

The Asymmetry Engine, Appendix B: The experiments

Martin Roark

Appendix B 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#experiments>. License: CC BY 4.0.

B:1 Every result in these nine experiments is a property of a stated model or a synthetic family of instances, and every public figure that grounds a parameter links to its source. Each description, with its parameters and seed, is enough to rewrite the program and recover its numbers. Where a sentence could be read as a fact about the world, it says *in the model*.

B.1 Sudoku

B:2 Counted in one unit, checking a filled 9×9 Sudoku costs 243 cell reads, building a puzzle backward costs about a dozen checks, and naive search peaks at 593,190 reads, 2,441 checks, at 55 blanks. These are the numbers of Experiment 2.1 in §2.2. The experiment asks what is cheap in the founding example of the verification asymmetry, checking or construction, and where on the difficulty dial the asymmetry peaks.

B:3 Every cost is charged in cell reads. A check reads 27 houses of 9 cells, so $c_{\text{ver}} = 243$. Construction fills a random valid grid by randomized backtracking, reading the row, column and box of each cell it visits, then carves blanks at random, which costs only writes; its floor, one scan per cell and no backtracking, is $81 \times 27 = 2,187$ reads. The naive solver fills cells in row order and reads 27 cells per value it tries; the minimum-remaining-values (MRV) solver reads 27 cells per open cell it evaluates and branches on the cell with the fewest candidates. Uniqueness is proved by an MRV search for a second solution, and a proper puzzle is carved with such a proof after every removal. Solvers first read the 81 cells to list the blanks, and writes enter no cost. The asymmetry is $\alpha = c_{\text{sol}}/c_{\text{ver}}$.

B:4

Parameter	Value	Basis
Unit	one cell read	the unit of the check, charged to every algorithm
Blank cells	10, 20, 30, 40, 45, 50, 55, 58, 61, 64	from trivial to the limit of uniqueness: a 9×9 puzzle needs at least 17 clues (McGuire, Tugemann and Civario)
Trials per level	10	medians stable to about ±20%
Naive budget	1.08×10^8 reads	stops pathological searches
Seed	20261003, Python's random	

B:5

Blanks	c_{con}	c_{sol} , naive	c_{sol} , MRV	Uniqueness proof	Proper puzzle	α , naive	α , MRV	$c_{\text{con}}/c_{\text{ver}}$	Share unique
10	3,456	1,310	351	351	n/a	5.4	1.4	14.2	1.0
20	2,795	3,011	648	648	n/a	12.4	2.7	11.5	1.0
30	3,186	5,805	1,958	1,998	19,562	23.9	8.1	13.1	0.5
40	2,835	16,173	5,225	5,265	48,263	66.6	21.5	11.7	0.5
45	2,862	55,013	11,921	13,028	n/a	226	49.1	11.8	0.2
50	3,213	108,054	20,115	20,534	275,333	445	82.8	13.2	0
55	2,984	593,190	27,338	27,878	468,437	2,441	112.5	12.3	0
58	3,132	407,606	31,307	33,548	n/a	1,677	128.8	12.9	0
61	3,497	425,169	35,033	38,516	n/a	1,750	144.2	14.4	0
64	3,105	278,924	39,731	40,055	n/a	1,148	163.5	12.8	0

B:6 Construction costs 11.5–14.4 checks at every level, far below search at the hard end. Naive α falls after its peak at 55 blanks because heavily carved puzzles have many solutions and any one will do; no carved puzzle is unique from 50 blanks on. MRV has no peak, so the peak belongs to the naive solver as much as to the puzzles, and MRV, 3–22 times cheaper, still costs 49–164 checks from 45 blanks on. A uniqueness proof costs about an MRV solve, 115 checks at 55 blanks, and a proper puzzle 0.8–3.4 naive or 9–17 MRV solves, so a reward that checks the rules, accepting any valid completion, never pays for uniqueness.

B:7

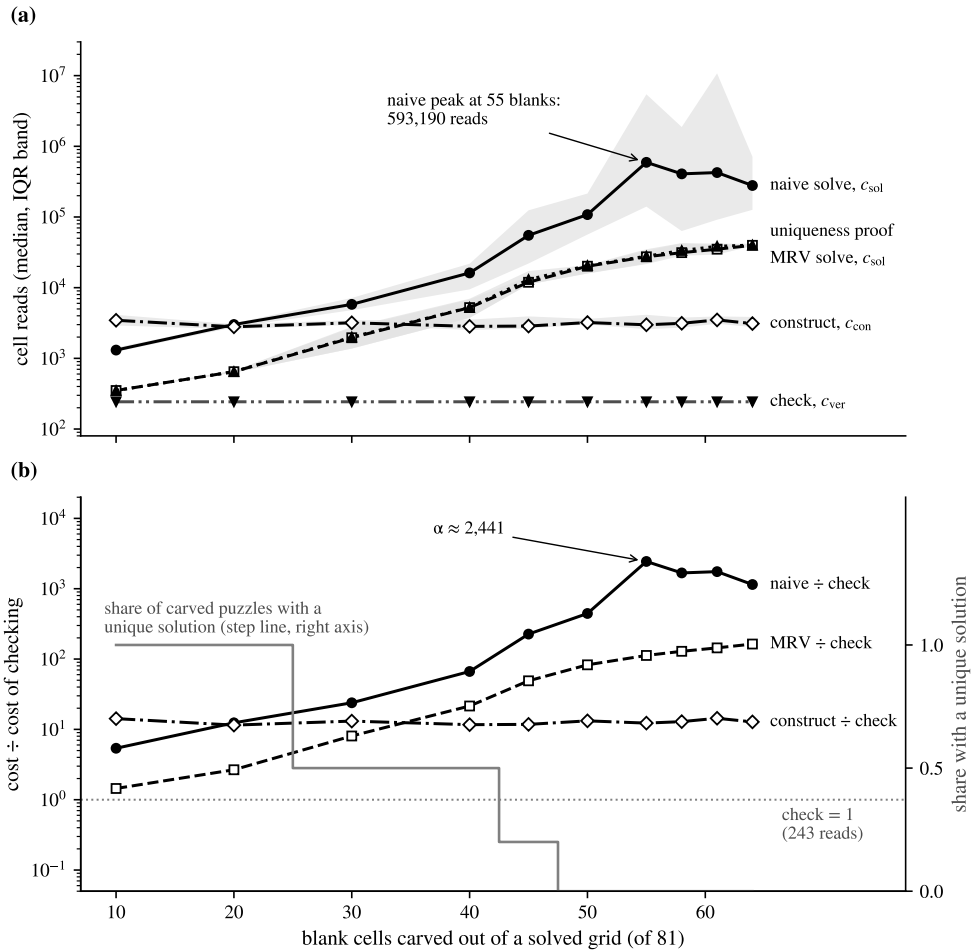


Figure B.1. Cell reads, median of ten trials with interquartile band, to check (c_{ver}), construct (c_{con}), solve (c_{sol} , naive and MRV) and prove the uniqueness of a 9×9 Sudoku against the number of blanks carved from a solved grid (a), and each cost divided by c_{ver} , which for the solvers is α , with the share of unique puzzles on the right axis (b). The check stays flat at 243 reads and construction near a dozen checks, while naive α peaks at 2,441 at 55 blanks, just after unique puzzles run out, and MRV α rises without a peak.

B:8 The counts are cell reads only: writes would add 5–7% to construction, and wall-clock time agrees on the order, 0.4 ms to construct against 0.02 ms to check. Counting the re-scan for the next blank, which is bookkeeping, would raise construction to 30–39 checks and the naive peak to 2,799. A check that stopped at the first conflict would read fewer than the naive solver's 27 cells per value, so naive α depends on the implementation by at most that factor; stronger solvers lower it, constraint propagation to about 10–20 (Figure 2.3). Ten trials give wide bands near the peak, and one naive run at 61 blanks hit the budget and is counted at it. A 9×9 grid understates the asymmetry: generalized $n^2 \times n^2$ Sudoku is NP-complete and finding another solution is ASP-complete (Yato and Seta), so unless $P = NP$ the worst case grows faster than any polynomial.

B:9 Ten puzzles per level from Python's `random` seeded with 20261003 reproduce the medians in about 20 seconds.

B.2 Planted 3-SAT

B:10 Built backward, a planted 3-SAT formula costs a fixed multiple of its check at every size, while search on it multiplies by a fixed factor per added variable: under balanced planting α grows from 26 at 50 variables to 3,246 at 190. Random 3-SAT can grow in size, peaks in difficulty at the satisfiability threshold ([Cheeseman, Kanefsky and Taylor](#); [Kirkpatrick and Selman](#)) and accepts a hidden solution. Experiment 2.2 in §2.3 asks how fast the asymmetry widens with size and where along the density knob the hard instances sit.

B:11 A formula has n variables and n times the clause density in clauses; each clause takes 3 distinct variables, each negated with probability $\frac{1}{2}$. A planted formula is built backward: draw a solution $y \in \{0, 1\}^n$ and reject every clause it falsifies, so a literal agrees with y with probability $4/7$. A balanced formula ([Jia, Moore and Strain](#)) keeps a clause with probability $1, q_h$ or q_h^2 as y makes one, two or three of its literals true, with $q_h = (\sqrt{5} - 1)/2$, so a literal agrees with y exactly half the time. Costs count ops: random draws, literal reads, variable writes and branching-score evaluations. Construction costs a draw per variable plus three draws and the needed reads per attempted clause, and checking reads each clause up to its first true literal.

B:12 The solver is DPLL ([Davis, Logemann and Loveland](#)) with unit propagation on two watched literals, chronological backtracking, and no pure-literal rule, restarts or clause learning. The MOMs rule (maximum occurrences in clauses of minimum size) scores each sign of a free variable at five per unsatisfied binary clause and one per ternary clause, branches on the variable with the largest 1,024 times the product of its two scores plus their sum, tries the more frequent sign first, and is charged every literal read of its clause scans. The naive rule takes the first unassigned variable and tries true first.

B:13

Parameter	Value
Seed	20261003, one NumPy <code>SeedSequence</code> per instance
Op budget per solve	10^7 (MOMs), 2×10^6 (naive); longer runs are kept as censored, and medians are reported only where fewer than half are censored
Density sweep	$n = 150$, MOMs; 23 densities, 2.0 to 7.0 in steps of 0.25 plus 4.125 and 4.375; 25 formulas per family per density; 1,725 solves
Size sweep	density 4.26; $n = 20, 30, \dots$; 25 formulas per family per n ; planted families by both rules, random by MOMs; each series grows until over a third of a level is censored, up to $n = 300$
Fits	least squares of the log median on n over $n \geq 50$ with at most a third censored; 95% intervals from 2,000 bootstrap resamples
Self-test	both rules agree with exhaustive search on all 300 random formulas with 8, 10 and 12 variables; every satisfying answer is checked; no planted formula was reported unsatisfiable; literal agreement 0.5715 (planted), 0.5002 (balanced)

B:14 At $n = 150$ no run was censored. $P(\text{sat})$ falls from 1.00 at density 4.0 to 0.44 at 4.25 and is 0 from 4.75; a logistic fit puts $P(\text{sat}) = \frac{1}{2}$ at 4.258 (95% interval 4.219–4.297), consistent with the threshold near 4.267 ([Mézard and Zecchina](#)). Random formulas cost a flat $5\text{--}6 \times 10^4$ ops up to density 3.75 and, in search nodes, peak at 2,792 against 68 at density 2.0 and 156 at 7.0.

B:15

Family	Median ops at density 4.25	Hardest density (bootstrap range)	Median ops there	Hardest over density 2.0
random	2.92×10^6	4.25 (4.25–4.50; parabola 4.31)	2.92×10^6	51× (11× over density 7.0)
planted	6.4×10^4 , 2.2% of random	5.25 (4.75–6.25), a weak maximum	3.2×10^5	5.5×
balanced	8.3×10^5 , 28% of random and 1.8× satisfiable random (4.7×10^5)	4.75 (4.25–4.75), on a plateau from 4.25	1.41×10^6	24×

B:16

Size sweep at density 4.26	Planted	Balanced	Random
Construction, linear fit	$24.1 n + 2$ ($R^2 = 0.99993$)	$43.2 n + 7$ ($R^2 = 0.99981$)	n/a
Check, linear fit	$6.70 n$ ($R^2 = 0.99991$)	$7.20 n$ ($R^2 = 0.99991$)	n/a
Construction over check, smallest to largest n	3.56 to 3.60	6.11 to 6.00	n/a
MOMs solve, growth of the median per variable (95% interval; range)	×1.017 (1.015–1.020; n 50–300)	×1.046 (1.041–1.050; n 50–190)	×1.051 (1.047–1.055; n 50–170)
MOMs solve, growth of median search nodes per variable	×1.012	×1.039	×1.043
Naive solve, growth of the median per variable	×1.072 (1.051–1.090; n 50–120)	×1.095 (1.068–1.128; n 50–90)	not run
α at the largest valid n , MOMs	342 at $n = 300$ (7 of 25 censored)	3,246 at $n = 190$ (7 of 25 censored)	n/a
α at the largest valid n , naive	1,152 at $n = 130$ (10 of 25 censored)	2,202 at $n = 100$ (11 of 25 censored)	n/a
α along the sweep, MOMs	25 at $n = 50$ to 342 at 300 (×1.010 per variable)	26, 178, 1,200 and 3,246 at $n = 50, 100, 150, 190$ (×1.036 per variable)	n/a

B:17 Construction and check match their expectations, $1 + \frac{8}{7} \times 4.75 \times 4.26 = 24.1$ ops and $\frac{11}{7} \times 4.26 = 6.69$ reads per variable. Plain planting leaks its solution: MOMs reads the 4/7 bias, and the median grows only ×1.017 per variable. One extra rejection test removes the bias for 43 instead of 24 ops per variable, and growth returns to ×1.046; a better branching rule lowers the factor, from ×1.07–1.09 to ×1.02–1.05, without removing it. A backward constructor is useful only if it is one-way for the solver that trains on it (Proposition 2.1). Balanced formulas pass the naive Sudoku α of 1,148 and 2,441 at $n = 150$ and 190, an indicative crossing because the units differ. At $n = 300$ the median planted run takes 327 search nodes while 12 of 25 take at least 698 and 7 hit the budget.

