The Maze as Machine

Why Current AI Is a Detour, What the Maintenance Ratio Says About It, and What Replaces the Model

J. Konstapel — Leiden, 18 August 2026

Context Header

Programme. Vacuum.Net is the model of the whole net. MAZE applies it. This piece extends MAZE from the human to the machine.

This piece. It asks one question: does the knot translation of Vacuum.Net apply to current artificial intelligence? It answers yes, with three corrections to the earlier Right Brain AI formulation, one measurement that can be run on already-published numbers, and one architecture that follows.

Settled. The axioms N1–N5. Theorem T7 and its corollaries. The maintenance ratio $R = r/\lambda$ and the bound $C \leq Rf$. The capacity ladder 1, 4, 13, 40, 121. The MAZE kernel in code, with exact tests.

Open. O2, the rewrite dynamics. Without it the kernel addresses but does not move.

Conventions. Five statuses: assumed, conditionally proved, corresponded, predicted, open. Every number needed to follow a calculation is in the text.

Corpus. Paths to the Knot (Complete Edition, 17 August 2026); The Vacuum.Net Theory, Foundational Paper, Edition 6 (9 August 2026); the Right Brain AI series (November–December 2025).

1. The Question

Right Brain AI was written in November 2025. It said that current language models are the left hemisphere only. It said they approach a ceiling that more computation cannot lift. It proposed an alternative built on resonance, on a nilpotent kernel, and on light.

That was intuition. It had no formalism under it.

The formalism now exists. Vacuum.Net gives five axioms. Paths to the Knot translates them into a general theory of maintenance, with two structural quantities and one ratio. The question is whether that machinery says anything about the machines we actually have.

It does. It confirms the ceiling. It relocates it. And it removes one of the three pillars of the original proposal.

2. A Trained Model Is a Maintained Pattern

Picture the net as a fishing net. Strands run through it. Where a strand touches itself there is a contact. A mesh is the smallest closed unit. What holds a mesh together is not the material of the twine. It is the fact that the crossings are continually re-laid.

A trained model is such a net.

The form is the set of connections that carry a capability. Flow is gradient traffic. Where flow runs, connections are re-laid. Where no flow runs, connections drift out of the pattern under updates driven by other material.

This is not an analogy imported for decoration. The field already measures both halves of it, under two names.

Decay. Train a model further on task B and its score on task A falls. The curve is published routinely. It is called catastrophic forgetting. In the vocabulary of the net it is λ, the rate at which an idle contact lapses.

Restoration. Mix a small fraction of the old material back into the training stream and the score on task A returns. This is called replay or rehearsal. In the vocabulary of the net it is r, the rate at which a lapsed contact is restored.

Both rates appear in the same papers. They are plotted next to each other. They are never divided by each other.

Status: corresponded. The identification of forgetting with λ and replay with r is declared, not proved. It can fail. What follows is what happens if it holds.

3. The Measurement

Define the maintenance ratio:

$R \equiv r / \lambda$

It is dimensionless. It compares restoration against lapse. It says nothing about how fast the system runs.

The maintenance condition from the seven-step construction is:

$f \geq C / R$

Here C is the number of contacts required by the maintained pattern, and f is the fraction of time available for repair — the fraction of the training stream given over to re-laying rather than to new material.

Turn it around. The minimum viable repair fraction is:

$f_{min} = C / R$

This is the step that matters, because f_min is already published. It is reported in every continual-learning paper as a practical setting: the smallest replay percentage that still holds the old tasks. It is reported as a hyperparameter. It is never read as a measurement of anything.

The calculation, in full

Suppose holding one task requires a replay fraction of 2 percent. Then, with C = 1:

R = C / f_min = 1 / 0.02 = **50**

Every further prediction follows from that single number.

- Holding 10 tasks: f = 10 / 50 = 0.20. Twenty percent of the stream must be replay.
- Holding 25 tasks: f = 25 / 50 = 0.50. Half the stream.
- Holding 40 tasks: f = 40 / 50 = 0.80.
- Holding 50 tasks: f = 50 / 50 = 1.00. The whole stream is replay. Nothing new can be learned.
- Above 50 tasks: no replay fraction exists. The pattern cannot be held at this R.

