

The Asymmetry Engine, Chapter 5: One over x

Martin Roark

Chapter 5 of *The Asymmetry Engine: How AI Closes the Gaps, and What Stays Scarce*, October 2026. The online edition, with the interactive figures and the links between chapters, is at <https://future-seems-so-good.com/blog/the-asymmetry-engine#latency>.

License: [CC BY 4.0](#).

5:1 Thinking is cheaper than waiting. When a person or an agent sits blocked on a model's answer, the waiting usually costs more than the compute that would remove it. The physics of decoding turns speed into a priced good. Across models, speed times active size is capped by memory bandwidth; on one chip, throughput falls to zero as a single stream's speed nears a ceiling set by a fixed cost per step. A stopwatch is a perfect verifier, so speed works as a reward wherever the clock decides, and the gap closes fast in games and code and slowly in regulated incumbents. Vendors began selling fast modes in 2026; set as the default, a fast mode is a nudge, and whether it pays depends on whose time is waiting.

5.1 The latency asymmetry

5:2 Every call to a model makes someone wait, and only the blocked part of the waiting costs anything. A programmer who reads the last answer while the next one streams is partly blocked; an agent whose next step needs the answer is fully blocked.

5:3

Definition 5.1 (Latency asymmetry). A call makes a principal whose time is worth w dollars a second wait, blocked for a share s_{blk} of the wait. A faster mode removes waiting Δt at extra cost Δc . The **latency asymmetry** of the call, and the gap it leaves over all calls, are

$$A_\ell = \frac{w s_{\text{blk}} \Delta t}{\Delta c}, \quad G_\ell = \sum_{\text{calls}} (w s_{\text{blk}} \Delta t - \Delta c)^+$$

Where $A_\ell > 1$, the waiting is worth more than the speed that would remove it, and G_ℓ is this asymmetry's term in (0.2). Value delivered after a latency ℓ discounts hyperbolically: an answer of quality Q_{ans} is worth $Q_{\text{ans}}/(1 + \ell/t_{1/2})$, where $t_{1/2}$ is the wait that halves its value.

5:4 The hyperbola is the standard model of how people value a delayed reward ([Ainslie 1975](#)). Most of the loss comes early: the first $t_{1/2}$ of waiting costs half the value, and the second costs another sixth. Once the wait is long, $Q_{\text{ans}}/(1 + \ell/t_{1/2}) \approx Q_{\text{ans}}t_{1/2}/\ell$: value falls as one over the wait, and capability and latency trade along hyperbolas of equal delivered value. At the frontier the stakes are large: a reasoning output of 100 million tokens takes a month at the roughly 40 tokens a second of current chips, and a day at 1,200 ([Fractile](#)).

5.2 Two hyperbolas

5:5 Decoding is serial: each new token for each user requires reading every active weight once. In Google's words, "each decoding step needs to read the entirety of the model's weights," while chips compute about two orders of

magnitude faster than they read memory ([Google Research](#)). Two more costs matter. Each stream's key–value cache must also be read at every step, and unlike the weights it is not shared across the batch. And each step pays fixed costs, such as communication among chips, that do not shrink with the batch.

5:6

Definition 5.2 (The decode step). One accelerator serves n streams at once. The model has N_{act} active parameters at b bytes each; each stream holds n_{ctx} context tokens at b_{kv} cache bytes each; the accelerator achieves memory bandwidth BW_{eff} , pays a synchronization cost t_{sync} per step and accepts $n_{\text{acc}} \geq 1$ tokens per stream per step. One **decode step** takes

$$t_{\text{step}}(n) \approx t_0 + t_{\text{kv}} n, \quad t_0 = t_{\text{sync}} + \frac{N_{\text{act}} b}{\text{BW}_{\text{eff}}}, \quad t_{\text{kv}} \approx \frac{n_{\text{ctx}} b_{\text{kv}}}{\text{BW}_{\text{eff}}}$$

and each stream decodes at $\nu = n_{\text{acc}}/t_{\text{step}}$ tokens a second. The **floor** t_0 is the part of the step that does not grow with the batch: synchronization plus one read of the active weights. With one token accepted per step it caps every stream at $\nu_{\text{max}} = 1/t_0$.

5:7 The step time linearizes the roofline of Experiment 10.1, and two laws follow from it, one across models and one on a single chip:

$$(a) \quad \nu \leq \frac{\text{BW}_{\text{eff}}}{N_{\text{act}} b}, \quad (b) \quad X_{\text{acc}}(\nu) = \frac{1 - \nu/\nu_{\text{max}}}{t_{\text{kv}}}, \quad \nu_{\text{max}} = \frac{1}{t_0} \quad (5.1)$$

▼ *Derivation: The two laws*

With $n_{\text{acc}} = 1$, $\nu = 1/(t_0 + t_{\text{kv}} n) \leq 1/t_0$, and $t_0 \geq N_{\text{act}} b / \text{BW}_{\text{eff}}$ gives (a). Solving the step time for the batch, $n = (1/\nu - t_0)/t_{\text{kv}}$, so $X_{\text{acc}} = n\nu = (1 - \nu t_0)/t_{\text{kv}}$, which is (b). When multi-token prediction or speculative decoding accepts $n_{\text{acc}} > 1$ tokens per step, both speeds scale by n_{acc} .

5:8 Law (a) caps any stream, whatever the batch: speed times active bytes is at most bandwidth, a hyperbola in N_{act} and a line of slope -1 on log–log axes. Law (b) gives an accelerator's throughput $X_{\text{acc}} = n\nu$ as a function of the speed each stream gets, falling linearly to zero at ν_{max} . The two laws price speed in two currencies. Across models, speed costs capability: twice the active bytes, at most half the speed. On one chip, speed costs money: a faster stream means a smaller batch, so the same hardware serves fewer tokens. Both are limits of one ceiling, $n_{\text{acc}}/t_0 = n_{\text{acc}}/(t_{\text{sync}} + N_{\text{act}} b / \text{BW}_{\text{eff}})$: the weight read dominates the floor for large models, and synchronization dominates for small ones or for weights spread over many chips. Fast and cheap sit at opposite ends of one curve. Where hardware can be added at equal cost per token, Epoch finds speed falling only with the square root of parameter count ([Epoch](#)).