B:18 A power law fits these sizes almost as well as an exponential (R^2 of 0.983 against 0.984 for random formulas, exponent 5.0), and the local rate falls from ×1.057 to ×1.046 across the range, so the exponential reading rests on theorems: resolution refutations of random 3-SAT above the threshold are exponentially long with high probability (Chvátal and Szemerédi; Beame, Karp, Pitassi and Saks), and simple DPLL takes exponential time at some satisfiable densities below it (Achlioptas, Beame and Molloy). Clause scans are 88–90% of the MOMs

solver's ops, and counting only propagation reads and writes gives α of 409 (balanced, $n = 190$) and 27 (planted, $n = 300$). Planted run times are heavy-tailed, as backtracking without restarts usually is ([Gomes, Selman, Crato and Kautz](#)). Spectral methods solve dense planted 3-SAT in polynomial time ([Flaxman](#)), so this hardness is evidence about DPLL without clause learning, and whether the generator is one-way stays open. $P(\text{sat})$ has a standard error near 0.1 mid-transition, censored runs bias it upward at $n \geq 170$, the bootstrap covers sampling noise only, and that informative training instances sit near the threshold is an inference, since the experiment measures difficulty.

B:19 The run takes about 4.5 CPU-minutes; each instance draws its own seed from 20261003, so op counts are identical across runs and worker counts.

B.3 The Ashby ceiling

B:20 Against a Zipf stream of a million request types, the best possible if-tree buys coverage at a logarithmic rate: 1,000 rules cover 52.0% of requests, 10,000 cover 68.0%, and 90% takes 237,100 rules, 237 engineer-years to build and 119 people to maintain. These are the numbers of Experiment 4.1 in §4.3. The experiment asks how far a best-case if-tree gets against a heavy-tailed stream, what it costs in engineer-hours and human-hours, and how (4.1) reads in bits.

B:21 Request types $j = 1, \dots, N_{\text{types}}$ arrive with the frequencies of (4.2), a type being a class of requests with one correct response. The if-tree is a best case: N_{rules} perfect rules for the most frequent types, ranking known, with coverage $\text{cov}(N_{\text{rules}})$. At two engineer-hours a rule, half rewritten every year, building N_{rules} rules takes $N_{\text{rules}}/1,000$ engineer-years and keeping them $N_{\text{rules}}/2,000$ full-time staff. A learned policy's success falls linearly in the logarithm of a type's rank, from a head accuracy at rank one to the head accuracy times one minus a tail penalty at the rarest type. The escalation design sends the top types to rules, the rest to the model and every detected failure to a person, at six minutes a request.

B:22 In the variety view the disturbance D is the request type and the outcome O the reply: a rule names its type, and every other request gets one default reply, the fallback of §4.4. The reply is a function of D , so $H(D | O) = H(D) - H(O) \geq H(D) - \log_2(N_{\text{rules}} + 1)$, which is (4.1) with $N_{\text{rules}} + 1$ responses. What the if-tree actually leaves is the uncovered mass times the entropy of the uncovered types, clause (ii) of Proposition 4.1; for a learned policy the sum runs over the mass it fails on.

B:23

Parameter	Value	Basis
Request types N_{types}	10^6 (also 10^5 and 10^7)	large enough that tails with $z \leq 1$ matter
Zipf exponent z	0.8, 1.0, 1.1, 1.3	the classic $z \approx 1$, one heavier and two lighter tails
Hours per rule; rewritten a year	2; half	assumptions; costs scale linearly with them
Learned levels (head accuracy, tail penalty)	L1 (0.90, 0.30), L2 (0.97, 0.10), L3 (0.995, 0.02)	stylized; GPT-4o passed about 61% of τ-bench retail tasks in 2024, below L1, and the best 2026 telecom scores on τ²-bench are near 99%, near L3
Volume	10^7 requests a day	the order of a large retailer's support desk

Parameter	Value	Basis
Time per escalated request	6 minutes	below the 11 minutes <u>Klarna</u> reported for human resolution, so human hours are understated
Rules in the escalation design	10^4	ten engineer-years; XCON had about 3,300 rules by 1983 (<u>Bachant and McDermott</u>)
Monte Carlo	10^6 requests per exponent; seed 20261003	

B:24

z	$H(D)$, bits	Rules for 50%	80%	90%	95%	99%
0.8	17.39	41,036	351,268	609,090	785,251	953,683
1.0	13.41	749	56,217	237,100	486,930	865,951
1.1	10.90	69	6,987	61,623	225,345	728,828
1.3	6.86	6	104	849	5,869	159,568

B:25

If-tree at $z = 1$	10^3 rules	10^4 rules	10^5 rules
Coverage	52.0%	68.0%	84.0%
Points added by the next engineer-year	4.81	0.66	0.069
Ashby's bound $H(D) - \log_2(N_{\text{rules}} + 1)$, bits	3.44	0.12	0
Variety the if-tree leaves, bits	8.51	6.02	3.12
Share of the rules' nominal bits used	49%	56% (7.39 of 13.29)	62%
Same share at $z = 0.8$	25%	38%	59%

B:26

Learned level, $z = 1$	Coverage	Rules with equal coverage	Their build, engineer-years	Their upkeep, staff	Variety left, bits	Rules with equal variety left
L1	77.05%	36,775	37	18	3.54	72,691
L2	92.35%	332,431	332	166	1.19	407,999
L3	98.55%	811,134	811	406	0.23	827,528

B:27

Design at $z = 1$ and 10^7 requests a day	Human-hours a day	Full-time staff
everyone handled by people	1,000,000	182,500
rules only: $10^3, 10^4, 10^5$ rules	479,913; 319,962; 159,982	87,584; 58,393; 29,197
model first: L1, L2, L3	229,487; 76,519; 14,544	41,881; 13,965; 2,654
10^4 rules, then the model, then people: L1, L2, L3	103,988; 35,462; 6,906	18,978; 6,472; 1,260

B:28

Further results	Value
Coverage sustained by 1, 10, 100 maintainers ($z = 1$)	56.8%, 72.8%, 88.8%
One engineer-year's coverage at $z = 0.8$ and $z = 1.3$	20.7% and 90.5%
Rule count at which Ashby's bound reaches zero, $z = 0.8, 1, 1.1, 1.3$	171,480; 10,855; 1,907; 115
Learned coverage at $z = 0.8, 1.1, 1.3$	L1 71.1%, 80.3%, 84.9%; L2 90.2%, 93.5%, 95.2%; L3 98.1%, 98.8%, 99.1%
Scale anchors at $z = 1$	ELIZA's largest script, about 50 keywords (<u>Weizenbaum</u>), covers 31.3%; XCON's 3,300 rules, 60.3%
A day's log of 10^6 requests ($z = 1$)	217,019 distinct types, 149,992 seen once
Unseen mass, Good–Turing estimate (<u>Good</u>) against truth	0.1500 against 0.1500 at $z = 1$; 0.250 at $z = 0.8$; 0.031 at $z = 1.3$
Plug-in entropy of that log	12.94 bits, 0.47 below $H(D)$
Coverage of 10^4 rules for $10^5, 10^6, 10^7$ types	$z = 0.8$: 59.5%, 36.2%, 22.4%; $z = 1$: 81.0%, 68.0%, 58.6%; $z = 1.3$: 97.3%, 95.9%, 95.3%
Rules for 90% of 10^7 types at $z = 1$	1,883,354
Rules multiplier to halve the uncovered mass, untruncated	$2^{1/(z-1)}$: 1,024 $\times$ at $z = 1.1$, 10.1 $\times$ at $z = 1.3$
Monte Carlo against closed form	coverage within 0.00078, standard scores within 1.99; learned and escalation rates within 2.16

B:29 Engineer-years run out before variety does: the 101st adds 70 times less coverage than the second, and 10^4 rules leave 6.02 bits where the bound allows 0.12, because rare rules rarely fire. The bound describes channel capacity, and the regulators that approach it generate their replies. The cheapest design puts rules at the head, a model in the body and people where the model fails (Proposition 4.2), and logs stay a sample behind the stream: a rule for every type seen in a day misses the 15% of tomorrow's traffic that is new.

B:30

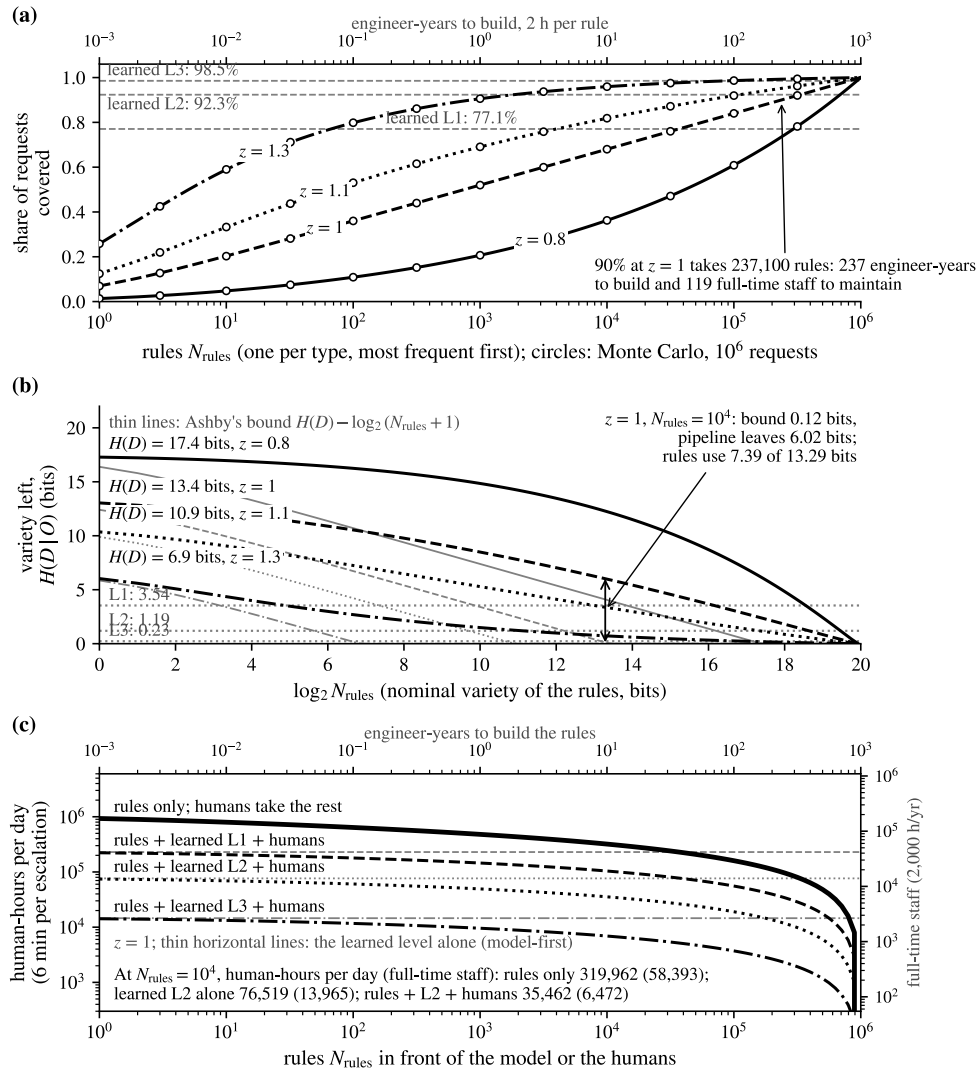


Figure B.2. Share of requests covered by the best N_{rules} rules against N_{rules} for Zipf exponents $z = 0.8, 1, 1.1$ and 1.3 , with Monte Carlo checks and the coverage of three learned levels at $z = 1$ (a); variety left, $H(D|O)$ in bits, against the rules' nominal variety $\log_2 N_{\text{rules}}$, with Ashby's bound for each z (b); and human-hours a day at $z = 1$ and ten million requests a day for rules alone, for rules in front of each learned level and for each level alone (c). At $z = 1$ each tenfold increase in rules buys about the same 16 points of coverage, ten thousand rules leave 6.02 bits where the bound allows 0.12, and rules in front of a learned model need the fewest people.