That is a hard ceiling on the number of capabilities a system can maintain at once, at fixed R. It is not a ceiling on size, speed, or budget.

The test

Take published continual-learning tables. Plot the minimum viable replay fraction against the number of tasks held.

The prediction is a straight line through the origin. Its slope is 1/R.

If the line curves, the prediction fails. If it does not pass through the origin, the prediction fails. If different architectures give different slopes but each gives a straight line, then R is an architectural property and can be engineered.

Status: predicted. No new experiment is required. The numbers exist. They have to be put on one denominator.

Note what this avoids. It never asks how many connections a model has. C is left as a count of maintained tasks, and R comes out of the slope. The absolute count, which nobody can supply, is not needed.

4. Why Computation Buys the Wrong Quantity

Now the part that turns a measurement into a diagnosis.

Re-laying is imperfect. As a pattern grows, a replacement contact is more likely to land outside the required arrangement. Let the probability of correct placement fall as C^(–a). Let μ be the number of contacts re-laid per unit of flow. The maintenance condition reads:

μ r > λ μ (1 – C^(–a)) C

μ appears on both sides. It cancels.

R > C – C^(1–a)

To leading order:

C ≤ R

The largest maintainable form is set by the ratio of restoration to lapse. The absolute re-laying rate has vanished from the result.

This is the whole argument, and it is short.

Computation buys μ. More processors, more parameters, more passes: all of it raises the rate at which connections are laid. And μ is exactly the quantity that drops out of the bound.

The industry is purchasing the term that cancels.

The observation

On 14 August 2026 a model was released that makes this visible. GLM-5.3 shares its base model with its predecessor. Nothing was added to the substrate. All gains came from extended post-training.

The reported result was roughly 50 percent better coding performance. On the vendor's own benchmark it scored 31.4 percent against 29.5 percent for the strongest closed competitor, while using about 50,000 output tokens per task against roughly 120,000.

Read that in the vocabulary of the net. Capacity unchanged. μ unchanged. Better maintenance of the pattern, and a shorter path to closure.

The vendor's benchmark numbers are its own and have not been independently reproduced. That does not affect the structural point, which rests on the architecture of the release rather than on the size of the gain: the base was not touched.

Status: corresponded. One case is not a demonstration. It is the first clean instance where the two quantities were separated in public.

5. Depth Buys Little

Closure gathers q constituent forms and connects them with ζq new contacts. The recurrence is:

$C(k+1) = q\,C(k) + \zeta\,q$

With $B = \zeta q/(q-1)$ and $A = C(0) + B$, the solution is $C(k) = A\,q^k - B$. Form grows geometrically with depth. The maintenance bound does not.

The maximum depth is therefore:

$$\mathbf{m = \lfloor \log((Rf + B)/A) / \log q \rfloor}$$

Depth grows with the logarithm of maintenance capacity. Doubling R does not buy two more levels. It buys a fraction of one.

In the ternary case, where each site carries three states and each closure gathers three, the capacity at depth n is:

$(3^n - 1)/2 =$ **1, 4, 13, 40, 121, 364**

These are not smooth. There is no depth two-and-a-half. A system either has the capacity to close another level or it does not.

The consequence for machines is direct. Stacking agents on agents, supervisors on supervisors, planners over executors — this is closure, and it obeys the logarithm. The tower is short whatever the budget. Systems that try to grow by adding layers of orchestration will find the ladder ends quickly, and it ends at a rung, not halfway between two.

Status: conditionally proved, on the axioms N1 and N5, and on the closure recurrence.

6. Where the Cost Actually Is

Before the architecture, the arithmetic of the detour.

A chain of steps between question and answer costs at every transition. If a fraction q survives each step, then after k steps a fraction q^k survives. Take q = 0.85, which is generous:

- 10 steps: 0.85^10 = 0.197. About 20 percent remains.
- 20 steps: 0.85^20 = 0.039. About 4 percent remains.

Waste sits in the number of steps. Optimising a step does not touch the exponent. Shortening the chain does.

Seven reductions follow, ordered by yield. All are documented practice. None requires a better model.

