DIGITAL THOUGHT DISRUPTION

Technical white paper

The Cost of a Useful Answer

API pricing and GPU capacity for enterprise inference

Paul Bryant | November 23, 2026 | DOI 10.5281/zenodo.24511860

An inference offer can incur charges and still yield no useful answer. This paper models the cost of delivering correct, filing-backed answers within a deadline. It compares hosted APIs, rented GPUs, and owned GPU capacity under steady load, failover, and burst traffic.

Executive summary

Under the reference assumptions, DeepInfra has a lower cost per useful answer than any of the modeled rental or ownership options. The comparison asks what it costs to deliver a correct, filing-backed answer within 20 seconds of the scheduled offer. Every original offer counts toward the joint pass rate. [1, 2, 3]

The comparison uses published rates for two hosted gpt-oss-120b routes and a Runpod Secure Cloud H100 PCIe rental. The reference case uses 8,192 uncached input tokens and 1,024 billed output tokens per offer, a 95% joint pass rate, and 0.50 useful answers per second per active GPU. It uses 730 paid hours per month and no failover reserve. These assumptions define the comparison. The token counts represent long-context pricing. Each filing fixture contains 33 to 52 words across the question and evidence. [1, 2, 3]

Key findings

The hosted rates differ sharply. Together AI costs $1.940 per 1,000 useful answers and DeepInfra costs $0.502 under the reference billing and pass assumptions. DeepInfra has the lower modeled cost across all 12 base rental and ownership fleet sizes, from 1 to 12 paid GPUs. [1, 2, 3]

Rental savings depend on fleet size and useful throughput. One rented H100 PCIe reaches its modeled capacity of 1.314 million useful answers per month before it reaches cost equality with Together AI. The first feasible interval uses two GPUs and spans 2.484 million to 2.628 million useful answers per month. At the upper bound, modeled savings are about $279 per month. Reserving for one replica failure shifts the first crossover to 9.008 million useful answers and eight paid GPUs. The reference rate of 0.50 useful answers per second per GPU is a model input that still needs to be measured on the H100 PCIe. In Table 6, none of the 12 fleet sizes beats Together AI on rental cost at 0.40, while one rented GPU does at 0.60.

Burst demand can leave paid capacity idle. For one million useful answers over a 160-hour window shaped by the BurstGPT API trace, the model assigns about 142,000 to one always-on GPU and about 858,000 to API overflow. Hybrid delivery costs $3,140.86 per month against $502.30 for API-only DeepInfra in that scenario. [2, 3, 5]

Contents

Executive summary 2

Scope and methodology 4

What counts as a useful answer 5

Filing workload and scoring 6

Hosted inference costs 7

Caching retries and billed reasoning 8

GPU capacity costs 9

Load and failover sensitivity 10

Burst demand and operating costs 11

Scheduling the GPU 12

Review and exception handling 13

Costs over 36 months 14

Capital risk and spare capacity 15

Reading public benchmarks 16

Proposed matched test 17

Reproducing the reported results 18

Using the model for a deployment decision 19

Conclusion 19

References 20

Scope and methodology

Scope and source selection

The workload answers questions using evidence from original annual filings by Apple, Microsoft, NVIDIA and Alphabet. It covers lookups, arithmetic, footnotes, abstentions and counterfactual controls. The pricing snapshot is November 22, 2026. [1, 2, 3, 12, 13, 14, 15]

Relation to prior work

The FinOps Foundation defines cost per successful outcome across all runs [6]. Thinkingbox uses observed token usage to estimate cost per successful attempt and measures consistent success across trials [7]. DistServe defines serving goodput under first-token and per-output-token latency constraints [8]. Do Tuan Anh compares sourcing options across token demand [9]. This paper compares the cost and capacity to deliver useful answers within an end-to-end deadline.

The comparison holds useful-answer demand equal across routes. The base case assumes steady traffic, the hybrid case varies arrivals, and the 36-month case scales GPU capacity with demand.

Model inputs

Input	Value	Basis
Tokens per offer	8,192 uncached input / 1,024 billed output	Scenario input
Joint pass rate	95% floor and modeled pass rate	Assumption
Useful answers per second per GPU	0.50 per active GPU	Assumption
Paid hours per month	730 base / 160 scheduled case	Scenario input
Traffic hours per month	730 base / 160 hybrid case	Scenario input
H100 PCIe Secure Cloud rental	$2.89 per GPU-hour	Published rate [3]
Shared monthly labor	$600 (4 hours at $150/h)	Scenario input
Installed capital	$40,000 per GPU over 36 months	Scenario input
Discount rate	8% effective annual	Scenario input
Rack and support	$250 per GPU per month	Scenario input
Power and electricity	0.65 kW per GPU, 1.4 facility multiplier, $0.12/kWh	Scenario input
Spare capacity displacement	$500 per GPU-month	Scenario input

Limits of the comparison

The study includes no H100 PCIe measurements. The rehearsal used fixtures and a local HTTP mock, so production quality remains unmeasured. The model assumes equal acceptance for the two hosted routes and holds labor constant regardless of fleet size. The BurstGPT trace records hosted language model usage [5]. The model uses that arrival shape as a sensitivity, separate from the filing workload.