B:31 The if-tree is a best case, and real rules misfire, conflict and decay. The environment is assumed: real streams have unknown exponents and fuzzy types, a template rule may cover a family of types and a multi-turn request may need many rules, and for $z \leq 1$ every result depends strongly on N_{types} . The learned policy is deterministic given the type, so inconsistent replies, which would raise Ashby's floor through $H_D(R)$, are absent; escalation assumes every failure is detected, while undetected ones become the silent errors behind the drop in quality that Klarna's chief executive acknowledged in 2025 (4:31). Labor parameters are round numbers that scale every hour. The entropy is uncertainty about the request type, to which the bound applies exactly because the regulators are deterministic. Inference is not counted: at Jev's list price a million short requests cost about ten dollars (§3.3).

B:32 The harmonic sums are exact, so the run takes about five seconds, with 10^6 Monte Carlo requests per exponent from seed 20261003.

B.4 Condorcet with clones

B:33 Clones of one model vote like a jury with a Condorcet ceiling: in the model, at single-agent accuracy 0.6 and latent error correlation 0.2, ten million clones are right 71.4% of the time, carry the information of 7.94 independent votes and decide like 7.37 independent voters. This is Experiment 7.2 in §7.5, the exact form of Proposition 7.3. It asks what N correlated clones are worth as a jury, how many independent voters they equal, whether a verifier changes the answer, and when a diverse crowd of weaker agents beats them.

B:34 Each binary question carries a shared difficulty factor Z_0 and each voter an idiosyncratic factor Z_j , independent standard normals, and voter j is right iff $\sqrt{\rho} Z_0 + \sqrt{1-\rho} Z_j \leq \Phi^{-1}(p)$. Given Z_0 , votes are independent with accuracy

$$\Pr[\text{correct} \mid Z_0] = \Phi\left(\frac{\Phi^{-1}(p) - \sqrt{\rho} Z_0}{\sqrt{1-\rho}}\right),$$

so the count of correct votes is binomial, and majority accuracy Acc_N , ties broken by a coin, is its average over Z_0 . Its limit is the Condorcet ceiling $\Phi(\Phi^{-1}(p)/\sqrt{\rho})$, approached at rate $1/N$, and the mirror $\text{Acc}_N(1-p) = 1 - \text{Acc}_N(p)$ sends sub-chance crowds to one minus the ceiling, which is zero only when $\rho = 0$.

B:35 Every $\mathbf{q}$ here is latent. Votes correlate less, $\rho_v = (\Phi_2(\Phi^{-1}(p), \Phi^{-1}(p); \rho) - p^2)/(p(1-p))$, with Φ_2 the bivariate normal distribution function; at $p = 0.6$, latent 0.1, 0.3, 0.6 and 0.9 give vote correlations 0.062, 0.191, 0.406 and 0.711. The effective jury is $n_{\text{eff}} = N/(1 + (N-1)\rho_v) \rightarrow 1/\rho_v$, the accuracy-equivalent jury is the number of independent voters with the same majority accuracy, and a beta-binomial jury at the same vote correlation has the same ceiling within 0.002. On generative tasks a share 0.1 lies beyond the family's reach, so a sound verifier keeping any correct sample tends to 0.9 while a vote tends to 0.9 times the ceiling. Abler clones beat a more diverse crowd in the limit iff their $\Phi^{-1}(p)/\sqrt{\rho}$ is larger. Integrals over Z_0 use 96-node Gauss–Legendre pieces on $[-12, 12]$, split where the conditional accuracy crosses the levels that matter, and SciPy's binomial tails; adaptive quadrature, an exact simulation and an agent-level simulation check them.

B:36

Parameter	Value	Basis
Single-agent accuracy p	0.6, and its 0.4 mirror	a weak voter, better than chance
Latent correlation $\mathbf{q}$	0, 0.01, 0.05, 0.2, 0.5	from independent voters to clones with shared blind spots
Crowd size N	odd, 1 to $10^7 + 1$	up to ten million clones
Share of generative tasks out of reach	0.1	assumption
Per-sample success on generative tasks	0.6, 0.2, 0.02, at $\mathbf{q} = 0.2$ (also 0 and 0.5)	easy, hard and very hard
Clones against a diverse crowd	(0.7, $\mathbf{q} = 0.4$) against (0.6, $\mathbf{q} = 0.05$)	abler but correlated, weaker but nearly independent
Map of which crowd wins	clone accuracy 0.5–0.95 (46 values) by clone $\mathbf{q}$ 0–0.8 (41 values), $N = 10^4$	

Parameter	Value	Basis
Monte Carlo	2×10^5 questions per point; 2×10^4 at agent level; seed 20261003	

B:37

Latent q	Vote ρ_v	$N = 11$	$N = 101$	$N = 1,001$	$N = 10^7 + 1$	Ceiling	First odd N within 0.001
0	0	0.7535	0.9791	1.0000	1	1	n/a
0.01	0.0062	0.7469	0.9443	0.9908	0.99435	0.99435	3,299
0.05	0.031	0.7249	0.8404	0.8679	0.87139	0.87139	3,537
0.2	0.126	0.6756	0.7087	0.7139	0.71447	0.71447	601
0.5	0.330	0.6316	0.6389	0.6398	0.63994	0.63994	105

B:38

Latent q	0.01	0.05	0.2	0.5
Vote ρ_v	0.00622	0.03116	0.1259	0.3297
Effective jury limit $1/\rho_v$ (reached at $N = 10^7 + 1$)	160.7	32.1	7.94	3.03
Accuracy-equivalent jury of 10^7 clones	156.7	30.9	7.37	2.66
Small- q approximation $\pi/(2\rho)$	157.1	31.4	7.85	3.14
First odd N with 99% of the possible gain	915	1,295	525	263
N times the distance to the ceiling, predicted and at $N = 10^7 + 1$	3.1739, 3.1740	3.5500, 3.5500	0.60475, 0.60475	0.10528, 0.10528
Accuracy at $p = 0.4$, $N = 10^7 + 1$ (0.0000 at $q = 0$)	0.0056	0.1286	0.2855	0.3601

B:39 In this model, a million clones of a 60% model whose votes correlate at 0.5 are right about 62% of the time: a vote correlation of 0.5 is a latent q of 0.711 at $p = 0.6$, whose ceiling is 0.6181, and $N = 10^6 + 1$ gives 0.61809. Reading the latent row for 0.5 as a vote correlation would give 0.640 and overstate the crowd. For an infinite crowd at $p = 0.6$ to reach 0.9, 0.99 or 0.999, q must be below 0.0391, 0.0119 or 0.0067; at $p = 0.7$, below 0.167, 0.0508 or 0.0288.

B:40

Generative tasks, $q = 0.2$	One agent	Best of N , $N = 1,001$	Best of N , $N = 10^7$	Majority vote, $N = 10^7$
per-sample success 0.6	0.54	0.9000	0.900	0.643
per-sample success 0.2	0.18	0.89998	0.900	0.0269
per-sample success 0.02	0.018	0.857	0.900	2.0×10^{-6}

B:41

N to reach 99% of the verifier's 0.9	$q = 0$	$q = 0.2$	$q = 0.5$
per-sample success 0.6	5.0	8.6	42.5
per-sample success 0.2	20.6	77.9	3,968
per-sample success 0.02	228	4,168	9.1×10^6

B:42

Clones (0.7, $q = 0.4$) against a diverse crowd (0.6, $q = 0.05$), and checks	Value
Who leads	clones up to $N = 27$; the crowd from $N = 29$ (0.78752 against 0.78773) at every checked size up to 10,001
Ceilings; accuracies at $N = 10^4$	0.7965 and 0.8714; 0.7965 and 0.8710
Correlation below which clones at 0.7 match the crowd in the limit	0.214, so accuracy $0.6 \rightarrow 0.7$ is worth 4.3 times the tolerable correlation
Share of the map the clones win; tie line at $N = 10^4$ against the limit	46.2%; within 0.0004 in accuracy
Exact Monte Carlo against quadrature	standard scores within 2.51 (majority) and 2.13 (verifier and vote)
Agent-level simulation	standard scores within 1.95; vote correlations within 1.3% of the formula
Gauss–Legendre against adaptive quadrature; mirror identity	agree to 2.4×10^{-14} ; holds to 1.7×10^{-14}
Tie identity (each even N scores like the odd N below it); monotonicity	holds to 6.7×10^{-16} ; accuracy rises along the 90-point grid of odd N for every q

B:43 Behind a sound verifier, correlation no longer caps the result and only slows the search, which is how Proposition 11.4 values clones as a search. Ability wins small committees and diversity wins large crowds: at large N ability enters only through $\Phi^{-1}(p)$ and diversity only through $1/\sqrt{p}$.

B:44

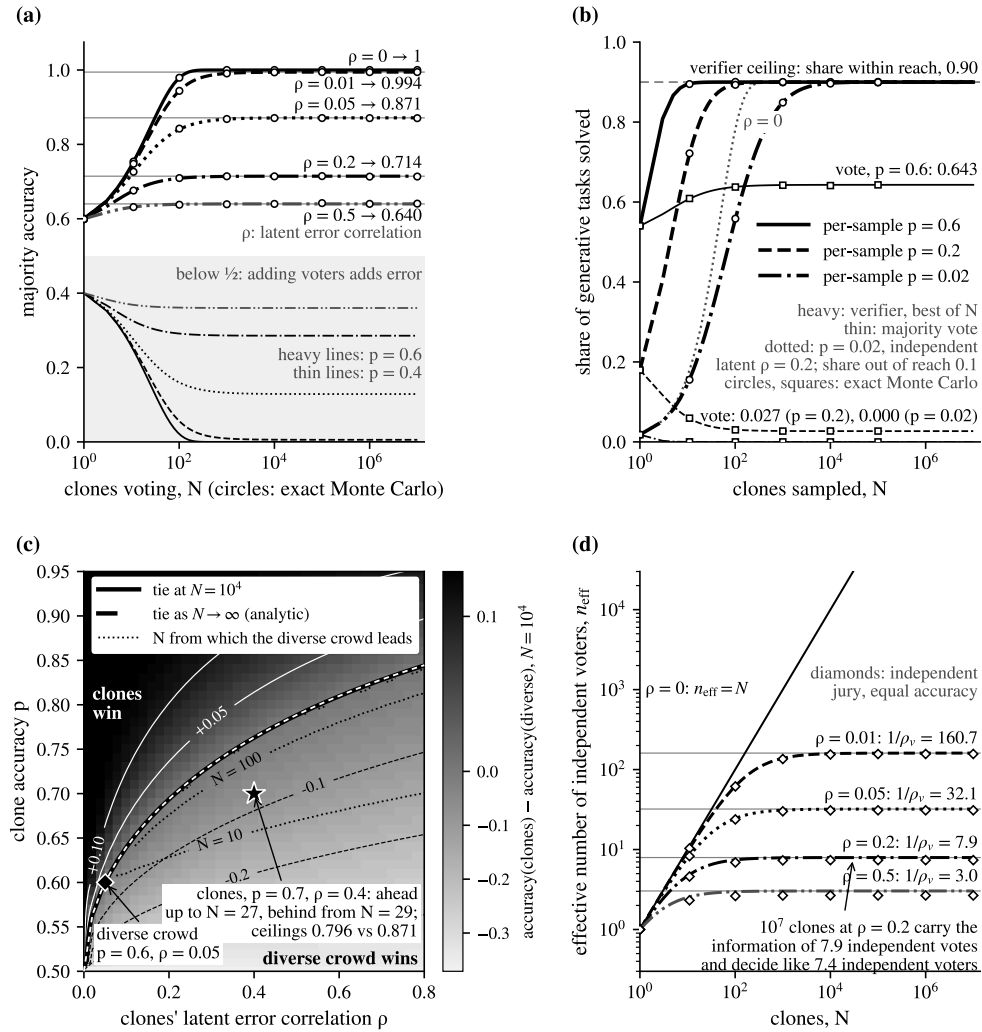


Figure B.3. Majority accuracy of N correlated clones against N for latent error correlations $\rho = 0, 0.01, 0.05, 0.2$ and 0.5 at single-agent accuracy 0.6 , with the 0.4 mirror below one half (a); share of generative tasks solved by a verifier keeping the best of N samples and by majority vote, for three per-sample success rates at latent $\rho = 0.2$ (b); accuracy of clones minus a diverse crowd ($p = 0.6, \rho = 0.05$) at $N = 10^4$ across clone accuracy and latent ρ , with the finite and limiting tie lines (c); and the effective number of independent voters n_{eff} against N , with its limits $1/\rho_v$ (d). In the model every correlated jury stops at its ceiling within a few thousand voters, the verifier lifts every curve to the 0.90 within reach, and ten million clones at latent $\rho = 0.2$ carry the information of 7.94 independent votes.