5:9 The model fits measurement. SemiAnalysis's InferenceX benchmark measured a GB200 NVL72 rack serving DeepSeek-R1 at 14,659 tokens a second per GPU in total when each user got 18 tokens a second, and at 1,149 when each got 164 ([InferenceX](#)). Fitting law (b) to those two points gives a floor $t_0 = 5.67$ ms, a ceiling $\nu_{\text{max}} = 176$ tokens a second and a peak $1/t_{\text{kv}} \approx 16,300$ tokens a second per GPU (Experiment 10.1). OpenAI reports minimum times between tokens on DeepSeek-R1 of 5.90 ms on GB300 and 1.43 ms on its own Jalapeño chip; their reciprocals, 169.5 and 699 tokens a second, match the fastest single streams measured on the two systems, 169 and about 700 ([OpenAI](#), from runs [SemiAnalysis](#) observed in OpenAI's lab).

5:10

Interactive in the online edition: <https://future-seems-so-good.com/blog/the-asymmetry-engine#fig:hyperbola>

Figure 5.1 (interactive). Single-stream speed against active bytes per token under a bandwidth line, and throughput per chip against the speed each user gets. Raise the bandwidth and watch the line climb; then lower the floor t_0 and watch which curve moves: only the fast end of the throughput curve, which is where decode chips compete.

5.3 Silicon against the floor

5:11 Law (b) makes every chip's fastest operating point its most expensive one, and decode silicon competes by moving the floor.

5:12

Proposition 5.1 (The price of speed). On a chip that obeys law (b) of (5.1), the energy per token at stream speed ν , and so the cost per token at fixed hardware rent, exceeds its throughput-optimal value by

$$\text{premium}(\nu) = \frac{\nu_{\max}}{\nu_{\max} - \nu}$$

On the GB200 fit the premium is 2.3 at 100 tokens a second, 6.7 at 150 and 24 at 169. It is convex, so a fixed speed-up costs more the nearer it starts to the ceiling.

▼ Proof

Power per accelerator is fixed, so energy per token is proportional to $1/X_{\text{acc}}(\nu) = t_{\text{kv}}/(1 - \nu/\nu_{\max})$. Dividing by its limit as $\nu \rightarrow 0$ gives the premium. ■

5:13 OpenAI's DeepSeek-R1 data put GB300 at 118 tokens a second per kilowatt at its fastest operating point, 169 tokens a second per user, against 11,781 at peak throughput ([OpenAI](#)): about a hundred times less work per watt, which under law (b) means running at 99% of $\nu_{\max}$. Convexity fits the year's prices: under a ceiling of 176 tokens a second, a 2.5-fold speed-up costs twice the throughput from 44 tokens a second and six times from 63, the two premiums Anthropic charged in 2026 for the same "up to 2.5x" fast mode.

5:14 Decode chips move weights into faster memory or into the logic itself. NVIDIA's Groq 3 LPU holds 500 MB of SRAM at 150 TB/s ([NVIDIA](#)); Cerebras's WSE-3 holds 44 GB on one wafer at 21 PB/s ([Cerebras prospectus](#)); Taalas's HC1 hard-wires Llama 3.1 8B into its silicon and fetches no weights at all ([Taalas](#)). Fast memory is small: a Groq 3 LPU holds 576 times less than a Rubin GPU, so DeepSeek V3 would need about 1,342 of them ([The Register](#)). Speed is bought with chip count, chip count brings communication, and so decode silicon must attack synchronization, the other half of the floor.

5:15 On SRAM the bandwidth line stops binding. Cerebras ran Kimi K2.6, a trillion-parameter mixture-of-experts model, at 981 tokens a second ([Cerebras](#)); at the 32 billion active parameters Moonshot reported for Kimi K2.5 ([Moonshot](#)), read at 8 bits, that is about 31 TB/s, a sliver of one wafer's 21 PB/s. Capacity binds instead, so the fast systems of 2026 are hybrids: GPUs or Trainium do prefill and hold the key-value cache, and SRAM chips run decode ([Amazon](#); [NVIDIA](#)).

5:16 Jalapeño's floor sits 4.1 times below GB300's. OpenAI says the chip delivers 12,258 tokens a second per kilowatt at GB300's fastest DeepSeek-R1 point, against 118, which is "fast-mode inference at efficiencies previously available only in batched mode," and 1.5–1.9 times GB300's peak throughput per kilowatt ([OpenAI](#)). These are OpenAI's own numbers, against Blackwell rather than its HBM4 peer Rubin ([SemiAnalysis](#)). If they hold at scale, the premium approaches 1 and speed becomes a default.

5:17 Inference silicon dates from Google's first TPU, "deployed in datacenters since 2015" ([Jouppi et al.](#)), but decode silicon at frontier scale arrived in 2026: Cerebras listed in May on a \$24.6 billion backlog mostly from OpenAI ([prospectus](#)), Jalapeño, built with Broadcom, appeared in June ([OpenAI](#)), and NVIDIA's Groq 3 LPX, after a reported payment of about \$20 billion for a non-exclusive Groq licence and most of its technical team ([EE Times](#)), entered full production in August, running Gemma 4 31B at 3,400 tokens a second with a 100,000-token context ([NVIDIA](#)). It waited for a stable transformer, for inference-dominated demand (Figure 3.3) and for a price on latency. The lag is the usual lag of a complement behind the good it serves.