How to reproduce the estimates

Every derived figure can be recomputed from the Model inputs table, the formulas in this paper and the cited public rates. The demand-shape figures can be recomputed from the public BurstGPT trace as described in Reproducing the reported results. Tables report USD and round displayed values after calculation.

What counts as a useful answer

The unit of work is one original scheduled offer. An answer passes when its result, period, unit and evidence citation meet the case rubric and the complete response is available within 20.000 seconds of scheduled arrival. An explicitly unanswerable case can pass with a timely abstention that meets its rubric. A retry retains the original clock. Errors, missing responses, and timeouts stay in the original offer count.

Filing case	Scoring outcome
Apple FY2025 net sales	USD 416,161 million for the year ended September 27, 2025, citing the supplied evidence from Note 2 of the Form 10-K. [12]
Correct result at 20.000 seconds	Passes if the period, units and evidence also pass.
Same result at 20.001 seconds	Fails the deadline and contributes zero useful answers.
Correct figure with the wrong fiscal period	Fails the answer rubric even if the response arrives on time.

Table 1. One filing lookup and three boundary examples. The clock starts at the scheduled arrival, so elapsed time can include queueing.

Score correctness and timing together

Consider 100 offers. If five fail the evidence and correctness check and five different offers miss the deadline, each separate pass rate is 95%. Only 90 offers pass both. A fixture in the scorer exercises this case. The joint rate has to come from the intersection of outcomes for the same offers.

Joint pass rate = useful answers ÷ original offers

Useful goodput = useful answers ÷ elapsed observation seconds

Cost per useful answer = total in-scope cost ÷ useful answers

Charge all billed attempts, including failures and billed retries. Add any operating or review costs included in the comparison. Count at most one useful answer per offer. Cost per useful answer is undefined when the sample yields zero useful answers.

The reference model uses a 95% joint pass rate for cost eligibility. A deployment choice also needs enough useful goodput under its actual load. A route that misses the quality and deadline requirement cannot satisfy it by lowering its tariff.

Filing workload and scoring

The corpus uses original 10-K filings from Apple, Microsoft, NVIDIA and Alphabet. The cases test whether a response can preserve the meaning of a financial fact across periods, units and evidence. A figure may be arithmetically correct and still answer the wrong question if it uses the wrong fiscal year, a different reporting basis, or percentages where the case asks for percentage points. [12, 13, 14, 15] Newer annual filings exist for NVIDIA (fiscal year ended January 25, 2026), Microsoft (fiscal year ended June 30, 2026), and Apple (fiscal year ended September 26, 2026) and are not used.

Case type	Cases	What makes the answer useful
Lookup	10	Return the specified line item in the expected units and period.
Calculation	8	Apply the requested arithmetic to the supplied values and reporting periods.
Footnote	5	Use a qualifying note or definition that changes the interpretation.
Abstention	4	Use the prescribed abstention when the supplied evidence cannot support the request.
Counterfactual	3	Use the stated synthetic change and keep it distinct from filed facts.

Table 2. The question set contains 27 filing-based cases and three labeled synthetic controls. A case in each category carries its own expected answer and evidence identifier.

How answers are scored

The scorer compares the expected response status, unit, period, and evidence identifier. Numeric answers are parsed as finite decimals and compared to the expected value. The gold values remain outside the input. A model receives the supplied evidence and the question. This tests whether it uses that evidence correctly, with retrieval quality outside the scope of the task.

Fixture and mock results

All 30 known-result fixtures and 10 targeted edge cases matched their expected scoring outcomes. In the rehearsal, a structured lookup and arithmetic baseline matched the expected values for all 21 cases that had a preassigned operation and structured inputs: 10 filing lookups, 8 filing calculations and 3 synthetic counterfactuals. The five footnote cases and four abstentions had no such operation. The remaining response fields came from the fixture. The exercise establishes a deterministic alternative for selected facts and calculations, with automatic routing and production quality left for measurement. These fixture, scorer and mock results come from an unpublished rehearsal of the scoring method and do not measure model quality or production capability.

A local HTTP mock produced 30 unique records, including 26 expected useful answers, two HTTP errors, one invalid response and one timeout. The mock exercised overlapping requests, response parsing and the retention of returned usage on failed attempts. Those counts describe transport and scorer behavior under controlled fixture inputs.