B:45 One Gaussian factor is a modeling choice: real errors have several factors and heavier tails, and no ρ here was measured on a real model. Questions are binary and need a strict majority, while plurality voting on open answers can win with fewer than half correct when wrong answers scatter. The verifier is sound, complete and free; if wrong samples could pass, false accepts would grow with N , and best-of- N also costs N samples and N checks. Reach is a point mass, where real coverage rises roughly log-linearly in samples. Diversity is free in the map, while diverse models are often weaker; the result that diverse groups beat able ones ([Hong and Page](#)) is contested ([Thompson](#)), and at small N one strong model aggregated with itself can beat a mix of weaker ones ([Li and others](#)). Voters do not talk, and deliberation can raise correlation ([Lorenz and others](#)) or lower error ([Becker, Brackbill and Centola](#)).

B:46 The quadrature runs in 20–40 seconds; the simulations use seed 20261003.

B.5 The topology model

B:47 Grown from rules that never encode the claim that intelligence licenses autonomy, an agent-based organization reproduces the ridge of Proposition 7.1: in a high-variety world top-down wins among weak agents and bottom-up among strong aligned ones, and a low-variety world is best run top-down at every level of intelligence. This is Experiment 7.1 in §7.3, whose surface is plotted in Figure 7.2. The experiment asks whether the surface of (7.1) emerges from plausible behavior, and what else that behavior shows.

B:48 Forty agents act for 160 periods in a 12-dimensional decision space, with Beta-distributed competence of mean $\bar{i}$ and concentration 12. Each period brings a common goal, a random unit vector, and for each agent a local situation, the goal plus a random unit vector scaled by the spread of local situations, normalized. Agents read goal and situation through signals that weight the truth by their competence and the rest by an error of which a share ϱ is a misconception common to all (Definition 7.1). The apex is the agent with the highest competence as observed with noise of standard deviation 0.05, or 50 for a random apex; it tailors directives for 5 agents a period from its reading of their situations and sends everyone else a generic directive from its reading of the goal. An agent's proposal mixes, with weight a , its reading of its situation plus the pooled reading of the goal, and with weight $1 - a$ a persistent personal agenda. Its effort $\mathbf{e}_j$ is the normalized mix of directive and proposal in proportions $1 - \tau$ and τ , executed with noise 0.6 times one minus its competence. With $\mathbf{b}_j$ its local situation, effectiveness is

$$F = \overline{\mathbf{b}_j \cdot \mathbf{e}_j} - c_{\text{opp}} \overline{\max(0, -\mathbf{e}_j \cdot \mathbf{e}_k)} - c_{\text{meet}} \tau,$$

local fit minus the cost of opposed efforts minus meetings, averaged over agents and over pairs. The conflict term charges only opposed efforts, as (7.2) requires, and the model never measures a summed norm, so it agrees with that law without testing it.

B:49

Parameter	Value
Agents, dimensions, periods	40, 12, 160
Mean competence $\bar{i}$	0.1 to 0.95 in steps of 0.05 (concentration 12)
Decentralization τ	0 to 1 in steps of 0.1; τ^* is the best of the eleven
Apex	tailors directives for 5 agents a period; selection noise 0.05 (merit) or 50 (random)
Misconception share ϱ	0.6 (sweep 0, 0.3, 0.6, 0.9)
$c_{\text{opp}}, c_{\text{meet}}$	0.6; 0.06 per unit of τ
Spread of local situations	0.4 (low variety), 1.6 (high variety)
Alignment a	0.2, 0.5, 0.8, 1.0
Execution noise	$0.6 \times (1 - \text{competence})$
Replicates	3 per cell (2 in the misconception sweep), seeded by grid point and replicate

B:50

High variety, merit apex	First $\bar{i}$ with $\tau^* > 0$	$\bar{i}$ at which τ^* reaches 1	τ^* at $\bar{i} = 0.95$
$a = 1.0$	0.25	0.60	1
$a = 0.8$	0.25	0.65	1
$a = 0.5$	0.30	never	0.5

High variety, merit apex	First i with $\tau^* > 0$	i at which τ^* reaches 1	τ^* at $i = 0.95$
$a = 0.2$	never	never	0

B:51

High variety	i	a	τ	F
mediocre, top-down	0.35	0.8	0	0.46
mediocre, bottom-up	0.35	0.8	1	0.34
brilliant but misaligned, bottom-up	0.8	0.2	1	0.06
brilliant and aligned, top-down	0.8	0.8	0	0.57
brilliant and aligned, bottom-up	0.8	0.8	1	0.73

B:52

Further results	Value
Low variety, best F at $i = 0.1, 0.5, 0.95$	0.24, 0.86, 0.94, all top-down, at every alignment
High variety, top-down F at $i = 0.1, 0.15, 0.35$, merit apex	0.14, 0.22, 0.46
The same, random apex	0.06, 0.04, 0.24; $\tau^* > 0$ already at $i = 0.1$
First i with $\tau^* > 0$ at misconception shares 0, 0.3, 0.6, 0.9	0.10, 0.15, 0.25, 0.35

B:53 At every alignment $\tau^* = 0$ for $i \leq 0.2$. Top-down mediocre agents beat bottom-up brilliant but misaligned ones, 0.46 against 0.06, a comparison that changes alignment as well as topology, and aligned talent does better bottom-up than top-down, 0.73 against 0.57. With a merit apex, top-down effectiveness plateaus near 0.57 however intelligent the agents, while bottom-up keeps rising, to 0.77 at $i = 0.95$. Among weak agents a random apex costs top-down half or more of its score and makes partial decentralization pay at once, so how the apex is chosen decides whether hierarchy pays there (7:23), and shared misconceptions raise the intelligence at which autonomy first pays.

B:54 The model is stylized: competence maps linearly to signal quality, the apex shares everyone's signal model, and the thresholds depend on the span of control, the two costs and q . With three seeds a cell, τ^* curves are noisy to about ± 0.1 , most under random selection; the monotonicity of τ^* in i and a , its zero at low variety and the selection effect survive every sweep. The threshold differs from Condorcet's one half because i weights the truth in a continuous signal; for votes, §B.4 keeps the sign flip at exactly one half for every correlation.

B:55 The full grid, 18 intelligence levels by four alignments and two varieties plus the random-apex and misconception sweeps, runs in about a minute.

B.6 Swarm scaling

B:56 A swarm that merges through one fixed verifier is capped at its capacity, and a swarm that staffs its own judges turns the cap into a tax set by the verification asymmetry. In the model, a hierarchy behind a shared merge gate is pinned at 424 merged units per unit time for every size above 950 agents, while a planner/worker/judge swarm of 10^5 agents reaches 7,930. This is Experiment 9.1 in §9.5. The experiment asks how useful throughput scales with N under five coordination topologies and what binds at large N .

Public account	What it reports	Where it enters the model
Cursor, Scaling long-running autonomous coding , January 2026	flat coordination through a locked file slowed twenty agents to the throughput of two or three; planners, workers and a judge worked	the locks topology and its calibration; the planner/worker/judge topology
Cursor, Towards self-driving codebases , February 2026	about 1,000 commits an hour over ten million tool calls; requiring correctness before every commit serialized the system, so it accepted a small, stable error rate (9:25)	the merge gate's capacity; lenient gating
Cursor, Agent swarms and the new model economics , July 2026	purpose-built version control at about 1,000 commits a second; an older run's hottest file collected 7,771 of its more than 70,000 conflicts, from 1,173 agents; review costs far less than the work it audits (9:24)	file popularity; the judges' cost
Anthropic, How we built our multi-agent research system , June 2025	a lead agent with parallel workers beat a single agent by 90.2% on an internal evaluation, token use explained 80% of the variance, and it used about 15 times the tokens of chat	parallel workers under a planner
Anthropic, Building a C compiler with a team of parallel Claudes , February 2026	sixteen agents coordinated through git and lock files, and a nearly perfect verifier carried the design (9:14)	high recall; the serial share of one monolithic task
The July 2026 swarm (§9.1)	a shared board, roles and norms nobody sanctioned	coordination only; nothing operational is modeled

B:58 Time is counted in the mean time an agent takes to produce one work unit, about a commit. A swarm splits into workers, planners or managers, who produce nothing, and judges, and works a backlog of ready units that never empties. A submitted unit is defective with probability 0.1; the verifier rejects a defective unit with probability equal to its recall, never rejects a good one, and sends rejected units back. Each merged unit depends on 3 parents among the last 1,000 merges. An accepted defective unit is a latent defect: a verified merge exposes each latent defect among its parents with probability half the recall, since tests cover half a parent's behavior, and exposure reverts the parent and every unit merged on top of it. Net throughput is merged minus reverted units per unit time, and after every merge the swarm pauses for its serial work, on average one serial unit per $(1 - s_{\text{ser}})/s_{\text{ser}}$ merges.

B:59 The five topologies:

- Flat broadcast: every resolved submission costs all agents a pause of $\mu(N - 1)$, which with the serial pause is (9.1) times the acceptance rate, peaking at $N^* = \sqrt{(1 - s_{\text{ser}})/\mu}$.
- Shared files with locks: each unit edits one of 1,000 files with Zipf popularity and pays 0.05 to read the board, holding its file's lock until its merge is accepted while others wait.
- Optimistic merges: no locks; a finished unit is redone with probability $1 - \exp(-0.5 N_{\text{ed}}/34.1)$, where N_{ed} agents are editing and 34.1, the inverse of the sum of squared file popularities, is the collision-equivalent number of files.
- Hierarchy: about one manager per seven workers in as many levels as a branching of 8 needs; each unit pays 8μ of coordination and 0.05 of assignment latency per level, and residual conflicts of 0.02 force redos.

- Planner/worker/judge: a planner tree of span 20 plans ahead, workers own disjoint scopes with residual conflicts of 0.02 and a handoff cost of 0.05, and each judge spends $1/\alpha$ per check, so verifier capacity is judges times α .

B:60 All but the last merge through one merge gate, a verifier plus version control that admits 500 submissions per unit time. Strict gating verifies every merge and makes agents queue; lenient gating merges a submission unverified after a wait of 0.25, Cursor's small but stable error rate. A mean-field model gives analytic curves, and its best judge share is

$$s_{\text{judge}}^* \approx \frac{1/(\alpha t_u)}{1 + \frac{1}{19} + 1/(\alpha t_u)}, \quad t_u = \frac{1.05}{0.98},$$

with t_u a worker's time per accepted unit including handoff and conflicts and 19 the planners' span less one: close to $1/(1 + \alpha t_u)$, so the cheaper the check, the smaller the tax.

B:61

Parameter	Value	Basis
Agents N	1 to 10^5	14 simulated sizes; mean field on a 90-point log grid
Defect rate	0.10	one unit in ten needs rework
Recall	0.9 (0.5 and 0.99 for latent defects)	a verifier must be nearly perfect
Unit times	lognormal, mean 1, shape 0.5	about $\pm 50\%$
Serial share s_{ser}	10^{-4} (also 0 and 10^{-3})	two units on the critical path per 20,000; Amdahl ceiling 9,100
Pairwise cost μ	0.002	gives $N^* = 22$
Merge gate	500 submissions per unit time	about Git's 1,000 commits an hour, if a unit is about 30 agent-minutes
Hierarchy	branching 8; 0.05 per level	span of control; one agent turn per level
Planner span	20	planners fan out widely
Judge cost	$1/\alpha$, $\alpha = 10$ (also 3 and 30)	review costs far less than the work it audits
Files	1,000, Zipf exponent 1, conflict constant 0.5	the top file carries 13.4% of units; Cursor's hottest file carried 11.1% of conflicts
Residual conflicts	0.02	what decomposition leaves
Dependencies	3 parents from the last 1,000 merges; tests cover half a parent	
Lenient wait; time step	0.25; 0.05	the step is also the granularity of lock grants
Seed	20261003	188 deterministic simulations

B:62

Topology	In the model	Simulation check
flat broadcast	peaks at $N^* = 22.4$ with 10.17 merged units per unit time, a speed-up of 11.4	10.18 at $N = 22$; 1.73 at 256

Topology	In the model	Simulation check
shared files with locks	plateaus at 6.34, the output of 7.5 single agents, from $N \approx 20$	6.23 at 22; 6.32 at 1,024, where 99.3% of agent time waits for locks
optimistic merges	peaks at 21.2 near $N = 71$, then collapses	21.26 at 64; 5.1 at 256, where 97.6% of finished units conflict
hierarchy behind the merge gate	pinned at 424 for every $N \geq 950$	at 10^5 it could submit 58,200 units per unit time; the gate admits 0.73%, and workers wait 99.2% of the time
planner/worker/judge, best judge count	683, 5,089 and 7,930 at $N = 1,024$, 16,384 and 10^5 , with 85, 1,334 and 8,146 judges	676, 5,046 and 7,811