One. Put the knowledge outside the model. A model that stores every fact in its weights must be large. A small model that looks facts up in a database need not be. Reported yield: a factor of 10 to 100 in model size at comparable answer quality.

Two. Do not use a model where a formula suffices. A birth chart computed from ephemerides. A tension score computed from index prices. A match computed from a table. These are exact, reproducible and nearly free. Yield: everything. The model leaves the chain entirely.

Three. Fewer bits per weight. Sixteen bits to four makes the model four times smaller with limited loss. On the alphabet $\{-1, 0, 1\}$ multiplication disappears altogether; addition and subtraction remain. That is not a percentage saving. It removes an arithmetic unit.

Four. Do not activate every parameter for every token. A mixture of experts activates a fraction of the net per token. Yield: a factor of 5 to 20 in computation at equal size.

Five. Distil. A large model trains a small one on its own output. On a bounded task the small model reaches comparable scores. Yield: a factor of 10 to 100. This is the route when one task must be done well rather than everything done adequately.

Six. Shorten the answer path. Fifty thousand tokens instead of a hundred and twenty thousand for the same result. Yield: a factor of 2 to 3, applied at every call.

Seven. Route. A small model answers most questions; the large one is called only when the small one fails. If 85 percent goes to a model ten times cheaper, the effective cost is 1 / (0.85/10 + 0.15) = 1 / 0.235 = **4.25 times less**.

They compound. Take mixture of experts at 8, token efficiency at 2.4, and routing at 4.25:

$8 \times 2.4 \times 4.25 = \mathbf{81.6}$

A factor of eighty, before distillation and before the first two reductions, which do not shrink the model but remove it. With those, two orders of magnitude is not an optimistic figure. It is the arithmetic.

The reason this does not happen is not technical. Selling capability by the unit of size gives no one an interest in demonstrating that a small model with a database does the same work.

Status: established practice. Each reduction is separately documented. The compounding is arithmetic.

7. Right Brain AI, Reconsidered

Three claims were made in 2025. The translation keeps one, rewrites one, and drops one.

The ceiling: rewritten

The original claim was that language models approach an energetic ceiling. The ceiling is real. It is not energetic.

Energy buys μ, and μ cancels. The bound is $C \leq Rf$. A system can be fast and fail; a slow system with adequate restoration persists. The diagnosis moves from thermodynamics to a ratio.

This is stronger than the original claim, and it is testable in a way the original was not. Section 3 is the test.

The nilpotent kernel: rewritten

The original claim was that a nilpotent algebra makes contradiction energetically forbidden, so that faults become physically impossible.

In the axioms this is N2 and correspondence C3. Only configurations that close persist. What does not close, does not last. There is no external selector and no energy barrier. It is selection by fit.

That distinction matters in practice, because selection by fit can be implemented in ordinary code. It already is: the gate in SWARP admits a proposal only when its residue falls below a threshold, and human and machine proposals pass through the same gate. What was posed as a property of exotic hardware turned out to be a rule that runs on ordinary hardware.

The optics: dropped

The original claim was that light is the necessary substrate, and an architecture of nineteen layers was built on it.

The knot framework does not support this. It says the opposite. Identity travels with the re-laying of a pattern into the same class, whatever the carrier. The clearest existing case in this very domain is quantisation: sixteen bits to four, the material changed drastically, the capability retained. The capability lives in the ratios, not in the medium.

Optics may still be a good engineering choice. It is not a requirement of the theory, and the theory supplies no argument for it. The nineteen-layer stack was one realisation among many.

What was already right, and is now arithmetic

The ternary alphabet. The cost of representing values in radix r is proportional to r / ln r:

- r = 2: 2 / ln 2 = 2.885
- r = 3: 3 / ln 3 = 2.731
- r = 4: 4 / ln 4 = 2.885
- continuous optimum: r = e = 2.718

Three is the integer nearest the optimum. Three beats two by about five percent in representation cost, and by considerably more in circuitry, because single-trit multiplication on $\{-1, 0, 1\}$ closes with no carry at all.