Hosted inference costs

The reference hosted scenario uses one billed attempt for every offer on a listed gpt-oss-120b endpoint. Each attempt has 8,192 uncached input tokens and 1,024 billed output tokens, including billable reasoning where applicable. Both routes use the same assumed 95% joint acceptance. Endpoint configurations, billed usage and observed quality can differ even when providers list the same model family. [1, 2, 4]

Hosted route	Input per 1M	Output per 1M	Cost per 1,000 useful answers
Together AI [1]	$0.150	$0.60	$1.94021
DeepInfra [2]	$0.037	$0.17	$0.50230

Table 3. Posted uncached and output tariffs and the modeled cost per 1,000 useful answers. DeepInfra posts $0.019 per million cached input tokens [2]. The reference model uses zero cached input. [1, 2]

Cost per offer

At DeepInfra's tariff, an attempt costs (8,192 × $0.037 + 1,024 × $0.17) ÷ 1,000,000, or $0.000477184. Dividing by 0.95 gives $0.000502299 per useful answer. The same calculation for Together AI gives $0.00184320 per attempt and $0.00194021 per useful answer. [1, 2]

One million original offers produce 950,000 useful answers under the reference pass assumption. The corresponding token bills are about $1,843.20 on Together AI and $477.18 on DeepInfra. A demand target of one million useful answers instead implies approximately 1,052,632 offers in expectation at the same joint pass rate. The capacity and 36-month sections use targets expressed in useful answers.

Use actual billed tokens

The price calculation represents a long-context billing scenario. The filing fixtures are short, with 33 to 52 whitespace-separated words in the supplied question and evidence before system instructions. A production request may include retrieved text, instructions and billed reasoning. Reprice measured acceptance and goodput with the exact token classes charged by the provider, including cache hits and attempts that return no useful answer.

A retry can raise the joint pass rate if it returns a useful answer within the original deadline. Its billed work raises the numerator even if it fails. The net effect depends on the number of additional offers that yield a useful answer and the extra charges.

Caching retries and billed reasoning

A blended input rate can change the hosted cost even when the posted uncached price remains the same. Let the cached share be the fraction of input tokens the provider actually invoices at the cache-read rate. In the DeepInfra example, the blended rate per million input tokens is $0.037 multiplied by the uncached share plus $0.019 multiplied by the cached share. Output billing and the 95% pass assumption stay fixed. [2]

Input billed as cached	Cost per 1,000 useful answers	Savings at 1M useful answers
0%	$0.50230	$0.00
25%	$0.46349	$38.80
50%	$0.42469	$77.61
75%	$0.38589	$116.41
100%	$0.34708	$155.22

Table 4. Illustrative cache billing at 8,192 input and 1,024 billed output tokens. The 100% row is a pricing bound under these inputs. Actual cache eligibility and billed classes depend on the request and provider. [2]

When a retry pays off

Suppose 95% of offers initially yield useful answers and 5% of offers receive a retry. If 60% of those retries produce a useful answer before the original deadline, overall acceptance rises to 98%. At the full attempt price, DeepInfra moves from $0.50230 to $0.51127 per 1,000 useful answers. If each retry instead costs one-quarter of the initial attempt, cost falls to $0.49301. Both alternatives use the same conditional recovery rate for illustration. [2]

The added spend is the number of retried offers times the billed retry cost. The additional output is the number of retries that turn a failed original offer into a useful answer. A retry policy can justify a higher unit cost if the increase in acceptance closes a service gap. Compare both outcomes when evaluating the policy.

Cost of extra output tokens

At DeepInfra’s $0.17 output tariff, 512 more billed tokens per original offer add about $0.092 per 1,000 useful answers at a 95% pass rate. Measure the effect of a higher reasoning budget on correctness and latency on the same offers before treating the added token spend as an improvement. [2]

GPU capacity costs

A Runpod Secure Cloud H100 PCIe at $2.89 per GPU-hour costs $2,109.70 for 730 paid hours. Four hours of shared monthly labor at $150/h bring the cost to $2,709.70. The model holds this labor cost constant regardless of fleet size, which favors larger fleets. The owned case uses $40,000 installed capital per GPU over 36 months at 8% effective annual, $250 per GPU per month for rack and support, and the same shared labor. Energy costs $79.72 per month at 0.65 kW per GPU, a facility power multiplier of 1.4, and $0.12/kWh. [3]

One GPU	Monthly cost	Together AI equality	DeepInfra equality
Rented	$2,709.70	1.397M / 0.531/s	5.395M / 2.053/s
Owned	$2,178.03	1.123M / 0.427/s	4.336M / 1.650/s

Table 5. Theoretical monthly price equalities and required useful answers per second over 730 hours. Owned cost includes $1,248.31 in monthly capital recovery.

Divide GPU fixed cost by the hosted unit cost minus the GPU variable unit cost, which is zero here. A fleet saves money where it can serve demand above that equality.