B:63

Further results	Value
Planner/worker/judge at $N = 10^5$	4,594 planners, 87,260 workers; 18.7× the gate-bound hierarchy; 87% of the Amdahl ceiling
Same swarm with serial share 0 or 10^{-3}	72,517 or 880
Output per agent with self-staffed judges and negligible serial work	about 0.72 units per unit time; behind one Git-class gate, agents beyond about 10^3 add nothing
Pairwise cost implied at $N = 100$	flat 2.0×10^{-3} ; locks 1.25×10^{-3} ; optimistic 3.3×10^{-4} ; hierarchy 4.0×10^{-5} ; planner/worker/judge 6.3×10^{-6} , 320× below flat
Best judge share, simulated (formula)	$\alpha = 3$: 22.8% (22.8%); $\alpha = 10$: 8.0% at $N = 100$, 8.4% at 10^3 and 10^4 (8.1%); $\alpha = 30$: 3.2% (2.9%)
Peak throughput per agent at $N = 10^2, 10^3, 10^4$	0.707, 0.667, 0.399 (simulated 0.695, 0.660, 0.397), the last held down by serial work
Misallocated judges at $N = 1,000, \alpha = 10$	1% judges: -87% (0.088 per agent); 30% judges: -22% (0.516)
Latent defects per 1,000 merged units, judges scaled, recall 0.5, 0.9, 0.99	27.1, 2.91, 0.252, flat in N (simulated 25.0–26.4, 2.6–3.3, 0.17–0.31)
Merged units reverted when those defects surface	8.3%, 2.2%, 0.22%
Lenient shared gate at $N = 10^5$	0.85% of merges verified; throughput 7,792 at recall 0.9
Mean field against simulation, throughput	median deviation 0.2% (flat), 1.3% (locks), 0.5% (optimistic), 0.2% (hierarchy), 0.8% (planner/worker/judge); worst 7.4%, at $N = 22$
Mean field against simulation, judge share and defects	median at most 0.6% (maximum 5.7%); 4.8%, 5.6%, 14.7% at recall 0.5, 0.9, 0.99, the last the Poisson noise of about 15 events a run

B:64

Lenient shared gate: latent defects per 1,000	$N = 1,000$	$N = 3,162$	$N = 10^5$, simulated	$N = 10^5$, mean field
recall 0.5	31.2	72.8	99.1	99.1
recall 0.9	7.9	57.3	98.7	98.5
recall 0.99	4.6	53.2	99.0	98.3

B:65 A fixed merge gate caps every topology that shares it, and past the knee more agents add only queue. Staffing the verifier moves the binding constraint to the serial share, so verification binds only once serial work is small. Misallocation is asymmetric, since too few judges lose throughput at a rate of α per missing judge and too many lose about one unit per extra judge, so a designer unsure of α should err toward more verification. A lenient gate turns a verification deficit into latent defects whatever the recall, 390 times as many at recall 0.99.

B:66 Optimistic merging is the one-file case of the conflict law of Proposition 9.1. If each task edits n_{edit} of N_{files} equally likely files, two tasks overlap with probability close to $n_{\text{edit}}^2/N_{\text{files}}$, so an agent conflicts with probability $p_{\text{conf}}(N) \approx 1 - \exp(-n_{\text{edit}}^2(N-1)/N_{\text{files}})$ and a shared repository carries about $N_{\text{files}}/n_{\text{edit}}^2$ agents. Under Zipf popularity the collision-equivalent count replaces N_{files} , which is why optimistic merging peaks near $34.1/0.5 \approx 68$ agents. The probability that two tasks both touch one given file, $(n_{\text{edit}}/N_{\text{files}})^2$, undercounts overlaps by a factor of N_{files} , and a coupling built on it overstates how many agents a repository carries.

B:67

Conflict law	Value
Direct simulation against the law ($N_{\text{files}} = 10^4$; $n_{\text{edit}} = 1, 3, 10$; $N = 3$ to 10^4 ; 24 points)	within 0.004
Carrying capacity at $n_{\text{edit}} = 3$	about 1,111
Share of cycles with a conflict at $N = 100, 1,000, 3,000$	8.5%, 59%, 93%
Peak of $N(1 - p_{\text{conf}})$ if conflicted work is lost	409 agent-equivalents at $N = 1,111$
Optimistic-merge peak if partitioning made all 1,000 files equally popular	$N = 2,000$
Hold time per unit that lets one shared lock slow 20 agents to 2–3	0.33–0.50 time units (exact mean-value analysis, unit think time)
Share of units the hottest file would need to carry for the same slowdown	about 35%, against 13.4% under Zipf

B:68 Cursor's collapse therefore implies a hotter coordination point than the base case, as its report that agents held locks too long suggests (Cursor).

B:69

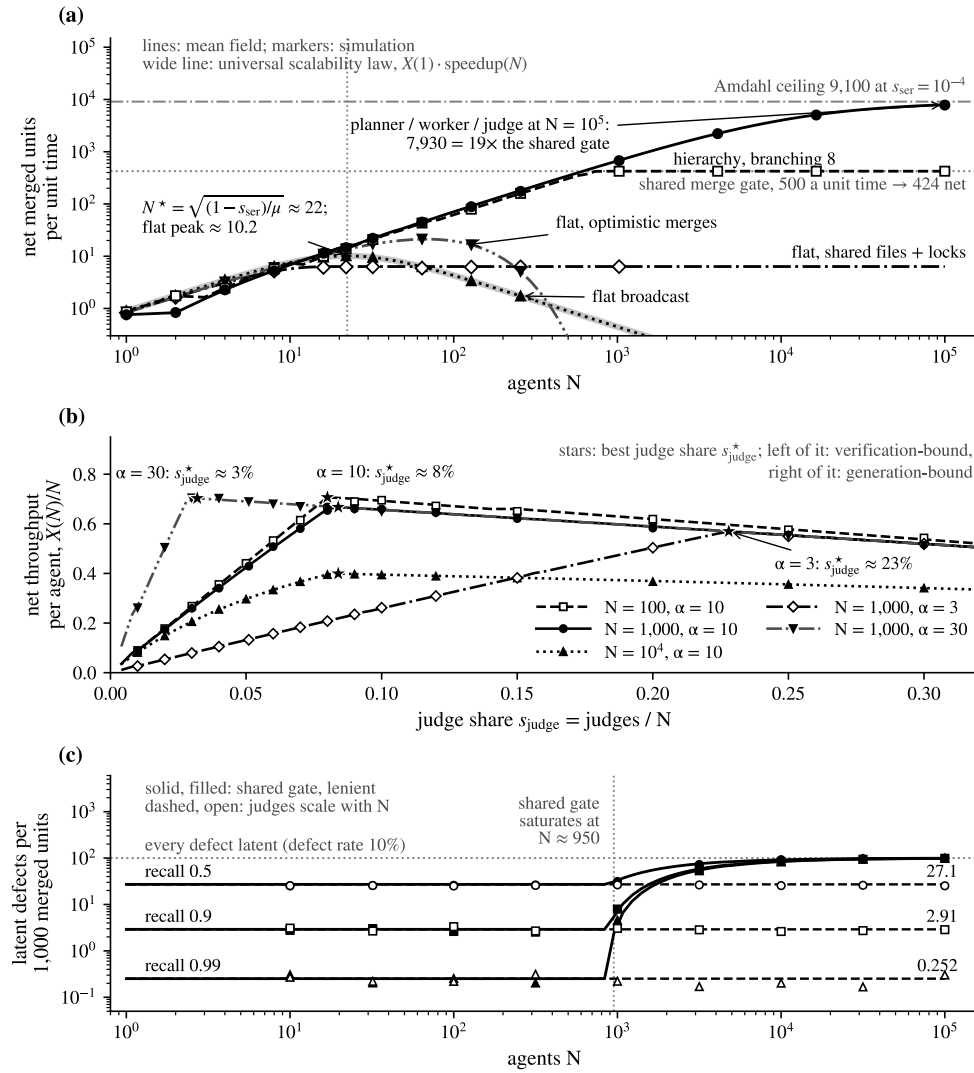


Figure B.4. Net merged units per unit time $X(N)$ against the number of agents N for five coordination topologies, with the pairwise peak, the shared merge gate and the Amdahl ceiling (a); throughput per agent $X(N)/N$ of a planner/worker/judge swarm against the judge share s_{judge} , for three swarm sizes at $\alpha = 10$ and for $\alpha = 3$ and 30 at 1,000 agents (b); and latent defects per 1,000 merged units against N for three recalls, with judges scaled to the swarm and behind a lenient shared gate (c). In the model the hierarchy behind the shared gate is pinned at 424 while the flat topologies stall far below it on their own coordination, the best judge share falls as checking gets cheaper, and past the gate's saturation near 950 agents latent defects climb toward the defect rate whatever the recall.

B:70 Every parameter is an assumption chosen to give the model its structure, none is fitted, and the companies' figures are self-reported. The dependency graph only propagates defects, and scheduling enters only through the serial share. Agents behave identically, so the failures Cursor reports (risk aversion without ownership, going stale over long runs, tunnel vision, planners that do not wake up), planning errors, false positives and correlated judge misses are absent, and each would lower the curves. The locks mean field assumes exponential service where the simulation uses lognormal, and reverting a parent reverts only its direct dependents. Merge capacity for planner/worker/judge is agent-scale, as in Cursor's purpose-built version control; on a Git-class gate it would share the hierarchy's 424.

B:71 The 188 simulations take about 50 seconds from seed 20261003, and the conflict-law simulation a few more.

B.7 The research loop

B:72 With compute fixed, the model gives a one-in-five chance that the first year of fully automated research compresses more than ten years of progress at the 2020–24 pace, matching [Davidson and Houlden](#), and year-one progress tracks the launch speed more than the returns to research. This is Experiment 11.1 in §11.4, plotted in Figure 11.3. It puts brakes into the research loop, a compute constraint, diminishing returns and physical limits, and asks how likely a runaway is, how fast it would be, and which uncertainty matters most.

B:73 The model is the law of motion (11.2), the structure of [Davidson and Houlden's](#) software intelligence explosion with a CES constraint from experiment compute ([Davidson](#); [Whitfill and Wu](#)). Software efficiency starts at $S_0 = 1$, and automated research labor equals it, $L = S$, because the same inference fleet runs more and faster agents as software improves; experiment compute is fixed or grows at g_C . With $\varepsilon_s = 1$ and fixed compute the law reduces to $\dot{S} \propto S^{1+\theta s_L-\beta}$, so each doubling comes faster than the last iff $r = \theta s_L/\beta > 1$; at the median $r = 1.2$ the exponent is about 1.05, and the knife-edge stays at $r = 1$ (Proposition 11.1). With $\varepsilon_s < 1$ automated labor's benefit is capped at a multiple of compute, the homeostat of Proposition 11.2. The factor $1 - \ln S / \ln S_{\max}$ stands for physical limits, and the base pace $g_0 = \ln 3$ a year is close to the 2020–24 trend, in which the compute needed for a fixed performance halved about every eight months ([Ho and others](#)).

B:74

Parameter	Prior	Basis
Returns to research r	log-uniform 0.4–3.6, median 1.2	Davidson and Houlden's range
Stepping on toes θ	uniform 0.4–0.8	parallel researchers duplicate work
Labor share s_L	0.5	
Launch speed ξ	log-uniform 2–32, median 8	research speed-up at full automation
Ceiling $\ln S_{\max}$	uniform 6–16 orders of magnitude	distance to effective limits
Elasticity ε_s	mixed: $(\varepsilon_s - 1)/\varepsilon_s$ uniform on $[-0.5, 0]$, so 0.67–1; Cobb–Douglas: 1; strong: on $[-0.5, -0.2]$, so 0.67–0.83	Davidson argues for 0.83–1; Epoch cites 0.7 from manufacturing (Erdil and Barnett)
Compute growth	none, or 2.5–4 times a year	fleet growth in gigawatts times gains per watt
Draws and integration	12,000 per scenario; Euler steps of 10^{-3} years over 4 years; NumPy seed 11	

B:75

Scenario	More than 3 years of progress in year one	More than 10 years	Median years in year one	Median orders of magnitude after a year	6 orders within two years
fixed compute, mixed ε_s	79%	20%	5.3	2.5	23%
fixed compute, Cobb–Douglas	82%	35%	6.4	3.1	47%

Scenario	More than 3 years of progress in year one	More than 10 years	Median years in year one	Median orders of magnitude after a year	6 orders within two years
fixed compute, strong constraint	79%	15%	5.1	2.4	17%
compute growing 2.5–4 times a year	87%	27%	6.3	3.0	37%
<u>Davidson and Houlden</u> , for reference	about 60%	about 20%	n/a	n/a	n/a

B:76 Rank correlations of year-one progress with each parameter, for fixed compute and mixed ε_s , are 0.80 for the launch speed, 0.52 for r , 0.14 for ε_s , 0.05 for the ceiling and -0.01 for θ : how many and how good the automated researchers are sets the first year, while r and ε_s set whether the explosion continues. In the phase portrait, $r = 3$ without complementarity ($\varepsilon_s = 1$) makes speed climb 36-fold before the ceiling, from 8 to 288 times the 2020–24 pace, $r = 0.5$ makes each gain slow the next, the median $r = 1.2$ keeps speed roughly flat, which is fast exponential progress, and a strong compute constraint bends even $r = 3$ down. Growing compute raises the chance of six orders of magnitude within two years from 23% to 37%; that growth exceeds the 2–2.4 times a year of gigawatts at the midpoints of the fleet priors in §B.8, so the build-out delivers it only with gains per watt of roughly 1.05–2 times a year. Correlated clones (§B.4) and verification throughput (§B.6) are brakes folded into θ and ξ , since each lowers effective labor.