This is no longer a proposal. Ternary-weight networks on exactly the alphabet $\{-1, 0, 1\}$ are current engineering practice. The trinity arrived in machine learning from the cost side, without anyone deriving it.

8. MAZE as Machine

Now the architecture.

Address, not distance

A language model has no addresses. It has distances. It knows that two things resemble each other. It does not know where either of them is.

Everything expensive about it follows from that. Similarity must be computed, and computing it requires the whole apparatus. There is no way to look something up, because there is no place.

MAZE inverts this. Theorem T7 states that every finite net state at depth n corresponds to exactly one trit string of length n, and every integer winding sum has exactly one finite balanced-ternary representation. One state, one address. No exceptions and no collisions.

Four properties follow, and each of them replaces machinery:

Identity across scale is a prefix. Shifting the string one place multiplies by three. The same pattern read at another depth is the same string with leading zeros. Coarse and fine are the same object, not two models.

The opposite is the mirror. Negation is swapping 1 and T throughout. There is no minus sign, because the sign is the sign of the leading nonzero trit. The counterpart of a pattern requires no separate computation.

Truncation is rounding. Cut the expansion after position m and the discarded tail is at most half of the last kept unit. Reading at a chosen depth is automatically the nearest value at that depth. No rounding convention is imported.

Comparison is arithmetic. Two states compare as two integers. Distance is subtraction. Compatibility is a mirror test. None of this is learned, and none of it can drift.

That is why it can be cheap. Not because the model is smaller. Because there is no model. The structure does the work. It is reduction one and reduction two of Section 6, taken to their limit.

What already exists

The kernel is written. It is 114 lines with no database dependency, and 96 lines of tests that reproduce the paper's numbers exactly. It uses arbitrary-precision integers so that the bijection stays exact at any depth. A turn is over, straight, or under: +1, 0, −1. A route is a sequence of turns. A route converts to an integer and back.

The worked example from the paper: the route (+1, −1, −1, −1, +1) gives

81 − 27 − 9 − 3 + 1 = **43**

Rest is address zero. There are 3^n routes at depth n, with a ceiling of (3^n − 1)/2.

The kernel currently has zero callers from the rest of the application. The map is drawn. The journey has not started. That order was deliberate.

What is missing

O2: the rewrite dynamics.

The kernel can address. It cannot move. There is no rule that says which rewrite follows which state. Without it, MAZE is a filing system with an exceptional index and no engine.

The form of the missing rule is known, and it is not the usual one. The question is not which quantity the net minimises. An action principle is an inherited external mover, and the net has no outside. The question is which rewrite dynamics is its own fixed point.

That is the real work. It belongs on paper before it belongs in code.

Status: open. This is the single item on which the architecture depends.

The first step

One existing object gets a MAZE address: the life course.

A life course becomes a route. Turns at a chosen depth, one string, one integer. Two life courses then compare as numbers rather than as narratives. The mirror gives the exact counterpart, which is the compatibility test, and it costs a trit-wise swap.

This is the smallest intervention that turns the map into a route. It requires no rewrite dynamics, because it addresses a history rather than predicting a next state. It uses the kernel that already exists and already passes its tests.

And it demonstrates, on one concrete case, the thing no language model can do: exact comparison of two structures across scale, with no training, no drift, and no similarity metric.

9. Status Ledger

Statement	Status
Axioms N1–N5	assumed
T7, unique address; corollaries T7.1–T7.3	conditionally proved
Maintenance bound C ≤ R; f ≥ C/R; depth m	conditionally proved
Capacity ladder 1, 4, 13, 40, 121	conditionally proved
Forgetting = λ; replay = r	corresponded
GLM-5.3 as an instance of μ held constant, R raised	corresponded
f_min against task count is a straight line of slope 1/R	predicted
Depth-ordered failure in machine systems	predicted
At equal total computation, distributed replay retains more than continuous training	predicted
The seven reductions and their compounding	established
O2, the rewrite dynamics	open
Optics as a requirement	withdrawn

10. Annotated References

Konstapel, J. (2026). ***The Vacuum.Net Theory: Foundational Paper*****, Sixth Edition. Leiden, 9 August 2026.** The axioms N1–N5, theorem T7 with the uniqueness proof, the ternary ground with the radix-economy calculation, and the five-status system. *Why read?* Sections 2, 5, 7 and 8 of this article rest on it directly. Part II.1 supplies the radix numbers; Part II.3 supplies T7 and the capacity ladder. *Reading advice:* Part II and Part X first. The rest can wait.