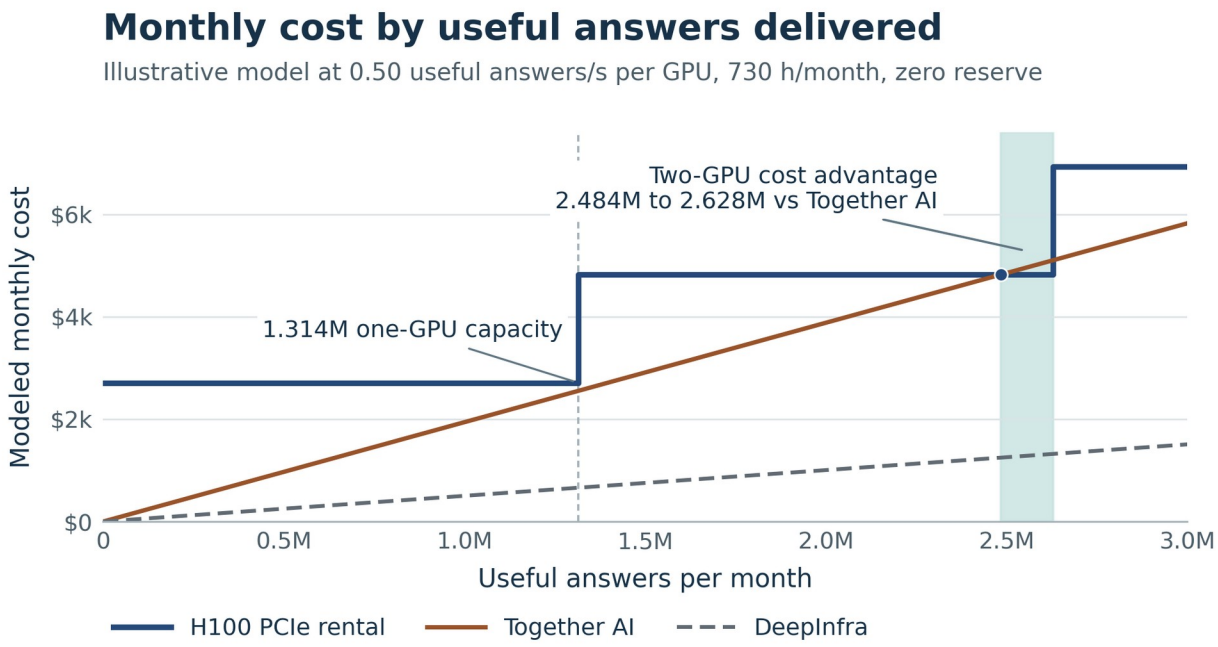


Figure 1. Modeled monthly rental and API costs at 0.50 useful answers/s per active GPU, 730 paid and traffic hours, and zero failover reserve. [1, 2, 3]

The model assigns an active GPU a capacity of 1,314,000 useful answers over 730 hours at the assumed goodput. Two rented replicas cost $4,819.40 and undercut Together AI at 2,483,958 to 2,628,000 useful answers. One owned GPU undercuts Together AI at 1,122,572 to 1,314,000. Demand above a fleet's capacity requires additional capacity.

Reserving capacity for one replica failure reduces active capacity at each paid fleet size. At the same goodput, the first rented crossover against Together AI moves to eight paid GPUs, with seven surviving and a range of 9,008,095 to 9,198,000 useful answers.

Load and failover sensitivity

Capacity is the number of replicas that remain available, multiplied by useful answers per second per replica and the usable traffic seconds. The rates below count useful answers only. The table assumes linear scaling across replicas, steady load during 730 hours, the same $2.89 rental rate, and $600 of shared monthly labor. All routes are assumed to meet the 95% pass floor. [1, 2, 3]

Useful answers/s per GPU	Capacity per active GPU	First cheaper fleet, zero reserve	First cheaper fleet, one reserve
0.40	1,051,200	None in 12 fleet sizes	None in 12 fleet sizes
0.50	1,314,000	2 paid GPUs, 2.484M to 2.628M	8 paid GPUs, 9.008M to 9.198M
0.60	1,576,800	1 paid GPU, 1.397M to 1.577M	4 paid GPUs, 4.659M to 4.730M
0.80	2,102,400	1 paid GPU, 1.397M to 2.102M	3 paid GPUs, 3.571M to 4.205M
1.00	2,628,000	1 paid GPU, 1.397M to 2.628M	2 paid GPUs, 2.484M to 2.628M

Table 6. First modeled rental fleet size with a cost advantage over Together AI. The intervals show monthly useful answers and use rounded values for display. DeepInfra remains below rental in the 12 paid fleet sizes evaluated at each displayed rate.

Throughput with a spare GPU

Two full-time rented GPUs incur $4,819.40 a month. Against Together AI, they need about 0.473 useful answers per second each if both can serve traffic at full utilization. Requiring the same paid pair to carry its load after one failure puts the entire requirement on one survivor, about 0.945 useful answers per second.

At 0.50 useful answers per second per active GPU, the two-GPU base interval saves at most $279.47 a month, reached at 2,628,000 useful answers. An additional recurring GPU-only cost of $279.48 a month removes the advantage for that fleet size under the other assumptions.