B:77 This is a one-sector model with no retraining lag, fixed chip technology and production, and a stylized ceiling. Years of progress are counted against the threefold-a-year software baseline, and capability benchmarks are not modeled. Ramp-up and retraining delays are omitted, which is why the chance of more than three years in one, 79%, exceeds Davidson and Houlden's 60%: the bulk of the distribution is an upper envelope, while the ten-year tail matches. The probabilities are only as good as the priors, which are contested, so three regimes of ε_s are reported.

B:78 The run integrates 12,000 draws per scenario from NumPy's generator seeded with 11 and takes about 70 seconds.

B.8 Agents per gigawatt

B:79 A decode roofline over 20,000 draws of frontier models gives medians of 0.37 million, 1.31 million and 4.9 million concurrent long-context agents per critical-IT gigawatt at 100 tokens a second on B200 HGX, GB300 NVL72 and Rubin NVL72, and a median 8.8 million agents for the two largest labs' inference fleets at the end of 2026. At a measured configuration the same roofline gives 1.6–6.1 times more streams per GPU than were measured, so its counts are an upper envelope, and the measured rows of §10.4 are the ones to use. This is Experiment 10.1 in §10.3. It checks four counts the book relies on: about one long-context agent per B200, ten million automated researchers at 10,000 tokens a second by the end of 2027, a ten-trillion-parameter model at 100 tokens a second, and the two hyperbolas of speed against size.

B:80 A serving instance of n_{inst} accelerators holds weights of $N_{\text{tot}}b$ bytes and serves n streams; each decode step accepts n_{acc} tokens per stream and takes

$$t_{\text{step}}(n) = t_{\text{sync}} + \max\left(\frac{N_{\text{tot}}b s_{\text{touch}}(n) + n n_{\text{ctx}} b_{\text{kv}}}{n_{\text{inst}} \text{BW} \eta_{\text{bw}}}, \frac{1.3 n n_{\text{acc}} (1 + n_{\text{in}}) (2N_{\text{act}} + \mathcal{F}_{\text{attn}})}{n_{\text{inst}} \mathcal{F}_{\text{acc}} \eta_f}\right),$$

where t_{sync} is the synchronization part of the floor t_0 of Definition 5.2, the weight read being counted with the share $s_{\text{touch}}(n) = s_{\text{act}} + (1 - s_{\text{act}})(1 - (1 - s_{\text{act}})^n)$ of mixture-of-experts weights a batch touches, and $s_{\text{act}} = N_{\text{act}}/N_{\text{tot}}$. Streams per accelerator are the largest n that holds per-stream speed $n_{\text{acc}}/t_{\text{step}}$ at its target within memory, $N_{\text{tot}}b + n n_{\text{ctx}} b_{\text{kv}} \leq 0.92 n_{\text{inst}} H_{\text{hbm}}$, and agents per critical-IT gigawatt are streams per accelerator times $10^9/P_{\text{acc}}$ times η_{srv} , the full form of (10.1).

B:81 Two laws frame the result, the hyperbolas of (5.1). Across models, single-stream speed times active bytes is at most the effective bandwidth. On one chip, with step time $t_0 + t_{\text{kv}}n$, throughput per accelerator falls linearly in per-stream speed to zero at $\nu_{\text{max}} = 1/t_0$; fitted to two published operating points of GB200 NVL72 on DeepSeek-R1, 14,659 tokens a second per GPU at 18 per user and 1,149 at 164 ([InferenceX](#)), it gives $t_0 = 5.67$ ms and $\nu_{\text{max}} = 176$ tokens a second, beside the 169 of GB300's fastest published point ([OpenAI](#)).

B:82

Parameter	Prior or value	Basis
GB300: HBM, bandwidth, FP4, power per GPU	288 GB, 8 TB/s, 15 PF, 1.9–2.2 kW	NVIDIA ; 110,000 GB300 draw about 220 MW in the SpaceX S-1
Rubin	288 GB, 19.2–22 TB/s, 35 PF dense, 2.3–3.0 kW	NVIDIA
B200 (HGX)	180 GB, 8 TB/s, 9 PF, 1.8–2.1 kW	NVIDIA
Synchronization t_{sync}	GB300 4–6 ms, Rubin 2.5–4.5 ms, B200 5.5–8 ms	the floor fitted above, taken as synchronization; a 5.90 ms minimum time between tokens on GB300 (OpenAI)
Model size N_{tot} ; active share	log-uniform 1.5–10 trillion; 1.5–5%	frontier mixture-of-experts range
Weights; cache bytes per token b_{kv}	FP4 (60%) or FP8; log-uniform 40–250 KB	compressed caches to grouped-query attention
Context n_{ctx} ; uncached input per output token n_{in}	log-uniform 40,000–160,000; log-uniform 1–6	long-horizon agent traffic
n_{acc} ; η_{bw} ; η_f ; η_{srv}	uniform 1.3–2.5; 0.6–0.85; 0.25–0.5; 0.55–0.8	speculation; achieved efficiency
OpenAI and Anthropic, critical-IT GW	10–12.5 (2026), 18–26 (2027), 35–70 (2028)	Dylan Patel: both above 5 GW at the end of 2026 (Dwarkesh) and both at 10 GW by the end of 2027 (Dwarkesh)
Inference share	uniform 0.35–0.6	the rest trains and experiments
Draws	20,000 per configuration; NumPy seed 7	

B:83

Agents per critical-IT GW	p10	Median	p90
B200 HGX, 100 tok/s	about 0 (14% of draws infeasible)	0.37M	1.8M
GB300 NVL72, 100 tok/s	0.21M	1.31M	4.6M
Rubin NVL72, 100 tok/s	1.8M	4.9M	12.4M
GB300 NVL72, 50 tok/s	1.9M	5.3M	13.9M
Rubin NVL72, 50 tok/s	2.7M	6.8M	16.8M

B:84

Two largest labs' inference fleets, 100 tok/s	End of 2026	End of 2027	End of 2028
Concurrent agents, median	8.8M	34M	82M
80% interval	2.1–27.6M	14–75M	15–236M

B:85

Speed check	Value
Ten million researchers at 10,000 tok/s	10 ¹¹ tok/s, about 30× the end-2027 median fleet's decode output and 12× the end-2028 median
Ten million agents at the end of 2027	the 4th-percentile outcome: 96% chance of at least that many
Ten-trillion-parameter model, about 400 billion active, FP4	200 GB read per token; about 20 TB/s per stream for 100 tok/s, a fraction of one Rubin NVL72's roughly 1.4 PB/s; about 2 PB/s for 10,000 tok/s
8-GPU B200 servers; rack-scale GB300 and Rubin	median 1.06 streams per GPU; 4–20
GPU streams under 2–6 ms floors	a few hundred tok/s, about 1,000 with speculation
Faster designs, small active footprints	Taalas's hard-wired Llama 3.1 8B at 15,000–16,000 tok/s on its public demo (EE Times); Groq 3 LPX at 3,400 on Gemma 4 31B (NVIDIA); Cerebras at 981 on the trillion-parameter Kimi K2.6 (Artificial Analysis)
GB300 work per watt, fastest point against peak throughput (DeepSeek-R1)	118 against 11,781 tok/s per kW, about 100× less (OpenAI)

B:86 The measured table of §10.4, for a 1.6-trillion-parameter model with 49 billion active, about 100,000 tokens of context and 100 tokens a second, gives 0.75–1.2, 3.4–5.4 and about 7–16 agents per GPU on B200 HGX, GB300 NVL72 and Rubin NVL72, the last over Rubin's range of $2.8\text{--}4\times 10^5$ accelerators a utility gigawatt. At that configuration the roofline gives 4.59, 8.53 and 27.9 streams per GPU, 3.8–6.1 times the measured value on B200, 1.6–2.5 times on GB300 and 1.7–4.0 times on Rubin. The medians above include 55–80% serving utilization and count critical-IT power; per utility gigawatt, the basis of the measurements and a quarter more power, they are 0.29M, 1.05M and 3.9M, against measured ranges of 0.3–0.5M, 1.3–2.1M and 2.8–4.4M. They land near those ranges only because the larger-model prior and utilization offset the roofline's optimism: the counts agree with measurement after offsets and confirm nothing independently. In the model the count of ten million agents is likely by the end of 2027, while a speed of 10,000 tokens a second is beyond every GPU system (11:55).

B:87

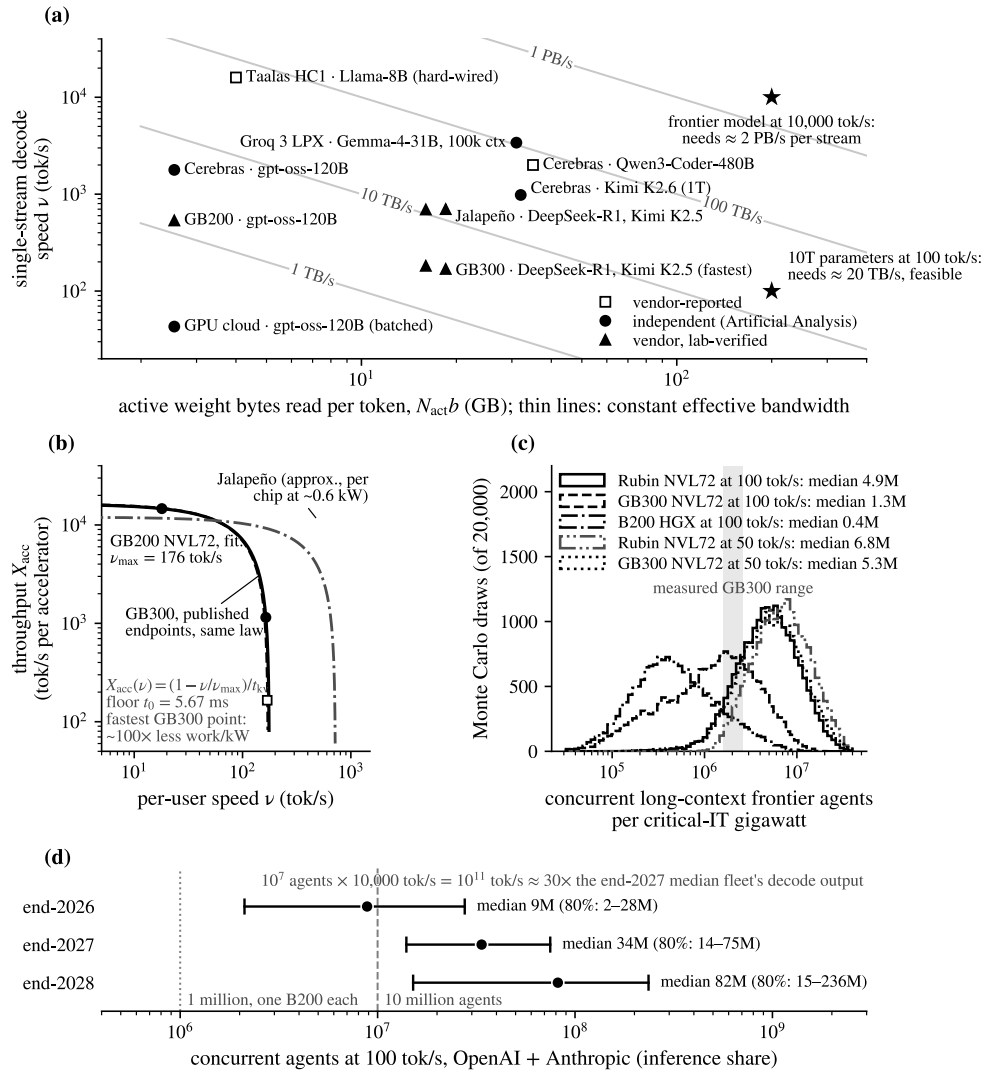


Figure B.5. Single-stream decode speed v against the active weight bytes read per token, $N_{\text{act}}b$, for measured systems, with lines of constant effective bandwidth (a); throughput per accelerator X_{acc} against per-user speed v for GB200 NVL72 (fitted), GB300 and Jalapeño (b); Monte Carlo distributions of concurrent long-context frontier agents per critical-IT gigawatt for five hardware and speed configurations, beside the measured GB300 range at 100 tok/s on the same basis (c); and the two largest labs' inference fleets at the end of 2026, 2027 and 2028, medians with 80% intervals (d). Speed times active size stays under a bandwidth line, throughput collapses as per-user speed nears the ceiling $v_{\text{max}} = 1/t_0$, and at 100 tok/s the model's GB300 median sits just left of the measured band, with only the 50 tok/s histograms to its right.

B:88 The roofline omits disaggregated prefill and decode, which would help; cache offload during tool waits, which keeps more agents alive than are decoding at any instant; and failures and stragglers. The synchronization time is calibrated on DeepSeek-R1's floor and assumed to carry over to larger models, and closed frontier models' sizes are unknown, hence the wide priors. Fleet gigawatts are critical-IT figures from public statements, good to about $\pm 30\%$, and the single-chip curves for GB300 and Jalapeño join published endpoints with the linear law.

