-aware routing**	Cache reuse and TTFT
Reusable prefixes get evicted from GPU cache	**KV cache offloading**	Reload cost and TTFT
Work can finish later	**Async processing / batch gateway**	Completion time and deadlines
Batch traffic delays interactive requests	**Priority + holdback**	Interactive latency and batch progress
One tenant crowds out peers at the same priority	**Tenant fairness**	Per-tenant latency and completions
Requests wait beyond a useful deadline	**Queue limits + wait budget**	Queue time, rejections, completions
Prefill interferes with decode	**P/D disaggregation**	Token latency, TTFT, transfer overhead
MoE experts need distributing across GPUs/nodes	**Wide expert parallelism**	Throughput, latency, communication cost

Try it yourself



interactive explainer

benchmark repo

Join Merge Forward!

Building a more sustainable open source future together!

#merge-forward on the CNCF Slack!

ESCAPE PARTY ROOM!

Not your average conference party!

Half **PARTY**, half **INVESTIGATION**.

Wed, Nov. 11 @ 7pm

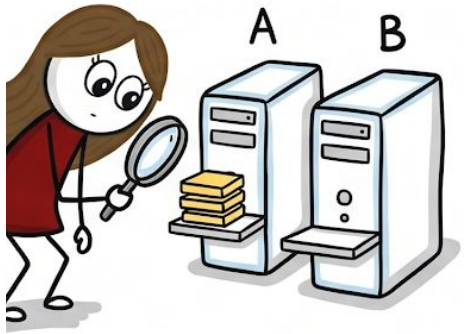
Enlist today

👈 (it's free!)



MERGE FORWARD | BUOYANT | Teleport | ProsperOps a Flexera company | dash0 | civo

Thank you!



alexagriffith.com/kubecon

llm-d course

Careers: Learn more about Red Hat