Shared labor and one-GPU costs

At full use and before shared labor, $2.89 per GPU-hour requires about 0.414 useful answers per second to match Together AI and 1.598 to match DeepInfra. Including $600 of monthly shared labor increases the one-device hurdles to 0.531 and 2.053, respectively. A queue, tenant limit, or common-mode failure can further reduce the capacity that meets the answer deadline. Measure the usable rate at each intended fleet configuration.

Burst demand and operating costs

Paid hours and traffic hours

The monthly model can pay for 730 hours even when offers arrive in a shorter window. At 0.50 useful answers per second, one active GPU can deliver at most 288,000 useful answers over 160 traffic hours. Variability within that window can lower utilization further. Idle capacity in one period cannot serve an offer that already missed its deadline in another.

The BurstGPT sensitivity takes 168 hourly bucket weights from the median-volume complete week in the v2.0 API trace. Its 132,596 arrivals supply relative weights, including buckets with no arrivals. The model distributes one million target useful answers over those weights and fits the 168 buckets into a 160-hour traffic window. Within each bucket, local service is capped at the active GPU's modeled capacity and the model assigns the excess to the API. [5]

One million useful answers	Estimate
Local answers with one always-on GPU	141,637
Useful answers assigned to API overflow	858,363
Peak overflow averaged over one modeled bucket	35.517 useful answers/s
GPU rental and shared labor	$2,709.70
DeepInfra overflow token cost	$431.16
Hybrid total	$3,140.86
DeepInfra API only	$502.30

Table 7. One GPU at 0.50 useful answers/s, 730 paid hours, 160 traffic hours and no failover reserve. The peak is an average over a rescaled bucket of about 57 minutes. Sizing quotas and assessing queues or deadline compliance require finer-grained observations. [2, 3, 5]

In this case the model assigns the GPU enough work to avoid about $71 of hosted token spend while incurring $2,709.70 in fixed rental and labor. Scheduled rental or already available GPU capacity would change those fixed costs and should be priced against its own operating window.

Scheduling the GPU

The reference hybrid case pays for one GPU around the clock while assigning traffic to 160 hours. An alternative buys only those 160 hours at the same hourly rate. It assumes the same GPU is ready for every modeled bucket, and holds labor, demand, throughput and overflow prices constant. This isolates the value of avoiding 570 hours of paid idle time. [2, 3, 5]

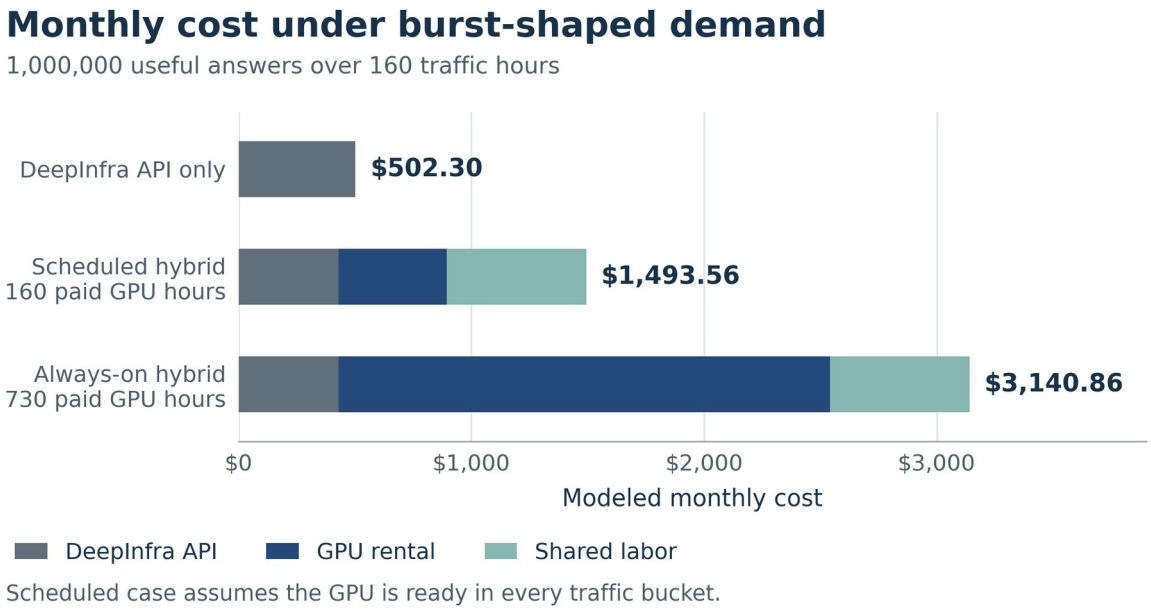


Figure 2. Modeled monthly costs to deliver one million useful answers under the BurstGPT shape. The scheduled case assumes the GPU is ready across the full traffic window. Both hybrid cases use the same local and overflow volumes. [2, 3, 5]

A shorter rental reduces fixed cost

At 160 billed hours, GPU rental is $462.40. With $600 in labor and $431.16 in overflow, scheduled hybrid totals $1,493.56. It saves $1,647.30 versus the 730-hour hybrid case. API-only delivery remains $991.26 cheaper because the model assigns about 141,637 useful answers to the GPU, displacing roughly $71.14 of API charges. [2, 3]

How steady traffic changes overflow

If the same million useful answers arrive uniformly over 160 hours, the model assigns 288,000 to the GPU and 712,000 to overflow. Scheduled hybrid would then cost $1,420.04 and always-on hybrid $3,067.34. The uniform case removes the within-window variation in the trace. Scheduling in production requires capacity to be ready at the start of the window, and realized startup time, minimum commitments and rental availability can affect the result.