Konstapel, J. (2026). ***Paths to the Knot: Complete Edition*****. Leiden, 17 August 2026.** The general theory of maintenance: form, depth, the ratio R = r/λ, the seven-step construction, and the mapping onto sixty-five disciplines. *Why read?* Sections 3, 4 and 5 are applications of its central results to one domain that is not among the sixty-five. *Reading advice:* the condensed theory paper covers the construction in twelve pages. The complete edition is for the disciplinary chapters.

Konstapel, J. (2025). The Right Brain AI series. constable.blog, 23 November – 17 December 2025. The original intuition: language models as left hemisphere, the energetic ceiling, resonance and the nilpotent kernel as alternative, optics as substrate. *Why read?* Section 7 revises it. The value of the series is that it stated the ceiling before there was a formalism to locate it.

Eigen, M. (1971). Selforganization of matter and the evolution of biological macromolecules. ***Naturwissenschaften*** **58(10), 465–523.** The error threshold: a bound on maintainable information under copying error and correction. *Why read?* It is the cleanest independent precedent for C ≤ R, derived in molecular genetics without any knowledge of this framework. If the bound is real, it should have been found twice, and it was.

Shannon, C. E. (1948). A mathematical theory of communication. ***Bell System Technical Journal*** **27, 379–423, 623–656.** Reliable retention over a noisy channel through continuous correction. *Why read?* The general form of the same object. Retention is an operation, not a property of a container.

McCloskey, M. & Cohen, N. J. (1989). Catastrophic interference in connectionist networks. ***Psychology of Learning and Motivation*** **24, 109–165.** The original description of what is now called catastrophic forgetting. *Why read?* This is where λ enters the machine literature. Note that it was named as a defect rather than as a rate.

Robins, A. (1995). Catastrophic forgetting, rehearsal and pseudorehearsal. ***Connection Science*** **7(2), 123–146.** Rehearsal: reintroducing a fraction of old material to restore lost performance. *Why read?* This is where r enters. The two papers are thirty years old and have never been divided by each other. *Reading advice:* read it directly after McCloskey & Cohen. The pair is the whole argument of Section 3.

Kirkpatrick, J. et al. (2017). Overcoming catastrophic forgetting in neural networks. ***PNAS*** **114(13), 3521–3526.** A widely used method for holding old capabilities while learning new ones, with task-count tables. *Why read?* This is the kind of table the straight-line test of Section 3 is run on. The numbers are already published.

Knuth, D. (1997). ***The Art of Computer Programming*****, Vol. 2, pp. 195–213.** Balanced-ternary arithmetic, including the equivalence of truncation and rounding. *Why read?* Section 8 uses that equivalence. Knuth remarks that the symmetric properties may one day prove important.

Ma, S. et al. (2024). The era of 1-bit LLMs: all large language models are in 1.58 bits. arXiv:2402.17764. Neural networks with weights restricted to $\{-1, 0, 1\}$. *Why read?* The ternary alphabet is not a proposal. It runs, and it was arrived at from the cost side.

Krinitsky, Mironov & Frolov (1963). Program-controlled machine Setun. The balanced-ternary computer built in Moscow in 1958. *Why read?* Hardware precedent. The notation has been executed before, and it worked.

Chaitin, G. (2005). ***Meta Math! The Quest for Omega*****.** A theory is a program shorter than the data it produces. *Why read?* It supplies the measure by which architectures can be compared without an external judge. It also proves that no one can demonstrate their own program to be the shortest, which is a useful discipline.

Hornborg, A. (2019). ***Nature, Society, and Justice in the Anthropocene*****.** Long chains, priced endpoints, and unpriced transitions. *Why read?* The economic twin of the exponent law in Section 6. The same structure, in a different vocabulary.

Next: the Dutch blog explaining this article.