5:18

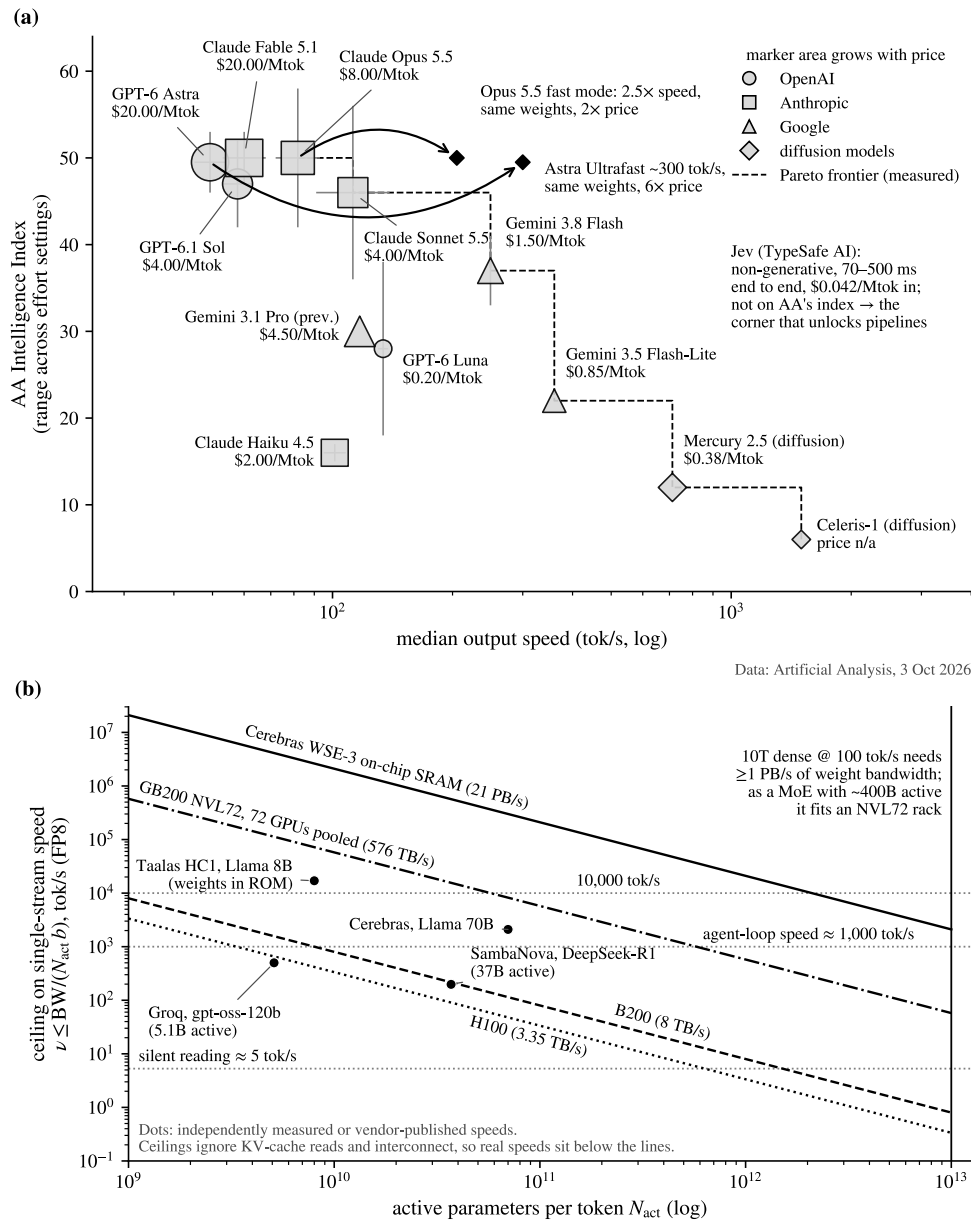


Figure 5.2. Capability on Artificial Analysis's index against median output speed for the October 2026 menu, with the measured frontier dashed and arrows for the same weights sold faster (a); the single-stream ceiling $\nu \leq \text{BW}/(N_{\text{act}}b)$ at 8 bits for four memory systems, with measured speeds as dots and dotted reference speeds for silent reading at about 5 tokens a second, an agent loop at about 1,000 and 10,000 (b). The frontier models stream at 45–93 tokens a second, 8.5 to 18 times silent reading's 5.3 tokens a second (238 words a minute, [Brysbaert 2019](#), at three-quarters of a word per token); only faster serving of the same weights moves right without moving down, and the SRAM dots sit far below their ceiling, where the floor binds.

5:19 Since the trade is physical, the market prices it. Anthropic launched fast mode on 7 February 2026 at six times the price for up to 2.5 times the output speed on Opus 4.6, \$30/\$150 per million input and output tokens against \$5/\$25 ([Simon Willison](#)). By 28 May the premium was twice the price, with Opus 4.8's fast mode at \$10/\$50 ([Anthropic](#)), and Opus 5.5 keeps it at \$8/\$40 against \$4/\$20, with the same weights and no faster first token ([Anthropic](#)). OpenAI's Fast mode also costs twice the price, Google's Priority tier 1.8 times ([OpenAI](#); [Google](#)), and on 29 September OpenAI launched Ultrafast for GPT-6 Astra, "up to 8x faster token generation (300 tokens

per second) in Codex and up to 6x in the API," at six times (\$60/\$300) ([OpenAI](#)). As of October 2026, speed is a priced dimension of intelligence, and its price is falling.

5:20 Per hour, the bill rises with the square of the speed, because the price per token and the tokens generated per hour both scale with the speed multiple. GPT-6 Astra's output costs about \$10 an hour at standard speed (\$50 per million tokens at about 54 tokens a second) and about \$324 an hour at Ultrafast (\$300 per million at up to 300): 5.6 times the speed for 33 times the bill, on list prices and [Artificial Analysis](#) speeds. For an agent's wage the premium is a quadratic tax on speed, one more reason inner loops run on the cheapest sufficient model (\$5.5).

5:21

Forecast 5.1 (The floor is the new bandwidth). Through 2027, single-stream speed on frontier mixture-of-experts models tracks the floor: GPU-only systems stay below about 400 tokens a second per stream on DeepSeek-R1-class models without speculation, while decode-specialized parts exceed 1,000.

Horizon 2027-12-31

Probability 70%

Check Falsified if, before 2028, a GPU-only system is independently measured above 400 tokens a second per stream on a mixture-of-experts model of at least 600 billion parameters without speculative decoding, or if by the end of 2027 no decode-specialized system has been independently measured above 1,000 tokens a second per stream on such a model.

5.4 Ten trillion parameters at a hundred tokens a second

5:22 No lab discloses a closed frontier model's size, and every public estimate I found is below ten trillion parameters, from Musk's "a 6 trillion parameter model" for Grok 5 ([Baron Investment Conference](#)) to a knowledge-probe estimate for GPT-5.5 revised from 9.7 trillion ([arXiv](#)) to about 1.5 ([eigenigma](#)). Ten trillion is the next size to test.

5:23 Take a ten-trillion-parameter mixture-of-experts model with about 400 billion active parameters at 4 bits, some 5.6 TB of weights. One Vera Rubin NVL72 rack holds 20.7 TB of HBM4 at about 1.4 PB/s ([NVIDIA](#)). One user is easy: each token reads only the active experts, about 0.2 TB, so the weight read takes well under a millisecond and the floor sets the speed at a few hundred tokens a second (Experiment 10.1).

5:24 A fleet is harder. At a hundred-odd users per rack, nearly every expert fires on every step, so the total parameter count sets the speed again: about 110–125 tokens a second per user on a Rubin rack from weights alone. Long contexts add every user's key–value cache to every step; at 100,000 tokens a user, a compressed cache leaves 98–109 tokens a second and a conventional grouped-query cache 55–61. A GB300 NVL72 rack, at 576 TB/s, gives about 50, and a dense model of the same size runs at 80–150 tokens a second on one Rubin rack at batch one. Ten trillion parameters at a hundred tokens a second is therefore feasible on 2026 Rubin racks, for sparse models with a compressed cache, at the edge. The binding cost is economic: each token needs about eight times the arithmetic of a model with 49 billion active parameters, so a gigawatt runs four to eight times fewer agents ((10.1)).