Review and exception handling

The price comparison covers usage and the modeled fleet costs. A workflow can also pay people to audit the result or resolve an exception. The measurement boundary matters. A post-delivery audit adds expense to the numerator. Review required before delivery adds both expense and elapsed time to the original offer, so it changes the joint pass rate as well.

Extra asynchronous audits per 1,000 useful answers	Added cost	DeepInfra total per 1,000 useful answers
0	$0.00	$0.502
0.5 average	$1.00	$1.502
1	$2.00	$2.502
2	$4.00	$4.502

Table 8. Review sensitivity at two minutes per audit and $60 per labor hour. The reference Together AI cost is $1.940 per 1,000 useful answers. The sensitivity assigns a difference in audit load to one route for illustration. [1, 2]

The tariff gap can fund less than one extra audit

The hosted rates in the reference case differ by $1.438 per 1,000 useful answers. At $2 an audit, that budget covers about 0.719 additional audits per 1,000 useful answers. A fractional figure is a rate over many answers. Across 100,000 useful answers, the same tariff gap funds about 72 such reviews.

Count fallback costs on the original offer

Consider an offer that first receives a failed result and then goes to a separate model or a human. Assign the charges from both routes to the original offer. If the delayed result fails the 20-second deadline, record the recovery as an operational outcome and keep it outside the useful-answer count. A separate service objective can account for that recovery value.

Workflow differences should be supported by records of why people intervened, how long they spent, and whether the work cleared the actual release deadline. A cheaper token route can remain cheaper with higher audit load if the added cost fits inside the observed price gap. A higher acceptance rate can also justify a higher cost when the alternative misses the required service level.

Costs over 36 months

The same 36-month workload is priced across routes. Monthly demand is 0.5 million useful answers in year one, 1.0 million in year two, and 1.5 million in year three. Sustaining that final level requires a second GPU at the assumed 1.314 million useful answers per active replica. The model keeps 730 hours per month, uniform load, and zero failover reserve.

Route	Present cost
DeepInfra with 25% lower token prices from month 13	$12,484.99
DeepInfra at reference token prices	$15,682.83
Existing spare GPUs, incremental cash costs	$22,567.20
Existing spare GPUs, including displacement cost	$43,524.07
Together AI at reference token prices	$60,577.47
Staged purchase	$107,339.19
Runpod on-demand rental	$107,651.44

Table 9. Present costs at an 8% effective annual discount rate. The price reduction is a sensitivity. The analysis assumes two GPUs are available for the spare-capacity route across the study horizon. [1, 2, 3]

When to buy more GPUs

The staged purchase buys one GPU at the start of month 1 and another at the start of month 25. Operating costs occur at month-end and the GPUs have zero residual value at the end of month 36. Purchase prices enter as cash outlays. The monthly capital recovery from the one-GPU comparison is excluded from these cash flows to avoid counting the same purchase twice.

The cash view of existing spare GPUs includes incremental energy and shared operations labor. The economic view also assigns $500 per GPU-month to the workload displaced by using that capacity. At the assumed goodput, demand and tariffs, the DeepInfra token-only case costs less than even the incremental spare-capacity case. The Together AI route costs more than both spare-capacity cases. [1, 2]

Effect of a lower API price

The 25% token price reduction lowers the DeepInfra present cost by about $3,198 versus a flat tariff. It makes the comparison more sensitive to how long hardware costs are committed and how early added capacity is paid for. The model leaves contract discounts, actual procurement quotes and residual value for the buyer to supply.

Capital risk and spare capacity

The reference present-cost difference between staged buying and full-time rental is $312.25 over 36 months. Both routes assume the same useful throughput and joint pass rate. The difference is small beside uncertainty in procurement, support charges, power, utilization, resale and any capacity required during a failure.

36-month ownership scenario	Present cost	Change from base buy
Base buy, $40,000 per GPU and zero residual	$107,339.19	$0.00
Capital raised to about $40,168 per GPU	$107,651.44	+$312.25
Base buy with $5,000 end value per GPU	$99,400.86	−$7,938.32

Table 10. Isolated capital and residual sensitivities for the staged buy. The middle row uses the unrounded installed cost that gives equality with rental. Added end value is a scenario, with other costs and deployment dates held constant.