B:89 The run draws 20,000 configurations per hardware and speed from NumPy's generator seeded with 7 and takes about 15 seconds.

B.9 The compute market

B:90 In a synthetic market of 200,000 workflow types at October 2026 list prices, 82.6% of workflows get the same answer from the tiny tier as from the frontier, yet removing that tier loses only 2.3% of surplus, because value per call decides what gets built. This is Experiment 3.1 in §3.5, plotted in Figure 3.1. The experiment asks which tier serves which work and how much is saturated, how much cheaper tiers and cheaper hands add in workflows and in surplus, and at what elasticity a falling price raises total spending.

B:91 Each workflow type t has a value per successful call u_t , n_t calls a year, prompt and completion lengths, a difficulty between 0 and 1, and a shape, a bounded decision or open generation. A tier m has list prices, which give a price per call $p_m(t)$, and a capability, which differs on decisions only for the tiny tier. The chance of a correct answer, $Q_m(t)$ as in (3.2), is logistic in capability minus difficulty with width 0.04, and a wrong call costs its value again to catch and redo, $c_{\text{fail},t} = u_t$. The yearly surplus before fixed costs is

$$\Pi_t(m) = n_t \left[Q_m(t) u_t - (1 - Q_m(t)) c_{\text{fail},t} - p_m(t) \right],$$

the surplus of (3.1) with success and failure added. A workflow takes its best tier and is built iff that surplus exceeds the hands c_{int} , the fixed cost of integrating it over a year, so the cheapest value per call worth building is about $p_m(t) + c_{\text{int}}/n_t$: the price of inference matters at high volume and the hands at low. A workflow is saturated at a tier when its difficulty is below that tier's capability, and the headline is the tiny tier's marginal share, the surplus lost when it leaves the full menu.

B:92 For the Jevons effect every price is multiplied by a common index, 1 at October 2026 list prices, and volume follows constant elasticity ε in that index; spend is then the index to the power $1 - \varepsilon$, as in (3.3), times a factor for what that condition leaves out, the workflows that cross the build threshold and those that move to a dearer tier. Every variant reuses the same random numbers, so each difference between variants comes from its parameters.

B:93

Menu	Tiers on offer	Hands
A	frontier	\$20,000
B	frontier, mid	\$20,000
C	frontier, mid, small	\$20,000
D	frontier, mid, small, tiny	\$20,000
E	as D, with agents building the pipelines	\$2,000 or \$200

B:94

Parameter	Value	Basis
Workflow types	200,000	
Frontier tier	\$10 in, \$50 out per million tokens; capability 0.97	GPT-6 Astra and Claude Fable 5.1 (Anthropic)
Mid tier	\$2, \$10; 0.85	Claude Sonnet 5.5 , GPT-6.1 Sol
Small tier	\$1, \$5; 0.70	Claude Haiku 4.5 , at 15–17 on the Artificial Analysis index against 53–58 at the top

Parameter	Value	Basis
Tiny tier	\$0.042 in, output free; 0.50, and 0.88 on decisions	<u>Jev</u> ; Jev matched a frontier model on triage readable off the input (200 of 200) and scored 0.79 against 1.00 on multi-step pay-or-hold checks (<u>distil labs</u>); capabilities are assumptions, and only their order is grounded
Prompt, completion	log-normal, medians 1,500 and 150 tokens, log standard deviation 0.7	average prompts grew from about 1,500 to over 6,000 tokens and completions from about 150 to 400 (<u>OpenRouter</u>)
Value per call	log-normal, median \$0.01, log standard deviation 2.0	assumption; at \$30 an hour, \$0.01 buys 1.2 seconds of human work; 2.3 million chats as the work of 700 agents imply about 3 minutes a chat (<u>Klarna</u>)
Daily volume	$\log_{10}$ normal, mean 3, standard deviation 1.2, truncated to $10\text{--}10^7$ calls	assumption; AT&T's 8 billion tokens a day, about 5 million calls at this prompt size, marks the top (<u>Bain</u>)
Decision-shaped share	0.55	an estimate that 40–70% of agent calls could go to specialized small models (<u>Belcak and others</u>)
Difficulty	Beta(1.6, 4.4), mean 0.27, for decisions; Beta(2.2, 2.8), mean 0.44, for generation	assumption that most enterprise calls are easy
Success width; failure cost	0.04; the call's value	assumptions, varied below
Hands C_{int}	\$20,000 a workflow over a year; swept \$2,000–\$320,000	about 0.1 engineer-year at \$200,000; 18% of US firms used AI in any function (<u>Census Bureau</u>), and one convenience sample ran 60% evaluated, 20% piloted, 5% in production (<u>MIT NANDA</u>)
Elasticity ϵ	0, 0.5, 1, 1.5, 2	within-model estimates of 1.08 and 1.11 bound the aggregate from above (<u>NBER</u>); the cross-model slope is 0.05–0.07 (<u>OpenRouter</u>); Bain's naive arc elasticity of 2.2 is not causal (<u>Bain</u>)
Price index	1 down to 10^{-3}	a fixed capability's price falls about 13 times a year (<u>Epoch</u>), so 1/13 is about a year and 1/169 two
Variant prices	Jev at \$0.42 in and out; GPT-6 Luna \$0.10, \$0.50; Claude Opus 5.5 \$4, \$20	a third-party gateway charges from \$0.42 per million input tokens with output free (<u>Jev pricing</u>), so the variant overstates its cost; <u>OpenAI</u> ; <u>Anthropic</u>
Latency variant	20% of workflows need an answer within 1 s, which only the tiny tier meets	Jev answers in 70–500 ms by the vendor's count and 236–276 ms at the median by a third party's (<u>systemonemodels.ai</u>); Haiku 4.5 at about 108 tokens a second needs about 1.4 s for 150 tokens; one second keeps flow uninterrupted (<u>Nielsen</u>)
Seed	20261003; all draws independent	

B:95

Tier, median call (1,500 in, 150 out)	Price per call	Compute asymmetry	Calls per \$10
frontier	\$0.0225	1	444
mid	\$0.0045	5	2,222
small	\$0.00225	10	4,444
tiny	\$0.000063	357	158,730

B:96 TypeSafe reports Jev as 444.6 times cheaper on its own workflow evaluations, in line with the 357 here. A million tiny-tier calls cost ten dollars only for prompts under about 240 tokens; at the median prompt they cost \$63, or \$630 through a third-party gateway (\$3.3).

B:97

Menu	Workflows built	Share built	Surplus, \$B a year	Inference spend, \$B a year	Tokens a day, T	Token share frontier / mid / small / tiny, %	Surplus per inference dollar	Effective compute asymmetry
A: frontier only	32,312	16.2%	150.4	19.96	3.92	100 / 0 / 0 / 0	7.5	1.0×
B: + mid	49,854	24.9%	171.1	8.29	7.69	1.6 / 98.4 / 0 / 0	20.6	4.7×
C: + small	54,431	27.2%	174.7	5.73	9.32	1.3 / 10.0 / 88.6 / 0	30.5	8.2×
D: + tiny	61,187	30.6%	178.9	2.16	12.57	1.0 / 5.9 / 10.4 / 82.8	82.7	29.1×
E: D with hands at \$2,000	112,011	56.0%	180.3	2.21	13.19	0.9 / 5.7 / 10.2 / 83.2	81.6	29.8×
E: D with hands at \$200	155,976	78.0%	180.5	2.22	13.33	0.9 / 5.7 / 10.1 / 83.3	81.4	30.0×

B:98

What the tiers and hands do	Value
Workflows saturated at the tiny, small and mid tiers	82.6% (67.9% if the same answer means success probability at least 0.99), 93.9%, 99.1%; only 0.9% need the frontier
Frontier-only market's spend on workflows the tiny tier answers equally well	82.5% of its inference bill, \$16.4B a year
All 200,000 workflows built	7.0B calls and 14.5T tokens a day, about half of Google's API traffic of 22B tokens a minute (Alphabet)
Surplus lost without the tiny tier (D to C)	2.3% (1.96–2.40% across five other seeds); without all three cheaper tiers, 15.9%
Tiny tier in D	82.8% of tokens and 60.1% of surplus served, 6.7% of inference revenue; frontier, mid and small earn 28.5%, 33.5%, 31.3%
Surplus gained from A to D	\$28.5B: \$18.1B cheaper inference on the 32,312 workflows built in A, \$10.4B from 28,875 new ones
Surplus gained from C to D	\$4.18B: \$3.56B (85%) savings on work the small tier served, \$0.62B (15%) from 6,756 new workflows
Median value per call	\$0.116 (built in A), \$0.041 (built in D), \$0.0035 (newly built by the tiny tier)
Hands cut to \$2,000 or \$200	50,824 or 94,789 more workflows; surplus +\$1.39B or +\$1.62B (+0.8%, +0.9%), of which new workflows add \$0.29B and \$0.41B
Building D at \$20,000 a workflow	about 6,100 engineer-years; the top 1% of built workflows hold 66.7% of surplus

What the tiers and hands do	Value
Phase map at 10 ⁴ calls a day (the median built workflow runs 11,437)	the tiny tier wins up to difficulty 0.43 on generation and 0.87 on decisions, where the mid and small tiers never win

B:99

Cheapest value per call worth building, difficulty 0.2	Tiny	Small	Frontier only
10 ³ calls a day	\$0.055	\$0.057	\$0.077
10 ⁴ calls a day	\$0.0055	\$0.0077	\$0.028
10 ⁶ calls a day	\$0.00012	\$0.0023	\$0.023

B:100 From A to D, inference spend falls 89% while surplus rises 19%, and surplus per inference dollar goes from 7.5 to 83: cheaper tiers move money from model providers to deployers. That surplus is the deployer's economic surplus; read as Marx's category, it is the temporary extra surplus value of §3.7. Cutting the hands 100-fold builds 2.5 times as many workflows and adds under 1% of surplus, because workflows that clear the threshold only when the hands get cheap are small by construction, and the value sits in the head, already built (§3.4).

B:101

Tiny tier's marginal share (share built in D), median value per call × hands	\$2,000	\$5,000	\$20,000	\$80,000	\$320,000
\$0.001	13.7% (27%)	13.5% (20%)	12.9% (11%)	11.9% (5%)	10.3% (2%)
\$0.003	6.4% (40%)	6.3% (31%)	6.1% (19%)	5.6% (10%)	5.0% (5%)
\$0.01	2.4% (56%)	2.4% (46%)	2.3% (31%)	2.2% (18%)	2.0% (10%)
\$0.03	0.9% (70%)	0.9% (60%)	0.9% (44%)	0.9% (29%)	0.8% (16%)
\$0.1	0.3% (82%)	0.3% (74%)	0.3% (59%)	0.3% (42%)	0.3% (27%)

B:102

One change from the baseline	Tiny tier's marginal share	Tiny tier's served share	All cheaper tiers' marginal share	Built, C → D	New workflows from the tiny tier
baseline	2.3%	60%	15.9%	27.2% → 30.6%	6,756
decision-shaped share 0.40	2.0%	49%	15.4%	27.1% → 30.1%	6,042
decision-shaped share 0.70	2.6%	78%	16.4%	27.4% → 31.1%	7,431
tiny capability on decisions 0.75	2.2%	58%	15.9%	27.2% → 30.5%	6,609
tiny capability on decisions 0.95	2.4%	61%	16.0%	27.2% → 30.6%	6,779

One change from the baseline	Tiny tier's marginal share	Tiny tier's served share	All cheaper tiers' marginal share	Built, C → D	New workflows from the tiny tier
tiny tier decides only, like Jev, and writes no text	1.8%	53%	15.5%	27.2% → 29.6%	4,831
tiny tier at \$0.42 in and out	1.6%	59%	15.3%	27.2% → 29.3%	4,216
small tier is GPT-6 Luna (\$0.10, \$0.50)	0.3%	56%	16.3%	30.8% → 31.2%	867
frontier is Opus 5.5 (\$4, \$20)	2.3%	60%	8.0%	27.3% → 30.7%	6,743
failure cost 0.25 of value	2.4%	62%	16.1%	27.4% → 30.8%	6,903
failure cost 4 times value	2.3%	59%	15.7%	26.9% → 30.2%	6,521
success width 0.02	2.4%	68%	16.4%	27.5% → 30.9%	6,910
success width 0.08	2.5%	54%	15.1%	26.5% → 29.8%	6,564
harder tasks (difficulty means +0.15)	2.4%	56%	14.6%	25.7% → 28.7%	6,103
value log standard deviation 1.5	6.4%	70%	39.3%	25.1% → 28.9%	7,484
value log standard deviation 2.5	0.7%	52%	4.8%	29.3% → 32.3%	5,951
value and volume correlated at −0.4	11.9%	73%	47.0%	21.1% → 25.8%	9,448
volume spread 0.8 in log ₁₀	2.0%	62%	14.7%	23.5% → 25.4%	3,839
longer prompts (6,000 in, 400 out)	6.5%	66%	31.9%	21.7% → 28.9%	14,584
20% of workflows need an answer within 1 s	18.8%	64%	30.1%	21.8% → 29.6%	15,704

B:103 Over the central block of the grid (median value \$0.003–\$0.03, hands \$5,000–\$80,000) the headline runs from 0.9% to 6.3%, and over the whole grid from 0.3% to 13.7%: it depends mostly on value per call and hardly on the hands, and the tiny tier's served share stays between 56% and 71%. The tiny tier decides where value per call is low, high volume comes with low value, prompts are long, or answers are due within a second, and support at very large scale has all four; it nearly vanishes beside a cheap generalist like GPT-6 Luna.