▼ *Derivation: The ten-trillion arithmetic*

A step serving n users of a mixture-of-experts model reads every expert any of them touched. With about 4% of experts active per token, a hundred-odd users touch nearly all of them ($0.96^{113} < 1\%$), so each step reads the whole $N_{\text{tot}}b \approx 5.6$ TB plus every user's cache:

$$\nu \approx \frac{\eta_{\text{bw}} \text{BW}}{N_{\text{tot}}b + n n_{\text{ctx}} b_{\text{kv}}}$$

where $\eta_{\text{bw}} = 0.45\text{--}0.5$ is the achieved share of a Rubin rack's 1.4 PB/s. At $n \approx 113$ users with $n_{\text{ctx}} = 10^5$, a compressed cache of about 70 KB per token adds 0.79 TB to each step (14%) and a grouped-query cache of about 516 KB adds 5.8 TB (104%). A hundred tokens a second at scale needs about 1.1–1.25 PB/s of raw bandwidth, which a Rubin rack just clears and a GB300 rack does not.

5:25 Ten thousand tokens a second is a different machine. Each token gets 100 μs . The step floors of Rubin and GB300 racks, 2.5–6 ms, are 25–60 times too long (Experiment 10.1), and the same model's active weights would have to stream at about 2 PB/s for each user, more than a whole Rubin rack's bandwidth. Only Taalas's HC1 reaches such per-token times, at 15,000–16,000 (public demo) to 17,000 (vendor) tokens a second for an 8B model cast in silicon ([EE Times](#)), and the fastest trillion-parameter run, Cerebras's, takes about 1 ms per token. A frontier model at ten thousand tokens a second needs SRAM-class designs and several tokens accepted per step.

5:26 Every lever of speed is a term of the decode step:

$$\nu = \frac{n_{\text{acc}}}{t_{\text{sync}} + (N_{\text{act}}b + n n_{\text{ctx}} b_{\text{kv}}) / \text{BW}_{\text{eff}}}$$

5:27 Speculative decoding raises n_{acc} , with "a 2X-3X acceleration ... with identical outputs" ([Leviathan et al.](#)). Silicon lowers t_{sync} . Quantization lowers b at a cost in fidelity (the fastest gpt-oss-120B endpoints keep 86–87% of reference accuracy; [Artificial Analysis](#)), sparsity and distillation shrink N_{act} , and cache compression shrinks b_{kv} . Nothing is free: the diffusion model Celeris-1 runs at about 1,500 tokens a second and scores 6 on Artificial Analysis's index, against 58 for the top model ([Artificial Analysis](#)). The search already uses the models it serves: Jalapeño went from design to tape-out in nine months, "accelerated by OpenAI's models" ([OpenAI](#)). Faster serving raises intelligence supply without new gigawatts, so it sits inside the research loop (§11.1).

5.5 Agent loops split intelligence in two

5:28 An agent that makes many sequential calls pays the latency once per call, so the discount of Definition 5.1 compounds with loop depth, and loop depth sets how large a model should be.

5:29

Proposition 5.2 (Loop depth sets model size). Let an answer's quality saturate in active size,

$Q_{\text{ans}}(N_{\text{act}}) = Q_{\infty} - k_Q N_{\text{act}}^{-z_Q}$, let latency grow with it, $\ell(N_{\text{act}}) = \ell_0 + c_{\ell} N_{\text{act}}$, and let an agent loop make m_{loop} sequential calls ($Q_{\infty}, k_Q, z_Q, \ell_0, c_{\ell}$ and m_{loop} are local). The size N_{act}^* that maximizes $Q_{\text{ans}}/(1 + m_{\text{loop}}\ell/t_{1/2})$ satisfies

$$Q'_{\text{ans}}(N_{\text{act}}^*) \left(1 + \frac{m_{\text{loop}} \ell(N_{\text{act}}^*)}{t_{1/2}}\right) = Q_{\text{ans}}(N_{\text{act}}^*) \frac{m_{\text{loop}} c_{\ell}}{t_{1/2}}$$

Deeper loops make the optimal model smaller. Faster hardware, a smaller c_{ℓ} , makes it bigger when the elasticity of quality in size is below one, which the first-order condition guarantees at any interior optimum.

▼ *Proof*

Maximize $\ln Q_{\text{ans}} - \ln(t_{1/2} + m_{\text{loop}}\ell)$; the second derivative at the optimum is $Q''_{\text{ans}}/Q_{\text{ans}} < 0$. The cross-derivatives in m_{loop} and c_{ℓ} are $-c_{\ell}t_{1/2}/(t_{1/2} + m_{\text{loop}}\ell)^2$ and $-m_{\text{loop}}(t_{1/2} + m_{\text{loop}}\ell_0)/(t_{1/2} + m_{\text{loop}}\ell)^2$, both negative. The second has the sign of $Q_{\text{ans}} - N_{\text{act}}Q'_{\text{ans}}$, and the first-order condition makes the elasticity $N_{\text{act}}Q'_{\text{ans}}/Q_{\text{ans}} = m_{\text{loop}}c_{\ell}N_{\text{act}}/(t_{1/2} + m_{\text{loop}}\ell) < 1$. ■

5:30 Inner loops with many steps (tool calls, retrieval, classification, interface responses) push toward models with few active parameters; outer loops with few steps (planning, design, judgment) can afford large, slow ones. This is Kahneman's System 1 and System 2, derived from a latency budget. It is why TypeSafe sells Jev as a "System One" model (§3.3) and why the cheapest swarms pair a frontier planner with fast workers (§9.6). Carried into hardware, the split points to two speeds: inner loops on SRAM or hard-wired silicon above 5,000 tokens a second per stream, outer loops on large models at 100–300, and a frontier stream at 10,000 only where SRAM-class hardware accepts several tokens per step (5:25).

5:31 Within one model, capability grows with the logarithm of waiting. On Artificial Analysis's index, Claude Opus 5.5 rises from 42 at its lowest effort, answering in 19 s, to 58 at maximum effort, in 685 s: about 4.5 points per factor of e in waiting, or 10 per decade. GPT-6.1 Sol rises from 42 at 11 s to 52 at 299 s, about 3 points per factor of e ([Artificial Analysis](#)). Under a deadline the choice sharpens: the tighter the deadline relative to the task's length, the less capable the model that wins.