Installed cost can reverse the ranking

The two purchases occur at the start of the study and two years later. A $1 increase in installed price per GPU therefore increases present cost by about $1.857. Raising both purchase prices by about $168.12 closes the $312.25 difference. Conversely, assigning $5,000 of end-month-36 value to each of the two purchased GPUs reduces present cost by approximately $7,938. Residual value is available only if the buyer can realize or credit it.

Spare fleet capacity still carries costs

In the demand ramp, an existing fleet makes 41.91 discounted GPU-months available for the workload. The $22,567.20 cash estimate includes energy and shared labor. The $500 per GPU-month displacement assumption adds about $20,957 of present cost. The displaced work may be negligible on an idle device or material on a fleet that fronts another service.

At the reference tariffs, the spare fleet matches Together AI at a displacement cost of about $907 per GPU-month. Against DeepInfra, the spare fleet's incremental cash cost alone exceeds the modeled hosted cost. Confirm how much capacity would actually be available at peak before using a spare-fleet estimate in a procurement decision. [1, 2]

Reading public benchmarks

Published benchmarks answer questions about memory fit, latency or token throughput under named configurations. The useful-answer model needs a different observation. It needs the number of original offers that yield useful answers at a given offered load.

Published evidence	What it measures	Use in this comparison
NVIDIA DGX Cloud Lepton [4]	Deployment requires at least one GPU with 80 GB of memory.	Supports testing a one-device configuration.
NVIDIA TensorRT-LLM [10]	H100 SXM DEP4 FP8 generated output token throughput for specified sequences.	Shows throughput sensitivity to input length under that topology.
InferenceX archived setup [11]	H100 vLLM FP4 configurations with tensor-parallel sizes 2, 4 and 8.	Identifies multi-device serving conditions that a single-device trial must distinguish.
DistServe [8]	Serving throughput within first-token and per-output-token latency constraints.	Supports measuring throughput under explicit service limits.

Table 11. The public results provide context for a test on one H100 PCIe. The priced device and acceptance rule define the comparison boundary.

Effect of longer inputs

NVIDIA’s DGX H100 SXM DEP4 FP8 table reports 1,169 generated output tokens per second per GPU for 8,192 input and 1,024 output tokens. With 32,768 input tokens and the same output length, it reports 333. The ratio is about 3.5 to 1 on those configurations. The comparison illustrates sensitivity to sequence shape even within a benchmark, before changing device type or serving topology. [10]

Compare throughput and answer quality

Normalizing output throughput by GPU count helps compare test configurations. A single H100 PCIe can have a different memory budget for batching and a different way of handling input processing, generation and queueing. Dividing the published token rate by an output length estimates a token-based request rate under additional assumptions. It leaves evidence quality and the end-to-end deadline unresolved.

Report first-token time and generation rate as diagnostics. Preserve response completion time from original scheduled arrival and the rubric decision on each offer to calculate the useful goodput used in this paper.

Proposed matched test

A matched run would serve a pinned gpt-oss-120b revision on one H100 PCIe 80 GB with tensor parallelism of one. Record the exact device, available memory, driver, runtime and model revision. Pin quantization, cache policy, batching, context limit and reasoning budget across the load points. Present the same task, evidence and acceptance criteria to any hosted endpoint included for comparison.

Test around the break-even load

The proposed test uses 30 separate warm-up requests, followed by ten seeded rounds of the 30 cases at each offered rate. Start at 0.25 offers per second, then use 0.56 and 2.17 offers per second to test around the one-GPU hurdle for Together AI and DeepInfra. The higher points come from dividing 0.531 and 2.053 useful answers per second by 0.95 and rounding upward. They are test targets derived from the price scenario.

Record	Role in the comparison
Original offer ID and schedule	Counts every offer and fixes the deadline.
Submission, worker start and completion	Includes waiting time and exposes queuing on the load generator.
Final answer, rubric fields and error	Supports the joint pass rate and measures how offers fail.
Provider usage and billing records	Prices all billed attempts, including failures, retries and any cached classes.

Table 12. The open-loop schedule submits offers independently of earlier completions. A 30-second transport budget can capture late responses. The business deadline remains 20 seconds from each offer.

Price the measured results

Report the joint pass rate and useful answers per second together at every offered load, including missing or timed-out offers. Report latency percentiles with their population, such as completed responses only. Compare observed usage with invoiced token classes and flag any missing metering.

The filing set is a controlled workload of repeated compact prompts. Broader claims need an independent holdout and traffic representative of the intended deployment. A matched long-context study also needs its own known answers and observed billing. Reprice compact prompts using their actual token mix before comparing measured goodput with an economic hurdle. The scenarios in this paper remain modeled.

Reproducing the reported results

Reconstruct the demand shape

Use BurstGPT v2.0's BurstGPT_1.csv. From 1,429,737 source rows, retain 1,275,518 API rows across model names, including rows with zero reported response tokens. Among seven complete, nonoverlapping trace-relative weeks after the first week, week 3 is the median by arrival count. Its 132,596 arrivals span the interval from 1,814,400 through the second before 2,419,200 from trace origin. There are 62 empty hourly buckets out of 168. [5]

For each bucket, divide its arrival count by 132,596, then multiply by the one-million useful-answer demand target. Give each bucket 160 ÷ 168 hours of time. Local service is the smaller of assigned demand and 0.50 times the bucket length in seconds. Overflow is the remainder. Sum those quantities and charge the API rate for overflow plus the fixed rental and labor for the GPU.