B:104

Elasticity ϵ	Spend $\times$, index 1/13	Spend $\times$, 1/169	Tokens $\times$, 1/169	Workflows built $\times$, 1/169	Spend $\times$, 1/1,000	Log-log slope of spend
0	0.173	0.024	1.09	1.05	0.0056	0.76

Elasticity ϵ	Spend $\times$, index 1/13	Spend $\times$, 1/169	Tokens $\times$, 1/169	Workflows built $\times$, 1/169	Spend $\times$, 1/1,000	Log-log slope of spend
0.5	0.638	0.319	14.9	2.05	0.183	0.25
1	2.32	4.17	195	2.86	5.80	-0.25
1.5	8.41	54.2	2,540	3.20	184	-0.75
2	30.4	704	33,000	3.26	5,800	-1.25

B:105

Further Jevons results	Value
Frontier's token share at $\epsilon = 1$, index 1 to 1/169	1.0% to 7.9%
Elasticity above which spend exceeds today's	0.72 at index 1/169; 0.67 at 1/13
Elasticity that reproduces Google's 330-fold token growth over the two years to May 2026 (Google)	1.10
Tokens at index 10^{-3} and $\epsilon = 2$	1.16 million times today's
Menu D's token flow	12.6T a day, 1.5×10^8 a second, mostly prefill on the small tiers; 195 times that at $\epsilon = 1$ and index 1/169

B:106 At $\epsilon = 1$ the price effect cancels, yet spend still rises 4.17-fold by 1/169, because more workflows clear the threshold and calls move up the tier ladder. The break-even elasticity is therefore 0.72, below the 1 of (3.3), because that build-and-upgrade factor grows roughly as the index to the power -0.25 . The best causal estimate, about 1.1 within models and an upper bound, lies above that threshold and the cross-model slope far below it, so the data leave the sign of the aggregate effect open (§3.6). The fit to Google's growth is not causal, since better models shifted demand in the same years, and constant elasticity is a local approximation, so the extreme cells are no forecast; at the far end only the build-out of §B.8 could serve the demand.

B:107 The distributions are assumptions, and the headline is sensitive to the tail: with a log standard deviation of 3.4 for value times volume, the top 1% of built workflows hold two-thirds of the surplus and dollar totals move about $\pm 10\%$ across seeds (\$169–207 billion for menu D), so shares and ratios are the outputs to read. Capability is one unmeasured scalar with no error floor. Value, volume and hands are independent and the hands uniform; hands that grew with value or difficulty, as real integrations often do, would bind the head too. Costs leave out routers, reasoning tokens, prompt caching, batch discounts and long-context surcharges. TypeSafe itself calls its measured savings optimistic and cannot rule out subsidized pricing (§3.3), and Claude 4.7 and later models produce about 30% more tokens for the same text ([Anthropic](#)). The baseline tiny tier is broader than Jev, also serving easy open-ended work at capability 0.5. The Jevons sweep moves all tiers together with one elasticity and ignores demand shifts from new capabilities, which drive most observed growth. The model holds verification fixed, while a cheap verifier would let a cascade run the tiny tier first and escalate on failure ([FrugalGPT](#); [RouteLLM](#)).

B:108 The run takes 40–50 seconds from seed 20261003.

References

1. Gary McGuire, Bastian Tugemann, Gilles Civario (2012). There is no 16-Clue Sudoku: Solving the Sudoku Minimum Number of Clues Problem. *arXiv*. <https://arxiv.org/abs/1201.0749>

2. Yato and Seta 2003. https://search.ieice.org/bin/summary.php?id=e86-a_5_1052&category=A&year=2003&lang=E&abst=
3. Cheeseman, Kanefsky and Taylor. <https://www.ijcai.org/Proceedings/91-1/Papers/052.pdf>
4. Scott Kirkpatrick, Bart Selman (1994). Critical Behavior in the Satisfiability of Random Boolean Expressions. *Science*. <https://doi.org/10.1126/science.264.5163.1297>
5. Hongyang Jia, Cristopher Moore, Doug Strain (2007). Generating Hard Satisfiable Formulas by Hiding Solutions Deceptively. *Journal of Artificial Intelligence Research*. <https://doi.org/10.1613/jair.2039>
6. Martin D. Davis, George W. Logemann, Donald W. Loveland (1962). A machine program for theorem-proving. *Communications of the ACM*. <https://doi.org/10.1145/368273.368557>
7. Marc Mézard, Riccardo Zecchina (2002). Random K -satisfiability problem: From an analytic solution to an efficient algorithm. *Physical review. E, Statistical physics, plasmas, fluids, and related interdisciplinary topics*. <https://journals.aps.org/pre/abstract/10.1103/PhysRevE.66.056126>
8. Vašek Chvátal, Endre Szemerédi (1988). Many hard examples for resolution. *Journal of the ACM*. <https://doi.org/10.1145/48014.48016>
9. Paul W. Beame, Richard M. Karp, Toniann Pitassi, Michael Saks (2002). The Efficiency of Resolution and Davis--Putnam Procedures. *SIAM Journal on Computing*. <https://doi.org/10.1137/S0097539700369156>
10. Achlioptas, Beame and Molloy. <https://utoronto.scholaris.ca/items/d7bc31bc-b6bd-4553-bd47-fac25150b3b4>
11. Carla Pedro Gomes, Bart Selman, Nuno Crato, Henry Kautz (2000). Heavy-Tailed Phenomena in Satisfiability and Constraint Satisfaction Problems. *Journal of Automated Reasoning*. <https://doi.org/10.1023/A:1006314320276>
12. Abraham D. Flaxman (2003). A spectral technique for random satisfiable 3CNF formulas. <https://dl.acm.org/doi/10.5555/644108.644166>
13. Shunyu Yao, Noah Shinn, Pedram Razavi, Karthik Narasimhan (2024). τ -bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains. *arXiv*. <https://arxiv.org/abs/2406.12045>
14. Artificial Analysis. <https://artificialanalysis.ai/evaluations/tau2-bench>
15. Klarna. <https://www.klarna.com/international/press/klarna-ai-assistant-handles-two-thirds-of-customer-service-chats-in-its-first-month/>
16. Bachant & McDermott 1984. <https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/download/445/381/0>
17. Joseph Weizenbaum (1966). ELIZA—A Computer Program For the Study of Natural Language Communication Between Man and Machine. *Communications of the ACM*. <https://doi.org/10.1145/365153.365168>
18. Irving John Good (1953). THE POPULATION FREQUENCIES OF SPECIES AND THE ESTIMATION OF POPULATION PARAMETERS. *Biometrika*. <https://doi.org/10.1093/biomet/40.3-4.237>
19. Lu Hong, Scott E. Page (2004). Groups of diverse problem solvers can outperform groups of high-ability problem solvers. *Proceedings of the National Academy of Sciences*. <https://doi.org/10.1073/pnas.0403723101>
20. Thompson 2014. <https://www.ams.org/notices/201409/rnoti-p1024.pdf>
21. Wenzhe Li, Yong Lin, Mengzhou Xia, Chi Jin (2025). Rethinking Mixture-of-Agents: Is Mixing Different Large Language Models Beneficial? *arXiv*. <https://arxiv.org/abs/2502.00674>
22. Jan Lorenz, Heiko Rauhut, Frank Schweitzer, Dirk Helbing (2011). How social influence can undermine the wisdom of crowd effect. *Proceedings of the National Academy of Sciences*. <https://doi.org/10.1073/pnas.1008636108>
23. Becker, Brackbill and Centola. <https://pmc.ncbi.nlm.nih.gov/articles/PMC5495222>
24. Cursor, January 2026. <https://cursor.com/blog/scaling-agents>
25. Cursor, February 2026. <https://cursor.com/blog/self-driving-codebases>
26. Cursor, July 2026. <https://cursor.com/blog/agent-swarm-model-economics>
27. Anthropic 2025. <https://www.anthropic.com/engineering/multi-agent-research-system>
28. Carlini 2026. <https://www.anthropic.com/engineering/building-c-compiler>
29. Forethought 2025. <https://www.forethought.org/research/how-quick-and-big-would-a-software-intelligence-explosion-be>
30. Forethought 2025. <https://www.forethought.org/research/will-compute-bottlenecks-prevent-a-software-intelligence-explosion>
31. Parker Whitfill, Cheryl Wu (2025). Will Compute Bottlenecks Prevent an Intelligence Explosion? *arXiv*. <https://arxiv.org/abs/2507.23181>
32. Anson Ho, Tamay Besiroglu, Ege Erdil, David Owen, Robi Rahman et al. (2024). Algorithmic progress in language models. *arXiv*. <https://arxiv.org/abs/2403.05812>
33. Epoch 2025. <https://epoch.ai/gradient-updates/most-ai-value-will-come-from-broad-automation-not-from-r-d>

34. InferenceX. <https://inference.semianalysis.com/blog/gb200-nv172-vs-b200-disagg-deepseek-r1-fp4-dynamo-trt>
35. OpenAI. <https://openai.com/index/jalapeno-first-results/>
36. 288 GB and 8 TB/s an accelerator. <https://www.nvidia.com/en-us/data-center/gb300-nv172/>
37. SpaceX S-1. <https://www.sec.gov/Archives/edgar/data/1181412/000162828026036936/spaceexplorationtechnologi.htm>
38. NVIDIA. <https://www.nvidia.com/en-us/data-center/vera-rubin-nv172/>
39. 180 GB. <https://www.nvidia.com/en-us/data-center/hgx/>
40. Dwarkesh Patel's podcast. <https://www.dwarkesh.com/p/dylan-patel-3>
41. March. <https://www.dwarkesh.com/p/dylan-patel>
42. EE Times. <https://www.eetimes.com/taalas-specializes-to-extremes-for-extraordinary-token-speed/>
43. NVIDIA. <https://nvidianews.nvidia.com/news/nvidia-groq-3-lpx-now-in-full-production-with-world-class-speed-for-agentic-ai>
44. Artificial Analysis. <https://artificialanalysis.ai/providers/cerebras>
45. GPT-6 Astra. <https://openai.com/index/gpt-6-astra/>
46. Anthropic. <https://platform.claude.com/docs/en/about-claude/pricing>
47. Claude Sonnet 5.5. <https://www.anthropic.com/claude-sonnet-5-5>
48. OpenAI. <https://openai.com/index/introducing-gpt-6-1-sol>
49. Claude Haiku 4.5. <https://www.anthropic.com/news/claude-haiku-4-5>
50. Artificial Analysis. <https://artificialanalysis.ai/leaderboards/models>
51. Jev. <https://typesafe.ai/blog/introducing-system-one-models-and-jev>
52. distil labs. <https://www.distillabs.ai/blog/jev-or-a-fine-tuned-small-model-we-built-a-pipeline-with-both-to-see-the-real-difference>
53. OpenRouter and a16z. <https://openrouter.ai/state-of-ai>
54. Bain. <https://www.bain.com/insights/how-token-economics-will-change-opex/>
55. Peter Belcak, Greg Heinrich, Shizhe Diao, Yonggan Fu, Xin Dong et al. (2025). Small Language Models are the Future of Agentic AI. *arXiv*. <https://arxiv.org/abs/2506.02153>
56. Census Bureau. <https://www2.census.gov/library/working-papers/2026/adrm/ces/CES-WP-26-25.pdf>
57. MIT NANDA. <https://www.grantthornton.sg/globalassets/1.-member-firms/singapore/pdf-articles/the-genai-divide---state-of-ai-in-business-2025.pdf>
58. Mert Demirel, Andrey Fradkin, Nadav Tadelis, Sida Peng (2025). The Emerging Market for Intelligence: Pricing, Supply, and Demand for LLMs. *National Bureau of Economic Research*. <https://www.nber.org/papers/w34608>
59. Epoch. <https://epoch.ai/publications/the-plunging-price-of-thought>
60. gateway pricing. <https://jevtypesafeai.com/pricing>
61. GPT-6 Luna. <https://openai.com/index/introducing-gpt-6-sol-and-luna/>
62. Anthropic. <https://www.anthropic.com/claude-opus-5-5>
63. systemonemodells.ai. <https://systemonemodells.ai/typesafe-ai/jev>
64. Nielsen. <https://www.nngroup.com/articles/response-times-3-important-limits/>
65. Alphabet. <https://blog.google/company-news/inside-google/message-ceo/alphabet-earnings-q2-2026/>
66. Google, 2026. <https://blog.google/innovation-and-ai/sundar-pichai-io-2026/>
67. Lingjiao Chen, Matei Zaharia, James Zou (2023). FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance. *arXiv*. <https://arxiv.org/abs/2305.05176>
68. Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu et al. (2024). RouteLLM: Learning to Route LLMs with Preference Data. *arXiv*. <https://arxiv.org/abs/2406.18665>