5:32

Proposition 5.3 (When buying speed pays). Let a standard mode decode at ν_s tokens a second at output price p_{out} per token, with the call's input costing r_{io} times its output, and let a fast mode run the same weights k_ν times faster at k_p times the price ($\nu_s, r_{\text{io}}, k_\nu$ and k_p are local; w and s_{blk} are those of Definition 5.1). (i) Metered, the fast mode pays on a call iff

$$w > w^* = \frac{(k_p - 1) p_{\text{out}} (1 + r_{\text{io}}) \nu_s}{s_{\text{blk}} (1 - 1/k_\nu)}$$

whatever the call's length, because the waiting removed and the extra price both scale with it. (ii) In the agentic regime, where model time dominates human think time, the faster model wins iff its chance of success per attempt times its speed exceeds the slower model's: speed is worth exactly its ratio. In the deliberative regime, where think time dominates, latency is irrelevant and quality wins. (iii) A call whose decode share is s_{dec} speeds up by $1/(1 - s_{\text{dec}}(1 - 1/k_\nu))$.

▼ *The exact condition in (ii)*

A session of length T alternates model latency with human think time t_{think} , so a model with latency ℓ gets $T/(\ell + t_{\text{think}})$ attempts and succeeds with probability $1 - (1 - p)^{T/(\ell + t_{\text{think}})}$ if each attempt succeeds with probability p . The faster model wins iff

$$\frac{\ln(1 - p_{\text{fast}})}{\ln(1 - p_{\text{slow}})} > \frac{\ell_{\text{fast}} + t_{\text{think}}}{\ell_{\text{slow}} + t_{\text{think}}}$$

For small p the left side is $p_{\text{fast}}/p_{\text{slow}}$; across $p \leq 0.6$ that approximation misjudges 2.7% of cases. The symbols $t_{\text{think}}, \ell_{\text{fast}}, \ell_{\text{slow}}, p_{\text{fast}}$ and p_{slow} are local.

5:33

Forecast 5.2 (Two-speed intelligence). By the end of 2029, a model of at least a trillion parameters streams above 5,000 tokens a second on SRAM-class or hard-wired silicon, while large models served on HBM GPUs stay below 1,000.

Horizon 2029-12-31

Probability 45%

Check Falsified if by the end of 2029 no model with at least a trillion parameters is independently measured above 5,000 tokens a second per stream on SRAM-class or hard-wired silicon, or if a GPU-only system is measured above 1,000 tokens a second per stream on such a model.

5.6 Fast as the default

5:34 A default decides which point on law (b) a user buys. Defaults move compute: when GPT-5's router replaced the model picker in August 2025, daily use of reasoning models rose from under 1% to 7% of free users and from 7% to 24% of paying ones ([Altman, via Simon Willison](#)). Why defaults stick, and when they serve the chooser, is Definition 6.2 and Proposition 6.1.

5:35 Vendors split on the speed default. Cursor made a pricier fast tier its default: Composer 2's fast variant costs three times the standard price, \$1.50/\$7.50 against \$0.50/\$2.50, and "We're making fast the default option"

(Cursor, March 2026). Claude Code keeps fast mode opt-in, "available via usage credits only" (Claude Code), and OpenAI sells Ultrafast in Codex as an upsell through a new Pro 500 plan (CNBC). No vendor discloses what fast tiers earn, so the claim that the speed nudge is lucrative is unverified.

5:36 Who should buy speed follows from Proposition 5.3. For Opus 5.5, at about 80 tokens a second and \$20 per million output tokens, with fast mode at twice the price for 2.5 times the speed, a fully blocked user on an output-dominated call gains whenever their time is worth more than about \$10 an hour, or \$19 when the call's input costs as much as its output. At February's six-times price the same thresholds started at \$60 an hour. Repricing moved flagship fast mode from the expert tail toward anyone actually waiting. These are illustrations from list prices and Artificial Analysis speeds, not measurements of users.

5:37 The decisive variable is s_{blk} . A programmer who reads and thinks while the model works, blocked a quarter of the time, quadruples every threshold, and with half of a call's latency outside decoding a "2.5x" mode makes the call only 1.43 times faster. For the average programmer, faster tokens have not yet meant faster work: in METR's 2025 trial, 16 experienced developers using early-2025 AI tools took 19% longer while believing they were 20% faster (METR), and Cognition found that at 950 tokens a second "previously negligible system delays emerged as dominant bottlenecks" (Cognition). The expert in flow, with a high w , s_{blk} near 1 and long agent chains that multiply Δt , is the long tail for whom speed pays, most of all where it carries an answer under Nielsen's 10 s, "about the limit for keeping the user's attention focused on the dialogue" (Nielsen).

5:38 A fast default is benign when most of the calls that keep it come from users above w^* , who would have chosen it anyway. For users below w^* who never opt out, it transfers $k_p - 1$ times the standard price of every call to the vendor, the case Thaler calls sludge (Proposition 6.2). Two forces push toward benign. The threshold scales with the absolute price, which for a fixed capability falls about 13 times a year (3:48), and competition erodes the premium, as the fall from six times to twice shows. Metered billing pulls the other way, since heavier defaults raise revenue.

5:39

Forecast 5.3 (Fast becomes the default). By the end of 2027, flagship fast modes cost less than 1.5 times the standard price, and at least two of Cursor, Codex and Claude Code serve their flagship model fast by default.

Horizon 2027-12-31

Probability 25%

Check Read the price lists and default settings of OpenAI, Anthropic, Cursor, Codex and Claude Code on 31 December 2027. The forecast holds if the fast mode of OpenAI's and of Anthropic's flagship model each costs less than 1.5 times its standard price and at least two of Cursor, Codex and Claude Code serve their flagship model fast by default; it fails otherwise.

5.7 Latency as a reward

5:40 Speed is a good reward for building products because a stopwatch is a perfect verifier: cheap, sound and complete in the sense of Definition 1.2, dense in time, nearly monotone in value and tied to cost. Its verifiability in (0.2) is close to 1, and the controlled evidence agrees. Adding 100–400 ms to Google's search results cut searches per user by 0.2–0.6%, and the effect persisted after the delay was removed (Brutlag 2009). At Bing, 500 ms of

delay cut revenue per user by 1.2% and 2 s cut it by 4.3% ([Schurman and Brutlag](#)). For programmers, IBM found transactions per hour rising from 180 to 371 as system response fell from 3 s to 0.3 s ([Doherty and Thadhani 1982](#)); the "400 ms Doherty threshold" is a later popularization.

5:41 The most famous numbers are the weakest. Amazon's "+100 ms costs 1% of sales" is one slide about an unpublished test from before 2002 ([Linden](#)), and Marissa Mayer's "+0.5 s costs 20% of traffic" came from a test that also tripled the number of results per page ([Linden](#)). The stopwatch can also be fooled: in Buell and Norton's labor-illusion experiments, people "can actually prefer websites with longer waits" when the site displays its effort ([Buell and Norton 2011](#)). Users reward the wait they perceive, which is why products stream their tokens.