The busiest source hour has 16,374 arrivals, about 20.75 times its hourly mean. At five-minute resolution, the largest bucket is 2,575, about 39.15 times its five-minute mean. Finer buckets expose more variation. Both values describe the source traffic, while the overflow estimates use the rescaled hourly weights. [5]

Recompute the figures

Recompute prices, replica limits, hybrid allocation and discounted costs from the Model inputs table and the formulas in this paper. Round only for display, after calculation.

Using the model for a deployment decision

Set the workload boundary before comparing prices. Start with representative offers, a measurable deadline, the evidence rubric, and the minimum joint pass rate. Estimate the useful demand by time window, then price the routes that can serve it. Allocate shared costs on the same basis for every eligible comparison.

Decision under consideration	Evidence to check
Hosted API only	Test acceptance at steady and burst load. Reconcile billed tokens, limits and any commitment discounts with useful answers.
Dedicated rental	Measure usable throughput under the deadline. Price the whole paid fleet, idle hours, shared labor and the required surviving capacity.
Purchased or existing GPUs	Compare cash cost with displacement and residual value. Check whether the fleet can be redeployed if demand or prices change.
Hybrid with API overflow	Confirm GPU readiness, local coverage by traffic window, and overflow capacity. Assign rerouting and retries to the original offer.

Table 13. Apply the same answer acceptance rule and original offer count to each route.

Test what could change the decision

Vary demand timing and rate, useful goodput, the cost of an extra paid replica, and the difference in review or exception effort. Include a route that uses existing spare capacity if sufficient availability can be demonstrated. Test a lower hosted tariff and a delayed second purchase before committing to the projected ramp.

For a team close to the rental crossover, the result should be a bounded operating interval with the minimum joint pass rate, required throughput, paid capacity, and a dollar margin. For a team well below the threshold, the next useful evidence is the cost and acceptance of a representative hosted pilot.

Conclusion

The hosted price gap matters more than the choice between renting and buying. Together AI costs about 3.9 times as much per useful answer as DeepInfra, which is cheaper than every modeled rental and ownership fleet size. The modeled GPU options compete only with Together AI, and the first rental crossover occupies a narrow interval that depends on the assumed useful throughput. Over the 36-month demand ramp in Table 9, rental ($107,651.44) and staged purchase ($107,339.19) both cost more than Together AI ($60,577.47), and only existing spare GPUs cost less among the GPU options.

References

[1] Together AI. gpt-oss-120b API pricing. Hosted price snapshot accessed November 22, 2026. Source

[2] DeepInfra. Entry for gpt-oss-120b in the text generation model listing. Published input, output and cached-input rates, accessed November 22, 2026. Listing pages and order can change. Source

[3] Runpod. GPU cloud pricing. Secure Cloud H100 PCIe rate accessed November 22, 2026. Source

[4] NVIDIA. Deploy GPT-OSS-120B. DGX Cloud Lepton deployment guide and GPU memory requirement. Source

[5] HPMLL. BurstGPT v2.0 release, January 15, 2026. API trace release and field definitions. Source

[6] FinOps Foundation. FinOps for AI Overview. Cost per successful outcome. Updated October 28, 2026. Source

[7] Zhuochun Li et al. One Success Isn't Reliability: Thinkingbox, a Sandbox and Benchmark for Agents in Stateful Business Workflows. Thinkingbox preprint v5, section D.6 and equation 14. Revised October 31, 2026. Source

[8] Yinmin Zhong et al. DistServe: Disaggregating Prefill and Decoding for Goodput-optimized Large Language Model Serving. USENIX OSDI 2024. Source

[9] Do Tuan Anh. The Economics of AI Inference: An Efficiency Curve for Choosing Between Credit Plans, Pay-per-Token APIs, and Self-Hosted Models. SSRN working paper, posted July 31, 2026. Source

[10] NVIDIA. TensorRT-LLM Performance Overview. Version 1.2.0rc7. H100 SXM serving measurements. Source

[11] SemiAnalysis. InferenceX NVIDIA configuration. Pinned gpt-oss-120b configuration, May 17, 2026, lines 4436 to 4457. Source

[12] Apple Inc. Form 10-K. Fiscal year ended September 27, 2025. SEC EDGAR. Source

[13] Microsoft Corporation. Form 10-K. Fiscal year ended June 30, 2025. SEC EDGAR. Source

[14] NVIDIA Corporation. Form 10-K. Fiscal year ended January 26, 2025. SEC EDGAR. Source

[15] Alphabet Inc. Form 10-K. Fiscal year ended December 31, 2025. SEC EDGAR. Source