5:42 Pavel Durov treats speed as one of Telegram's core technical ideas: "we prioritize speed," he says, because users notice differences of 50 ms and "the difference is subconscious" ([Lex Fridman podcast, September 2025](#)); half a second of delay, multiplied across a billion users, is "centuries, millennia lost." The discipline is organizational. Telegram passed a billion monthly users in March 2025 ([TechCrunch](#)) with a core engineering team Durov puts at "about 40 people" running "almost 100,000" servers through automation; in 2024 he counted about 30 engineers, a number security researchers called a red flag ([TechCrunch](#)). "Fastest messaging app" is Telegram's own description, and I found no independent benchmark. Elon Musk's speed is a different loop, engineering cycle time: "If a schedule's long, it's wrong, if it's tight, it's right" ([Spaceflight Now, 2019](#)), and the fourth step of his design "algorithm" is to accelerate cycle time ([Everyday Astronaut, 2021](#)). I found no statement of his tying product latency to revenue. Product latency is the user's loop delay; cycle time is the organization's.

5:43 Speed below a threshold feels like magic, and the threshold has a number: Nielsen puts 0.1 s as "about the limit for having the user feel that the system is reacting instantaneously" ([Nielsen](#)). The steering bound explains why. A person correcting a model through a delay is a brake acting on old information, and by the $g = 0$ case of Proposition 4.3 a correction applied through a lag Δ stays stable only while its strength times Δ is below $\pi/2$, and avoids overshoot only below $1/e$: a loop corrects about once per delay. With ten-second turns a person corrects a model's course every ten seconds; at 100 ms, ten times a second. Magic is a loop whose machine delay falls below the human's. The bound runs the other way too: when the machine is the fast party, "man and machine operate on two distinct time scales" and "do not gear together without serious difficulties" ([Wiener 1960](#)), so a fast fleet's purpose must be right in advance (§13.8).

5.8 Banks, games and the supply of effort

5:44 Sectors differ by the strength of their reward signal. A mobile game learns from retention curves within days: Supercell soft-launches each game in test markets and kills even well-reviewed ones that miss their numbers ([Supercell](#)), and its small cells decide for themselves; in its chief executive's words, "the less the management ... had to do with the game, the better" ([Game Developer](#)). Free-to-play revenue is retention, so the studio's alignment with its players is high and its inertia low. A large bank faces slow outcome signals, because customers rarely switch; weak alignment, because spread income flows in a concentrated market such as Brazil's (15:33); and high inertia from regulation, security and legacy systems. In (0.2), three of the four terms differ between the game and the bank, and none of them is the engineers' skill.

5:45 Some inertia is deliberate. After a wave of fraud and express kidnappings, Brazil's central bank capped transfers between individuals at R\$1,000 between 20:00 and 06:00 and made limit increases wait at least 24 hours

(Resolução BCB nº 142, September 2021), having found signs of fraud in about one Pix transfer in 100,000 (Radioagência Nacional). The central bank inserted latency on purpose: where verification is hard, delay is its price. Pix itself, the rail the central bank built and mandated, is the case of §15.5.

5:46 Speed wins users quickly and money slowly: Nubank serves most Brazilian adults yet holds a small share of the banking profit pool, because inertia keeps the profits for years after the users have moved (15:36). I expect the gap to close mainly when agents that move a customer's balances do the switching.

5:47

Definition 5.3 (Effort-limited quality). Let a product's residual defects decay with verified engineering effort E_{eng} as

$$D_{\text{def}}(E_{\text{eng}}) = D_{\text{def},0} e^{-E_{\text{eng}}/E_0}$$

with $D_{\text{def},0}$ the initial defect mass and E_0 the effort per e -fold improvement (all symbols local). A product has **effort-limited quality** when the effort it receives, capped by headcount, hours and coordination, holds E_{eng} near E_0 .

5:48 Products stay rough for want of effort. A product at E_0 keeps 37% of its defects; ten times the effort leaves 0.005%. Abundant intelligence acts on the exponent: if engineering effort becomes cheap, the polish that today needs a Telegram-grade team becomes the default, and the binding constraint moves to verifying quality (§2.8) and to the hands that integrate it (§3.4). Three limits bound the model: effort can be aimed only where "better" is verifiable; users' attention caps how much improvement they absorb (Definition 6.1); and weak incentives, as in banking, withhold effort even when it is available. Coordination caps human effort (§9.3), and the cap is ultimately biological (§6.1).

5:49 A trillion-dollar valuation rewards distribution more than polish: Google paid \$26.3 billion in 2021 for default placements that the court called "extremely valuable real estate" (US v. Google). Even the optimists build in the limit: Dario Amodei expects decreasing marginal returns to intelligence once the outside world binds (§13.8).

5:50

Forecast 5.4 (Speed moves users first, money last). Through 2030, entrants in concentrated, regulated sectors win users faster than profits: Nubank's share of the Brazilian banking system's net income stays below half its share of Brazilian adults.

Horizon 2030-12-31

Probability 90%

Check Divide Nu Holdings' Brazilian net income by the banking system's net income in the central bank's latest annual data published by the horizon, and compare the result with Nubank's latest reported share of Brazilian adults. The forecast fails if the first is at least half the second; it holds otherwise.

5.9 Three gaps on one chart

5:51 Three asymmetries now have numbers: the verification asymmetry α (Definition 2.1), the compute asymmetry A_c ((3.1)) and the latency asymmetry A_ℓ . Placing task families on all three log axes maps the gaps

and predicts the order in which they close. For placement, A_ℓ is read as the time the status-quo route takes over the time the machine route takes.

5:52 The corners have names. The self-play corner, with a high verification asymmetry, holds formal proofs, competitive programming and puzzles (§2.5). The cheap-and-fast corner, with high compute and latency asymmetries, holds document extraction, customer support and real-time voice, where price and speed release the value (§3.4). The root has a low verification asymmetry and needs the frontier: AI research, the most valuable point in the space because improving it moves every other point (§11.1). The axes interact: verification governs how fast a cheaper model can be trained to clear a threshold, compute whether it is worth deploying, latency whether its answer arrives while it still has value.

5:53 Each family closes at the rate (0.2) assigns it, and the placements predict the order: fastest in digital, verifiable work done by small teams, slowest in regulated incumbents, medicine and management. Three placements invert intuition. AI research sits at $\alpha < 1$, because checking a research idea costs a training run, and closes early only because the labs aim their own fleets at it. Vulnerability discovery has a very high α but closes eighth, because its binding term is the throughput of verifying and fixing (§12.4, Experiment 9.1). Drug discovery closes last despite its value, because its verifier is a clinical trial, with α near 10^{-2} (§14.4).

5:54

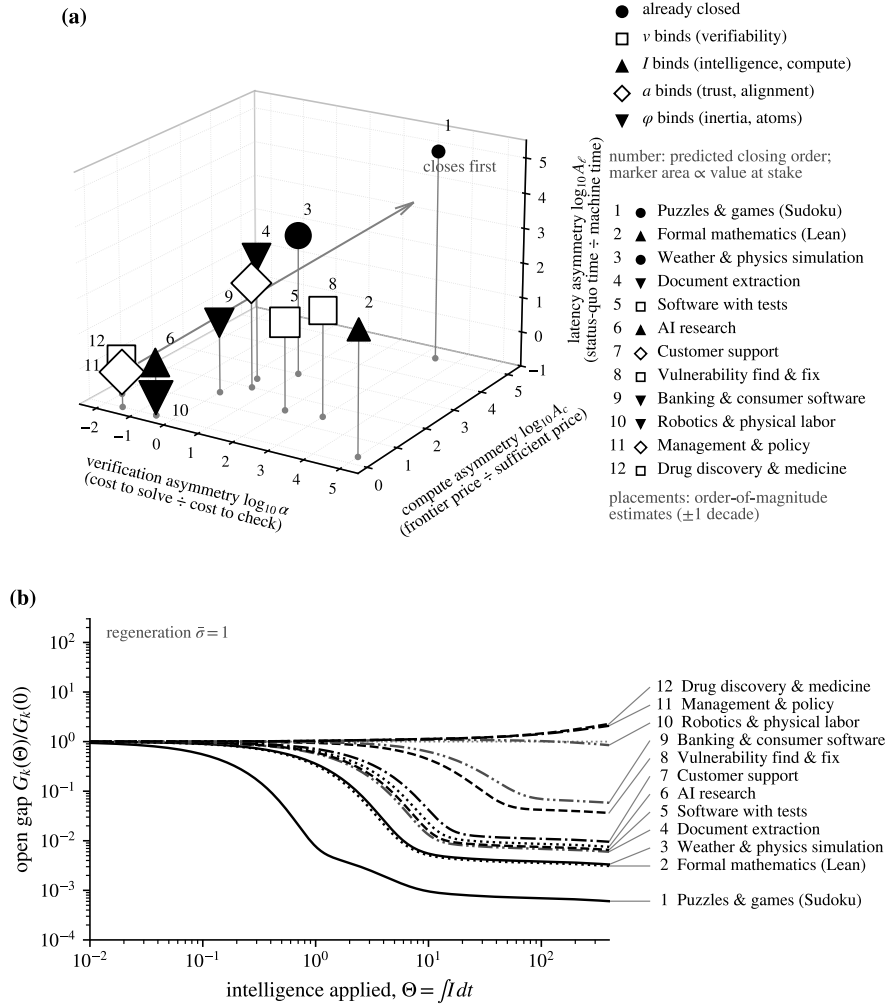


Figure 5.3. Twelve task families placed by the logarithms of their verification asymmetry α , compute asymmetry A_c and latency asymmetry A_ℓ , the last read as status-quo time over machine time, numbered by predicted order of closing, with marker shape giving the term of the master equation that binds each and marker area the value at stake (a); each family's open gap relative to its start as intelligence is applied, with regeneration matching closing and line style repeating the binding term (b). The six fastest families fall below 1% of their starting gap, customer support far later, and vulnerability work and banking level off at 3.6% and 5.8%, while management, policy, drug discovery and medicine grow to about twice theirs, so the three slowest families go from holding 90% of the open gap to holding 99.8%.

▼ The twelve placements

Each placement is an order-of-magnitude estimate, good to about one decade. A family's closing rate multiplies the factors of (0.3) by a serving efficiency $1 + \log_{10} A_c$, with verifiability read off the verification asymmetry as $\alpha/(1 + \alpha)$, allocation 1 for every family except AI research (10), alignment a as licensed autonomy, and inertia φ of 1 for digital work, 3 for organizational, 10 for regulated and 30 for atoms. Value at stake is in dollars a year, from headcount times wage or market size. Sudoku's α is measured (Experiment 2.1), the price gap on extraction comes from Experiment 3.1 and the verifier ceiling on software from Experiment 9.1; the rest are judgments the listed chapters defend.

#	Family	$\log_{10} \alpha$	$\log_{10} A_c$	$\log_{10} A_\ell$	Value at stake	a	φ	Binds	Chapter
1	Puzzles and games (Sudoku)	3.4	5	5	$\$10^8$	1.0	1	closed	Chapter 2
2	Formal mathematics (Lean)	5	0.3	2.5	$\$10^{10}$	0.9	1	I	Chapter 14
3	Weather and physics simulation	1	3	3	$\$5 \times 10^{10}$	0.9	3	closed	Chapter 14
4	Document extraction	0.3	2.4	2.5	$\$3 \times 10^{11}$	0.8	3	φ	Chapter 3
5	Software with tests	2	1.4	1.5	$\$10^{12}$	0.7	3	v	Chapter 9
6	AI research	-0.5	0	0.5	$\$3 \times 10^{11}$	0.6	3	I	Chapter 11
7	Customer support	0.5	2	2	$\$4 \times 10^{11}$	0.5	3	a	Chapter 4
8	Vulnerability find and fix	3	1.5	2	$\$2 \times 10^{11}$	0.4	10	v	Chapter 12
9	Banking and consumer software	0	1.5	1	$\$5 \times 10^{11}$	0.5	10	φ	§5.8, Chapter 15
10	Robotics and physical labor	-0.5	0	-0.5	$\$2 \times 10^{13}$	0.6	30	φ	Chapter 15
11	Management and policy	-1.5	0	0	$\$5 \times 10^{12}$	0.3	10	a	Chapter 7, Chapter 8
12	Drug discovery and medicine	-2	0.5	0	$\$2.5 \times 10^{11}$	0.4	10	v	Chapter 14

References

1. George W. Ainslie (1975). Specious reward: A behavioral theory of impulsiveness and impulse control. *Psychological Bulletin*. <https://doi.org/10.1037/h0076860>
2. Fractile. <https://www.fractile.ai/news/fractile-raises-220m-to-build-the-next-generation-of-inference-hardware>
3. Google Research. <https://research.google/blog/looking-back-at-speculative-decoding/>
4. Epoch. <https://epoch.ai/publications/inference-economics-of-language-models>
5. InferenceX. <https://inference.semianalysis.com/blog/gb200-nvl72-vs-b200-disagg-deepseek-r1-fp4-dynamo-trt>
6. OpenAI. <https://openai.com/index/jalapeno-first-results/>
7. SemiAnalysis. <https://inference.semianalysis.com/blog/openai-jalapeno-better-than-nvidia>
8. NVIDIA. <https://www.nvidia.com/en-us/data-center/lpx/>
9. Cerebras prospectus. <https://www.sec.gov/Archives/edgar/data/2021728/000162828026035214/cerebras-424b4.htm>
10. Taalas. <https://taalas.com/products/>
11. The Register. <https://www.theregister.com/systems/2026/08/24/what-nvidias-first-groq-3-lpu-benchmarks-tell-us-about-its-20b-gamble/5291880>
12. Cerebras. <https://www.cerebras.ai/blog/cerebras-kimi-k2-Enterprise>
13. Moonshot. <https://github.com/MoonshotAI/Kimi-K2.5/>
14. Amazon. <https://press.aboutamazon.com/aws/2026/3/aws-and-cerebras-collaboration-aims-to-set-a-new-standard-for-ai-inference-speed-and-performance-in-the-cloud>
15. NVIDIA. <https://developer.nvidia.com/blog/inside-nvidia-groq-3-lpx-the-low-latency-inference-accelerator-for-the-nvidia-vera-rubin-platform/>
16. Norman P. Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal et al. (2017). In-Datcenter Performance Analysis of a Tensor Processing Unit. *arXiv*. <https://arxiv.org/abs/1704.04760>
17. OpenAI. <https://openai.com/index/openai-broadcom-jalapeno-inference-chip/>

18. EE Times. <https://www.eetimes.com/fallout-from-nvidia-groq-deal-validates-ai-chip-startup-landscape/>
19. NVIDIA. <https://nvidianews.nvidia.com/news/nvidia-groq-3-lpx-now-in-full-production-with-world-class-speed-for-agentic-ai>
20. Marc Brysbaert (2019). How many words do we read per minute? A review and meta-analysis of reading rate. *Journal of Memory and Language*. <https://doi.org/10.1016/j.jml.2019.104047>
21. three-quarters of a word per token. <https://help.openai.com/en/articles/4936856>
22. Simon Willison. <https://simonwillison.net/2026/Feb/7/claude-fast-mode/>
23. Anthropic. <https://www.anthropic.com/news/claude-opus-4-8>
24. Anthropic. <https://platform.claude.com/docs/en/build-with-claude/fast-mode>
25. OpenAI. <https://developers.openai.com/api/docs/guides/fast-mode>
26. Google. <https://ai.google.dev/gemini-api/docs/pricing>
27. OpenAI. <https://developers.openai.com/api/docs/guides/ultrafast-mode>
28. Artificial Analysis. <https://artificialanalysis.ai/leaderboards/models>
29. Baron Investment Conference. <https://singjupost.com/fireside-chat-elon-musk-at-ron-barons-32nd-baron-investment-conference-transcript/>
30. Bojie Li (2026). Incompressible Knowledge Probes: Estimating Black-Box LLM Parameter Counts via Factual Capacity. *arXiv*. <https://arxiv.org/abs/2604.24827>
31. eigenigma. <https://eigenigma.io/en/articles/estimating-parameter-counts-of-claude-5-and-gpt-5-6/>
32. NVIDIA. <https://www.nvidia.com/en-us/data-center/vera-rubin-nv172/>
33. EE Times. <https://www.eetimes.com/taalas-specializes-to-extremes-for-extraordinary-token-speed/>
34. Yaniv Leviathan, Matan Kalman, Yossi Matias (2022). Fast Inference from Transformers via Speculative Decoding. *arXiv*. <https://arxiv.org/abs/2211.17192>
35. Artificial Analysis. <https://artificialanalysis.ai/models/gpt-oss-120b/providers>
36. Altman, via Simon Willison. <https://simonwillison.net/2025/Aug/10/sam-altman/>
37. Cursor. <https://cursor.com/blog/composer-2>
38. Claude Code. <https://code.claude.com/docs/en/fast-mode>
39. CNBC. <https://www.cnbc.com/2026/09/29/openai-devday-2026-live-updates.html>
40. METR. <https://metr.org/blog/2025-07-10-early-2025-ai-experienced-os-dev-study/>
41. Cognition. <https://cognition.com/blog/swe-1-5>
42. Nielsen. <https://www.nngroup.com/articles/response-times-3-important-limits/>
43. Brutlag 2009. <https://research.google/blog/speed-matters/>
44. Schurman and Brutlag. <https://www.w3.org/2013/Talks/0610-performance/>
45. Doherty and Thadhani 1982. <https://jlelliotton.blogspot.com/p/the-economic-value-of-rapid-response.html>
46. Linden. <https://slideum.com/doc/5840687/presentation>
47. Linden. <https://glinden.blogspot.com/2006/11/marissa-mayer-at-web-20.html>
48. Ryan W. Buell, Michael I. Norton (2011). The Labor Illusion: How Operational Transparency Increases Perceived Value. *Management Science*. <https://pubsonline.informs.org/doi/10.1287/mnsc.1110.1376>
49. Lex Fridman podcast, September 2025. <https://lexfridman.com/pavel-durov-transcript/>
50. TechCrunch. <https://techcrunch.com/2025/03/19/telegram-founder-pavel-durov-says-app-now-has-1b-users-calls-whatsapp-a-cheap-watered-down-imitation/>
51. TechCrunch. <https://techcrunch.com/2024/06/24/experts-say-telegrams-30-engineers-team-is-a-security-red-flag/>
52. Spaceflight Now, 2019. <https://spaceflightnow.com/2019/09/29/elon-musk-wants-to-move-fast-with-spacexs-starship/>
53. Everyday Astronaut, 2021. <https://everydayastronaut.com/starbase-tour-and-interview-with-elon-musk/>
54. Some Moral and Technical Consequences of Automation. <https://gwern.net/doc/reinforcement-learning/safe/1960-wiener.pdf>
55. Supercell. https://en.wikipedia.org/wiki/Supercell_%28video_game_company%29
56. Game Developer. <https://www.gamedeveloper.com/business/less-management-more-success-inside-supercell-s-upside-down-organization>

57. Resolução BCB nº 142. <https://in.gov.br/web/dou/-/resolucao-bcb-n-142-de-23-de-setembro-de-2021-347046831>
58. Radioagência Nacional. <https://agenciabrasil.ebc.com.br/radioagencia-nacional/economia/audio/2021-08/banco-central-vai-reduzir-limite-de-transferencias-com-pix-noite>
59. US v. Google. <https://www.courthousenews.com/wp-content/uploads/2024/08/google-antitrust-monopoly-opinion.pdf>